Learning Weakly Structured Representations for Text-to-Text Generation

Tom Hosking

Doctor of Philosophy
Institute for Language, Cognition and Computation
CDT Natural Language Processing
School of Informatics
The University of Edinburgh
2024

Abstract

Text-to-text generation refers to a class of problems that involve transforming one piece of text to another, such as paraphrase generation, summarisation and automatic translation. Deep learning approaches to text-to-text generation first map a natural language utterance to some learned representation, perform some processing within this representation space, then map the modified representation back to natural language. Currently, the majority of such models use an unstructured sequence of dense vector embeddings that is fully learned from data as the representation. This data-driven approach has proven successful and requires little guidance from a model designer, but the resulting representations are not easily interpretable and do not exploit known properties of the task under consideration (e.g., for paraphrase generation, the meaning and form of an input sentence should be treated separately).

In this thesis, we hypothesise that choosing a *weakly structured* representation is a better approach. The structure should encode the aspects of the tasks that are known, but remain sufficiently flexible that the unknown aspects may be learned. We argue that discrete and hierarchical representations make some aspects of text-to-text generation tasks more feasible, enabling models that are attributable and scale to longer inputs. Finally, we hypothesise that structure alone is not sufficent, and that some degree of supervision is needed to assign meaning to a structured representation. We focus on two text-to-text generation tasks to gather support for these hypotheses: paraphrase generation, where a model must generate an output sentence with the same meaning but different surface form to a given input sentence; and opinion summarisation, which involves generating a textual summary that aggregates popular opinions from customer reviews about hotels or other products.

We begin by proposing a model for paraphrase generation that represents the meaning and surface form of an input separately, with the surface form represented as a set of discrete codes learned through Vector Quantisation (VQ-VAE). We show that this weakly structured choice of representation enables us to generate high quality paraphrases by keeping the semantic representation constant and varying the syntactic representation, supporting our first hypothesis. We use a denoising objective based on distant supervision to induce the separation between representations. Next, we address the lack of a tractable factorisation in VQ-VAE, and introduce Hierarchical Residual Quantisation (HRQ-VAE), a method for learning *hierarchical* discrete representations of input data, and show that it learns more informative representations than VQ-VAE.

We then combine the hierarchical representations of HRQ-VAE with separated encoding spaces for paraphrase generation, showing that the more richly structured choice of representation leads to improved quality of generated paraphrases. To demonstrate that HRQ-VAE can be beneficial for more complex text-to-text tasks, we apply it to opinion summarisation, representing sentences from customer reviews as paths through a learned hierarchy. We show that we can generate informative summaries of these reviews that are attributable and scale to large numbers of reviews, by identifying which paths in the hierarchy are frequently attested across each set of reviews. Finally, we combine the scalability and attributability of hierarchical representations with the fluency and coherence of Large Language Models, and use an encoder based on HRQ-VAE to build a hierarchical index over review sentences that may then be used to retrieve clusters of sentences containing popular opinions. We use distant supervision based on entailment relations to induce a semantic ordering to the learned hierarchy and show that the hierarchy directly enables the scalability and attributability of our model.

Overall, our experiments act as support in favour of our hypotheses that weakly structured representations are beneficial for text-to-text generation, that discrete and hierarchical representations are a powerful choice of structure, and that distant supervision is needed to assign meaning to the structures.

Lay Summary

Text-to-text generation is about taking one piece of text and turning it into another. This could mean creating a paraphrase, a summary, or translating it into a different language. Current systems try to understand the text by turning it into a kind of code or *representation*, doing some work on that representation, and then turning it back into natural language.

Most systems use a type of representation that is essentially a big soup of numbers that is learned automatically from data. This works well, but it is hard to understand what the representation means, and it doesn't make use of the fact we know some things about how the task should be done. For example, when creating a paraphrase, we should treat the meaning and structure of a sentence separately.

In this thesis, we argue that a *weakly structured* representation is better. Weak structure means capturing the important aspects of the task that we understand, while remaining flexible enough to let models learn the rest. We argue that a structure that is discrete and hierarchical can make some parts of text-to-text generation easier. This type of code allows us to understand what evidence the system is using to generate its output, and allows it to handle much longer input texts.

We test our hypotheses on two tasks: paraphrase generation and opinion summarisation. For paraphrase generation, we propose a system that separates the meaning and structure of a sentence into different representations. Then, we can create new sentences with the same meaning but different structures by leaving the representation of the meaning fixed and changing the representation of the structure. For opinion summarisation, we design a model that places sentences from customer reviews into a hierarchy. Then we can identify common themes in the reviews and create informative summaries by looking at which parts of the hierarchy are most frequently used. By combining our methods with existing Large Language Models, we can create summaries that capture important opinions in customer reviews that are also fluent. Our experiments show that using a structured code improves text-to-text generation tasks.

Acknowledgements

I would first like to thank Mirella Lapata, my primary supervisor, for her guidance over the years. Her passion for guiding, supporting and nurturing students to become excellent independent researchers is inspirational. Thank you for being so generous with your time, for always pushing us to be the best we can, and for always having our best interests at heart. I also thank Hao Tang, my second supervisor, for his support throughout the thesis and for his uncanny ability to find a simple solution to every knotty problem. Thank you to Sebastian Riedel, my MSc dissertation supervisor, for getting me hooked on research in the first place and for supporting my first trip to an academic conference where I really caught the NLP bug. Thanks must also go to the rest of the Bloomsbury AI team: Max, Patrick and Guillaume, I hope to work with you again.

I owe many thanks to all the people at Edinburgh who made the PhD such an enjoyable and stimulating time. To Stefanos Angelidis, who patiently listened and encouraged me to explore some of the early ideas in the thesis. To Seraphina Goldfarb-Tarrant, for always bringing a fresh perspective and getting me re-enthused about a topic when I've had enough. To Henry Conklin, for innumerable fascinating discussions over coffee during the pandemic and for bringing a much-needed linguistics perspective. NLP needs more Henrys. To Georgia Carter, for all the tunes and venting. To Nina Markl, for making NLP a better and kinder place. To James Owers, for your boundless and infectious enthusiasm. To all those in Mirella's research group over the years: Yumo Xu, Reinald Amplayo, Ratish Puduppully, Hao Zhang, Nelly Papalampidi, Tom Sherborne, Parag Jain, Agostina Calabrese, Irina Saparina, Laura Perez-Beltrachini. To Sally Galloway, our infinitely patient CDT administrator. To John Soulsby, for many excellent coffee outings and bike rides to decompress. And to Nile Valley, for quite literally sustaining me through the PhD.

I thank my examiners, Kathleen McKeown and Antonio Vergari, for their thoughtful and detailed feedback.

Finally, this thesis would not have been possible without my wife, Alice. Thank you for everything.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

[illegible]

(*Tom Hosking)*

Table of Contents

Chapter 1

Introduction

Text-to-text generation may be defined as “the process of producing a natural language text in order to meet specified communicative goals” based on a piece of input text (Dong et al., 2022). Text-to-text generation enables access to information in a human-interpretable format, with a wealth of possible applications. For example, summarisation involves condensing a long input article into a shorter, easier to read output summary that contains the most important information (e.g., Banko et al., 2000; Rush et al., 2015; See et al., 2017; Goyal et al., 2022)[1]; machine translation transforms an input sentence from one language to another while preserving the original meaning (Brown et al., 1993; Koehn et al., 2007; Kalchbrenner and Blunsom, 2013); question answering seeks to generate the answer to an input query, optionally grounding the answer in an associated document(s) (Hirschman et al., 1999; Ng et al., 2000; Rajpurkar et al., 2016; Seo et al., 2017; Fan et al., 2019); question generation considers the inverse problem, taking an answer as input and generating a possible question that might lead to that answer (Heilman and Smith, 2010; Du et al., 2017; Hosking and Riedel, 2019; Narayan et al., 2023). In this thesis, we restrict our focus to generating text from textual inputs, but the overarching ideas apply when generating text from any modality.

Text-to-text generation involves a number of challenges, some of which are shared with *understanding* natural language. In both cases, the meaning of a sentence or utterance is a complex (and unknown) function of the parts of the sentence; indeed, it is not clear how granular the parts are, with English words having shared lemmas and other languages having even richer morphological structure. In many languages, including English, word order matters. While understanding natural language requires

[1]A complete literature review is beyond the scope of this thesis; we include here a selection of papers that were influential on the author.

learning a mapping from surface form to 'meaning', for text-to-text tasks this mapping must also be invertible, allowing us to map from meaning back to surface form.

In this thesis, we will consider the effects of the choice of mapping between the surface form of textual inputs and outputs, and the model-internal representations used to perform a task. A text-to-text system must first 'read' the input, then 'understand' and process it, before finally 'writing' a generated output. We will examine how the choice of internal representations affects models' ability to process inputs and perform useful tasks. This is analogous to the study of algorithms and data structures in classical computer science; the way that data is organised by a system will heavily impact the choice of algorithm used to process it, and vice-versa. Designing an appropriate data structure with particular properties may lead to a natural solution to a problem. For example, Dijkstra's algorithm for finding the shortest path between nodes of a weighted graph employs a min-priority queue data structure for efficiently keeping track of the shortest currently known paths. The CKY algorithm for parsing uses a chart to store intermediate results, enabling an efficient dynamic programming solution to the problem. We will show that choosing intermediate representations with an appropriate structure can facilitate text-to-text generation tasks.

The symbolic-connectionist spectrum Broadly speaking, approaches to generating natural language (and indeed other machine learning problems) fall somewhere on a spectrum between two extrema. At one end, the connectionist approach uses neural networks as function approximators. In theory, neural networks may be learned entirely from data, can approximate any function to arbitrary precision (given sufficient, potentially infinite, parameters capacity and computational cost; Hornik et al., 1989), and require minimal design. However, they are also difficult to interpret, may not generalise in the desired manner, and may require large quantities of data to find a useful solution to a problem. At the other end of the spectrum, symbolic approaches are based on rules acting on symbolic units. They are therefore inherently interpretable, generalise in a known way, and are data-efficient (Russell and Norvig, 2010; Garcez and Lamb, 2023). However, they are brittle; any input sample that does not match the expectations of the system designer is likely to fail completely. They also require significant effort to design, with more complex and expressive systems requiring correspondingly more effort.

Current neural approaches to text-to-text generation fall towards the connectionist end of this spectrum. They are based on neural networks that are learned from data, but the splitting of input text into tokens and the self-attention mechanism used in Transformers, a currently popular architecture (Vaswani et al., 2017), provide a weak inductive bias that narrows the possible solution space and makes them tractable in practice. Nonetheless, most current text-to-text models represent natural language utterances or tokens as points in a dense vector space. Whether or not it is possible to 'cram the meaning of a whole [...] sentence into a single [...] vector' (Mooney, 2014), an unstructured dense vector seems unlikely to be the optimal representation to achieve it. In this thesis, we argue for a more balanced approach. We hypothesise that a representation structure that accounts for the properties of the task that we do understand, while remaining flexible enough to learn the parts that we do not, is beneficial for text-to-text generation. Note that the models proposed in this thesis will be neural networks and learned end-to-end, but with additional constraints on the model-internal representations that lead to a higher degree of structure than other current approaches.

1.1 Structured Representations

One could define a good representation as one that contains all the important information about a sample (or all the information required for the task) and discards measurement error or irrelevant details of the input. In other words, we are trying to find invariant (or equivariant[2]) features or transformations of the data; if the desired model output is invariant to a given feature, then it may safely be discarded or ignored. Note that the meaning of 'good' and 'important' will depend on the specific context and task.

Invariance For images, rotation, scale and translation of an image clearly should not affect the semantic content of that image. Representations of images should therefore be invariant/equivariant to affine transformations. There are many examples in the vision literature of approaches that exploit this property. For example, Lenc and Vedaldi (2016) learn features of images that are invariant to affine transformations directly from supervised data using regression. Chen et al. (2016) propose InfoGAN, a generative model of images that is trained to 'fool' a separate discriminative model into thinking that its generated images are real. The GAN approach assumes that the labels of objects in images (e.g., 'dog' or 'table') are the important information, and leaves the discriminator (and therefore generator) invariant to all the remaining details (e.g., where

[2]Invariance means that the output does not change when the input is changed. Equivariance means that the output changes in the same way as the input was changed.

the dog is or what colour it is).

In an attempt to unify the wide range of approaches to invariance, Bronstein et al. (2021) argue that

> “while learning generic functions in high dimensions is a cursed estimation problem, most tasks of interest are not generic, and come with essential pre-defined regularities arising from the underlying low-dimensionality and structure of the physical world."

In other words, they suggest that identifying the invariances of a problem leads to a natural solution to that problem. They consider a wide range of domains, including vision, strategy games and biological sequence modelling, all of which are governed by fixed rules. However, natural language is notably absent from their set of domains.

Invariance in natural language What transformations is natural language invariant to? There is no clear or obvious answer to this question; word order is important in many languages, and the addition or modification of a single word can completely change the meaning of a sentence. Natural language is a fundamentally human phenomenon, with no single author, design principle or fixed rules. One could argue that the meaning of a natural language utterance and the way that it is expressed are independent, but even this high-level assertion is not strictly true. The field of pragmatics is concerned with studying the meaning conveyed beyond semantics, and the tone of an utterance or choosing to omit information may change how it is interpreted by a reader or listener. The difficulty of finding consistent invariances of natural language explains the absence of prior work on the topic.

We exploit some specific types of invariance throughout this thesis, but note that they are specific to the tasks under consideration. In Chapter 3 and Chapter 5, where the goal is to generate paraphrases of a sentence with the same meaning but different surface form, we attempt to learn two representations that are invariant to meaning and form respectively. This invariance is achieved by introducing a degree of structure to the representations. In Chapter 6 and Chapter 7 we consider the task of opinion summarisation, where large numbers of sentences about a hotel or product must be aggregated according to their underlying semantic content. Then, learning a representation with a structure that is invariant to the particular choice of phrasing allows us to identify how many people share a given opinion (and therefore which opinions are particularly frequent), irrespective of the way in which they are expressed. In both tasks, the invariant property is the semantic meaning of a sentence. However, for opinion summarisation the threshold for what ‘counts’ as equivalent meaning is rather looser, requiring similarity in topic and sentiment rather than precise semantics.

1.2 Text-to-Text Generation

Demonstrating conclusively that a degree of structure is beneficial for *all* text-to-text generation tasks would be beyond the scope of a thesis. We therefore constrain our work to two main text generation problems, paraphrase generation and opinion summarisation.

Paraphrase generation involves taking a natural language utterance as input, and generating as output another utterance that has the same meaning but a different surface form (Madnani and Dorr, 2010). The ability to generate multiple diverse paraphrases of an input sentence has potential utility in improving the robustness of other natural language understanding systems, either through data augmentation during training or rewriting input queries at evaluation time (Dong et al., 2017; Iyyer et al., 2018).

Opinion summarisation involves generating a textual summary from a large number of customer reviews about a product, hotel or other *entity* (Hu and Liu, 2004; Ganesan et al., 2010). Popular products on Amazon may receive thousands of reviews, which is an infeasibly large number for a user to read when trying to decide whether to buy a product. The ability to generate an automatic summary that aggregates the frequent and popular opinions, as well as any details that particularly differentiate an entity from its competitors, is therefore very useful.

We choose these two tasks as they have practical applications while still remaining relatively constrained, and therefore feasible to experiment on and evaluate. They also present some additional challenges that make them particularly interesting. A successful paraphrase generation system must preserve the meaning of the input utterance, which implicitly involves determining what that meaning is. The ability to determine the meaning of an utterance, independent of the particular phrasing, is arguably one of the core problems of Natural Language Processing (NLP) research. Opinion summarisation operates over a large quantity of text that is highly semantically redundant, requiring an approach that scales efficiently and is able to separate unnecessary details from the overarching trends.

1.3 Research Questions

In this thesis, we consider three core hypotheses.

Hypothesis I Weakly structured representations are beneficial for text-to-text generation.

We use the term *weak structure* to refer to representations that are structured enough to encode the aspects or invariances of the tasks that we understand, yet remain flexible enough to learn the remaining aspects: for paraphrase generation, where the meaning should remain constant and the surface form should be varied, we argue that the meaning and form should therefore be represented separately; opinion summarisation requires aggregating shared high-level opinions that may differ in the specific details or phrasing, and so we argue that a discrete hierarchical representation, which enables counting up opinions at varying levels of abstraction, should be beneficial. At the same time, we leave enough flexibility in the structures so that the models are able to learn the aspects for which we do not have a good understanding; there is no clear concensus on the true or correct structure of semantics or syntax (e.g., Huck and Goldsmith, 1996; Friederici, 2011; Carnie et al., 2014), and so we opt to allow the models to learn these details from data. In each case, the weak structure encodes a high-level invariance that is relevant to the task: the meaning of a set of paraphrases is invariant to changes in surface form, and the topic and sentiment of sentences in reviews are invariant with respect to whichever specific details about an entity are mentioned.

We will show that using representations that exhibit weak structure leads to models that are able to generate paraphrases with better semantic preservation and higher syntactic diversity (Chapters 3 and 5), or are able to generate more informative summaries of customer reviews that are attributable and scale to large numbers of inputs (Chapters 6 and 7). While the specific nature of the improvements varies by task, in each case the additional structure is beneficial.

Hypothesis II Discrete and hierarchical representations can be used to make text-to-text generation problems feasible.

In this thesis, discrete representations refer to any representation that is discretised, i.e. can take one of a fixed number of values[3]. We use the term *hierarchical* to refer to a representation that is ordered, such that different parts of the representation refer

[3]By contrast, continuous dense vector representations can take an infinite number of possible values

to high- and low-level details about an input sample. A core contribution of this thesis is Hierarchical Residual Quantisation (HRQ-VAE), a technique for learning discrete hierarchical representations of sentences (Chapter 4). In Chapters 5 to 7 we apply HRQ-VAE to paraphrase generation and opinion summarisation. We find that discrete representations are a natural fit for representing the *syntactic structure* of sentences, and the discreteness directly enables us to easily predict alternative surface forms when generating paraphrases. Using a hierarchical representation further simplifies this prediction problem, allowing us to first predict a high-level syntactic form before gradually refining the level of detail, and leading to improved quality when generating paraphrases.

We also find that discrete hierarchical representations are a natural fit for representing the *meaning* of sentences, where the top level in the hierarchy might correspond to the overall topic of sentiment of a sentence and lower levels may encode more specific details. The discreteness of such representations directly allows us to 'count' opinions from product and hotel reviews, thereby identifying which opinions occur frequently and should form part of a summary of the reviews. Since this aggregation takes place in a (fairly small) discrete space, it is highly efficient, and can handle large numbers of reviews as input. The hierarchical nature of the representation allows us to perform this counting operation at varying degrees of specificity, so that the resulting summaries convey an appropriate level of detail.

Hypothesis III Given a structured representation, some degree of supervision is required to assign meaning to the structure.

Introducing a structured representation to the model is not sufficient; if we want particular types of information to be assigned to particular parts of a structured representation, we find that this assignment must be introduced explicitly (Locatello et al., 2019). In Chapters 3 and 5 we use a denoising objective to assign meaning to the parts of the representation, whereby the model must reconstruct a target sentence from one input with the correct meaning but different surface form, and another input with the correct surface form but different meaning. In Chapters 6 and 7, where we want the levels of the hierarchical representation to correspond to a hierarchy of meaning, we automatically construct pairs of sentences with similar meanings but different surface forms, and train the model to place these pairs close together in the hierarchy. In both cases, we find that these sources of *distant supervision* lead to structured representations where the parts of the structure have an associated meaning.

1.4 Outline

The remainder of the thesis is organised as follows:

- In Chapter 2 we describe the background material relevant to our experiments. We give a brief overview of prior work on representation learning for NLP. Then, we introduce the text-to-text generation framwork, and describe current modelling approaches including Transformers and Large Language Models. We provide a description of Variational Autoencoders and Vector-Quantised Variational Autoencoders (VQ-VAE), which form the basis for the majority of the models introduced in the thesis. We conclude by introducing the text-to-text generation tasks that will act as the case studies for our experiments, and give some background on evaluation methods for these tasks.

- In Chapter 3 we introduce SEPARATOR, a paraphrase generation method that uses an encoder-decoder model to map an input sentence into a latent encoding space, and then back to an output paraphrase. A principled information bottleneck and a careful choice of training scheme (Section 3.3.2) lead to an encoding space that separately represent the meaning and surface form of an input sentence, with VQ-VAE (a popular discrete representation learning approach) used to learn a discrete representation of the surface form. This separation enables us to paraphrase the input sentence, varying the surface form of the output by directly manipulating the syntactic encoding of the input and keeping the semantic encoding constant. Extensive experiments and a human evaluation show that we are able to generate paraphrases with a better tradeoff between semantic preservation and syntactic novelty compared to previous methods. This chapter acts as the first piece of evidence in support of our core hypotheses, and was previously published in Hosking and Lapata (2021).

- In Chapter 4 we address the lack of a tractable factorisation of the joint distribution over codes in VQ-VAE, and introduce Hierarchical Residual Quantisation (HRQ-VAE), a method for learning hierarchical discrete latent representations of input data by recursively quantising the residual error between an input embedding and its discretised approximation. We introduce the theoretical foundations of the approach and give practical details for stable training of models that use HRQ-VAE. We report validation experiments on MNIST, a dataset of handwritten digit images, and show that HRQ-VAE learns more informative representations than VQ-VAE. HRQ-VAE forms the basis for the models used in the Chapters 5 to 7.

- In Chapter 5 we combine the factorised representation spaces from Chapter 3 with HRQ-VAE and describe CALYPSO, a method for generating paraphrases that learns hierarchical representations of the syntactic structure of input sentences. This hierarchical encoding is easier to predict at test time than the more weakly structured representation used by SEPARATOR, leading to a higher quality of generated paraphrases. The contributions in this chapter appeared in Hosking et al. (2022).
- In Chapter 6 we demonstrate that HRQ-VAE may also be used for more complex tasks, by applying it to unsupervised opinion summarisation. We introduce HERCULES, a method that encodes sentences from customer reviews into a hierarchical discrete latent space, then identifies common opinions based on the frequency of their encodings. HERCULES is attributable, meaning that each output is accompanied by supporting evidence, and our approach scales to large numbers of input reviews while also generating abstrative summaries that are more informative than prior work. HERCULES exploits the discrete and hierarchical properties of the learned representation to aggregate opinions from reviews about hotels and Amazon products, and therefore acts as the third piece of evidence in support of our core hypotheses. The work in this chapter was previously published in Hosking et al. (2023b).
- In Chapter 7 we combine the scalability and attributability of structured representations with the fluency of LLMs. We introduce HIRO, an unsupervised opinion summarisation method that uses a hierarchical discrete latent space based on HRQ-VAE to identify clusters of sentences from reviews that contain popular opinions. Passing these retrieved clusters to a Large Language Model leads to fluent and coherent summaries. This chapter acts as the final piece of evidence in support of our hypotheses, and demonstrates how methods based on weakly structured representations are compatible with powerful (but unstructured) Large Language Models. The work in this chapter was published in (Hosking et al., 2024).
- In Chapter 8 we offer concluding remarks, summarising our contributions and describing additional themes and findings that arose over the course of the thesis. We also lay out possible directions for future work.

Chapter 2

Background

In this chapter, we provide a brief overview of background material relevant to this thesis. We begin by describing prior work on representation learning and structured latent variables. We give an overview of text-to-text generation including encoder-decoder models and Large Language Models (LLMs). We give a brief introduction to Variational Autoencoders (VAEs) and Vector-Quantised VAEs (VQ-VAE), which form the basis for HRQ-VAE (Chapter 4). Finally, we describe the tasks that will form the focus of this thesis, including the methods used for evaluation.

2.1 Notation

Throughout this thesis, we make use of the notation listed below. Any other notation used will follow standard conventions or will be defined when used.

$\mathbf{x}$	— an input sentence as a sequence of tokens
$\mathbf{y}$	— an output sentence as a sequence of tokens
$\hat{\mathbf{y}}$	— an estimated output sentence as a sequence of tokens
$\mathbf{e} \in \mathbb{R}^d$	— a dense vector encoding
$\mathbf{z} \in \mathbb{R}^d$	— a dense latent vector
$q_d \in \{1, \ldots, K\}$	— a discrete code
$q_{1:D} \in \{1, \ldots, K\}^D$	— a sequence of D codes
$\mathbf{C}(q_d) \in \mathbb{R}^d$	— a dense embedding of code q_d

$p_\theta(x)$ — a probability distribution over variable x

$\mathcal{L}$ — a loss function or training objective

$\phi(a|b)$ — an approximate posterior of a given b

2.2 Representation Learning

One might argue that all of deep learning is concerned with learning good representations, but the structure of the representations is not always an explicit consideration. We do not attempt to provide a thorough summary of all prior work on representation learning, but describe some directions that are particularly relevant to this thesis.

2.2.1 Explicit Representation Learning

Mikolov et al. (2013) propose word2vec, a method for representing words as dense vector embeddings such that words with similar meanings are located in similar regions of embedding space. Word2vec exploits the distributional hypothesis — the assumption that words used in similar context have similar meanings — to learn this mapping from a large discrete space to a smaller, dense space. As such, it represents an early example of evidence in support of the core hypothesis of this thesis; representing different words as unique discrete objects does not account for similarities between words, and choosing a dense continuous vector is more appropriate.

Although all neural models are to some extent concerned with learning representations, some prior work has explicitly focused on *how* they are learned. In particular, contrastive learning aims to learn representations of input samples, such that two similar input examples are encoded to similar representations, and dissimilar examples are assigned dissimilar encodings (Chopra et al., 2005; Schroff et al., 2015; Song et al., 2016; Gutmann and Hyvärinen, 2010). In other words, the goal is a representation space that is *somewhat ordered*; samples that are similar in some way should be mapped to similar regions of the space.

Contrastive learning has been widely used for learning sentence and document embedding models, that may then be used for information retrieval (Reimers and Gurevych, 2019; Karpukhin et al., 2020; Gao et al., 2021; Izacard et al., 2022, inter alia); given an input query, and a set of documents, the aim is to identify which documents are related to the query or may be used to find an answer to the query.

Then, a natural solution is to find a common embedding space for both queries and documents, where queries and corresponding relevant documents are mapped to similar regions of embedding space. At test time, a nearest-neighbours lookup is performed to retrieve relevant documents (Douze et al., 2024). Such embedding models may either be trained on labelled pairs of related and unrelated sentences (or pairs of queries and documents), or with some form of distant supervision. For example, Izacard et al. (2022) extract sentences from Wikipedia documents, with sentences from the *same* document considered as positive pairs, and from *different* but plausibly related documents as negative pairs.

There has been some prior work on using contrastive learning techniques to improve models' ability to perform a task for inputs in multiple different languages. In this case, the desired invariance is with respect to the language label of the input: English, French, or Chinese utterances with same meaning should be mapped to the same output, and therefore to the same internal representation. For example, Pan et al. (2021) apply contrastive training to Machine Translation, using pairs of sentences in different languages with the same meaning as positive examples, and sentences with different meanings as negatives.

Although not strictly based on contrastive learning, Sherborne et al. (2023) introduce an additional objective to encourage language independent representations for semantic parsing; their model is trained to minimise the divergence between the distributions over encodings for each language label, to prevent each language occupying its own region of the representation space. They find that this explicit 'language invariance' objective leads to improved generalisation capabilities for a model trained primarily on English data.

We draw inspiration from the distant supervision techniques used for constrastive learning in this thesis (specifically, the training objective used in Chapter 3), and directly use contrastive learning to train a content selection module for opinion summarisation in Chapter 7.

2.2.2 Strong Structure

It is reasonable to consider whether the good representations should be designed instead of learned. These rigid or *strongly structured* representations received extensive attention before the advent of neural approaches, with possible candidates for representations including Lambda calculus (Church, 1941), Universal Dependencies (UD, de Marneffe and Manning, 2008; Nivre et al., 2020) and Abstract Meaning Representation (AMR, Banarescu et al., 2013). Models based on these representations are inherently more interpretable, but significant effort is required to design a representation that is both expressive and structured. For example, Lambda calculus representations are lexicalised (i.e., they use the vocabulary of a language to represent its semantics) and therefore are unable to infer any relations between semantically related but lexically distinct words.

We experimented with both UD and AMR during exploratory work before this thesis, but found that they were both insufficiently expressive yet difficult for a model to learn. Given a set of sentences that are paraphrases of each other, we found that the AMR parses for each were significantly different and therefore did not capture only the shared meaning of the sentences, but also some details specific to the particular phrasing.For example, we explored using AMR to perform paraphrasing by first parsing an input sentence, then generating an output sentence based on the AMR parse. We found that the AMR parses for two sentences that are paraphrases of each other were very different, indicating that AMR is unsuccessful at abstracting the semantics of a sentence from its realisation. Despite this, we found that they were often also missing crucial information required to reconstruct an output sentence that faithfully conveyed the original meaning. We believe this highlights the challenges involved in designing (rather than learning) a representation structure to convey the meaning of sentences.

A survey on neurosymbolic techniques in NLP that use strongly structured representations to perform reasoning (Hamilton et al., 2022) found a relatively low number of papers, with disagreement on the nature of reasoning, and requirements to hand craft rules and logic. Neurosymbolic approaches are potentially still promising — an approach based on rigorous logic with interpretable representations remains an attractive goal, but the practicalities of implementing them are highly complex and current approaches fall far short of methods that are fully learned. Our approach in this thesis does not meet the requirements of a true neurosymbolic approach (since it uses representations that are neither fully symbolic nor easily interpretable), but draws inspiration from that direction, and aims to improve on fully learned neural approaches by introducing a degree of *weak* structure.

In this thesis, we argue that representations should be structured enough to capture aspects of the tasks that we understand, while remaining flexible enough to learn the aspects that are unknown. For example, for paraphrase generation we know from the task definition that the meaning of the input sentence should be preserved, while the surface form should be changed. Therefore, it is natural that the meaning and form should be encoded separately. However, there is no clear concensus on the true or correct structure of semantics or syntax (e.g., Huck and Goldsmith, 1996; Friederici, 2011; Carnie et al., 2014), and so it is also natural to allow the models to learn these details from data. We refer to this type of partly-structured, partly-learned representation as *weakly structured*.

2.3 Variational Autoencoders

An autoencoder (AE) is a model that is trained to map an input sample $\mathbf{x}$ from an arbitrary space to a learned representation $\mathbf{z} \in \mathbb{R}^d$, which usually has lower complexity or dimensionality compared to the input space, and back again to a reconstruction $\hat{\mathbf{x}}$ of the input. Autoencoding is usually performed to achieve dimensionality reduction; for example, PCA (Pearson, 1901) can be viewed as the solution to a particular setting of a linear autoencoder. Dimensionality reduction has the useful property that it performs some degree of *abstraction*, learning to include information that is important and discarding the minutiae or measurement error included in a specific sample. This dimensionality reduction resembles an information bottleneck; forcing a model to use a compressed representation of an input sample leads it to ignore unwanted noise in the data and prioritise the important properties that give the best approximate reconstruction. For example, given a set of images of handwritten digits from zero to nine (Lecun et al., 1998), a linear autoencoder with latent representation $\mathbf{z} \in \mathbb{R}^{10}$ might be used to perform unsupervised digit classification.

While early autoencoders like PCA used shallow linear mappings between data and representations, Hinton and Salakhutdinov (2006) extended the approach to use a network with multiple non-linear layers. Encoder-decoder models that use a single dense vector encoding (e.g., Seq2Seq LSTMS, or Transformers with a pooling layer) are also autoencoders.

Variational Autoencoders (VAEs) extend autoencoders by assuming that the learned representation $\mathbf{z}$ is a continuous latent random variable (Kingma and Welling, 2014). Then, the mapping is from a sample $\mathbf{x}$ to a *distribution* over latent representations $p(\mathbf{z}|\mathbf{x})$ and vice-versa, with the observed distribution given by

$$p(\mathbf{x}) = \int d\mathbf{z}\, p(\mathbf{x}|\mathbf{z})p(\mathbf{z}), \tag{2.1}$$

(a) Posterior (encoder) (b) Generative model (decoder)

Figure 2.1: Graphical model for a VAE with a continuous latent variable.

and the posterior over $\mathbf{z}$ given by

$$p(\mathbf{z}|\mathbf{x}) = \frac{p(\mathbf{z},\mathbf{x})}{p(\mathbf{x})} \tag{2.2}$$

$$= \frac{p(\mathbf{z},\mathbf{x})}{\int d\mathbf{z}\, p(\mathbf{x}|\mathbf{z})p(\mathbf{z})}. \tag{2.3}$$

This corresponds to the graphical model shown in Figure 2.1.

Assuming the mapping is parameterised by some set of parameters θ, the optimal parameter set is the one that maximizes the probability of generating real data samples,

$$\theta^* = \arg\max_\theta \prod_{i=1}^{n} p_\theta(\mathbf{x}^{(i)}), \tag{2.4}$$

where $\mathbf{x}^{(i)}$ is the i'th sample from the dataset $\mathbf{X}$. In general, the integrals in Equations (2.1) and (2.3) are intractable. Therefore we instead introduce an *approximate posterior* $\phi_\psi(\mathbf{z}|\mathbf{x})$, and minimise the KL divergence between this approximate posterior and the true posterior,[1]

$$\theta^* = \arg\min_\theta \mathrm{KL}\big[\phi_\psi(\mathbf{z}|\mathbf{x}) \,\|\, p_\theta(\mathbf{z}|\mathbf{x})\big], \tag{2.5}$$

$$\mathrm{KL}\big[\phi_\psi(\mathbf{z}|\mathbf{x}) \,\|\, p_\theta(\mathbf{z}|\mathbf{x})\big] = \int d\mathbf{z}\, \phi_\psi(\mathbf{z}|\mathbf{x}) \log \frac{\phi_\psi(\mathbf{z}|\mathbf{x})}{p_\theta(\mathbf{z}|\mathbf{x})} \tag{2.6}$$

$$= \mathbb{E}_{\mathbf{z}\sim\phi_\psi}\left[\log \frac{p_\theta(\mathbf{z}|\mathbf{x})}{\phi_\psi(\mathbf{z}|\mathbf{x})}\right] \tag{2.7}$$

$$= \mathbb{E}_{\mathbf{z}\sim\phi_\psi}\left[\log \frac{p_\theta(\mathbf{x}|\mathbf{z})p_\theta(\mathbf{z})}{\phi_\psi(\mathbf{z}|\mathbf{x})p_\theta(\mathbf{x})}\right] \tag{2.8}$$

$$= \mathbb{E}_{\mathbf{z}\sim\phi_\psi}\left[\log \frac{p_\theta(\mathbf{x}|\mathbf{z})p_\theta(\mathbf{z})}{\phi_\psi(\mathbf{z}|\mathbf{x})}\right] - \log p_\theta(\mathbf{x}). \tag{2.9}$$

We cannot compute $\log p_\theta(\mathbf{x})$; however, since the KL divergence is non-negative, we

[1]This derivation is based on material from Weng (2018), Blei et al. (2017), and Murphy (2023).

have

$$\log p_\theta(\mathbf{x}) \geq \underbrace{\mathbb{E}_{\mathbf{z}\sim\phi_\psi}\left[\log\frac{p_\theta(\mathbf{x}|\mathbf{z})p_\theta(\mathbf{z})}{\phi_\psi(\mathbf{z}|\mathbf{x})}\right]}_{\text{ELBO}} + \mathrm{KL}\big[\phi_\psi(\mathbf{z}|\mathbf{x}) \parallel p_\theta(\mathbf{z}|\mathbf{x})\big], \tag{2.10}$$

where the term labelled ELBO is the Evidence Lower-Bound. Maximising the ELBO therefore lower-bounds the (log) evidence, and we use it as an alternative (tractable) objective,

$$\begin{aligned}\text{ELBO} =&\mathbb{E}_{\mathbf{z}\sim\phi_\psi}\left[\log\frac{p_\theta(\mathbf{x}|\mathbf{z})p_\theta(\mathbf{z})}{\phi_\psi(\mathbf{z}|\mathbf{x})}\right] &(2.11)\\ =&\mathbb{E}_{\mathbf{z}\sim\phi_\psi}\left[\log p_\theta(\mathbf{x}|\mathbf{z}) + \log\frac{p_\theta(\mathbf{z})}{\phi_\psi(\mathbf{z}|\mathbf{x})}\right] &(2.12)\\ =&\mathbb{E}_{\mathbf{z}\sim\phi_\psi(\mathbf{z}|\mathbf{x})}\big[p_\theta(\mathbf{x}|\mathbf{z})\big] - \mathrm{KL}\big[\phi_\psi(\mathbf{z}|\mathbf{x}) \parallel p_\theta(\mathbf{z})\big]. &(2.13)\end{aligned}$$

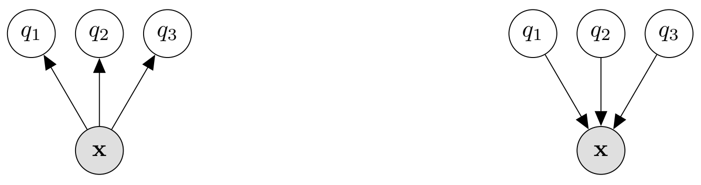
Intuitively, maximising the ELBO jointly maximises the likeihood of generating real data $\mathbf{x}$ from samples of $\mathbf{z}$ while also ensuring the approximate posterior $\phi_\psi(\mathbf{z}|\mathbf{x})$ stays close to the prior $p_\theta(\mathbf{z})$. Note that we have not yet assumed any specific parameterisation for $\mathbf{z}$ or the distributions $p_\theta(\mathbf{x}|\mathbf{z})$ and $\phi_\psi(\mathbf{z}|\mathbf{x})$. If the latent variable $\mathbf{z}$ is instead discrete, the ELBO can be derived in essentially the same way, with the integrals replaced by sums. Note that in the remainder of the thesis, we drop the θ subscript for brevity, so that in general $p_\theta(\cdot) \equiv p(\cdot)$ and $\phi_\psi(\cdot) \equiv \phi(\cdot)$.

In practice, the posterior $\phi_\psi(\mathbf{z}|\mathbf{x})$ is parameterised by a model called the *encoder*, and $p_\theta(\mathbf{x}|\mathbf{z})$ is parameterised by a *decoder*. Optimizing the ELBO (Equation (2.13)) involves backpropagating through a random sample, which is not generally differentiable. Some form of reparameterisation trick is therefore often used. For example, if $\mathbf{z}$ is a Gaussian $\mathbf{z} \sim \mathcal{N}(\mu, \sigma)$, we can instead express $\mathbf{z}$ as a deterministic function of an auxiliary random variable,

$$\mathbf{z} = \mu + \sigma \odot \epsilon \tag{2.14}$$

$$\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{1}), \tag{2.15}$$

where $\odot$ is the Hadamard product. Gradient can then flow to μ and σ. Differentiable sampling of discrete random variables is more involved, and we discuss this in more detail in the next section.



(a) Posterior (encoder) (b) Generative model (decoder)

Figure 2.2: Graphical model for VQ-VAE with $H = 3$ discrete latent codes, assuming independence between the codes q_i.

2.4 Vector-Quantised Variational Autoencoders

Vector-Quantised Variational Autoencoders (VQ-VAE, van den Oord et al., 2017) use a discrete latent variable or variables $q \in \{1, \dots, K\}$ to represent input data, corresponding to the graphical model shown in Figure 2.2. However, since neural models generally operate in continuous space, a method is needed to map between continuous and discrete encodings, and vice-versa.

Let $\mathbf{C} \in \mathbb{R}^{K \times d}$ be a *codebook*, with $\mathbf{C}(i) \in \mathbb{R}^d$ the i'th embedding vector. Then, the VQ-VAE encoder deterministically maps dense encoding vectors $\mathbf{z} \in \mathbb{R}^d$ to codes q by selecting the nearest neighbour,

$$q = \arg\min_i ||\mathbf{z} - \mathbf{C}(i)||_2. \tag{2.16}$$

The input to the decoder network is then a 'reconstructed' estimate $\tilde{\mathbf{z}} = \mathbf{C}(q)$, given by using the codebook to embed q back into the dense vector space.

Since the code lookup is deterministic, VQ-VAE is not strictly a variational model. Furthermore, this lookup process is not differentiable, and so the authors propose using the straight-through estimator, which essentially copies the gradient from the decoder input $\tilde{\mathbf{z}}$ to the encoder output $\mathbf{z}$. Then, they propose two additional loss terms to update the codebook: the quantisation loss

$$\mathcal{L}_{quant} = \|\text{sg}[\mathbf{z}] - \mathbf{C}(q)\|_2^2, \tag{2.17}$$

and commitment loss

$$\mathcal{L}_{commit} = \|\mathbf{z} - \text{sg}[\mathbf{C}(q)]\|_2^2, \tag{2.18}$$

where $\text{sg}[\cdot]$ is the stop gradient operator.

van den Oord et al. (2017) found that it was beneficial to update the codebook using an exponential moving average (EMA) approach instead of the quantisation loss, as it leads to more stable training. At each training step, the embedding for each code in the codebook is updated to be a weighted average of the existing embedding and the input encodings that were mapped to that code. We refer to van den Oord et al. (2017, Appendix A) for the full details.

Using a single latent code q to represent the full information in an input sample is a tight constraint, and in practice models will often split the input encoding into multiple heads (for language data) or channels (for images), resulting in multiple latent codes $q_i, i \in \{1, \ldots, H\}$. If we assume independence between these codes (also known as the mean-field assumption), then we reach the graphical model shown in Figure 2.2.

In Chapter 4 we propose an extension to VQ-VAE, that represents input data as a *hierarchically ordered* sequence of latent codes. Additionally, our proposed approach uses the Gumbel reparameterisation trick (Jang et al., 2017; Maddison et al., 2017) instead of the straight-through estimator, leading to more stable training with a simplified objective.

2.5 Text-to-Text Generation

Text-to-text generation encompasses a wide range of problems and applications. Summarisation involves condensing long articles into a shorter, easier to read summary that contains the most important information (Banko et al., 2000; Rush et al., 2015; See et al., 2017; Goyal et al., 2022). Machine translation transforms an input sentence from one language to another while preserving the original meaning (Brown et al., 1993; Koehn et al., 2007; Kalchbrenner and Blunsom, 2013). Question answering seeks to generate responses to an input query, optionally grounding the answer in an associated document(s). These answers may be small spans of text containing a few words (Hirschman et al., 1999; Ng et al., 2000; Rajpurkar et al., 2016; Seo et al., 2017), or they may be longer-form explanations (Fan et al., 2019). Question generation considers the inverse problem: given an answer, generate a possible question that might lead to that answer (Heilman and Smith, 2010; Du et al., 2017; Hosking and Riedel, 2019; Narayan et al., 2023). Paraphrase generation involves taking a natural language utterance as input, and generating as output another utterance that has the same meaning but a different surface form (Barzilay and McKeown, 2001; Bowman et al., 2016; Mallinson et al., 2017). Opinion summarisation involves generating a textual summary from a large number of customer reviews about a product, hotel or other *entity* (Erkan and Radev, 2004; Ganesan et al., 2010; Angelidis et al., 2021; Amplayo et al., 2021b).

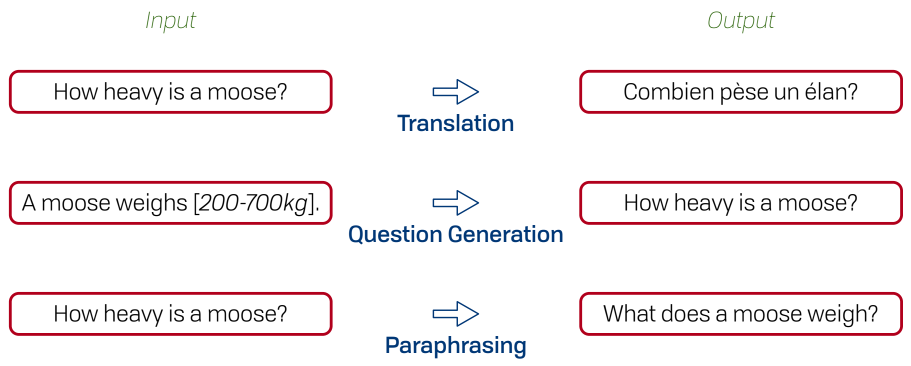


Figure 2.3: Examples of text-to-text generation tasks. While the specifics of each task are different, in each case the input and output modality is text, and a successful system must be able to first interpret the input, then process it, before finally generating fluent and valid output language.

We show some examples of text-to-text generation problems in Figure 2.3. Each text-to-text generation task involves their own specific challenges and requirements, but they share some common traits. In each case, the input and output modalities are both text, requiring models to be able to both 'read' and process the input, and 'write' well-formed output language. Improvements and innovations from one task may often be applicable to other areas, making text-to-text generation a useful research direction beyond the direct application of each specific task.

2.5.1 Encoder-Decoder Models

An encoder-decoder model is an application of autoencoders to text-to-text generation, that first uses an encoder model to map a sequence of input tokens $\mathbf{x}$ to a representation $\mathbf{e}$, then a decoder model maps $\mathbf{e} \in \mathbb{R}^d$ to a sequence of output tokens $\mathbf{y}$. Such models may either be trained as true autoencoders, where $\mathbf{x} = \mathbf{y}$ and the model is trained to reconstruct the input sentence; or, they may be trained as *denoising* autoencoders, where $\mathbf{x}$ and $\mathbf{y}$ are different sequences but with some common properties, e.g., the same meaning but different word choice.

Neural sequence-to-sequence (Seq2Seq) models are a class of encoder-decoder architecture (Sutskever et al., 2014) based on Recurrent Neural Networks (Rumelhart et al., 1986) using Long-Short Term Memory (LSTM) networks (Hochreiter and Schmidhuber, 1997; Sundermeyer et al., 2012) or Gated Recurrent Units (GRUs) (Cho et al., 2014b). Seq2Seq models represent a sequence of input tokens (i.e., an input sentence) as a single dense vector, which is then used as the initial state of a RNN decoder that generates an output sentence.

The single dense vector used by Seq2Seq models is a highly compressed bottleneck between the encoder and decoder. Bahdanau et al. (2015) therefore propose Attention, whereby a distribution is induced over the representations of the input tokens, and a weighted average over all input representations is fed to the decoder at each time step. This allows the model to 'see' representations from the full input sequence during decoding, increasing the capacity of the bottleneck and leading to improved quality of generated text.

Transformers Transformers (Vaswani et al., 2017) are a neural architecture for sequence modelling based primarily on attention mechanisms, that have dominated the fields of natural language understanding and generation since their invention, and are state-of-the-art at time of writing.

Figure 2.4 depicts the Transformer architecture as defined by Vaswani et al. (2017). The encoder and decoder are stacked Transformer layers, each comprising multi-head attention and feedforward sublayers with skip connections. Each multi-head attention sublayer receives input queries, keys, and values (Q, K, and V respectively), and outputs a contextualised representation of V weighted by the dot-product interaction between Q and K,

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_H)W^O \tag{2.19}$$

$$\text{head}_h = \text{Attention}(QW_h^Q, KW_h^K, VW_h^V) \tag{2.20}$$

$$\text{Attention}(Q, K, V) = \text{softmax}\Big(\frac{QK^T}{\sqrt{d_k}}\Big)V. \tag{2.21}$$

Each head, $h \in 1, \ldots, H$, uses dimensionality $d_h = d/h$ and splits the input encodings into subspaces. $W_h^Q, W_h^K, W_h^V \in \mathbb{R}^{d \times d_h}$, $W^O \in \mathbb{R}^{d \times d}$ are learned parameters. The different heads may specialise in some way, with these specialisations combined through successive layers.

The output of the multi-head attention sublayer is then passed through a fully-connected linear network with ReLU activation,

$$\text{FeedForward}(X) = \text{ReLU}(XW_i + b_i)W_j + b_j, \tag{2.22}$$

where $W_i \in \mathbb{R}^{d \times d_f}, W_j \in \mathbb{R}^{d_f \times d}, b_i \in \mathbb{R}^{d_f}, b_j \in \mathbb{R}^d$ are learned parameters, d_f is the dimensionality of the 'hidden layer' and $\text{ReLU}(x) = \max(x, 0)$ is the Rectified Linear activation function (Glorot et al., 2011).

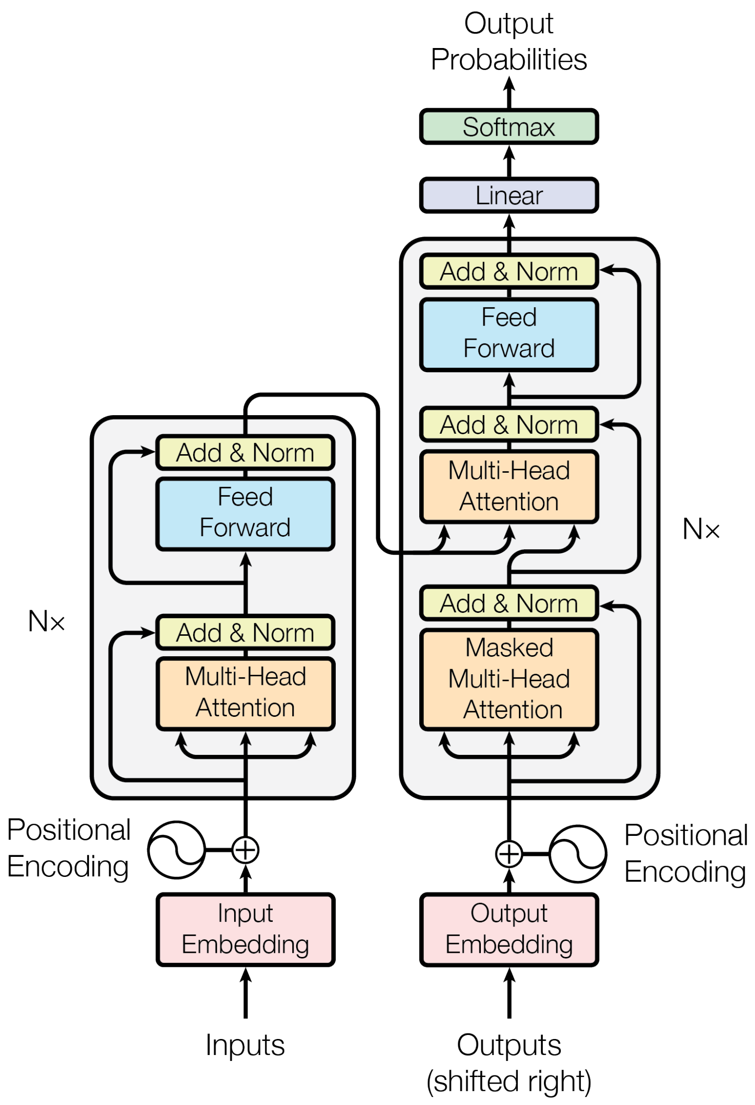


Figure 2.4: Illustration of the Transformer architecture, from Vaswani et al. (2017). A single layer comprises multi-head attention and feedforward layers with additional skip connections. These layers are stacked to produce a complete encoder-decoder model. Each multi-head attention layer receives input queries, keys, and values (Q, K, and V respectively), and outputs a contextualised representation of V weighted by the dot-product interaction between Q and K. See `arxiv.org/abs/1706.03762` for authors' permission for reproduction of graphics for academic purposes.

Each encoder layer computes a contextual representation for each input token using multi-head self-attention, with the final encoder layer resulting a sequence of encodings $\mathbf{e}_{h,t}$. In a standard Seq2Seq Transformer, the decoder conditions upon this encoding sequence with an additional multi-head attention sublayer.

Transformers' use of attention allows them to model long-term dependencies within language, while remaining parallelisable and therefore efficient to train. When combined with subword tokenisation methods such as BPE to avoid the out-of-vocabulary problem (Sennrich et al., 2016), they are a powerful architecture for text-to-text generation. The models in this thesis all make use of Transformer encoders and decoders as their main backbone.

Multi-head Pooling A Transformer encoder outputs a sequence of dense encodings, but some model architectures (e.g., Variational Autoencoders) are usually defined based on a single encoding vector. In these cases, we use a multi-head pooling layer, originally proposed by Liu and Lapata (2019), to pool the sequence of encodings into a single dense vector. Then, the Transformer decoder attends to this vector, effectively treating it as a sequence of length one.

For each head h in the multi-head pooling layer, we calculate a distribution $\alpha_{h,t}$ over time indexes t using attention,

$$\alpha_{h,t} = \frac{\exp a_{h,t}}{\sum_{t' \in |\mathbf{x}|} \exp a_{h,t'}}, \tag{2.23}$$

$$a_{h,t} = \mathbf{k}_h^T \mathbf{e}_{h,t}, \tag{2.24}$$

with $\mathbf{k}_h \in \mathbb{R}^{d/H}$ a learned parameter. We then take a weighted average of a linear projection of the encodings, to give pooled output $\tilde{\mathbf{e}}_h$,

$$\tilde{\mathbf{e}}_h = \sum_{t' \in |\mathbf{x}|} \alpha_{h,t'} V_h \mathbf{e}_{h,t'}, \tag{2.25}$$

with $V_h \in \mathbb{R}^{D/H \times D/H}$ a learned parameter. It is this pooled output $\tilde{\mathbf{e}}_h$ that is used as the input 'sequence' to the decoder.

2.5.2 Large Language Models

During the course of this thesis, a new paradigm of powerful models based on pretraining has emerged, in the form of Large Language Models (LLMs). Originally used to develop encoder-only models such as ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019), pretraining involves training a model with a high parameter count in a self-supervised manner on large quantities of unlabelled text. Pretraining was then applied to encoder-decoder models (e.g., T5 (Raffel et al., 2020) and BART (Lewis et al., 2019)) and decoder-only models (e.g., GPT, Radford and Narasimhan, 2018). This process of compressing a huge volume of training data seems to endow LLMs with strong capabilities, including surprisingly strong performance on tasks that they were not explicitly trained to do (Radford et al., 2019; Brown et al., 2020). A trend of increasing scale has exploded in the last few years, with current commercial LLMs having parameter counts in the 100s of billions and training costs exceeding 100 million US dollars for the larger models (Knight, 2023).

LLMs may additionally be trained on annotated examples to improve their ability to follow instruction *prompts* and perform tasks. This can be achieved by mixing annotated examples with the pretraining data (Raffel et al., 2020), with a separate fine-tuning stage (Thoppilan et al., 2022), or by using reinforcement learning techniques to maximise expected human evaluation scores (Ziegler et al., 2020; Ouyang et al., 2022a). This class of LLMs are broadly referred to as *instruction tuned* LLMs.

Although LLMs are highly performant, they are also extremely expensive to train. Open-weight[2] models are available, but inference is still costly; the popular Mistral 7B model requires a minimum of 20GB of GPU memory for inference.[3] Furthermore, the training data for commercially trained models is generally not released, leading to suspicions that they may have been trained on the test splits of popular datasets (Roberts et al., 2024; Golchin and Surdeanu, 2024; Oren et al., 2024). We include comparisons to a current LLM, Mistral 7B, throughout the thesis. However, this is not a truly fair comparison, with a roughly 100x difference in parameter count between our models and the LLM, combined with the possibility that the LLM was trained on the test splits of each dataset. In Chapter 7 we present a method that aims to combine the performance of LLMs with the efficiency and scalability of previous approaches.

2.6 Tasks

In this thesis, we focus primarily on two text-to-text generation tasks: paraphrase generation and opinion summarisation.

[2]Open-weight means that the trained weights of the models are public, but the training data and code to reproduce the model are not.

[3]Calculated using `https://huggingface.co/spaces/hf-accelerate/model-memory-usage`.

2.6.1 Paraphrasing

Paraphrase generation involves taking a natural language utterance as input, and generating as output another utterance that has the same meaning but a different surface form. The ability to generate multiple diverse paraphrases of an input query has potential utility in improving the robustness of other natural language understanding systems, either through data augmentation during training or rewriting input queries at evaluation time (Dong et al., 2017; Iyyer et al., 2018).

We begin by noting that defining what counts as the 'same meaning' is quite subtle, and likely depends on context. The field of pragmatics is concerned with studying the meaning conveyed beyond semantics, and the tone of an utterance or choosing to omit information may change how it is interpreted by a reader or listener. For example, the questions 'How heavy is a moose?' and 'A moose is how heavy?' convey slightly different intent; in the first case, the querent likely wants to know the weight of a moose, but the second phrasing could be interpreted as an expression of surprise on learning the average moose weight. In this thesis, we consider two possible *groundings* of paraphrases. We firstly consider factual questions, where two questions may be considered paraphrases if they lead to the same answer. Secondly, we consider image captions, where paraphrases are different sentences describing the same image content.

Paraphrase generation is technically challenging because it implicitly involves some of the most fundamental problems in NLP: a paraphrase generation model must extract the underlying meaning of an input sentence, independent of the particular phrasing, and represent this meaning in some machine-readable representation; then, it must generate well-formed and fluent output language, that accurately reflects the meaning contained in the extracted representation. This description of the task gives a hint as to the desired invariance of the representation; it should be invariant to syntactic and pragmatic choices of phrasing that do not affect the meaning of the utterance.

2.6.2 Opinion Summarisation

Opinion summarisation involves generating a textual summary from a large number of customer reviews about a product, hotel or other *entity*. Popular products on Amazon may receive thousands of reviews, which is an infeasible number for a user to read when trying to decide whether to buy a product. The ability to generate an automatic summary that aggregates the frequent and popular opinions, as well as any details that particularly distinguish a product from its competitors, is therefore very useful.

The ideal summary should accurately reflect the distribution of opinions in the input reviews. If the vast majority of customers enjoyed the service at a restaurant, but one reviewer found it to be rude, then the summary should probably be positive. If half of the reviews are positive and half were negative, the summary should reflect this mixture of opinions.

However, it is also worth considering what a summary will be used for. Most likely, a user will be looking to decide between multiple options, and is looking at the reviews (or summary thereof) to help make up their mind[4]. In this case, a useful summary should also help *differentiate* between entities, even if that comes at the cost of less accurately representing the overall sentiment.

Opinion summarisation shares some of the same challenges as paraphrase generation: a system must be able to extract the underlying meaning or opinion from a review; and, it must be able to generate a fluent textual summary. The desired invariance is also the same as paraphrase generation; the representation should be invariant to the specific phrasing of an opinion. However, opinion summarisation additionally requires that the model be *scalable* and able to handle a large number of input reviews. Finally, it also involves determining which information to include and which information to omit.

2.6.3 Evaluation

Reference-based metrics In order to successfully determine whether a proposed text-to-text approach offers an improvement compared to prior methods, it is crucial to be able to evaluate system output. Reference based metrics aim to compare generated outputs with *references* that have been curated by hand, or through some form of distant supervision. BLEU and ROUGE (Papineni et al., 2002; Lin, 2004) aim to improve on simple string matching by comparing the degree of n-gram overlap between generated and reference outputs. BLEU is defined as a weighted geometric mean of all the n-gram precisions for $1 \geq n \geq 4$, multiplied by a brevity penalty that penalises overly short generations. Intuitively, it captures how many of the n-grams in the generated output appear in the human reference output. The precise definition is rather involved, and we refer to Papineni et al. (2002) for the full details. By contrast, ROUGE measures recall, or how many of the n-grams in the reference appear in the generated output.

BLEU may also be used to measure the similarity between a generated output and the original *input*. While BLEU$(output, references)$ is a proxy for the quality of

[4]We thank Bonnie Webber for this insight.

the output, Self-BLEU $=$ BLEU$(output, input)$ is used in the context of paraphrase generation as a measure of the diversity introduced by a system. A good paraphrasing system should generate output that concurrently preserves the meaning of the input and introduces diversity compared to the input, leading Sun and Zhou (2012) to propose iBLEU. iBLEU is defined as a weighted difference between BLEU and Self-BLEU,

$$\begin{aligned}\text{iBLEU} = \alpha\ &\text{BLEU}(output, references) \\ -(1-\alpha)\ &\text{BLEU}(output, input),\end{aligned} \tag{2.26}$$

where α is a constant that weights the tradeoff between fidelity to the references and variation from the input. Intuitively, iBLEU rewards output that is similar to the references, but penalizes output that is similar to the inputs (and therefore has low diversity).

A range of methods have proposed taking advantage of the comparative strength of discriminative NLP models to assist with evaluation. Primarily designed for evaluating summarisation systems, Deutsch et al. (2021) and Fabbri et al. (2022) propose automatically generating questions about the outputs, attempting to automatically answer the questions based on both generated and reference output, and use the overlap between the answers as an indication of the semantic consistency between the generated outputs. FActScore (Min et al., 2023) uses a LLM to extract atomic facts from a long generated output, and compares these facts to an external database. Mahon and Lapata (2024) extended FActScore to the case where there is no external database, instead comparing the set of facts extracted from generated and reference outputs. However, metrics based on discriminative models are much more costly to compute than the simpler n-gram metrics, yet still require good quality references.

Both BLEU and ROUGE may not correctly account for minor lexical differences that may or may not affect the meaning of the output. For example, the addition of the word ‘not’ may completely change the meaning of a sentence. While their ease of computation makes them useful for model development, BLEU and ROUGE have been shown to correlate poorly with human judgements of quality (Callison-Burch et al., 2006; Tay et al., 2019; Fabbri et al., 2021; Freitag et al., 2022). All reference-based metrics rely on the availability of high-quality references, which may not be available in every language or domain.

In this thesis, we use iBLEU and ROUGE for paraphrase generation and opinion summarisation respectively, as metrics for model development.

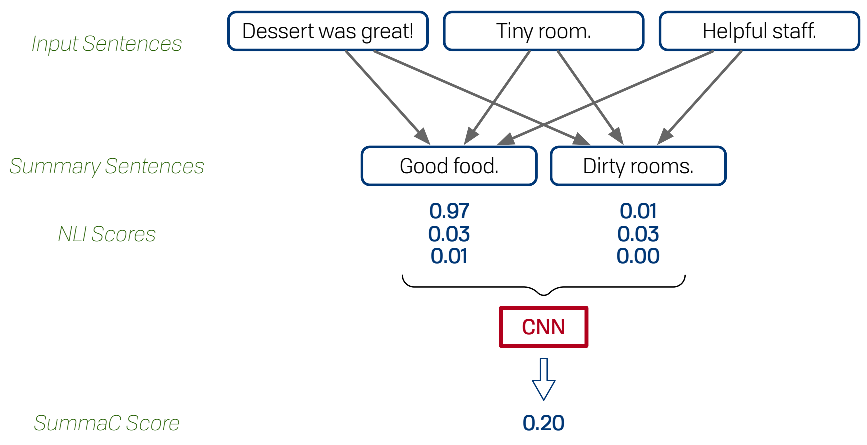


Figure 2.5: A simplified example showing the SummaC metric. A NLI model is used to calculate the entailment scores between each input-output sentence pair, then a shallow convolutional neural network (CNN) aggregates these component scores to give the overall SummaC score.

Reference-free metrics For tasks where the output should be semantically consistent with the input, a Natural Language Inference (NLI) model may be used to evaluate the degree to which generated output is entailed by the original input.

Entailment is directional; ‘Tibbles is a kitten’ entails that ‘Tibbles is a cat’, but not vice-versa. In the context of paraphrase generation, two sentences that are paraphrases of each other should also entail *each other*. We can use an NLI model to measure the whether a generated paraphrase is entailed by the input sentence (forward entailment) and vice-versa (backward entailment), and take the mean of the forward and backward scores (Zhang et al., 2024a) as a measure of semantic equivalence. We note that entailment is, strictly speaking, a binary property, but NLI models predict a score from 0 to 1. This continuous score may be interpreted as the confidence of the model.

When the input and/or output are formed of multiple sentences, it does not make sense to check whether the full input entails the full output. SummaC (Laban et al., 2022) considers entailment at a more granular level, segmenting inputs and outputs into sentences and measuring the entailment scores between each pair of input and output sentences. Then, a shallow convolutional neural network is used to aggregate these granular scores into an overall document-level score that indicates how strongly the input document supports the generated output. Figure 2.5 shows a simplified example of this process. InFusE (Zhang et al., 2024a) futher extends SummaC by comparing entailment scores in both a forward and backward manner at a more granular sub-sentence level.

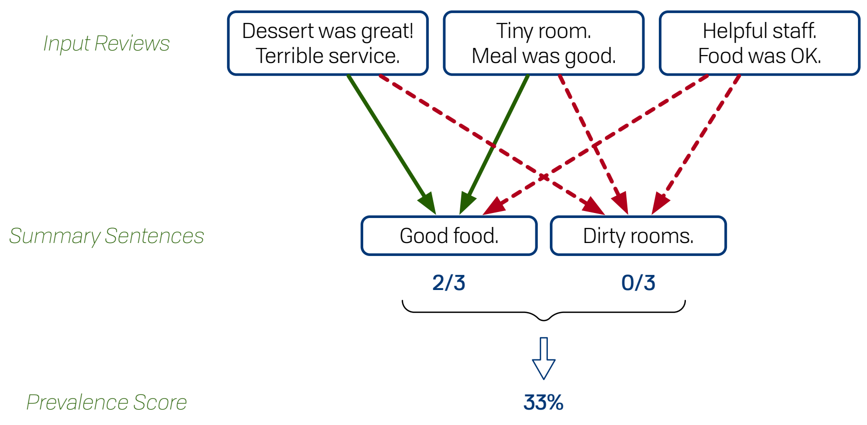

Input Reviews
Dessert was great! Terrible service.
Tiny room. Meal was good.
Helpful staff. Food was OK.
Summary Sentences
Good food.
Dirty rooms.
2/3
0/3
Prevalence Score
33%

Figure 2.6: A simplified example demonstrating the prevalence metric. Given three input reviews and two output sentences, a NLI model is used to determine whether each output sentence is supported (solid green) or not supported (dotted red) by each input review. The binary labels are aggregated across reviews and sentences to give the final prevalence score of 33%.

In the context of opinion summarisation, each sentence within the summary should be supported by many of the input reviews. Malon (2023) therefore propose 'prevalence', which uses SummaC to measure the entailment score between each summary sentence and each input *review*. This score is thresholded and meaned, giving the average number of reviews that support each generated sentence. We show an example of this calculation in Figure 2.6.

In this thesis, we use bidirectional NLI scores to automatically measure meaning preservation in paraphrase generation, and we use SummaC and prevalence for evaluating faithfulness in opinion summarisation.

In Chapter 7 we identify a major failure case of both SummaC and Prevalence. Trivial statements like "A moose is an animal" will be scored as strongly entailed by NLI models, regardless of the premise given to the model. Therefore, generated summaries that make very generic statements will achieve high SummaC and Prevalence scores. For example, using the statement "The rooms are clean and comfortable" as the summary for every entity in SPACE, a dataset of hotel reviews (Angelidis et al., 2021), achieves a prevalence score of 72% — for comparison, the human-written reference summaries score only 44%. This statement is clearly not a helpful summary, and we propose a modified version of prevalence in Chapter 7 to account for this failure mode.

Human Evaluation Human evaluation remains the preferred approach for evaluating model performance. Best-worst scaling, whereby annotators are asked for *comparative* judgements about system outputs rather than absolute ratings, has been shown to lead to improved rates of annotator agreement (Louviere and Woodworth, 1990; Kiritchenko and Mohammad, 2017; Novikova et al., 2018). However, human evaluation is not without its own pitfalls; Thomson et al. (2024) found that many human evaluation studies in NLP contain errors and are not executed correctly, while Hosking et al. (2023a) and Sharma et al. (2024) found that human annotators struggle to evaluate the factuality of generated text and may be biased by confounding factors like assertiveness or sycophancy.

In this thesis, we consider human evaluation to be the best way of measuring model performance. We employ best-worst scaling for our evaluations, where annotators are asked which of two generated outputs they think is better. Different tasks will have different desiderata: for paraphrase generation, outputs should be fluent while preserving the meaning of the original input utterance and introducing diversity to the surface form; for opinion summarisation, the generated summaries should be fluent, informative and accurately represent the opinions in the input reviews. The ideal balance between these factors is unknown and will likely depend on the specific downstream application of a system, and so we solicit pairwise preferences across each dimension.

Chapter 3

Factorising Meaning and Form for Paraphrase Generation

In Chapter 1 we hypothesised that weakly structured representations are beneficial for text-to-text generation (Hypothesis I). In this chapter, we offer our first contribution of evidence to support this hypothesis, using *paraphrase generation* as a case study. A paraphrase of an utterance is “an alternative surface form in the same language expressing the same semantic content as the original form” (Madnani and Dorr, 2010). Paraphrase generation therefore involves producing an output sentence that has the same semantic meaning but difference surface form, whether in terms of syntactic structure or lexical choice, as a given input sentence. This task description already gives a hint as to a natural choice of representation — the goal is to preserve semantic information and vary syntactic and lexical information. We therefore evaluate whether learning separate representations for the meaning and form of an input sentence can enable the generation of higher quality paraphrases, that introduce more diversity in the output surface form while better preserving the meaning of the input. We focus primarily on a method for introducing syntactic variation to the outputs, since this has been more challenging for the field so far.

In this chapter, we propose a method for generating paraphrases of English sentences that retain the original meaning but use a different surface form. Our model combines a principled information bottleneck with a careful choice of training objective, to induce a latent encoding space that disentangles meaning and form. We use a Vector-Quantised Variational Autoencoder (VQ-VAE, van den Oord et al., 2017) to represent the surface form as a set of discrete latent variables, allowing us to use a classifier to select a different surface form at test time (Hypothesis II). Crucially, our method does not require access to an external source of target exemplars during inference. Extensive experiments and a human evaluation show that we are able to generate paraphrases with a better tradeoff between semantic preservation and syntactic novelty compared to previous methods.

3.1 Introduction

We focus primarily on generating paraphrases of questions in English, for three reasons: (a) the concept of a paraphrase is more clearly defined for questions compared to generic utterances, as question paraphrases should lead to the same answer; (b) the space of possible surface forms is smaller for questions, making the task more achievable, and (c) better dataset availability. However, our approach does not otherwise make any assumptions specific to questions, and we include experiments on a dataset of image captions to evaluate our method on an additional domain.

For questions, a paraphrase should have the same intent as the original question, and should therefore lead to the same answer as the original. The clusters of questions shown in in Table 3.1 have different phrasings, but could all be answered by the same response. Their meanings are therefore grounded in a common (hypothetical) answer. Question paraphrases are of significant interest, with a range of applications: they may be used for data augmentation (Iyyer et al., 2018), to create additional training data for question answering systems; query rewriting (Dong et al., 2017) involves generating multiple equivalent phrasings for inputs to question answering systems and using the 'concensus' answer as the response; duplicate question detection (Shah et al., 2018) identifies question paraphrases in online forums, preventing redundant and unnecessarily duplicated work answering them individually.

Prior approaches to paraphrasing use information bottlenecks with VAEs (Bowman et al., 2016) or pivot languages (Wieting and Gimpel, 2018) to try to extract the semantics of an input utterance, before projecting back to a (hopefully different) surface form. However, these methods have little to no control over the preservation of the input meaning or variation in the output surface form. Other work has specified the surface form to be generated (Iyyer et al., 2018; Chen et al., 2019a; Kumar et al., 2020), but has so far assumed that the set of valid surface forms is known a priori.

In this chapter, we propose SEPARATOR, a method for generating paraphrases that exhibit high variation in surface form while still retaining the original meaning.

[1]Males normally weigh from 380 to 700 kg, and females typically weigh 200 to 490 kg.

How is a dialect different from a language?
The differences between language and dialect?
What is the difference between language and dialect?
What is the weight of an average moose?[1]
Average weight of the moose?
How much do moose weigh?
How heavy is a moose?
What country do parrots live in?
In what country do parrots live?
Where do parrots naturally live?
What part of the world do parrots live in?

Table 3.1: Examples of question paraphrase clusters, drawn from Paralex (Fader et al., 2013). Each member of the cluster has essentially the same semantic *intent*, but a different *surface form*. Each cluster exhibits variation in word choice, syntactic structure and even question type. Our task is to take one of these surface forms as input, and generate another alternative surface form from the same cluster.

Our key innovations are: (a) to train a model to reconstruct a target sentence from an input *paraphrase* with the same meaning, and an *exemplar* with the same surface form, and (b) to separately encode the form and meaning of sentences as discrete and continuous latent variables respectively, enabling us to modify the output surface form while preserving the original sentence meaning. Crucially, unlike prior work on syntax controlled paraphrasing (Iyyer et al., 2018; Chen et al., 2019a; Kumar et al., 2020), we show that we can generate diverse paraphrases of an input sentence at test time by inferring a different discrete syntactic encoding, without needing access to reference exemplars.

3.2 Related Work

Paraphrasing Prior work on generating paraphrases has looked at extracting sentences with similar meaning from large corpora (Barzilay and McKeown, 2001; Bannard and Callison-Burch, 2005; Ganitkevitch et al., 2013), or identifying paraphrases from sources that are weakly aligned (Dolan et al., 2004; Coster and Kauchak, 2011).

More recently, neural approaches to paraphrasing have shown promise. Several models have used an information bottleneck to try to encode the semantics of the input, including VAEs (Bowman et al., 2016) and VQ-VAEs (van den Oord et al., 2017; Roy and Grangier, 2019). Such models are trained to reconstruct one paraphase from another, with syntactic diversity introduced through by sampling from the posterior in latent space. This stochasticity makes them difficult to control, with no guarantee that the sampled latent encoding will correspond to the same meaning as the input, or will correspond to a valid output sentence.

Fu et al. (2019) introduce a latent bag-of-words model that represents an input sentence as a bag-of-words (which is predicted from the input and is necessarily the same set of words). Then, a decoder conditions on this bag-of-words to generate an output paraphrase. Their approach is more interpretable than VAE or VQ-VAE models but still offers little control over the output. Guo et al. (2021) extend this idea, coupling an encoder-decoder model that uses a set of words as the latent representation with a round-trip translation model to enforce semantic consistency. At decoding time, they combine the predictions from both models to balance semantic preservation with syntactic diversity.

Other work has relied on the strength of neural machine translation (MT) models, translating an input into a *pivot* language and then back into English (Mallinson et al., 2017; Wieting and Gimpel, 2018; Hu et al., 2019). These approaches exploit the property that MT models trained on particular language pairs are biased towards a particular choices of phrasing.

Kumar et al. (2019) improve the diversity of paraphrases generated by an encoder-decoder model, by using techniques from function maximisation to efficiently search a large number of possible output candidates. They show that this approach can select for more diverse outputs than standard beam search. Li et al. (2019) introduce multiple encoders for different granularities (i.e., lexical, phrasal, sentential), and aggregate their encodings during generation to encourage high level variation in the output. Lin and Wan (2021) use multiple rounds of paraphrasing to improve diversity, applying the same paraphrase model to its own output to iteratively generate paraphases that are less and less like the original.

Syntactic Templates The idea of generating paraphrases by controlling the structure of the output has seen recent interest, but most work so far has assumed access to a template oracle during inference.

Iyyer et al. (2018) use linearised parse trees to represent the syntactic structure of sentences, and train a two-stage model to generate an output sentence with this syntactic form. They create training data automatically, generating possible paraphrases with backtranslation and using a parser to extract their parse trees. However, they do not investigate how target parse trees might be obtained for novel inputs at test time.

Chen et al. (2019a,b) use a multi task objective to train a model to generate output that follows an input template, reconstructing a sentence from its own syntactic encoding and the semantic encoding from a known paraphrase. However, their approach is limited by their use of automatically generated paraphrases for training, and their reliance on the availability of oracle templates during inference.

Bao et al. (2019) propose learning separate spaces for meaning and form, using discriminators to assign meaning to these subspaces; one discriminator attempts to predict the bag-of-words of the input sentence from the semantic encoding, and another discriminator is trained to predict the linearised parse tree of the input from the syntactic encoding. However, they rely on adding stochastic noise to samples from the latent space to induce variation in the output form, which is not guaranteed to produce a valid target surface form. Their results show good fidelity to the references, but low variation compared to the input.

Goyal and Durrett (2020) propose an approach that first generates a target syntactic 'plan', then generates an output paraphrase based on this plan and the original input. However, they use the artifically generated dataset ParaNMT-50m (Wieting and Gimpel, 2018) for their training and evaluation, which displays low output variation according to our results.

Similar to Iyyer et al. (2018), Kumar et al. (2020) show strong performance using full parse trees as templates. They encode the parse tree directly (rather than linearising it), but focus on generating output with the correct parse and do not consider the problem of template prediction.

Huang and Chang (2021a) propose training a model with a similar 'denoising training' approach to ours, but using constituency parses instead of exemplars, and a 'bag-of-words' instead of reference paraphrases. Their approach has the advantage of not requiring paraphrase clusters during training, but they do not attempt to solve the problem of predicting valid syntactic forms and rely on the availability of oracle parse trees during inference.

Russin et al. (2020) modify the architecture of an encoder-decoder model, introducing an inductive bias to encode the structure of inputs separately from the lexical items to improve compositional generalisation on an artificial semantic parsing task. Wieting et al. (2020) propose a method for learning separated encoding spaces that is similar to ours, but with the goal of factorising meaning and *language* (e.g., French or English). Yang et al. (2021) propose using contrastive learning techniques to separately encode the meaning and form of input sentences, with a training objective similar to ours that involves reconstructing a target from two inputs with the correct meaning and surface form. However, they do not propose a method for predicting valid syntactic exemplars during inference, and their method requires access to oracle exemplars during evaluation.

Work since this chapter The main content in this chapter was published in Hosking and Lapata (2021). Since then, there has continued to be interested in generating diverse paraphrases.

Ormazabal et al. (2022) revisit paraphrase generation based on MT systems. They observe that an encoding of the semantics of an input sentence should encode as much information as possible about its reference translation, and as little information as possible about itself. They train an encoder to minimise the translation loss while maximising the reconstruction loss, resulting in an encoding that primarily represents the meaning of the input. This semantic encoding is then used to generate an output paraphrase.

Luo et al. (2023) propose leveraging pretrained Language Models for paraphrase generation. They encode the input sentence with the Flan-T5 encoder, add additional learned embeddings to the sequence to modify the output surface form, then decode using the Flan-T5 decoder. Similar to SEPARATOR, their learned embeddings are discretised and therefore may be predicted, but their use of a pretrained encoder and decoder result in superior quality paraphrases compared to our work.

Xue et al. (2023) build on our work in this chapter (and Chapter 5), and propose a encoder-decoder model similar to ours that uses a continuous semantic representation and a discrete syntactic representation. They propose training an auxiliary model to generate sentences from lists of keywords, and show that they can generate low-quality paraphrases by shuffling the order of these keywords. They then use this auxiliary model to generate large quantities of paraphrase pairs to train their main paraphrasing model, removing our requirement for labelled paraphrase pairs during training.

3.3 Problem Formulation

The task is to learn a mapping from an input sentence, represented as a sequence of tokens x, to paraphrase(s) y which have different *surface form* to x, but convey the same *meaning*. Examples of paraphrases are shown in Table 3.1.

Our proposed approach, which we call SEPARATOR, uses an encoder-decoder model to transform an input sentence into a latent encoding space, and then back to an output paraphrase. We hypothesize that a principled information bottleneck (Section 3.3.1) and a careful choice of training scheme (Section 3.3.2) lead to an encoding space that separately represents the meaning and surface form. This separation enables us to paraphrase the input sentence, varying the surface form of the output by directly manipulating the syntactic encoding of the input and keeping the semantic encoding constant (Section 3.3.3). We assume access to reference *paraphrase clusters* during training (e.g., Table 3.1), sets of sentences with different surface forms that have been collated as having the same meaning or *intent*.

Our model is a variant of the standard encoder-decoder framework (Cho et al., 2014a), and consists of:

(a) a vanilla Transformer sentence encoder (Vaswani et al., 2017), that maps an input

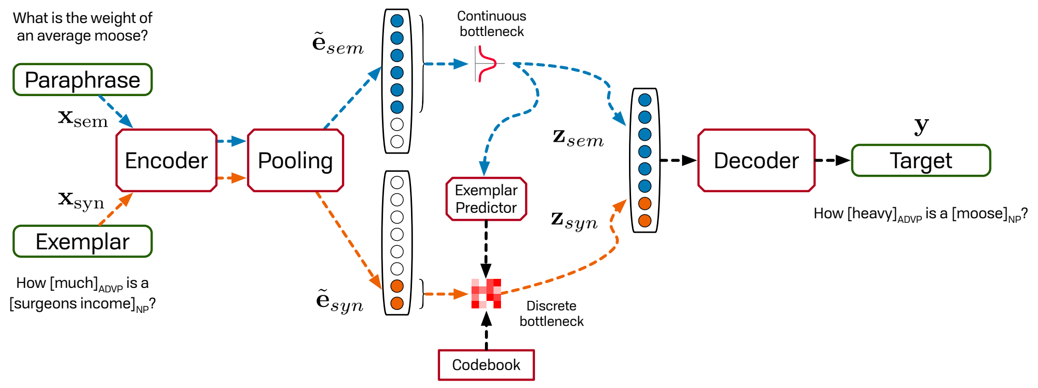

What is the weight of an average moose?
Paraphrase
$\mathbf{x}_{\text{sem}}$
Encoder
$\mathbf{x}_{\text{syn}}$
Exemplar
How [much]ADVP is a [surgeons income]NP?
Pooling
$\tilde{\mathbf{e}}_{sem}$
$\tilde{\mathbf{e}}_{syn}$
Continuous bottleneck
Exemplar Predictor
Discrete bottleneck
Codebook
$\mathbf{z}_{sem}$
$\mathbf{z}_{syn}$
Decoder
$\mathbf{y}$
Target
How [heavy]ADVP is a [moose]NP?

Figure 3.1: Overview of our approach. The model is trained to reconstruct a target sentence ($\mathbf{y}$) from one input with the same *meaning* but different surface form ($\mathbf{x}_{sem}$) and another input with the same *surface form* but different meaning ($\mathbf{x}_{syn}$). This induces separate latent encoding spaces for meaning and form, allowing us to vary the output form while keeping the meaning constant. Using a discretised space for the syntactic encoding makes it tractable to predict valid surface forms at test time.

sentence $\mathbf{x}$ to a multi-head sequence of encodings,

$$\mathbf{e}_{h,t} = \text{ENCODER}(\mathbf{x}); \tag{3.1}$$

(b) a principled choice of information bottleneck, with a continuous variational path and a discrete vector-quantised path, that maps the encoding sequence to a pair of latent vectors,

$$\mathbf{z}_{sem}, \mathbf{z}_{syn} = \text{BOTTLENECK}(\mathbf{e}_{h,t}), \tag{3.2}$$

represented in more detail in Figure 3.1;

(c) a vanilla Transformer decoder, that attends over the latent vectors to generate a sequence of output tokens,

$$\hat{\mathbf{y}} = \text{DECODER}(\mathbf{z}_{sem}, \mathbf{z}_{syn}). \tag{3.3}$$

The separation between $\mathbf{z}_{sem}$ and $\mathbf{z}_{syn}$ is induced by our proposed training scheme, shown in Figure 3.1 and described in detail in Section 3.3.2.

3.3.1 Model Architecture

While the encoder and decoder used by the model are standard Transformer modules, our bottleneck is more complex and we now describe it in more detail.

Let the encoder output be

$$\{\mathbf{e}_{h,1}, \ldots, \mathbf{e}_{h,|\mathbf{x}|}\} = \textsc{Encoder}(\mathbf{x}), \tag{3.4}$$

where $\mathbf{e}_{h,t} \in \mathbb{R}^{D/H_T}$, $h \in 1, ..., H_T$ with H_T the number of transformer heads, $|\mathbf{x}|$ the length of the input sequence and D the dimension of the transformer. We first pool this sequence of encodings to a single vector, using the multi-head pooling described in Liu and Lapata (2019). For each head h, we calculate a distribution over time indexes $\alpha_{h,t}$ using attention,

$$\alpha_{h,t} = \frac{\exp a_{h,t}}{\sum_{t' \in |\mathbf{x}|} \exp a_{h,t'}}, \tag{3.5}$$

$$a_{h,t} = \mathbf{k}_h^T \mathbf{e}_{h,t}, \tag{3.6}$$

with $\mathbf{k}_h \in \mathbb{R}^{D/H}$ a learned parameter.

We then take a weighted average of a linear projection of the encodings, to give pooled output $\tilde{\mathbf{e}}_h$,

$$\tilde{\mathbf{e}}_h = \sum_{t' \in |\mathbf{x}|} \alpha_{h,t'} V_h \mathbf{e}_{h,t'}, \tag{3.7}$$

with $V_h \in \mathbb{R}^{D/H \times D/H}$ a learned parameter.

Transformer heads are assigned either to a *semantic* group H_{sem}, that will be trained to encode the meaning of the input, $\tilde{\mathbf{e}}_{sem} = [\ldots; \tilde{\mathbf{e}}_h; \ldots], h \in H_{sem}$, or to a *syntactic* group H_{syn}, that will be trained to represent the surface form $\tilde{\mathbf{e}}_{syn} = [\ldots; \tilde{\mathbf{e}}_h; \ldots], h \in H_{syn}$ (see Figure 3.1).

The space of possible semantic meanings is extremely large and may be reasonably approximated by a continuous vector space. However, the possible surface forms are discrete and smaller in number. We therefore model the semantic encoding as a continuous latent variable $\mathbf{z}_{sem}$, and use a VQ-VAE for the syntactic encoding $\mathbf{z}_{syn}$, as shown in the upper and lower parts of Figure 3.1, respectively. For simplicity, we assume that $\mathbf{z}_{sem}$ is Gaussian, but future work could consider more principled choices for the semantic encoding.

Vector Quantisation Let q_h be discrete latent variables corresponding to the syntactic quantiser heads, $h \in H_{syn}$.[2] Each variable can be one of K possible latent codes, $q_h \in [0, K]$. The heads use distinct codebooks, $\mathbf{C}_h \in \mathbb{R}^{K \times D/H}$, which map each discrete code to a continuous embedding $\mathbf{C}_h(q_h) \in \mathbb{R}^{D/H}$. Given sentence $\mathbf{x}$ and its

[2]The number and dimensionality of the quantiser heads need not be the same as the number of transformer heads.

pooled encoding $\{\tilde{\mathbf{e}}_1, ..., \tilde{\mathbf{e}}_H\}$, we independently quantise the syntactic subset of the heads $h \in H_{syn}$ to their nearest codes from $\mathbf{C}_h$ and concatenate, giving the syntactic encoding

$$\mathbf{z}_{syn} = [\mathbf{C}_1(q_1); \ldots ; \mathbf{C}_{|H_{syn}|}(q_{|H_{syn}|})]. \tag{3.8}$$

The quantiser module is trained through backpropagation using straight-through estimation (Bengio et al., 2013), with an additional loss term to constrain the embedding space as described in van den Oord et al. (2017) and Section 2.4,

$$\mathcal{L}_{cstr} = \lambda \sum_{h \in H_{syn}} \left\| \left(\tilde{\mathbf{e}}_h - \mathrm{sg}(\mathbf{C}_h(q_h)) \right) \right\|_2, \tag{3.9}$$

where the stop gradient operator $\mathrm{sg}(\cdot)$ is defined as identity during forward computation and zero on backpropagation, and λ is a weight that controls the strength of the constraint. We follow the soft EM and exponentially moving averages training approaches described in earlier work (Roy et al., 2018; Angelidis et al., 2021) and in Chapter 2, which we find improve training stability.

Variational Bottleneck For the semantic path, we introduce a learned Gaussian posterior, that represents the encodings as smooth distributions in space instead of point estimates (Kingma and Welling, 2014). Formally, $\phi(\mathbf{z}_{sem}|\mathbf{e}_{sem}) \sim \mathcal{N}(\boldsymbol{\mu}(\mathbf{e}_{sem}), \boldsymbol{\sigma}(\mathbf{e}_{sem}))$, where $\boldsymbol{\mu}(\cdot)$ and $\boldsymbol{\sigma}(\cdot)$ are learned linear transformations. To avoid vanishingly small variance and to encourage a smooth distribution, a prior is introduced, $\mathbf{p}(\mathbf{z}_{sem}) \sim \mathcal{N}(\mathbf{0}, \mathbf{1})$. The VAE objective is the standard evidence lower bound (ELBO), given by

$$\mathrm{ELBO} = -\mathrm{KL}[\phi(\mathbf{z}_{sem}|\mathbf{e}_{sem})||p(\mathbf{z}_{sem})] + \mathbb{E}_\phi[\log p(\mathbf{e}_{sem}|\mathbf{z}_{sem})]. \tag{3.10}$$

We use the usual Gaussian reparameterisation trick, and approximate the expectation in Equation (3.10) by sampling from the training set and updating via backpropagation (Kingma and Welling, 2014). The VAE component therefore adds an additional KL term to the overall loss (Section 2.3),

$$\mathcal{L}_{KL} = -\mathrm{KL}[\phi(\mathbf{z}_{sem}|\mathbf{e}_{sem})||p(\mathbf{z}_{sem})]. \tag{3.11}$$

The final combined training objective is therefore given by

$$\begin{aligned} \mathcal{L} &= \mathcal{L}_{\mathbf{y}} + \mathcal{L}_{cstr} + \mathcal{L}_{KL} && (3.12) \\ &= \mathcal{L}_{\mathbf{y}} + \lambda \sum_{h \in H_{syn}} \left\| \left(\tilde{\mathbf{e}}_h - \mathrm{sg}(\mathbf{C}_h(q_h)) \right) \right\|_2 - \beta \mathrm{KL}[\phi(\mathbf{z}_{sem}|\mathbf{e}_{sem})||p(\mathbf{z}_{sem})] && (3.13) \end{aligned}$$

where $\mathcal{L}_{\mathbf{y}}(\mathbf{x}_{sem}, \mathbf{x}_{syn})$ is the cross-entropy loss of teacher-forcing the decoder to generate $\mathbf{y}$ from $\mathbf{z}_{sem}(\mathbf{x}_{sem})$ and $\mathbf{z}_{syn}(\mathbf{x}_{syn})$, and β controls the strength of the KL term. Note that $\mathcal{L}$ is no longer an ELBO, but the additional terms (and weights) are nonetheless beneficial for stable training using gradient descent.

In summary, $\textsc{Bottleneck}(\mathbf{e}_{h,t})$ maps a sequence of token encodings to a pair of vectors $\mathbf{z}_{sem}, \mathbf{z}_{syn}$, with $\mathbf{z}_{sem}$ a continuous latent Gaussian, and $\mathbf{z}_{syn}$ a combination of discrete code embeddings.

3.3.2 Factorised Reconstruction Objective

We now describe the training scheme that causes the model to learn separate encodings for meaning and form — $\mathbf{z}_{sem}$ should encode only the semantics of the input, while $\mathbf{z}_{syn}$ should capture any information about the surface form of the input. Although we refer to $\mathbf{z}_{syn}$ as the *syntactic encoding*, it will not necessarily correspond to any specific syntactic formalism. We also acknowledge that meaning and form are not completely independent of each other; arbitrarily changing the form of an utterance is likely to change its meaning. However, it is possible for the same meaning to have multiple phrasings, and it is this 'local independence' that we intend to capture.

We create triples $\{\mathbf{x}_{sem}, \mathbf{x}_{syn}, \mathbf{y}\}$, where $\mathbf{x}_{sem}$ has the same meaning but different form to $\mathbf{y}$ (i.e., it is a paraphrase, as in Table 3.1) and $\mathbf{x}_{syn}$ is a sentence with the same form but different meaning (i.e., it shares the same syntactic *template* as $\mathbf{y}$), which we refer to as an *exemplar*. In this context, 'template' refers to the type of syntactic form, whereas 'exemplar' refers to a particular instantiation of that template. We describe the method for retrieving these exemplars in Section 3.3.3. The model is then trained to generate a target paraphrase $\mathbf{y}$ from the semantic encoding $\mathbf{z}_{sem}$ of the input paraphrase $\mathbf{x}_{sem}$, and from the syntactic encoding $\mathbf{z}_{syn}$ of the exemplar $\mathbf{x}_{syn}$, as demonstrated in Figure 3.1.

Early experiments showed that, while the model was able to separately encode meaning and form, the 'syntactic' encoding space showed little ordering. That is, local regions of the encoding space did not necessarily encode templates that *co-occurred* with each other in paraphrase clusters. We therefore propose *template dropout*, where exemplars $\mathbf{x}_{syn}$ are replaced with probability $p_{td} = 0.3$ by a sentence with a different template from the same paraphrase cluster. This is intended to provide the model with a signal about which templates occur in similar semantic contexts, and thus reduce the distance between their encodings.

Input	How heavy is a moose?
Chunker output	How [heavy]$_{\text{ADVP}}$ is a [moose]$_{\text{NP}}$?
Template	How ADVP is a NP ?
Exemplar	How much is a surgeon's income?
Input	What country do parrots live in
Chunker output	What [country]$_{\text{NP}}$ do [parrots]$_{\text{NP}}$ [live]$_{\text{VP}}$ in ?
Template	What NP do NP VP in ?
Exemplar	What religion do Portuguese believe in?

Table 3.2: Examples of the exemplar retrieval process for training. The input is tagged by a chunker, ignoring stopwords. An exemplar with the same template is then retrieved from the training corpus.

3.3.3 Specifying the Syntactic Form

Exemplar Construction As shown in Figure 3.1, our approach requires exemplars during training to induce the separation between latent spaces. These exemplars should have the same surface form as the target sentence but different meaning, such that the model learns to encode only syntactic information in $\mathbf{z}_{syn}$. Exemplars are not generally available, and must therefore be constructed or retrieved automatically.

During training, we retrieve exemplars $\mathbf{x}_{syn}$ from the training data following a process which first identifies the underlying syntax of $\mathbf{y}$, and finds a sentence with the same syntactic structure but a different, arbitrary meaning. We use a shallow approximation of syntax, to ensure the availability of equivalent exemplars in the training data. An example of the exemplar retrieval process is shown in Table 3.2; we first apply a chunker (FlairNLP, Akbik et al., 2018) to $\mathbf{y}$, then extract the chunk label for each tagged span, ignoring stopwords. This gives us the *template* that $\mathbf{y}$ follows. We then select a sentence at random from the training data with the same template to give $\mathbf{x}_{syn}$. If no other sentences in the dataset use this template, we create an exemplar by replacing each chunk with a random sample of the same type.

We experimented with a range of approaches to determining sentence templates, including using part-of-speech tags and (truncated) constituency parses. We found that using chunks and preserving stopwords gave a reasonable level of granularity while still grouping together sentences with a similar form. The templates (and corresponding exemplars) need to be granular enough that the model is forced to use them, but abstract enough that the task is not impossible to learn.

Prediction at Test Time We also need to specify the desired surface form at *test time*, either by supplying an exemplar as input or by directly predicting the latent codes. The output should have a different surface form to the input but remain fluent.

In general, we do not assume access to reference exemplars at test time and yet the decoder must generate a paraphrase from semantic *and* syntactic encodings. Since our representation spaces are separated, we can directly *predict* the syntactic encoding, without needing to retrieve or generate an exemplar. Furthermore, by using a discrete representation for the syntactic space, we reduce this prediction problem to a simple classification task. It is important to note that not all surface forms are valid or *licensed* for a given meaning, and the classification must therefore be conditioned on the meaning of the utterance.

Formally, for an input sentence $\mathbf{x}$, we learn a distribution over licensed discrete codes $q_h, h \in \tilde{H}_{syn}$. We assume that the heads are independent, so that

$$p(q_1, \ldots, q_{\tilde{H}_{syn}}|\mathbf{x}) = \prod_i p(q_i|\mathbf{x}). \tag{3.14}$$

We use a small fully connected network with the semantic and syntactic encodings of $\mathbf{x}$ as inputs, giving

$$p(q_h|\mathbf{x}) \;=\; \text{MLP}(\mathbf{z}_{sem}(\mathbf{x}), \mathbf{z}_{syn}(\mathbf{x})). \tag{3.15}$$

The network is trained to maximize the likelihood of all other syntactic codes licensed by each input. We calculate the discrete syntactic codes for each sentence in a paraphrase cluster, and minimize the cross-entropy loss of the network with respect to these codes. At test time, we set $q_h = \text{argmax}_{q'_h}\left[p(q'_h|\mathbf{x}_{test})\right]$.

3.4 Experimental Setup

Datasets A paraphrase is 'an alternative surface form in the same language expressing the same semantic content as the original form' (Madnani and Dorr, 2010), but it is not always clear what counts as the 'same semantic content'. Our approach requires access to reference paraphrases; we evaluate on two English datasets of question paraphrases which have clear grounding for the meaning of each sentence. Each cluster of paraphrases is grounded to a (hypothetical) *answer* they share. Paralex (Fader et al., 2013) is a dataset of question paraphrase clusters scraped from WikiAnswers. Quora Question Pairs (QQP)[3] and is sourced from the community question answering forum

[3]https://www.kaggle.com/c/quora-question-pairs

	Paralex		**QQP**		**MSCOCO**	
	Clusters	Questions	Clusters	Questions	Clusters	Captions
Train	222,223	1,450,759	55,611	138,965	113,287	566,742
Dev	27,778	183,273	5,255	12,554	5,000	25,011
Test	27,778	182,818	5,255	12,225	5,000	25,014

Table 3.3: Summary statistics for our cleaned version of (Fader et al., 2013), and our partitioning of QQP.

Quora. We additionally evaluate how well SEPARATOR performs on a different domain, using MSCOCO 2017 (Lin et al., 2014), a set of images that have been captioned by multiple annotators. We evaluate on the public validation set, randomly selecting one caption for each image to use as input and using the remaining four as references.

We observed that a significant fraction of the questions in Paralex included typos or were ungrammatical. We therefore filter out any questions marked as non-English by a language detection script (Lui and Baldwin, 2012), then pass the questions through a simple spellchecker. While this destructively edited some named entities in the questions, it did so in a consistent way across the whole dataset. There is no canonical split for Paralex, so we group the questions into clusters of paraphrases, and split these clusters into train/dev/test partitions with weighting 80/10/10. Similarly, QQP does not have a public test set. We therefore partitioned the clusters in the validation set randomly in two, to give us our dev/test splits. For MSCOCO, we evaluate on the public validation set. Each image is associated with five captions, so we randomly select one caption for each image to use as input and using the remaining four as references. All scores reported are on our test splits, and we trained and evaluated all baseline models ourselves for consistency.

Summary statistics for our partitions of Paralex, QQP and MSCOCO are shown in Table 3.3. Questions in QQP were 9.7 tokens long on average, compared to 8.2 for Paralex, while the image captions in MSCOCO average 11.3 tokens in length. We show some examples of paraphrase clusters from each of the datasets in Table 3.4

We also show the distribution of different question types in Figure 3.2; QQP contains a higher percentage of *why* questions, and we found that the questions tend to be more subjective compared to the predominantly factual questions in Paralex.

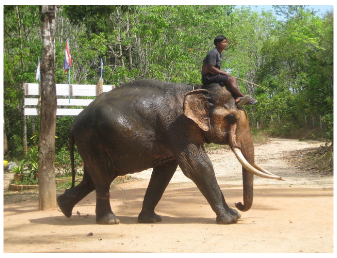
Paralex	Where was christianity spread and what location? Why was christianity able to spread? How did christianity spread westward? How did missionaries help to spread christianity? What did constantine do to spread christianity? How did christianity spread from europe to the rest of the world? How did christianity become accepted? How did monasteries help to continue the spread of christianity?
QQP	What are the most important books ever written? What are some of the best books ever written? The best book you have ever read What is the best book or book series you ever read and why? What's a good book to read? What is the best book ever made? What is the most important book you have ever read?
MSCOCO	 A man rides on top of the head of an elephant. A man riding on an elephant head in the road A man sitting on the head of an elephant. A person riding on an elephants head walking on a dirt road A man riding on the head of an elephant on a road.

Table 3.4: Examples of paraphrase clusters from each of the three datasets used in our experiments: Paralex, QQP, and MSCOCO. The question paraphrases are grounded in a shared long-form answer (not shown), while the MSCOCO paraphrases are grounded in a common image.

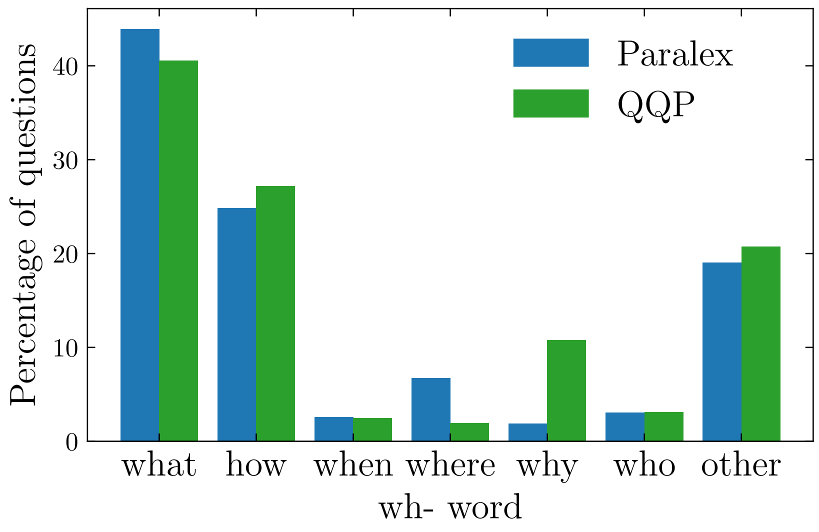


Figure 3.2: Distribution of wh- words for the datasets used in our experiments. QQP contains a much higher percentage of *why* questions.

Model Configuration Following previous work (Kaiser et al., 2018; Angelidis et al., 2021), our quantiser uses multiple heads ($H = 4$) with distinct codebooks to represent the syntactic encoding as 4 discrete categorical variables q_h, with $\mathbf{z}_{syn}$ given by the concatenation of their codebook embeddings $\mathbf{C}_h(q_h)$. We use a relatively small codebook size of $K = 256$, relying on the combinatoric power of the multiple heads to maintain the expressivity of the model. We argue that, assuming each head learns to capture a particular property of a template (see Section 3.5.3, Head Specialisation), the number of variations in *each property* is small, and it is only through combination that the space of possible templates becomes large.

We include a detailed list of hyperparameters in Appendix B.1. Our code is available at `http://github.com/tomhosking/separator`.

Comparison Systems We compare SEPARATOR against several related systems, described in more detail in Section 3.2. These include a model which reconstructs $\mathbf{y}$ only from $\mathbf{x}_{sem}$, with no signal for the desired form of the output. In other words, we derive both $\mathbf{z}_{sem}$ and $\mathbf{z}_{syn}$ from $\mathbf{x}_{sem}$, and *no separation* between meaning and form is learned. This model uses a continuous Gaussian latent variable for both $\mathbf{z}_{syn}$ and $\mathbf{z}_{sem}$, but is otherwise equivalent in architecture to SEPARATOR. We refer to this as the **VAE** baseline. We also experiment with a vanilla autoencoder or **AE** baseline by removing the variational component, such that $\mathbf{z}_{sem}, \mathbf{z}_{syn} = \tilde{\mathbf{e}}_{sem}, \tilde{\mathbf{e}}_{syn}$.

We include our own implementation of the **VQ-VAE** model described in Roy and

Grangier (2019). They use a quantised bottleneck for *both* $\mathbf{z}_{sem}$ and $\mathbf{z}_{syn}$, with a large codebook K = 64,000, H = 8 heads and a residual connection within the quantiser. For QQP, containing only 55,611 training clusters, the configuration in Roy and Grangier (2019) leaves the model overparameterised and training did not converge; we instead report results for K = 1,000. By contrast, SEPARATOR uses only K = 256 codes.

ParaNMT (Wieting and Gimpel, 2018) translates input sentences into a pivot language (Czech, chosen arbitrarily), then back into English. Although this system was trained on high volumes of data (including Common Crawl), the training data contains relatively few questions, and we would not expect it to perform well on the two datasets of question paraphrases.

'Diverse Paraphraser using Submodularity' (**DiPS**; Kumar et al. 2019) uses efficient optimisation to search a wider space of possible outputs and thereby increase the diversity of samples from a standard encoder-decoder model.

Latent bag-of-words (**LBoW**; Fu et al. 2019) uses an encoder-decoder model with a discrete bag-of-words as the latent encoding.

SOW/REAP (Goyal and Durrett, 2020) uses a two stage approach, deriving a set of feasible syntactic rearrangements that is used to guide a second encoder-decoder model.

BTmPG (Lin and Wan, 2021) uses multi-round generation to improve diversity and a reverse paraphrasing model to preserve semantic fidelity. We use the results after 10 rounds of paraphrasing.

We implement a simple **tf-idf** baseline (Jones, 1972), retrieving the sentences from the training set with the highest similarity to the input. We include a basic **copy** baseline as a lower bound, that simply uses the input sentence as the output.

We compare to an instruction-tuned LLM, **Mistral 7B** Instruct v0.2, one of the strongest performing open-weight LLMs available at the time of writing. The LLM was prompted in a zero-shot manner according to Prompt A.1. However, it is not an entirely fair comparison; LLMs were developed later than the other models, and use orders of magnitude more data and computational resources during training. For example, while Mistral has 7 billion trainable parameters, Separator has 70 million, fewer than 1% of Mistral. The training data is also unknown, and it is possible that the model was trained on the evaluation splits of one or more of the datasets in our experiments.

Encoding	*Cluster type* **Paraphrase**	**Template**
$\mathbf{z}_{sem}$	94.3	9.6
$\mathbf{z}_{syn}$	95.2	9.2
$\mathbf{z}$	96.0	9.6

(a) VAE Baseline

Encoding	*Cluster type* **Paraphrase**	**Template**
$\mathbf{z}_{sem}$	94.4	5.3
$\mathbf{z}_{syn}$	6.5	86.6
$\mathbf{z}$	30.7	84.9

(b) SEPARATOR

Table 3.5: Retrieval accuracies (%) for each encoding for semantic and syntactic clusters, indicating which types of information are encoded in which parts of the representation. The VAE baseline is trained only on paraphrase pairs and receives no signal for the desired form of the output. SEPARATOR is able to learn separate encodings for meaning and form, with negligible loss in semantic encoding performance.

3.5 Experiments

Our experiments were designed to answer three research questions: (a) Does SEPARATOR effectively factorise meaning and form? (b) Does SEPARATOR successfully generate diverse paraphrases (while preserving the meaning of the input)? (c) What does the underlying quantised space encode (i.e., can we identify any meaningful syntactic properties)? We address each of these questions in the following sections.

3.5.1 Verification of Separation

Inspired by Chen et al. (2019b), we use a *semantic textual similarity* task and a *template detection* task to confirm that SEPARATOR does indeed lead to encodings $\{\mathbf{z}_{sem}, \mathbf{z}_{syn}\}$ in latent spaces that represent different types of information.

Using the test set of Paralex, we construct clusters of sentences that share the same meaning $\mathcal{C}_{sem}$ (i.e., paraphrase clusters like those in Table 3.4), and clusters that share the same template $\mathcal{C}_{syn}$ (see Table 3.6 for examples). For each cluster $C_q \in \{\mathcal{C}_{sem}, \mathcal{C}_{syn}\}$, we extract one sentence at random $\mathbf{x}_q \in C_q$, compute its encodings $\{\mathbf{z}_{sem}, \mathbf{z}_{syn}, \mathbf{z}\}$, where $\mathbf{z}$ refers to the combined encoding, i.e., $[\mathbf{z}_{sem}; \mathbf{z}_{syn}]$, and its cosine similarity to the encodings of all other sentences in the test set. We take the sentence with maximum similarity to the query $\mathbf{x}_r, r = \text{argmax}_{r'}(\mathbf{z}_q.\mathbf{z}_{r'})$, and compare the cluster that it belongs to, C_r, to the query cluster $\mathbb{I}(C_q = C_r)$, giving a *retrieval* accuracy score for each encoding type and each clustering type. For the VAE, we set $\{\mathbf{z}_{sem}, \mathbf{z}_{syn}\}$ to be the

Template	**Cluster**
How many NP does NP have?	How many shoes does Imelda Romualdez have?
	How many students does Dayton university have?
	How many national titles does Florida have?
	How many siblings does Nancy Hart have?
	How many moons does Neptune have?
What is the NP in the NP?	What is the problem in the odyssey?
	What is the population in the california coast?
	What is the widest suspension bridge in the world?
	What is the lenses' function in the eyeball?
	What is the temperature in the oceanic crust?
Are NP ADJP?	Are wooden spoons dishwasher safe?
	Are nightshade berries edible?
	Are high fiber carbohydrates healthy?
	Are andean countries tropical?

Table 3.6: Examples of clusters $\mathcal{C}_{syn}$ of questions from Paralex that share the same syntactic template.

same heads of $\mathbf{z}$ as the separated model.

Table 3.5 shows that our approach yields encodings that successfully factorise meaning and form, with negligible performance loss compared to the VAE baseline; paraphrase retrieval performance using $\mathbf{z}_{sem}$ for the separated model is comparable to using $\mathbf{z}$ for the VAE.

3.5.2 Paraphrase Generation

Automatic Evaluation While we have shown that our approach leads to disentangled representations, we are ultimately interested in generating diverse paraphrases for *unseen data*. That is, given some input sentence, we want to generate an output sentence with the same meaning but different form.

We use iBLEU (Sun and Zhou, 2012) as our primary metric, a variant of BLEU (Papineni et al., 2002; Post, 2018) that is penalised by the similarity between the output and the *input* (see Equation (2.26)). Following the recommendations of Sun and Zhou (2012), we set $\alpha = 0.8$, with a sensitivity analysis shown in Figure 3.3.

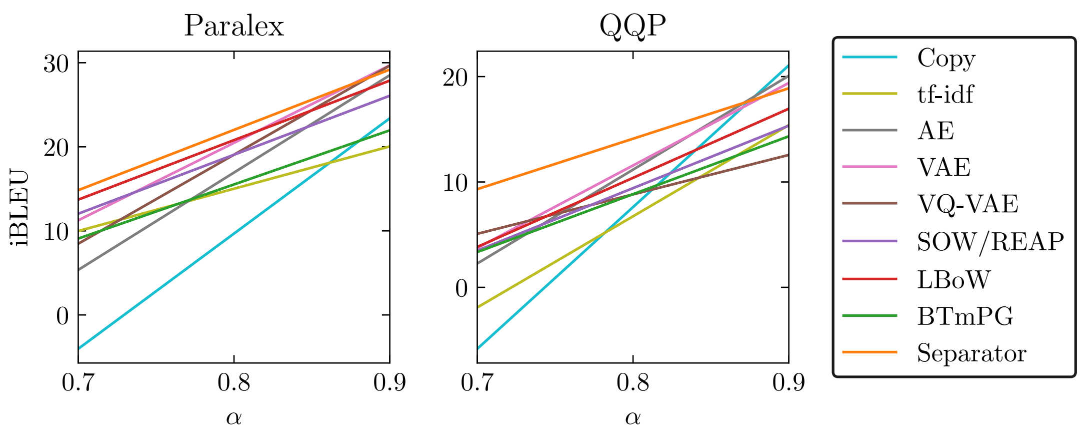


Figure 3.3: iBLEU scores for all comparison systems, for a range of values of α. For almost every choice of alpha between 0.7 and 0.9, Separator achieves the highest iBLEU scores.

We also report the usual BLEU(output, references) as well as Self-BLEU(output, input). The latter allows us to evaluate the degree of diversity introduced by the models. The Paralex test set contains 5.6 references on average per cluster and 4.0 for MSCOCO, while QQP contains only 1.3. This leads to lower BLEU scores for QQP in general, since the models are evaluated on whether they generated the specific paraphrase(s) present in the dataset.

Reference-based metrics rely on the availability of high-quality and diverse reference paraphrases, which are not always available. Sentences that are paraphrases of each other should entail each other, so we also use a *reference-free* metric based on Natural Language Inference (NLI). We use an NLI model (DeBERTa v3, trained on Debiased NLI; He et al., 2021; Wu et al., 2022) to measure the degree to which the generated paraphrase is entailed by the input sentence (forward entailment) and vice-versa (backward entailment), and report the mean of the forward and backward scores as 'NLI' (Zhang et al., 2024a).

Tables 3.7 and 3.8 show that the Copy, VAE and AE models display relatively high BLEU scores, but achieve this by 'parroting' the input; they are good at reconstructing the input, but introduce little variation in surface form, reflected in the high Self-BLEU scores. This highlights the importance of considering similarity to both the references *and* to the input. The tf-idf baseline achieves surprisingly strong iBLEU scores on Paralex; the large dataset size gives a higher chance of finding a paraphrase cluster with a similar meaning to the query in the training set.

The other comparison systems (in the third block in Tables 3.7 and 3.8) achieve lower

	Paralex			
Model	BLEU ↑	Self-BLEU ↓	**iBLEU** ↑	NLI ↑
Copy	37.1	100.0	9.7	97.4
tf-idf	25.1	**25.3**	15.0	26.1
AE	**39.7**	69.4	17.9	87.2
VAE	39.2	53.2	20.8	78.3
VQ-VAE	36.0	52.3	18.3	74.1
SOW/REAP	33.1	37.1	19.1	62.8
LBoW	26.4	27.9	15.5	8.9
BTmPG	28.4	36.0	15.5	51.6
DiPS	25.7	28.3	14.9	28.9
ParaNMT	27.5	52.0	11.6	**92.6**
SEPARATOR	36.3	35.4	**22.0**	60.8
Mistral 7B	13.4	14.1	7.9	88.1
Oracle$_{\text{SEPARATOR}}$	52.0	24.4	36.7	50.9

Table 3.7: Automatic evaluation results for paraphrase generation, on Paralex. Best scores are bolded. SEPARATOR achieves the highest iBLEU scores, indicating the best tradeoff between output diversity and fidelity to the reference paraphrases.

	QQP			
Model	BLEU ↑	Self-BLEU ↓	**iBLEU** ↑	NLI ↑
Copy	34.5	100.0	7.6	98.6
tf-idf	24.1	62.5	6.7	59.7
AE	**29.5**	61.8	11.3	89.8
VAE	21.3	39.8	9.1	71.7
VQ-VAE	28.3	56.8	11.3	84.6
SOW/REAP	17.4	30.4	7.9	55.1
LBoW	23.1	41.2	10.3	22.9
BTmPG	20.9	36.5	9.4	59.8
DiPS	18.8	28.6	9.3	33.2
ParaNMT	25.7	57.6	9.0	**94.6**
SEPARATOR	23.9	**23.5**	**14.5**	59.9
Mistral 7B	8.9	12.6	4.6	90.9
Oracle$_{\text{SEPARATOR}}$	40.9	26.4	27.4	62.8

Table 3.8: Automatic evaluation results for paraphrase generation, on QQP. Best scores are bolded. SEPARATOR achieves the highest iBLEU scores, indicating the best tradeoff between output diversity and fidelity to the reference paraphrases.

	MSCOCO			
Model	BLEU ↑	Self-BLEU ↓	**iBLEU** ↑	NLI ↑
Copy	19.9	100.0	-4.1	98.6
tf-idf	18.3	38.4	6.9	42.2
AE	**27.6**	39.3	14.2	61.7
VAE	27.3	24.1	**17.0**	43.9
VQ-VAE	25.9	28.4	15.1	41.2
SOW/REAP	12.5	**6.5**	8.7	30.9
LBoW	21.6	16.5	14.0	27.1
BTmPG	21.3	13.8	14.3	24.5
DiPS	19.0	14.4	12.3	28.6
ParaNMT	15.4	50.6	2.2	**91.3**
SEPARATOR	20.6	12.8	13.9	20.4
Mistral 7B	9.7	16.0	4.6	93.6
Oracle$_{\text{SEPARATOR}}$	38.1	9.7	28.6	22.3

Table 3.9: Automatic evaluation results for paraphrase generation, on MSCOCO. Best scores are bolded. SEPARATOR was originally designed primarily for questions, and performs less well on the broader range of domains covered by image captions in MSCOCO compared to other systems.

Self-BLEU scores, indicating a higher degree of variation introduced, but this comes at the cost of much lower scores with respect to the references. SEPARATOR achieves the highest iBLEU scores on the question datasets (Paralex and QQP), indicating the best balance between fidelity to the references and novelty compared to the input.

However, SEPARATOR performs less well compared to other systems on the image captions in MSCOCO (Table 3.9), which cover a wider range of domains. The space of possible syntactic forms for image captions is much wider than for questions, and the discretised represention used by SEPARATOR is perhaps insufficiently expressive to capture this range.

Mistral 7B achieves low scores for both BLEU and Self-BLEU, but some of the highest NLI scores. This indicates that it is succesfully generating outputs that are similar in meaning but have different surface form to the input, but that these outputs are also dissimilar to the reference paraphrases. While the other comparison systems were trained on the specific datasets being considered, Mistral 7B is evaluated zero-shot and so may generate outputs with a different style.

The last rows in Tables 3.7 and 3.8 (Oracle) report results when our model is given a valid exemplar to use directly for generation, thus bypassing the code prediction problem. For each paraphrase cluster, we select one sentence at random to use as input, and select another to use as the target. We retrieve a sentence from the training set with the same template as the target to use as an *oracle exemplar*. This represents an upper bound on our model's performance. While SEPARATOR outperforms existing methods, our approach to predicting syntactic codes (using a shallow fully-connected network) is relatively simple. SEPARATOR using oracle exemplars achieves by far the highest iBLEU scores in Tables 3.7 and 3.8, demonstrating the potential expressivity of our approach when exemplars are guaranteed to be valid. A more powerful code prediction model could close the gap to this upper bound, as well as enabling the generation of multiple diverse paraphrases for a single input sentence.

ParaNMT consistently achieves the highest NLI scores, indicating that it best preserves the meaning of the input sentence, but also displays some of the highest Self-BLEU scores. We define *dissimilarity* as $100 -$ Self-Bleu, so that a dissimilarity score of 100 indicates no overlap between the input and the generated paraphrase. Then, we plot NLI scores against dissimilarity for Paralex and QQP in Figure 3.4, where the ideal system would fall in the top-right corner. In both cases, Separator demonstrates the best trade-off between dissimilarity and meaning preservation compared to other (non-LLM) systems.

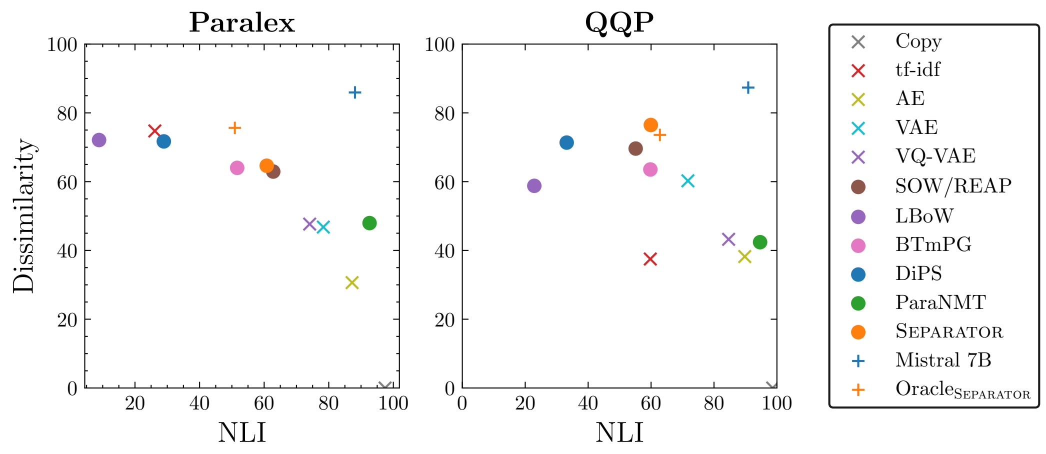


Figure 3.4: Dissimilarity (defined as 100 − Self-Bleu) against NLI scores for all models tested. The ideal model would be placed at the top-right of the plot. While Mistral 7B outperforms other systems, it was trained using many orders of magnitude more data and computational resources. SEPARATOR offers the best balance between meaning preservation and dissimilarity of the non-LLM systems.

The instruction tuned LLM, Mistral 7B, outperforms all comparison systems by some margin for all three datasets, achieving the highest NLI scores concurrently with some of the highest dissimilarity. However, there is a large disparity in parameter count and training resources required for the different models. Mistral 7B uses 7 billion parameters, was trained using extensive and costly pretraining, and was instruction-tuned on a wide range of NLP tasks, mostly likely including paraphrasing. By contrast, Separator uses fewer than 70 million parameters (1% of Mistral 7B) and can be trained in less than 24 hours on a single consumer-grade 12GB GPU.

Human Evaluation In addition to automatic evaluation we elicited judgements from crowdworkers on Amazon Mechanical Turk (AMT). Specifically, they were shown a sentence and two paraphrases thereof (corresponding to different systems) and asked to select which one was preferred along three dimensions: the *dissimilarity* of the paraphrase compared to the original sentence, how well the paraphrase reflected the *meaning* of the original, and the *fluency* of the paraphrase. Annotators were asked to rate the outputs with the following instructions:

- **Dissimilarity** — Does the rewritten version use different words or phrasing to the original? You should choose the system that uses the most different words or word order.

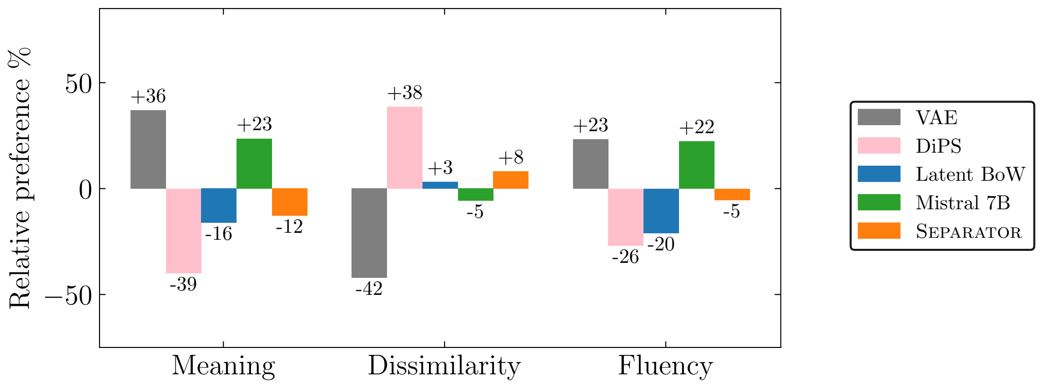


Figure 3.5: Results of our human evaluation. Although the VAE baseline is the best at preserving meaning, it is the worst at introducing variation to the output. SEPARATOR offers the best balance between dissimilarity and meaning preservation, and is more fluent than both DiPS and Latent BoW.

- **Meaning** — To what extent is the meaning expressed in the original question preserved in the rewritten version, with no additional information added? Which of the questions generated by a system is likely to have the same answer as the original?
- **Fluency** — Which system output is the most fluent and grammatical?

A screenshot of the annotation interface is shown in Appendix A.3.

We evaluated a total of 200 questions sampled equally at random from both Paralex and QQP, and collected 3 ratings for each sample. Following best-worst scaling (Louviere and Woodworth, 1990), we assigned each system a score of $+1$ when it was selected, -1 when the other system was selected, and took the mean over all samples. Negative scores indicate that a system was selected less often than an alternative. We chose the four best performing models according to Tables 3.7 and 3.8 for our evaluation: SEPARATOR, DiPS (Kumar et al., 2019), Latent BoW (Fu et al., 2019) and VAE.

We note that since this study, some doubt has been cast within the community on the quality of studies performed on AMT, and the human evaluations in Chapters 6 and 7 use Prolific for participant recruitment instead. However, a reproduction study of our claims found that they were reproducible (Watson and Gkatzia, 2024).

Figure 3.5 shows that although the VAE baseline is the best at preserving meaning, it is also the worst at introducing variation to the output. SEPARATOR introduces more variation than the other systems evaluated and better preserves the original sentence meaning, as well as generating significantly more fluent output (using a one-way ANOVA with post-hoc Tukey HSD test, $p<0.05$).

3.5.3 Analysis

Head Specialisation When predicting latent codes at test time, we assume that the code for each head may be predicted independently of the others, as working with the full joint distribution $p(q_1, \ldots, q_H)$ would be intractable. We now examine whether different heads within the syntactic encoding represent distinct syntactic properties. We focus on question paraphrases, since the space of possible syntactic forms is comparatively constrained, making it easier to identify possible syntactic properties that could be encoded.

First, we define four syntactic properties $f_1, \ldots, f_4$ that questions may exhibit: which wh- word is used for the question (who, what, when etc.); whether the question word is fronted ('what does a moose eat?' vs 'a moose eats what?'); the length in words of the question; and, whether the question contains a prepositional phrase. This is clearly not an exhaustive set of properties, but nonetheless allows some insight into what each head might encode. Following Angelidis et al. (2021), we compute the probability of a question property f_i taking a particular value a, conditioned by head h and quantised code k_h as

$$P(f_i|h, k_h) = \frac{\sum_{x \in \mathcal{X}} \mathbb{I}(q_h(x) = k_h)\mathbb{I}(f_i(x) = a)}{\sum_{x \in \mathcal{X}} \mathbb{I}(q_h(x) = k_h)}, \tag{3.16}$$

where $\mathbb{I}(\cdot)$ is the indicator function, and examples of values a are shown in Figure 3.6. We then calculate the mean entropy of these distributions, to determine how property-specific each head is:

$$\mathcal{H}_h = \frac{1}{K} \sum_{k_h} \sum_{a} P(a|h, k_h) \log P(a|h, k_h). \tag{3.17}$$

Heads with lower entropies are more predictive of a property, indicating specialisation. Figure 3.6 shows our analysis for four syntactic properties: head #2 has learned to control the high level output structure, including the question type or *wh- word*, and whether the question word appears at the beginning or end of the question. Head #3 controls which type of prepositional phrase is used. The length of the output is not determined by any one head, implying that it results from other properties of the surface form.

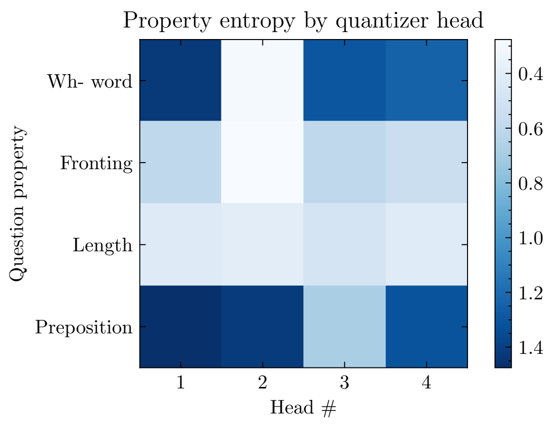


Figure 3.6: Predictive entropy by head for various question properties - lower entropy (indicated by lighter colour) indicates higher predictive power. The results show that SEPARATOR has some degree of interpretability. Head #2 has learned to control the high level output structure, including the question type or *wh- word*, and whether the question word appears at the beginning or end of the question. Head #3 controls which type of prepositional phrase is used.

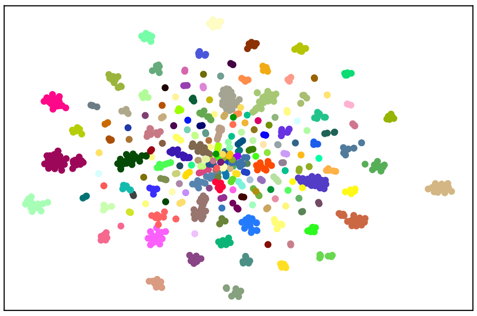
In summary, we find that SEPARATOR successfully learns separate encodings for meaning and form. SEPARATOR is able to generate question paraphrases with a better balance of diversity and meaning preservation compared to prior work. Although we are able to identify some high-level properties encoded by each of the syntactic latent variables, further work is needed to learn interpretable and disentangled syntactic encodings.

Structure of the Encoding Space It is reasonable to ask whether the code prediction network is required, and whether it might instead be possible to simply perturb the continuous syntactic encoding $\mathbf{z}_{syn}$ to vary the output surface form. Figure 3.7 shows that the semantic encodings $\mathbf{z}_{sem}$ are tightly clustered by paraphrase, but syntactic encodings $\mathbf{z}_{syn}$ are much less clearly ordered, and the set of valid syntactic forms for each semantic cluster overlaps significantly. In other words, regions of licensed templates for each input are not contiguous, and naively perturbing a syntactic encoding for an input sentence is not guaranteed to lead to a valid template. Template dropout seems to improve the arrangement of encoding space, but is not sufficient to allow us to 'navigate' encoding space directly.

Qualitative Analysis We give some example output in Table 3.10; while the other systems mostly introduce lexical variation, SEPARATOR is able to produce output with

Input	What is the most known singer?
VAE	What is the most known singer?
DiPS	What was the most known famous singer?
SOW/REAP	What is the most famous singer?
Latent BoW	What is the most famous singer?
Mistral 7B	Which singer is the most renowned or famous?
SEPARATOR	Who is the most famous singer in America?
Input	What is the income for a soccer player?
VAE	What is the salary for a soccer player?
DiPS	What is the median income in soccer?
SOW/REAP	What is US cer?
Latent BoW	What is the salary of a soccer [UNK]?
Mistral 7B	What is a soccer player's earnings?
SEPARATOR	How much is a soccer players' salary?
Input	What has been the economic impact from Brexit referendum so far?
VAE	What has been the economic impact of Brexit referendum so far?
DiPS	What will be a impact of Brexit referendum?
SOW/REAP	How do I spend my virginity?
Latent BoW	How did Brexit referendum impact the Brexit referendum?
Mistral 7B	How has the Brexit vote affected the economy up to this point?
SEPARATOR	How much will the Brexit referendum cost?
Input	What are the basics I should know before learning Hadoop?
VAE	What are the basics should I know before learning Hadoop?
DiPS	How do I know before I want to learn Hadoop?
SOW/REAP	How can I know before learning Hadoop?
Latent BoW	What are the basics of learning Hadoop?
Mistral 7B	Before starting to learn Hadoop, what are the essential concepts I need to be familiar with?
SEPARATOR	How much should I know before learning Hadoop?

Table 3.10: Examples of output generated by various approaches for a given input, from Paralex and QQP. SEPARATOR is able to generate semantically equivalent questions with a different syntactic form to the input.


(a) Semantic encodings

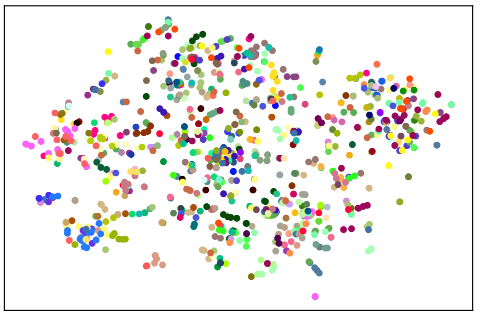

(b) Syntactic encodings

Figure 3.7: Visualisations of $\mathbf{z}_{sem}$ and $\mathbf{z}_{syn}$ using t-SNE (van der Maaten and Hinton, 2008), coloured by paraphrase cluster. The semantic encodings are clustered by meaning, as expected, but there is little to no local ordering in the syntactic space; valid surface forms of a particular sentence do not necessarily have syntactic encodings near to each other.

markedly different syntactic structure to the input, and can even change the question type while successfully preserving the original meaning. The LLM, Mistral 7B, is also able to succesfully generate questions that preserve the original meaning with a different surface form.

A downside of our approach is the use of an information bottleneck; the model must learn to compress a full sentence into a single, fixed-length vector. This can lead to loss of information or corruption, with the output occasionally repeating words or generating a number that is slightly different to the correct one, as shown in Table 3.11. We also occasionally observe instances of the well documented posterior collapse phenomenon, where the decoder ignores the input encoding and generates a generic high probability sequence.

3.6 Summary

In this chapter, we presented SEPARATOR, a method for generating paraphrases that balances high variation in surface form with strong meaning preservation.

SEPARATOR acts as the first piece of evidence to support Hypothesis I, that "weakly structured representations are beneficial for text-to-text generation". By selecting a representation structure that is a good fit for the task – specifically, using separate encoding space for the semantics and syntax of a sentence, and using a discrete encoding for the syntactic space – we are able to generate paraphrases with a better balance between

	Numerical error
Input	Replace starter on a 1988 Ford via?
Output	How do you replace a starter on a 1992 Ford?
	Repetition
Input	What brought about the organization of the Republican political party?
Output	What is the political party of the Republican party?
	Ignoring encoding
Input	What do Hondurans do for a living?
Output	What do Hondurans eat?

Table 3.11: Examples of failure modes. The compressed bottleneck means that SEPARATOR can sometimes hallucinate, generating incorrect numbers. It may also exhibit posterior collapse, ignoring the input encoding.

semantic consistency and syntactic diversity compared to prior work. We hypothesised that discrete representations can be used to make some text-to-text problems feasible (Hypothesis II); choosing a discrete representation for the syntactic encoding allowed us to easily predict licensed surface forms during inference, by reducing it to a classification problem. Finally, we showed that the denoising autoencoder objective (training the model to reconstruct a paraphrase from two inputs that each contain part of the requisite information) is required to assign meaning to the structure (Hypothesis III), encoding semantic and syntactic information in distinct subspaces.

However, the discrete bottleneck used in SEPARATOR has a major drawback: the joint distribution over codes $p(q_1, \ldots, q_H)$ does not conveniently factorise, making it difficult to predict licensed syntactic structures during inference. This lack of known dependency structure also means that the codes q_h are not guaranteed to be disentangled, making the model brittle to template prediction errors during inference. In the next chapter, we propose a novel variant of VQ-VAE that goes some way towards addressing these drawbacks.

Chapter 4

Hierarchical Residual Quantisation

In the previous chapter we showed that using a discrete latent variable, learned with Vector Quantisation, to encode the syntax of natural language sentences allowed us to predict alternative (but still valid) syntactic forms to use for an output paraphrase. However, the statistical independences between the resulting codes $q_{1:H}$ are unknown, leading to a joint probability $p(q_{1:H})$ that does not neatly factorise. This makes it difficult to predict codes that correspond to licensed syntactic forms at inference time, and makes the model somewhat brittle to prediction errors, since high-level and granular details are mixed together.

To address these shortcomings, we propose *Hierarchical Residual Quantisation* (HRQ-VAE), a method for learning discrete representations that uses a known autoregressive factorisation of the joint distribution,

$$p(q_{1:D}) = p(q_1) \times \prod_{d=2}^{D} p(q_d | q_{<d}), \tag{4.1}$$

where d indexes the depth in the hierarchy, and is analogous to the head index h. In other words, the value of the codes at each level are dependent on all previous levels.

In this chapter, we will describe the theoretical foundations of the approach, followed by practical training details. We report validation experiments on MNIST (Lecun et al., 1998), a dataset of handwritten digit images whose properties and true generating process are comparatively well understood, showing that HRQ-VAE learns more informative representations than VQ-VAE.

4.1 Related Work

Vector Quantisation (VQ), the process of mapping a dense vector encoding to one or more discrete *codes*, has a rich history in the field of signal processing (Gray, 1984; Gersho and Gray, 1991; Vasuki and Vanathi, 2006). VQ has been used extensively for applications ranging from speech processing (Wong et al., 1981) to approximate nearest-neighbours search in information retrieval (Jégou et al., 2011). Residual Vector Quantisation (RVQ, Juang and Gray, 1982) extends VQ by applying it recursively to the residual errors between encodings and quantised approximations, effectively decomposing a dense vector into a sum of fixed embeddings.

With the advent of neural networks, VQ-VAE (van den Oord et al., 2017) combines the discrete codes of VQ with the end-to-end training of neural models, learning the codebook simultaneously with an encoder and decoder. Quantisation is, in general, not smoothly differentiable, and so VQ-VAE uses the straight-through estimator to allow gradient to flow to the encoder. As discused in Section 2.4, the authors include additional terms in the training objective (the commitment and quantisation loss) as well as using a EMA update for the codebook (see van den Oord et al. (2017, Appendix A)). Sønderby et al. (2017) propose replacing the straight-through estimator with the Gumbel softmax reparameterisation trick, allowing gradient to flow from the decoder to both the encoder and codebook, resulting in more stable training and improved reconstruction error. SQ-VAE (Takida et al., 2022) generalise this approach, modelling the dequantization process as sampling from either a Gaussian or von Mises-Fisher distribution, with improved performance on vision and speech tasks.

There has been some prior work on learning *hierarchical* discrete representations of input data. VQ-VAE 2 (Razavi et al., 2019) and Huang et al. (2023) extend VQ-VAE to represent images as a hierarchy of codes. However, in that context the hierarchy refers to a 'stacked' architecture, where the output of one variational layer is passed through a CNN and then another variational layer that can be continuous (Vahdat and Kautz, 2020) or quantised (Williams et al., 2020; Liévin et al., 2019; Willetts et al., 2021). Unlike these approaches, we induce a *single* latent space that has hierarchical properties.

RQ-VAE (Lee et al., 2022) propose decomposing a dense vector encoding into a sequence of codes representing iterative refinements on the approximation. However, their method uses a single shared codebook at all levels, so the codes are not hierarchically ordered and the resulting representations do not have a coarse-to-fine structure. This property will prove to be crucial for the models proposed in Chapters 5 to 7. Concurrent with our work, SoundStream (Zeghidour et al., 2022) propose learning a discrete hierarchical representation using *different* codebooks, and is closely related to our method (and concurrently proposed depth dropout as a technique for ensuring intermediate nodes in the hierarchy are still meaningful). However, all these prior methods use the complex and unstable straight-through estimator with EMA updates during training; we describe an alternative approach in this chapter based on the Gumbel reparameterisation (Jang et al., 2017; Maddison et al., 2017).

Beyond quantisation techniques, there has been work on other approaches to learning ordered or structured representations. Vendrov et al. (2016) propose learning an embedding space where the ordering of two pairs of samples could be inferred from their relative positions, but their method requires supervision of the correct ordering. Opper et al. (2023) describe a method for learning embeddings that explicitly include structural information. However, they focus on learning representations of known structures, rather than on learning an ordering within the embedding space itself. Li et al. (2023) learn a tree-based index for passage retrieval concurrently with the dense embedding space, showing that this leads to improved retrieval performance. However, their method does not learn a single embedding space that is hierarchically structured.

A separate line of work has looked at using the properties of hyperbolic geometry to encourage autoencoders to learn hierarchical representations. Mathieu et al. (2019) show that a model endowed with a Poincaré ball geometry is able to recover hierarchical structure in datasets, and Surís et al. (2021) use this property to deal with uncertainty in predicting events in video clips. However, their work is limited to continuous encoding spaces.

4.2 Generative Model

Let $\mathbf{x}$ be a sample from data, e.g. a sequence of tokens or an image. We assume that the information contained in $\mathbf{x}$ may be encoded as a sequence $q_{1:D}$ of discrete latent variables or *codes* $q_d \in \{1, \ldots, K\}$, with $d \in \{1, \ldots, D\}$. Further, we assume that the $q_{1:D}$ should be ordered *hierarchically*, such that q_1 represents high level information about the input (e.g., the topic or digit label) whereas q_D represents fine-grained information (e.g., the specific phrasing or image rotation). The codes $q_{1:D}$ can be viewed as a single *path* through a hierarchy or tree. The structure of the tree is fixed (it has depth D and branching factor K) but the mapping from data $\mathbf{x}$ to paths $q_{1:D}$ (and vice versa),

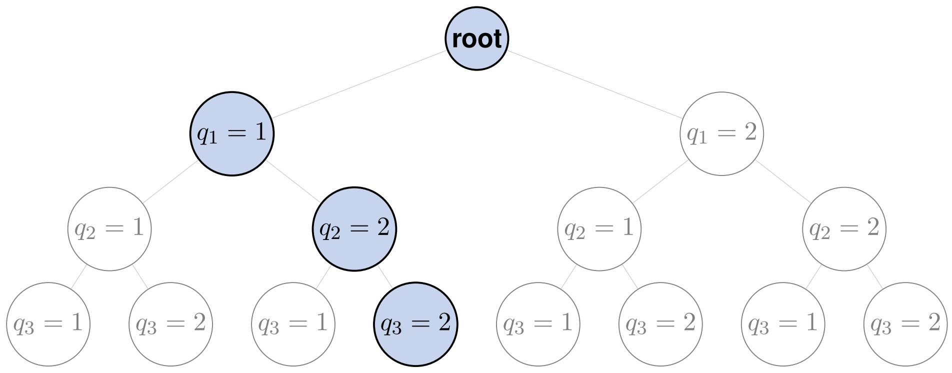


Figure 4.1: An idealised depiction of a hierarchy with $D = 3$ and $K = 2$. The path corresponding to codes $q_{1:D} = \{1, 2, 2\}$ is highlighted.

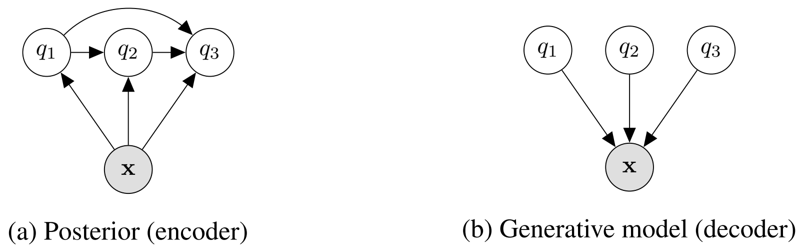


Figure 4.2: Proposed generative model for encoding input samples as paths through a hierarchy.

and therefore the meaning of each node in the tree, are learned. Figure 4.1 depicts an example of such a tree with $D = 3$ and $K = 2$, highlighting the path $q_{1:D} = \{1, 2, 2\}$. The hierarchy is induced by the dependence relations on the encoder side, i.e., given $\mathbf{x}$ we assume that the q_d are dependent on both $\mathbf{x}$ and $q_{<d}$. For the decoder, we assume that the q_d are independent of each other.

This corresponds to the generative model shown in Figure 4.2. It leads to the factorisation

$$p(\mathbf{x}) = \sum_{q_{1:D}} p(\mathbf{x}|q_{1:D}) \times \prod_{d=1}^{D} p(q_d), \tag{4.2}$$

while the posterior factorises as

$$\phi(q_{1:D}|\mathbf{x}) = \phi(q_1|\mathbf{x}) \times \prod_{d=2}^{D} \phi(q_d|q_{<d}, \mathbf{x}). \tag{4.3}$$

Recall from Chapter 2 that during training we optimise the Evidence Lower-Bound (ELBO), defined as the divergence between the true generative distribution and the expectation under the approximate posterior,

$$\text{ELBO} = \mathbb{E}_{q_{1:D} \sim \phi}\left[\log \frac{p(\mathbf{x}|q_{1:D})p(q_{1:D})}{\phi(q_{1:D}|\mathbf{x})} \right] \tag{4.4}$$

$$= \mathbb{E}_{q_{1:D} \sim \phi}\Big[\log p(\mathbf{x}|q_{1:D}) + \log \frac{p(q_{1:D})}{\phi(q_{1:D}|\mathbf{x})} \Big] \tag{4.5}$$

$$\tag{4.6}$$

This leads to the final training objective

$$\mathcal{L} = \mathbb{E}_{q_{1:D} \sim \phi}\big[\log p(\mathbf{x}|q_{1:D}) \big] - \beta_{KL} \sum_{d=1}^{D} \text{KL}\big[\phi(q_d|q_{<d}, \mathbf{x}) \parallel p(q_d) \big], \tag{4.7}$$

where $q_d \sim \phi(q_d|q_{<d}, \mathbf{x})$, $\text{KL}\big[\cdot \parallel \cdot\big]$ is the KL divergence and β_{KL} determines the weight of the KL term. Intuitively, this objective jointly maximises the likelihood of generating real data $\mathbf{x}$ from samples of $q_{1:D}$ while also ensuring the approximate posterior $\phi_\theta(q_{1:D}|\mathbf{x})$ stays close to the prior $p_\theta(q_{1:D})$. We choose a uniform prior over $p(q_d)$ for simplicity – future work could consider using a learned prior.

4.3 Neural Parameterisation

Let $\mathbf{z} \in \mathcal{R}^{\mathbb{D}}$ be the (deterministic) output of the encoder network $\mathbf{z} = \textsc{Encoder}(\mathbf{x})$, that we wish to decompose as a sequence of discrete hierarchical codes. Let $q_d \in \{1, \dots, K\}$ be discrete latent variables corresponding to the *codes* at different levels in the hierarchy, $d \in \{1, \dots, D\}$. These levels are comparable to the different channels in VQ-VAE (van den Oord et al., 2017) or multiple codebooks in QT (Angelidis et al., 2021), but each level is now dependent on the value of the previous level. Each level uses a distinct codebook, $\mathbf{C}_d \in \mathbb{R}^{K \times \mathbb{D}}$, which maps each discrete code to a continuous embedding $\mathbf{C}_d(q_d) \in \mathbb{R}^{\mathbb{D}}$.

Since the $q_{1:D}$ are intended to represent hierarchical information, the posterior distribution over codes at each level $\phi(q_d|q_{<d}, \mathbf{z})$ is parameterised by a softmax distribution, with the scores s_d given by the L2 distance from each of the codebook embeddings to the residual error between the input and the cumulative embedding from all previous levels,

$$s_d(q_d) = -\left(\left[\mathbf{z} - \sum_{d'=1}^{d-1} \mathbf{C}_{d'}(q_{d'}) \right] - \mathbf{C}_d(q_d) \right)^2. \tag{4.8}$$

During inference, we set $q_d = \arg\max(s_d)$.

$$q_d = \arg\min(s_d) \tag{4.9}$$

Illustrated in Figure 4.3, the embeddings at each level can be viewed as refinements of the (cumulative) embedding so far, or alternatively as selecting the centroid of a subcluster within the current cluster. The posterior network $\phi(q_d|q_{<d}, \mathbf{z})$ iteratively *decomposes* an encoding vector into a path through a hierarchy of clusters whose centroids are the codebook embeddings.

Importantly, it is not necessary to specify a path to the complete depth D; a *subpath* $q_{1:d}$ ($d < D$) still results in a valid embedding $\mathbf{z}$. We can therefore control the specificity of an encoding by varying its depth.

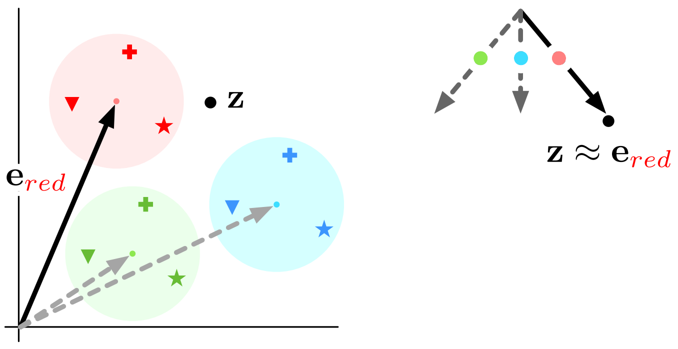
Given a sequence of discrete codes $q_{1:D}$, we deterministically construct its continuous representation with the composition function $f_{q\rightarrow\mathbf{z}}(\cdot)$,

$$\mathbf{z} = f_{q\rightarrow\mathbf{z}}(q_{1:D}) = \sum_{d=1}^{D} \mathbf{C}_d(q_d). \tag{4.10}$$

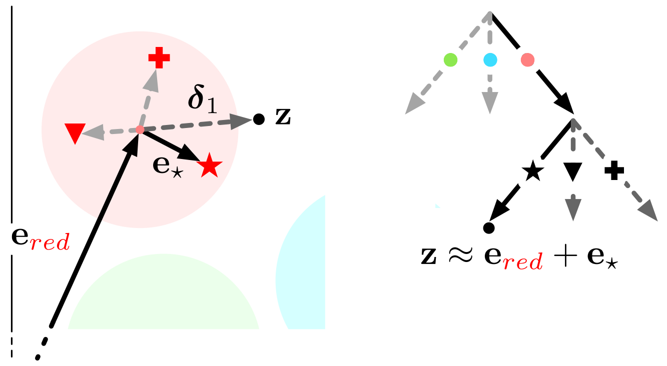
HRQ-VAE can be viewed as an extension of VQ-VAE (van den Oord et al., 2017), with two significant differences: (1) the codes are hierarchically ordered and the joint distribution $p(q_1, \ldots, q_D)$ admits an autoregressive factorisation; and (2) the HRQ-VAE composition function is a sum, compared to concatenation in VQ or a complex neural network in VQ-VAE 2 (Razavi et al., 2019). Under HRQ, latent codes describe a path through the learned hierarchy within a shared encoding space. The form of the posterior $\phi(q_d|q_{<d}, \mathbf{z})$ and the composition function $f_{q\rightarrow\mathbf{z}}(\cdot)$ could be applied to any encoding space; it is a general technique non-specific to any particular task. Additionally, HRQ-VAE does not make any assumptions about the particular encoder or decoder architecture, or about the data it models; any architecture that maps a data sample $\mathbf{x}$ to a dense vector encoding $\mathbf{z}$ (and vice-versa) may be used.

4.4 Training Techniques

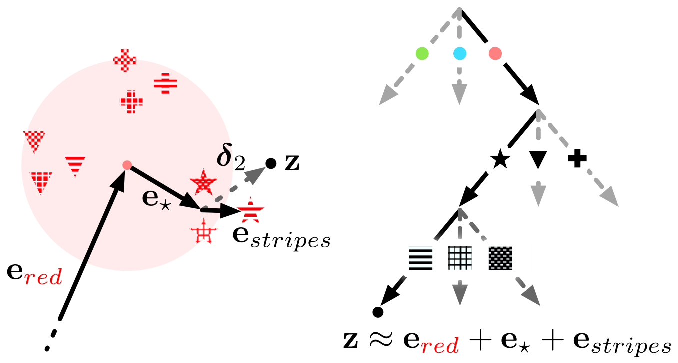
Gradient estimator VQ-VAE (van den Oord et al., 2017) uses the straight-through estimator to update the encoder during training, and requires additional loss terms to update the codebook. Since this training process is often unstable, an EMA update scheme is generally used to update the codebook. This approach has two major drawbacks: it requires significant additional implementation complexity; and, the encoder and



(a) HRQ-VAE compares the input to a jointly learned codebook of embeddings that become increasingly granular at lower depths of hierarchy. In this simplified example, with a depth of 3 and a codebook size of 3, the nearest top-level (colours) embedding to an input vector $\mathbf{z}$ is $\mathbf{e}_{red}$



(b) Then, the residual error $\boldsymbol{\delta}_1 = \mathbf{z} - \mathbf{e}_{red}$ is compared to the 2nd level of embeddings (shapes), with the nearest being $\mathbf{e}_\star$



(c) Finally, the residual error $\boldsymbol{\delta}_2$ is compared to the 3rd level codebook (patterns), where the closest is $\mathbf{e}_{stripes}$. The quantised encoding of $\mathbf{z}$ is then $\mathbf{z} \approx \mathbf{e}_{red} + \mathbf{e}_\star + \mathbf{e}_{stripes}$.

Figure 4.3: An illustration of how HRQ-VAE maps an input encoding vector $\mathbf{z}$ to a decomposition of hierarchical discretised encodings. HRQ-VAE compares the input to a jointly learned codebook of embeddings that become increasingly granular at lower depths of hierarchy.

codebook are not updated jointly. The straight-through estimator updates the encoder according to gradient from the decoder only, and the codebook is then adjusted to fit the encoder. As we will show in Section 4.5, this results in a suboptimal codebook.

Instead, we use the Gumbel reparameterisation (Jang et al., 2017; Maddison et al., 2017) to sample from the distribution over $q_{1:D}$ during training, as proposed by Sønderby et al. (2017). This allows gradient to flow to the encoder *and* the codebook from the reconstruction loss, resulting in more stable training and a better learned codebook.

Initialisation Smaller perturbations in encoding space should result in more fine grained changes in the information they encode. Therefore, we encourage *ordering* between the levels of hierarchy (such that lower levels encode finer grained information) by initialising the codebook with a decaying scale, such that later embeddings have a smaller norm than those higher in the hierarchy. Specifically, the norm of the embeddings at level d is weighted by a factor $(\alpha_{init})^{d-1}$.

Inspired by the findings from Łańcucki et al. (2020), we initialise the codebook on a hypersphere by normalising the magnitude of each embedding. The embeddings should have a smaller magnitude than the input encodings, to encourage the model to use the full set of codes, and to avoid the radial distance component dominating the angular component. We found that the model is highly sensitive to the initialisation of the codebook; the initial codes should be located in roughly the same region of space as the output of the encoder, but should have sufficient variation so as to be informative for the decoder.

Depth Dropout To encourage the hierarchy within the encoding space to correspond to hierarchical properties of the output, we introduce *depth dropout*, whereby the hierarchy is truncated at each level during training with some probability p_{depth}. The output of the quantiser is then given by

$$\mathbf{z} = \sum_{d=1}^{D} \left(\mathbf{C}_d(q_d) \prod_{d'=1}^{d} \gamma_{d'} \right), \tag{4.11}$$

where $\gamma_h \sim \text{Bernoulli}(1 - p_{depth})$. This means that the model is sometimes trained to reconstruct the output based only on a *partial* encoding of the input, and should learn to cluster similar outputs together at each level in the hierarchy.

Norm Loss For some applications (e.g., opinion summarisation, Chapters 6 and 7), it is particularly important that the deeper levels in the hierarchy correspond to more fine-grained embeddings, to avoid the possibility of clusters overlapping. We therefore include an additional loss $\mathcal{L}_{NL}$ to encourage deeper embeddings to remain fine-grained during training,

$$\mathcal{L}_{NL} = \frac{\beta_{NL}}{D} \sum_{d=2}^{D} \big[\max \big(\gamma_{NL} \frac{||\mathbf{C}_d||_2}{||\mathbf{C}_{d-1}||_2}, 1 \big) - 1 \big]^2,$$

where γ_{NL} determines the relative scale between levels and β_{NL} controls the strength of the loss. This ensures that the embeddings $\mathbf{C}_d$ are smaller than those at higher levels $\mathbf{C}_{<d}$, since this is otherwise not guaranteed. HRQ-VAE uses a very narrow bottleneck, and we found that models may exploit the available capacity of lower codebooks to improve the overall expressivity of the model, at the cost of the quality of the learned hierarchy.

These additional training techniques ensure that the representations learned are hierarchical both in the shared embedding space and in the information they represent. However, a structured representation space is not sufficient; the model must in general also be trained to assign *meaning* to each level of the hierarchy. In future chapters, we show how distant supervision may be used to learn a meaningful hierarchy.

4.5 Validation Experiments

We now experimentally validate that HRQ-VAE is able to learn useful representations and compare it to VQ-VAE. Note that our goal is not to compete with prior work in terms of compression ratios or reconstruction error, and the experiments in this section are intended only to show that HRQ-VAE is a comparable technique. Other approaches either employ a straight-through estimator, which we found to be unstable and makes hyperparameter tuning challenging; or, they lack crucial properties (e.g., a shared representation space or hierarchically ordered codes) that are crucial for the techniques proposed in Chapters 5 to 7. We therefore limit our comparisons to the method used in Chapter 3, VQ-VAE.

Experimental Setup We implement HRQ-VAE in Pythae (Chadebec et al., 2022),[1] a library comprising reference implementations of a wide range of Variational Autoencoder methods, with shared encoder/decoder implementations allowing for direct comparison of the different choices of bottleneck. We perform experiments on MNIST

[1] `https://github.com/clementchadebec/benchmark_VAE`

(Lecun et al., 1998), a dataset of images of handwritten digits. While MNIST is no longer considered a challenging task, it has a well understood generating distribution (e.g., the true number of digits is known) and therefore allows us to validate how HRQ-VAE behaves with a minimum of possible confounding factors. Additionally, the comparatively small size of the dataset allows for rapid and efficient experimentation.

We train a small autoencoder network based on a 7-layer ResNet encoder and decoder (He et al., 2016) with a single latent vector $\mathbf{z} \in \mathbb{R}^{16}$, using the squared reconstruction error of the image as the training objective. No data augmentation techniques were used. We set the codebook size $K = 10$, corresponding to the 10 possible digits, with $D = 3$. Recall that each level in the hierarchy has an associated codebook of size K, leading to a total of $K \times D$ embeddings for HRQ-VAE. Although the code at a given level q_d is dependent on the previous codes $q_{<d}$, its embedding $\mathbf{C}_d(q_d)$ is independent. We compare to VQ-VAE with an equivalent total number of embeddings ($K = 30$), as well as with $K = 10$ and $K = 128$. We also compare to a non-hierarchical version of HRQ-VAE (i.e., $D = 1$) to compare training a VQ model using the straight-through estimator against sampling using Gumbel softmax. For comparison, we include the reconstruction errors for a *continuous* autoencoder (AE) and variational autoencoder (VAE) trained using the same hyperparameters.

Reconstruction Error and Stability We begin by confirming that HRQ-VAE can be trained in a stable manner, and is able to successfully encode the content of the input images. Table 4.1 shows the reconstruction error and standard deviations across five different random seeds, for a range of bottleneck configurations.

HRQ-VAE with $K = 10$ codes and $D = 3$ levels of depth leads to comparable reconstruction error as a VQ-VAE with equivalent capacity ($K = 30$), with both setups achieving low variation across random seeds. The VQ-VAE without EMA updates results in much worse reconstruction error, indicating the importance of this component. HRQ-VAE with a single level of depth achieves slightly worse error than the equivalent VQ-VAE model, but still outperforms VQ-VAE without EMA updates. We additionally present some examples of output images from both VQ-VAE and HRQ-VAE in Figure 4.4, which clearly shows that HRQ-VAE is able to generate higher quality reconstructions with fewer total parameters. Overall, these results indicate that HRQ-VAE is able to learn informative representations, and trains in a stable manner. HRQ-VAE is significantly more straightforward to implement compared to VQ-VAE with EMA updates, and we suggest that HRQ-VAE with $D =$

	Model	**Variant**	D	K	**Err.** $\downarrow$	**Std.** $\downarrow$
K=30	VQ-VAE		–	30	34.30	0.28
	HRQ-VAE		3	10	**27.65**	**0.10**
	VQ-VAE	(No EMA)	–	30	51.13	1.52
K=10	VQ-VAE		–	10	**40.48**	**0.37**
	HRQ-VAE		1	10	41.09	0.54
	VQ-VAE	(No EMA)	–	10	52.07	1.37
K=128	VQ-VAE		–	128	**29.92**	0.67
	HRQ-VAE		1	128	31.30	**0.14**
	VQ-VAE	(No EMA)	–	128	43.73	8.68
Continuous	VAE		–	–	22.13	0.12
	AE		–	–	6.90	0.06

Table 4.1: Reconstruction errors (Err.) and standard deviations across 5 random seeds (Std.) for a range of configurations of VQ-VAE and HRQ-VAE, as well as continuous autoencoders (AE/VAE). HRQ-VAE with 3 levels achieves better reconstruction errors than a VQ-VAE model with equivalent capacity, and trains with comparable stability.

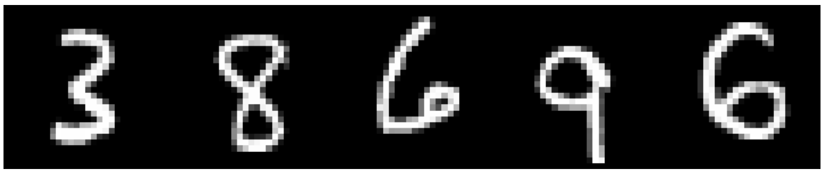
1 offers a compelling alternative method to learn quantised latent representations. The reconstruction errors for the continuous bottlenecks are nonetheless much lower, indicating that any improvement in interpretability does come at a cost of expressivity.

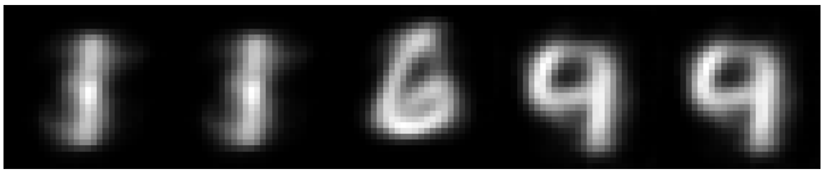
Representation Quality Next, we evaluate whether the representations learned by HRQ-VAE are *interpretable*. Although the model is trained to reconstruct the images in an unsupervised manner, the true digit labels are known. We can therefore compare the learned codes or *clusters* to the label and evaluate whether the model has learned a useful clustering of the input space.

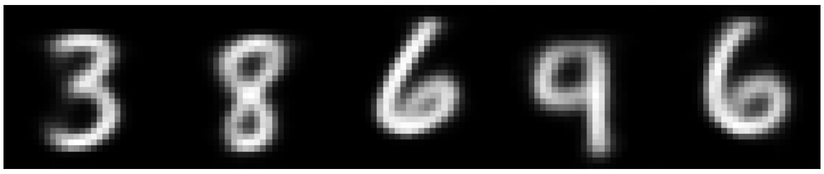
Figure 4.5 shows confusion matrices for VQ-VAE and HRQ-VAE models, with the value of the latent code q_1 against the true digit label. We also include two baselines, where k-means clustering (Lloyd, 1982; Pedregosa et al., 2011) with $k = 10$ has been applied to the embeddings from a vanilla autoencoder (AE) and a Variational Autoencoder (VAE), and the 'codes' correspond to the assigned cluster index. The codes (i.e., the rows of the plot) have been reordered using a greedy heuristic to maximise the weight along the diagonal. While VQ-VAE has learned to group together some digits, other codes correspond to a wide range of digit labels (e.g., $q_1 = 2$ is not clearly associated with any particular label). By contrast, HRQ-VAE has successfully learned to group together digits with the same label. VQ-VAE must learn to encode all the information about the input image into a single code, but HRQ-VAE encodes high-level information (such as digit label) in the root levels of the hierarchy and low-level information (rotation, scale etc.) in the leaf levels. We also report the squared error between the confusion matrics in Figure 4.5 and the identity matrix (corresponding to a perfect clustering) in Table 4.2. HRQ-VAE achieves the lowest error of all models evaluated, indicating that it learns a more meaningful clustering of the encoding space.

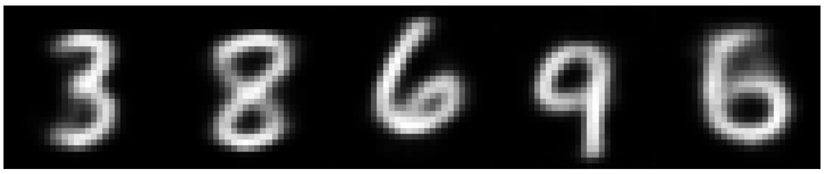
Analysis We conclude by offering some analysis of how HRQ-VAE performs under different hyperparameter settings. Figure 4.6 shows the reconstruction error for a range of different bottleneck capacities for both HRQ-VAE and VQ-VAE. We set $K = 10$ as before for HRQ-VAE, and use an equivalent capacity for VQ-VAE by setting $K = 10 \times D$. Although VQ-VAE achieves a slightly lower reconstruction error for a single level of depth, HRQ-VAE far outperforms it for $D \geq 2$ as the combinatoric nature of HRQ-VAE allows it to represent a larger number of possible outputs.

We also evaluate the importance of the training techniques used for both VQ-VAE and HRQ-VAE, with a set of ablation studies shown in Table 4.3. We set $D = 3$


(a) Inputs


(b) VQ-VAE ($K = 30$)


(c) VQ-VAE ($K = 128$)


(d) HRQ-VAE ($D = 3$, $K = 10$)

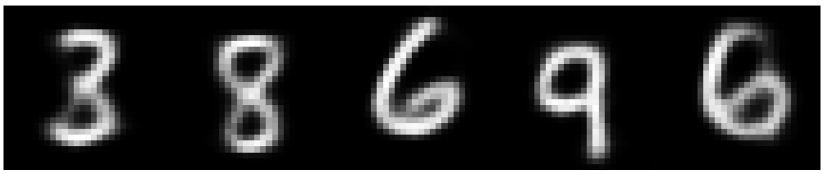
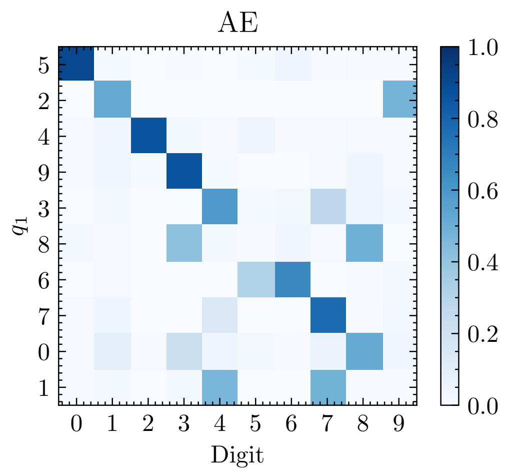
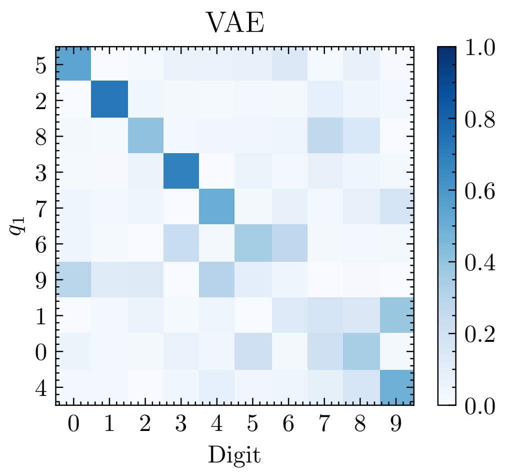

(e) HRQ-VAE ($D = 8$, $K = 10$)

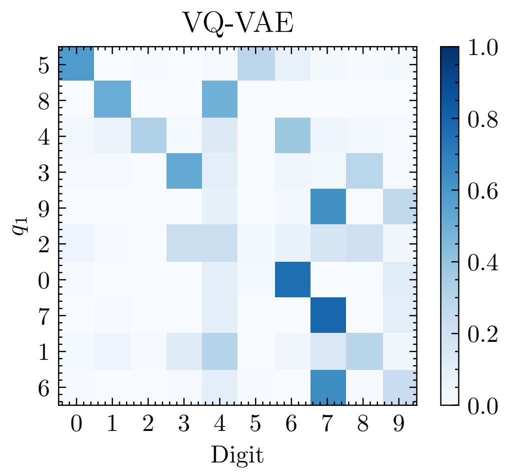
Figure 4.4: Examples of input images and reconstructions from VQ-VAE and HRQ-VAE models with a range of capacities. HRQ-VAE is able to generate higher quality reconstructions with fewer total parameters.



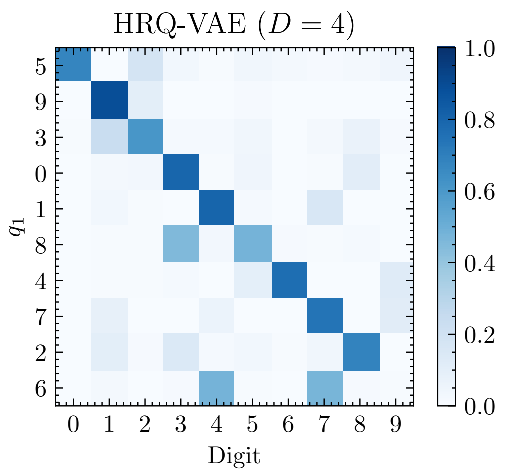
(a) AE + k-means



(b) VAE + k-means



(c) VQ-VAE



(d) HRQ-VAE with $D = 4$

Figure 4.5: Confusion matrices for true digit label compared to latent codes q_1, where dark cells indicate higher cluster membership. The codes have been ordered using a greedy heuristic to be as close as possible to the identity. HRQ-VAE learns an encoding structure that more closely resembles the true digit labels compared to other methods.

Model	**Err.** $\downarrow$
AE + k-means	4.35 $\pm$0.76
VAE + k-means	4.13 $\pm$0.75
VQ-VAE	5.60 $\pm$0.53
HRQ-VAE	**3.30** $\pm$0.36

Table 4.2: Classification errors for unsupervised digit recognition. The error shown is the sum of the squared differences between the confusion matrices in Figure 4.5 and the identity. HRQ-VAE learns a clustering over the inputs that is more meaningful than comparison methods. The differences between HRQ-VAE and VQ-VAE are significant according to a t-test ($p < 0.05$).

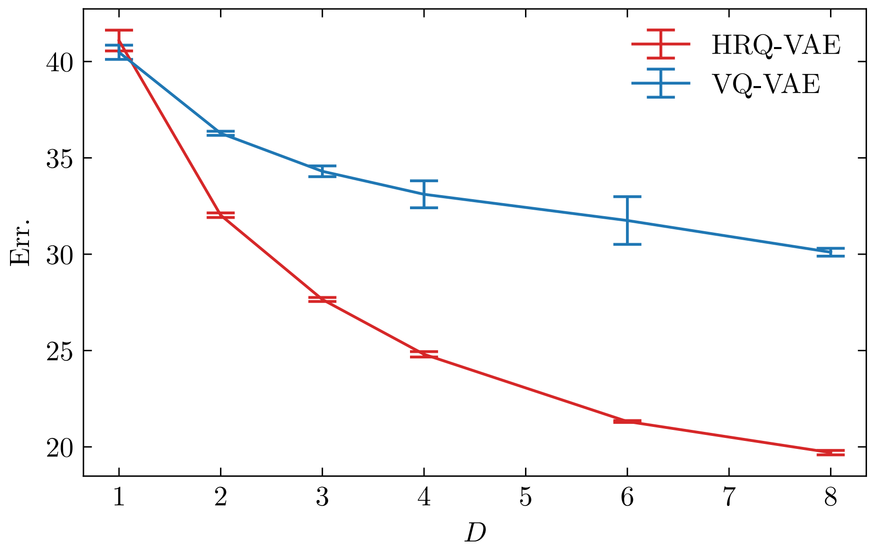


Figure 4.6: Reconstruction error against depth for HRQ-VAE, across 5 random seeds. Lower scores are better. Increasing the depth of the representation generally leads to lower reconstruction error, indicating that HRQ-VAE successfully exploits the additional capacity.

Model	**ΔErr. ↓**	**Std. ↓**
VQ-VAE	34.30	0.28
No EMA	+16.83	1.52
No Commitment Loss	+14.29	0.60
No Quantisation Loss	+0.03	0.11
HRQ-VAE	27.65	0.10
No Norm Loss	-0.08	0.19
No KL	+2.45	+7.62
No Init Decay	-0.56	0.17
No Temp. Scheduling	-0.00	0.12
No Depth Dropout	-4.16	0.15

Table 4.3: Change in reconstruction errors (ΔErr.) and standard deviations across 5 random seeds (Std.) for a range of configurations of VQ-VAE and HRQ-VAE. VQ-VAE performs much worse without the EMA updates or commitment loss, while HRQ-VAE suffers when the KL term is omitted. Norm loss, initialisation decay, temperature scheduling and depth dropout all have a negligible or negative effect on reconstruction error, but we found that they were important to ensure that the learned encodings remained hierarchically ordered.

for HRQ-VAE and $K = 30$ for VQ-VAE. VQ-VAE performs much worse without the EMA updates or commitment loss, while HRQ-VAE suffers when the KL term is omitted. Norm loss, initialisation decay and temperature scheduling all have a negligible effect, which depth dropout has a negative effect on reconstruction error for HRQ-VAE, but we found that they were important to ensure that the learned encodings remained hierarchically ordered.

4.6 Summary

In this chapter, we introduce Hierarchical Residual Quantisation VAEs, a method for representing data as a path through a learned discrete hierarchy. The method extends VQ-VAE by using residual quantisation to iteratively decompose an input encoding into progressively more fine-grained components, and encodes the input as a sequence of embedding indices or *codes*. The model is trained end-to-end, using the Gumbel reparameterisation trick to allow gradient to flow from decoder to encoder while simultaneously updating the embeddings. We perform an intial evaluation on a dataset of handwritten digit images with well-understood properties, MNIST, and show that HRQ-VAE is able to achieve much better reconstruction errors than VQ-VAE with equivalent parameter counts, with comparable training stability. We also demonstrate that HRQ-VAE is able to learn meaningful structure from the data without supervision, clustering digits with the same label under the same top-level code in the hierarchy.

While the small scale and limited complexity of MNIST makes it valuable for rapid experimentation and validation of our method, in the next chapter we combine the separated representation spaces from Chapter 3 with HRQ-VAE, and apply our hierarchical residual quantisation technique to paraphrase generation.

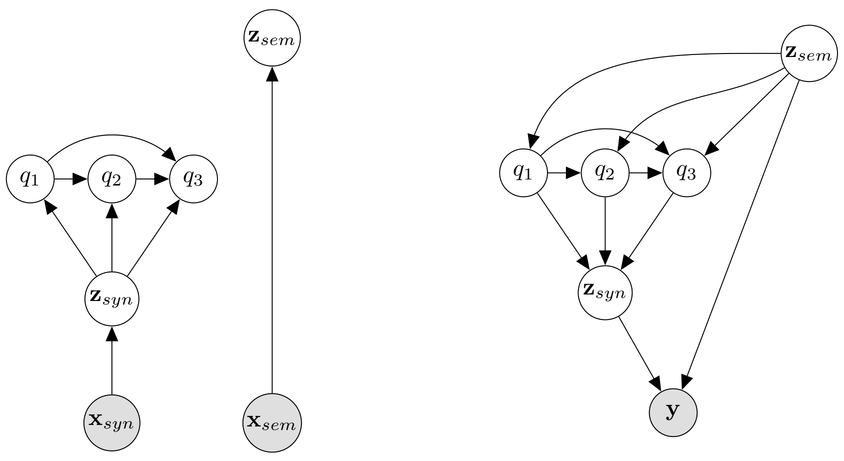
Chapter 5

Hierarchical Syntactic Sketches for Paraphrase Generation

In the previous chapter, we introduced Hierarchical Residual Quantisation (HRQ-VAE), a method for learning discrete hierarchical encodings. We demonstrated that HRQ-VAE is able to learn informative representations for a simple dataset of handwritten digits, and that the learned representations were organised in a meaningful way. HRQ-VAE was motivated by a drawback of the SEPARATOR model introduced in Chapter 3: the discrete syntactic representation used does not have a known decomposition of the joint probability over codes, making inference of licensed templates challenging. We now address this problem, combining the separated semantic and syntactic representation spaces from Chapter 3 with the hierarchical quantisation technique from Chapter 4.

In this chapter, we propose a generative model of paraphrase generation, that encourages syntactic diversity by conditioning on an explicit syntactic sketch. We introduce CALYPSO,[1] a method for generating paraphrases that represents the syntactic structure of sentences as a sequence of discrete latent variables. This hierarchy of codes is learned through end-to-end training, and represents fine-to-coarse grained information about the input. By using HRQ-VAE to represent the syntactic form of an input sentence as a path through the discrete hierarchy, we can learn richer representations of the syntactic structures and more easily predict syntactic sketches at test time, supporting Hypothesis II. Extensive experiments, including a human evaluation, confirm that CALYPSO learns a hierarchical representation of the input space, and generates paraphrases of higher quality than previous systems.

[1]Our model uses sketches; Calypso was an Ancient Greek painter, mentioned in Pliny the Elder's 'Natural History'.



(a) Posterior (encoder)

(b) Generative model (decoder)

Figure 5.1: The generative models underlying our approach. Given some semantic content $\mathbf{z}_{sem}$, we predict a hierarchical set of syntactic codes q_d that describe the output syntactic form at increasing levels granularity. These are combined to give a syntactic embedding $\mathbf{z}_{syn}$, which is fed to the decoder along with the original semantic content to generate the output sentence $\mathbf{y}$. During training, the encoder is driven by a paraphrase $\mathbf{x}_{sem}$ and a syntactic exemplar $\mathbf{x}_{syn}$. Note that $\mathbf{z}_{syn}$ is not a random variable, but is nonetheless shown here to illustrate the encoding and decoding procedure.

5.1 Introduction

Humans use natural language to convey information, mapping an abstract idea to a sentence with a specific surface form. Recall that a paraphrase is an alternative surface form of the same underlying semantic content.

While autoregressive models of language (including paraphrasing systems) predict one token at a time, there is evidence that in humans some degree of planning occurs at a higher level than individual words (Levelt, 1993; Martin et al., 2010). Prior work on paraphrase generation has attempted to include this inductive bias by specifying an alternative surface form as additional model input, either in the form of target parse trees (Iyyer et al., 2018; Chen et al., 2019a; Kumar et al., 2020), exemplars (Meng et al., 2021), or syntactic codes (Shu et al., 2019; Hosking and Lapata, 2021, see Chapter 3). Most of these approaches suffer from an 'all or nothing' problem: the target surface form must be fully specified during inference. However, predicting the complete syntactic structure is almost as difficult as predicting the sentence itself, negating the benefit of the additional planning step.

In this chapter, we propose a generative model for paraphrase generation, that combines the diversity introduced by an explicit syntactic target with the tractability of models trained end-to-end. Shown in Figure 5.1, the model begins by assuming the existence of some semantic content $\mathbf{z}_{sem}$. Conditioned on this semantic information, the model predicts a syntactic 'sketch' in the form of a hierarchical set of discrete codes $q_{1:D}$, that describe the target syntactic structure with increasing granularity. The sketch is combined into an embedding $\mathbf{z}_{syn}$, and fed along with the original meaning $\mathbf{z}_{sem}$ to a decoder that generates the final output utterance $\mathbf{y}$. Choosing a discrete representation for the sketch means it can be predicted from the meaning as a simple classification task, and the hierarchical nature means that the joint probability over the codes admits an autoregressive factorisation, making prediction more feasible.

The separation between $\mathbf{z}_{sem}$ and $\mathbf{z}_{syn}$ is induced by a training scheme introduced in Chapter 3 (Hosking and Lapata, 2021; Huang and Chang, 2021b) and inspired by prior work on separated latent spaces (Chen et al., 2019b; Bao et al., 2019), whereby the model must reconstruct a target output from one input with the correct meaning, and another input with the correct syntactic form. To learn the discretised sketches, we employ HRQ-VAE (introduced in Chapter 4), a variant of Vector-Quantised Variational Autoencoders (VQ-VAE, van den Oord et al., 2017) that learns a *hierarchy of embeddings* within a shared vector space, and represents an input encoding as a path through this hierarchy. HRQ-VAE leads to a decomposition of a dense vector into embeddings of increasing granularity, representing high-level information at the top level before gradually refining the encoding over subsequent levels.

Our contributions are summarised as follows:

- We propose a generative model of natural language generation, CALYPSO, that induces a syntactic sketch to account for the diversity exhibited by paraphrases. We present a parameterisation of our generative model that is a novel method for learning hierarchical discretised embeddings over a single latent encoding space. These embeddings are trained end-to-end and jointly with the encoder/decoder.

- We use CALYPSO to induce hierarchical sketches for paraphrase generation, demonstrating that the known factorisation over codes makes them easier to predict at test time, and leads to higher quality paraphrases.

5.2 Latent Syntactic Sketches

5.2.1 Motivation

Let $\mathbf{y}$ be a sentence, represented as a sequence of tokens. We assume that $\mathbf{y}$ contains semantic content, that can be represented by a latent variable $\mathbf{z}_{sem}$. Types of semantic content might include the description of an image, or a question intent. However, the mapping from semantics to surface form is not unique: in general, there is more than one way to express the semantic content. Sentences with the same underlying meaning $\mathbf{z}_{sem}$ but different surface form $\mathbf{y}$ are *paraphrases*. Standard approaches to paraphrasing (e.g., Bowman et al. 2016) map directly from $\mathbf{z}_{sem}$ to $\mathbf{y}$, and do not account for this diversity of syntactic structure.

Following recent work on syntax-guided paraphrasing (Chen et al., 2019a; Hosking and Lapata, 2021), and inspired by evidence that humans plan out utterances at a higher level than individual words (Martin et al., 2010), we introduce an intermediary *sketching* step, depicted in Figure 5.1b. We assume that the output sentence $\mathbf{y}$ is generated as a function both of the meaning $\mathbf{z}_{sem}$ *and* of a syntactic encoding $\mathbf{z}_{syn}$ that describes the structure of the output. Moreover, since natural language displays hierarchical organization in a wide range of ways, including at a syntactic level (constituents may contain other consituents), we also assume that the syntactic encoding $\mathbf{z}_{syn}$ can be decomposed into a hierarchical set of discrete latent variables $q_{1:D}$, and that these q_d are conditioned on the meaning $\mathbf{z}_{sem}$. This contrasts with popular model architectures such as VAE (Bowman et al., 2015) which use a *flat* internal representation in a dense Euclidean vector space.

Intuitively, our generative model corresponds to a process where a person thinks of a message they wish to convey; then, they decide roughly how to say it, and incrementally refine this decision; finally, they combine the meaning with the syntactic sketch to 'spell out' the sequence of words making up the sentence.

Our approach is an extension of SEPARATOR, introduced in Chapter 3. SEPARATOR used VQ-VAE to represent the syntactic structure of a sentence as a set of discrete codes, but the lack of a known decomposition over the joint distribution $p(q_{1:H})$ made it difficult to predict valid syntactic forms during inference. By contrast, HRQ-VAE has a known decomposition over the joint probability, making it easier to predict syntactic sketches at test time. Additionally, the hierarchical nature of HRQ-VAE allows the model to learn more expressive sketches.

5.2.2 Factorisation and Objective

The graphical model in Figure 5.1b factorises as

$$p(\mathbf{y}, \mathbf{z}_{sem}) = \sum_{q_{1:D}, \mathbf{z}_{syn}} p(\mathbf{y}|\mathbf{z}_{sem}, \mathbf{z}_{syn}) \times p(\mathbf{z}_{syn}|q_{1:D}) \tag{5.1}$$

$$\times p(\mathbf{z}_{sem}) \times p(q_1|\mathbf{z}_{sem}) \prod_{d=2}^{D} p(q_d|q_{<d}, \mathbf{z}_{sem}). \tag{5.2}$$

Although $q_{1:D}$ are conditionally dependent on $\mathbf{z}_{sem}$, we assume that $\mathbf{z}_{sem}$ may be determined from $\mathbf{y}$ without needing to explicitly calculate $q_{1:D}$ or $\mathbf{z}_{syn}$. We also assume that the mapping from discrete codes $q_{1:D}$ to $\mathbf{z}_{syn}$ is a deterministic function $f_{q\rightarrow\mathbf{z}}(\cdot)$. The posterior therefore factorises as

$$\phi(\mathbf{z}_{sem}, \mathbf{z}_{syn}|\mathbf{y}) = \phi(\mathbf{z}_{sem}|\mathbf{y}) \times \phi(\mathbf{z}_{syn}|\mathbf{y}) \tag{5.3}$$

$$\times \phi(q_1|\mathbf{z}_{syn}) \times \prod_{d=2}^{D} \phi(q_d|q_{<d}, \mathbf{z}_{syn}). \tag{5.4}$$

The separation between $\mathbf{z}_{sem}$ and $q_{1:D}$, such that they represent the meaning and form of the input respectively, is induced by the training scheme. During training, the model is trained to reconstruct a target $\mathbf{y}$ using $\mathbf{z}_{sem}$ derived from an input with the correct meaning (a paraphrase) $\mathbf{x}_{sem}$, and $q_{1:D}$ from another input with the correct form (a syntactic exemplar) $\mathbf{x}_{syn}$. We showed in Chapter 3 that the model therefore learns to encode primarily semantic information about the input in $\mathbf{z}_{sem}$, and primarily syntactic information in $q_{1:D}$. Exemplars are retrieved from the training data following the process described in Section 5.2.3. The setup is shown in Figure 5.1a; in summary, during training we set $\phi(\mathbf{z}_{sem}|\mathbf{y}) = \phi(\mathbf{z}_{sem}|\mathbf{x}_{sem})$ and $\phi(q_d|\mathbf{y}, q_{<d}) = \phi(q_d|\mathbf{x}_{syn}, q_{<d})$. The final objective is given by

$$\begin{aligned}\text{ELBO} =& \mathbb{E}_{\phi}\big[-\log p(\mathbf{y}|\mathbf{z}_{sem}, q_{1:D})) \\ & - \log p(q_1|\mathbf{z}_{sem}) - \sum_{d=2}^{D} \log p(q_d|q_{<d}, \mathbf{z}_{sem})\big] \\ & + KL\big[\phi(\mathbf{z}_{sem}|\mathbf{x}_{sem}) || p(\mathbf{z}_{sem})\big],\end{aligned} \tag{5.5}$$

where $q_d \sim \phi(q_d|\mathbf{x}_{syn})$ and $\mathbf{z}_{sem} \sim \phi(\mathbf{z}_{sem}|\mathbf{x}_{sem})$.

5.2.3 Exemplar Retrieval Process

Our approach requires exemplars during training to induce the separation between latent spaces. We follow the approach introduced in Chapter 3 (Hosking and Lapata, 2021),

Input	How heavy is a moose?
Chunker output	How [heavy]$_{\text{ADVP}}$ is a [moose]$_{\text{NP}}$?
Template	How ADVP is a NP ?
Exemplar	How much is a surgeon's income?
Input	What country do parrots live in
Chunker output	What [country]$_{\text{NP}}$ do [parrots]$_{\text{NP}}$ [live]$_{\text{VP}}$ in ?
Template	What NP do NP VP in ?
Exemplar	What religion do Portuguese believe in?

Table 5.1: Examples of the exemplar retrieval process for training, repeated from Chapter 3. The input is tagged by a chunker, ignoring stopwords. An exemplar with the same template is then retrieved from the training corpus.

whereby exemplars are extracted from the training corpus automatically. For each target sentence $\mathbf{y}$, we retrieve exemplars $\mathbf{x}_{syn}$ from the training data by first identifying the underlying syntax of $\mathbf{y}$, and finding a question with the same syntactic structure but a different, arbitrary meaning. We use a shallow approximation of syntax, to ensure the availability of equivalent exemplars in the training data. An example of the exemplar retrieval process is shown in Table 5.1; we first apply a chunker (FlairNLP, Akbik et al., 2018) to $\mathbf{y}$, then extract the chunk label for each tagged span, ignoring stopwords. This gives us the *template* that $\mathbf{y}$ follows. We then select a question at random from the training data with the same template to give $\mathbf{x}_{syn}$. If no other questions in the dataset use this template, we create an exemplar by replacing each chunk with a random sample of the same type.

5.3 Neural Parameterisation

For simplicity, we assume a Gaussian distribution for $\mathbf{z}_{sem}$, with prior $p(\mathbf{z}_{sem}) \sim \mathcal{N}(\mathbf{0}, \mathbf{1})$. The encoders $\phi(\mathbf{z}_{sem}|\mathbf{x}_{sem})$ and $\phi(\mathbf{z}_{syn}|\mathbf{x}_{syn})$ are Transformers (Vaswani et al., 2017), and we use an autoregressive Transformer decoder for $p(\mathbf{y}|\mathbf{z}_{sem}, \mathbf{z}_{syn})$. The mapping $f_{q\rightarrow\mathbf{z}}(\cdot)$ from $q_{1:D}$ to $\mathbf{z}_{syn}$ and the posterior network $\phi(q_d|q_{<d}, \mathbf{z}_{syn})$ are based on HRQ-VAE (Chapter 4). We repeat the relevant details here for convenience.

5.3.1 Hierarchical Residual Quantisation

Let $\mathbf{z}_{syn} \in \mathcal{R}^{\mathbb{D}}$ be the output of the encoder network $\phi(\mathbf{z}_{syn}|\mathbf{y})$, that we wish to decompose as a sequence of discrete hierarchical codes. Recall that $q_d \in \{1, \ldots, K\}$ are discrete latent variables corresponding to the codes at different levels in the hierarchy, $d \in \{1, \ldots, D\}$. Each level uses a distinct codebook, $\mathbf{C}_d \in \mathbb{R}^{K \times \mathbb{D}}$, which maps each discrete code to a continuous embedding $\mathbf{C}_d(q_d) \in \mathbb{R}^{\mathbb{D}}$.

The distribution over codes at each level is a softmax distribution, with the scores s_d given by the distance from each of the codebook embeddings to the residual error between the input and the cumulative embedding from all previous levels,

$$s_d(q) = -\left(\left[\mathbf{x} - \sum_{d'=1}^{d-1} \mathbf{C}_{d'}(q_{d'})\right] - \mathbf{C}_d(q)\right)^2. \tag{5.6}$$

These embeddings therefore represent iterative refinements on the quantisation of the input. The posterior network $\phi(q_d|q_{<d}, \mathbf{z}_{syn})$ iteratively *decomposes* an encoding vector into a path through a hierarchy of clusters whose centroids are the codebook embeddings.

Given a sequence of discrete codes $q_{1:D}$, we deterministically construct its continuous representation with the composition function $f_{q\to\mathbf{z}}(\cdot)$,

$$\mathbf{z}_{syn} = f_{q\to\mathbf{z}}(q_{1:D}) = \sum_{d=1}^{D} \mathbf{C}_d(q_d). \tag{5.7}$$

Initialisation Decay Smaller perturbations in encoding space should result in more fine grained changes in the information they encode. Therefore, we encourage *ordering* between the levels of hierarchy (such that lower levels encode finer grained information) by initialising the codebook with a decaying scale, such that later embeddings have a smaller norm than those higher in the hierarchy. Specifically, the norm of the embeddings at level d is weighted by a factor $(\alpha_{init})^{d-1}$. The model is highly sensitive to the initialisation of the codebook; the initial codes should be located in roughly the same region of space as the output of the encoder, but should have sufficient variation so as to be informative for the decoder (see Table 5.9).

Depth Dropout To encourage the hierarchy within the encoding space to correspond to hierarchical properties of the output, we introduce *depth dropout*, whereby the hierarchy is truncated at each level during training with some probability p_{depth}. The output of the quantiser is then given by

$$\mathbf{z}_{syn} = \sum_{d=1}^{D} \left(\mathbf{C}_d(q_d) \prod_{d'=1}^{d} \gamma_{d'} \right), \tag{5.8}$$

where $\gamma_h \sim \text{Bernoulli}(1 - p_{depth})$. This means that the model is sometimes trained to reconstruct the output based only on a *partial* encoding of the input, and should learn to cluster similar outputs together at each level in the hierarchy.

5.3.2 Sketch Prediction Network

During training the decoder is driven using sketches sampled from the encoder, but at test time exemplars are unavailable and we must predict a distribution over syntactic sketches $p(q_{1:D}|\mathbf{z}_{sem})$. Modelling the sketches as hierarchical ensures that this distribution admits an autoregressive factorisation.

Similar to Chapter 3, we use a simple network to infer valid codes at each level of hierarchy. However, since the codes are now ordered hierarchically we use a recurrent model, using the semantics of the input sentence and the cumulative embedding of the predicted path so far as input, such that q_d is sampled from

$$p(q_d|\mathbf{z}_{sem}, q_{<d}) = \mathtt{Softmax}(\mathtt{MLP}_d(\mathbf{z}_{sem}, \mathbf{z}_{<d})), \tag{5.9}$$

where

$$\mathbf{z}_{<d} = \sum_{d'=1}^{d-1} \mathbf{C}_{d'}(q_{d'}). \tag{5.10}$$

This MLP is trained jointly with the encoder/decoder model, using the outputs of the posterior network $\phi(q_d|\mathbf{x}_{syn}, q_{<d})$ as targets. To generate paraphrases as test time, we sample from the sketch prediction model $p(q_d|\mathbf{z}_{sem}, q_{<d})$ using beam search and condition generation on these predicted sketches.

5.3.3 Training Setup

We use the Gumbel reparameterisation trick (Jang et al., 2017; Maddison et al., 2017; Sønderby et al., 2017) for the discrete codes and the standard Gaussian reparameterisation for the semantic representation. To encourage the model to use the full codebook, we decay the Gumbel temperature τ, according to the schedule

$$\tau = \max\big(\tau_0 \times \exp(-\frac{t}{\gamma_{temp}}), \tau_{min}\big), \tag{5.11}$$

where τ_0, τ_{min} are the initial and final temperatures, and γ_{temp} controls the rate of decay, in line with Jang et al. (2017). We approximate the expectation in Equation (5.5) by sampling from the training set and updating via backpropagation (Kingma and Welling, 2014). The norm loss described in Section 4.4 was not used. The full model was trained jointly by optimizing the ELBO in Equation (5.5).

5.4 Experimental Setup

Datasets As in Chapter 3, we evaluate on three datasets of English paraphrases that are grounded in some common meaning. Paralex and QQP are datasets of questions, where each paraphrase cluster shares a common (hypothetical) *answer*, while MSCOCO is a dataset of image captions where each caption is grounded by the *image* that it describes. We use the splits released by Hosking and Lapata (2021), as used in Chapter 3. We give examples of paraphrase clusters and dataset statistics in Section 3.4.

Model Configuration Hyperparameters were tuned on the Paralex development set, and reused for the other evaluations. We set the depth of the hierarchy $D = 3$, and the codebook size $K = 16$. The Transformer encoder and decoder consist of 5 layers each, and we use the vocabulary and token embeddings from BERT-Base (Devlin et al., 2019). We use an initialisation decay factor of $\alpha_{init} = 0.5$, and a depth dropout probability $p_{depth} = 0.3$. A full set of hyperparameters is given in Appendix B.1, and our code is available at `https://github.com/tomhosking/hrq-vae`.

Comparison Systems We compare to the same range of systems as in Chapter 3, which we briefly describe again here. As baselines, we consider three popular architectures: a vanilla autoencoder (**AE**) that learns a single dense vector representation of an input sentence; a Gaussian Variational AutoEncoder (**VAE**, Bowman et al., 2015), which learns a distribution over dense vectors; and a Vector-Quantised Variational AutoEncoder (VQ-VAE, van den Oord et al., 2017), that represents the full input sentence as a set of discrete codes. All three models are trained to generate a sentence from one of its paraphrases in the training data, and are not trained with an autoencoder objective. We implement a simple **tf-idf** baseline (Jones, 1972), retrieving the question from the training set with the highest cosine similarity to the input. Finally, we include a basic copy baseline as a lower bound, that simply uses the input sentences as the output.

We also compare to a range of other paraphrasing systems. **ParaNMT** (Wieting and

Gimpel, 2018) translates input sentences into a pivot language (Czech), then back into English. Although this system was trained on high volumes of data (including Common Crawl), the training data contains relatively few questions, and we would not expect it to perform well on the two datasets of question paraphrases. 'Diverse Paraphraser using Submodularity' (**DiPS**; Kumar et al. 2019) uses efficient optimisation techniques to search a wider space of possible outputs and thereby increase the diversity of samples from a standard encoder-decoder model. Latent bag-of-words (**BoW**, Fu et al., 2019) uses an encoder-decoder model with a discrete bag-of-words as the latent encoding. **SOW/REAP** (Goyal and Durrett, 2020) uses a two stage approach, deriving a set of feasible syntactic rearrangements that is used to guide a second encoder-decoder model. **BTmPG** (Lin and Wan, 2021) uses multi-round generation to improve diversity and a reverse paraphrasing model to preserve semantic fidelity. We use the results after 10 rounds of paraphrasing. **SEPARATOR** (Hosking and Lapata, 2021), described in Chapter 3, uses separated, non-hierarchical encoding spaces for the meaning and form of an input, and an additional inference model to predict the target syntactic form at test time. All comparison systems were trained and evaluated on our splits of the datasets.

We additionally compare to an instruction-tuned LLM, **Mistral 7B** Instruct v0.2, one of the strongest performing open-weight LLMs available at the time of writing. The LLM was prompted in a zero-shot manner according to Prompt A.1. Recall that it is not an entirely fair comparison; LLMs were developed later than the other models, and use orders of magnitude more data and computational resources during training. For example, while Mistral has 7 billion trainable parameters, CALYPSO has 70 million, fewer than 1% of Mistral. The training data is also unknown, and it is possible that the model was trained on the evaluation splits of one or more of the datasets in our experiments.

As an upper bound, we select a sentence from the evaluation set to use as an **oracle** syntactic exemplar for both CALYPSO and SEPARATOR, conditioning generation on a sketch that is known to represent a valid surface form.

5.5 Results

Our experiments were designed to test two primary hypotheses: (1) Does CALYPSO learn a *hierarchical* decompositions of an encoding space? and (2) Does our choice of generative model enable us to generate *high quality* and *diverse* paraphrases?

5.5.1 Inspecting the Hierarchy

Figure 5.2 shows a t-SNE (van der Maaten and Hinton, 2008) plot of the syntactic encodings $\mathbf{z}_{syn}$ for 10,000 examples from Paralex. The encodings are labelled by their quantisation, so that colours indicate top-level codes q_1, shapes denote q_2, and patterns q_3. The first plot shows clear high level structure, with increasingly fine levels of substructure visible as we zoom into each cluster. This confirms that the discrete codes are ordered, with lower levels in the hierarchy encoding more fine grained information.

To confirm that intermediate levels of hierarchy represent valid points in the encoding space, we generate paraphrases using oracle sketches, but truncate the sketches at different depths. Masking one level (i.e., using only q_1, q_2) reduces performance by 2.5 iBLEU points on Paralex, and two levels by 5.5 (iBLEU is an automatic metric for assessing paraphrase quality; see Section 5.5.2). Although encodings using the full depth are the most informative, *partial* encodings still lead to good quality output, with a gradual degradation. This implies both that each level in the hierarchy contains useful information, and that the cluster centroids at each level are representative of the individual members of those clusters.

5.5.2 Paraphrase Generation

Metrics As in Chapter 3, our primary metric is iBLEU (Sun and Zhou, 2012) that measures the fidelity of generated outputs to reference paraphrases as well as the level of diversity introduced (see Equation (2.26)). Following Sun and Zhou (2012) and in line with Chapter 3, we set $\alpha = 0.8$. We also report BLEU$(outputs, references)$ as well as Self-BLEU$(outputs, inputs)$. The latter allows us to examine the extent to which models generate paraphrases that differ from the original input.

Reference-based metrics rely on the availability of high-quality and diverse reference paraphrases, which are not always available. Utterances that are paraphrases of each other should entail each other, so we also use a *reference-free* metric based on Natural Language Inference (NLI). As in Chapter 3, we use an NLI model (DeBERTa v3, trained on Debiased NLI; He et al., 2021; Wu et al., 2022) to measure the degree to which the generated paraphrase is entailed by the input utterance (forward entailment) and vice-versa (backward entailment), and report the mean of the forward and backward scores as 'NLI' (Zhang et al., 2024a).

To evaluate the diversity between multiple candidates generated by the *same system*,

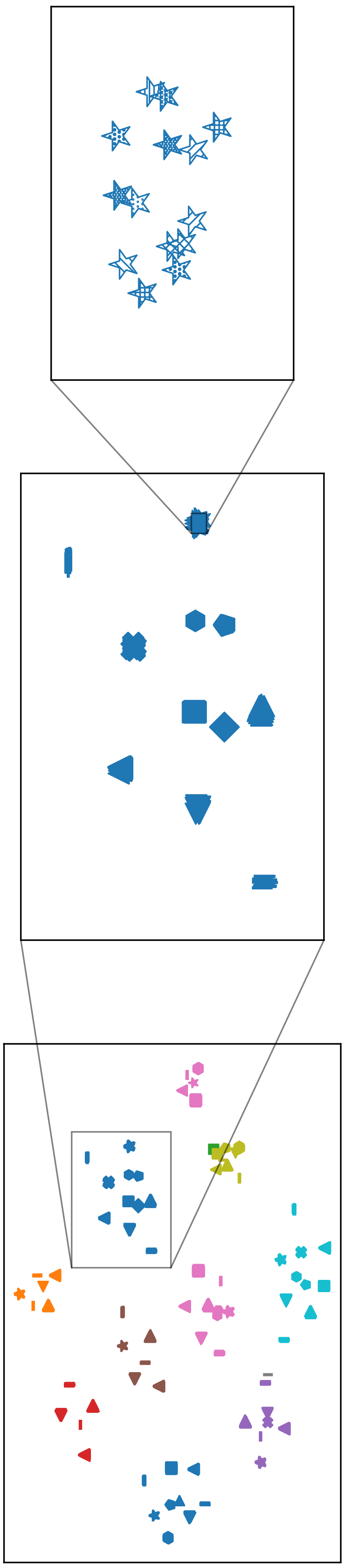


Figure 5.2: t-SNE visualisation of the syntactic encodings $\mathbf{z}_{syn}$ for 10k examples from Paralex: colours indicate top-level codes q_1, shapes indicate the second level, and patterns are used to label the third level. Deeper levels in the hierarchy represent finer grained information in encoding space.

	Paralex			
Model	BLEU ↑	Self-BLEU ↓	**iBLEU** ↑	NLI ↑
Copy	37.1	100.0	9.7	97.4
tf-idf	25.1	**25.3**	15.0	26.1
AE	**39.7**	69.4	17.9	87.2
VAE	39.2	53.2	20.8	78.3
VQ-VAE	36.0	52.3	18.3	74.1
SOW/REAP	33.1	37.1	19.1	62.8
LBoW	26.4	27.9	15.5	8.9
BTmPG	28.4	36.0	15.5	51.6
DiPS	25.7	28.3	14.9	28.9
ParaNMT	27.5	52.0	11.6	**92.6**
Mistral 7B	13.4	14.1	7.9	88.1
SEPARATOR	36.3	35.4	22.0	60.8
CALYPSO	39.5	33.3	**24.9**	64.2
Oracle$_{\text{SEPARATOR}}$	52.0	24.4	36.7	50.9
Oracle$_{\text{CALYPSO}}$	50.6	28.1	34.8	57.0

Table 5.2: Top-1 paraphrase generation results for Paralex. CALYPSO achieves the highest iBLEU scores, indicating the best tradeoff between quality and diversity. Paired bootstrap resampling (Koehn, 2004) indicates that CALYPSO achieves significantly higher iBLEU scores than all other systems (p< 0.05)

we report pairwise-BLEU (Cao and Wan, 2020),

$$\text{P-BLEU} = \mathbb{E}_{i \neq j}[\text{BLEU}(outputs_i, outputs_j)].$$

This measures the average similarity between the different candidates, with a lower score indicating more diverse hypotheses.

Automatic Evaluation Shown in Tables 5.2 to 5.4, the results of the automatic evaluation highlight again the importance of measuring both paraphrase quality and similarity to the input: the Copy baseline is able to achieve high BLEU scores despite simply duplicating the input. The VAE baseline is competitive but tends to have a high Self-BLEU score, indicating that the semantic preservation comes at the cost of low syntactic diversity. CALYPSO achieves both higher BLEU scores and higher iBLEU

	QQP			
Model	BLEU ↑	Self-BLEU ↓	**iBLEU** ↑	NLI ↑
Copy	34.5	100.0	7.6	98.6
tf-idf	24.1	62.5	6.7	59.7
AE	29.5	61.8	11.3	89.8
VAE	21.3	39.8	9.1	71.7
VQ-VAE	28.3	56.8	11.3	84.6
SOW/REAP	17.4	30.4	7.9	55.1
LBoW	23.1	41.2	10.3	22.9
BTmPG	20.9	36.5	9.4	59.8
DiPS	18.8	28.6	9.3	33.2
ParaNMT	25.7	57.6	9.0	**94.6**
Mistral 7B	8.9	12.6	4.6	90.9
SEPARATOR	23.9	**23.5**	14.5	59.9
CALYPSO	**33.1**	40.4	**18.4**	77.7
Oracle$_{\text{SEPARATOR}}$	40.9	26.4	27.4	62.8
Oracle$_{\text{CALYPSO}}$	50.5	36.8	33.0	72.2

Table 5.3: Top-1 paraphrase generation results for QQP. CALYPSO achieves the highest iBLEU scores, indicating the best tradeoff between quality and diversity. Paired bootstrap resampling (Koehn, 2004) indicates that CALYPSO achieves significantly higher iBLEU scores than all other systems ($p < 0.05$)

	MSCOCO			
Model	BLEU ↑	Self-BLEU ↓	**iBLEU** ↑	NLI ↑
Copy	19.9	100.0	-4.1	98.6
tf-idf	18.3	38.4	6.9	42.2
AE	27.6	39.3	14.2	61.7
VAE	27.3	24.1	17.0	43.9
VQ-VAE	25.9	28.4	15.1	41.2
SOW/REAP	12.5	**6.5**	8.7	30.9
LBoW	21.6	16.5	14.0	27.1
BTmPG	21.3	13.8	14.3	24.5
DiPS	19.0	14.4	12.3	28.6
ParaNMT	15.4	50.6	2.2	**91.3**
Mistral 7B	9.7	16.0	4.6	93.6
SEPARATOR	20.6	12.8	13.9	20.4
CALYPSO	**27.9**	16.6	**19.0**	36.1
$\text{Oracle}_{\text{SEPARATOR}}$	38.1	9.7	28.6	22.3
$\text{Oracle}_{\text{CALYPSO}}$	35.8	12.8	26.1	31.5

Table 5.4: Top-1 paraphrase generation results for MSCOCO. CALYPSO achieves the highest iBLEU scores, indicating the best tradeoff between quality and diversity. Paired bootstrap resampling (Koehn, 2004) indicates that CALYPSO achieves significantly higher iBLEU scores than all other systems (p< 0.05).

scores than the comparison systems, indicating that it is able to generate higher quality paraphrases without compromising on syntactic diversity. In particular, CALYPSO generates higher quality paraphrases than SEPARATOR (Chapter 3); the main difference between the two models is that CALYPSO uses hierarchical latent codes, indicating that the use of a more richly structured representation has led to better performance on the task.

Similar to our findings in Chapter 3, Mistral 7B achieves low scores for both BLEU and Self-BLEU, but some of the highest NLI scores. This indicates that it is succesfully generating outputs that are similar in meaning but have different surface form to the input, but that these outputs are also dissimilar to the reference paraphrases. While the other comparison systems were trained on the specific datasets being considered, Mistral 7B is evaluated zero-shot and so may generate outputs with a different style to the references. It is possible that a greater degree of diversity could be introduced by including a small number of few-shot examples in the prompt. It may also be possible to control the syntactic form of the output by including some exemplars as part of the input instruction. Future work could investigate whether LLMs may be used for controllable paraphrasing via prompting.

We plot NLI scores against *Dissimilarity* (defined as $100 -$ Self-Bleu) for all three datasets in Figure 5.3, where the ideal system would fall in the top-right corner. CALYPSO improves on SEPARATOR in all three cases, indicating that the richer choice of representation enables the model to generate more diverse paraphrases that better preserve the original meaning of the input.

The examples in Table 5.5 demonstrate that CALYPSO is able to introduce significant syntactic variation while preserving the original meaning of the input. However, there is still a gap between generation using predicted sketches and 'oracle' sketches (i.e., when the target syntactic form is known in advance), indicating ample scope for improvement. Mistral 7B generates fluent paraphrases, but is still not infallible; the last example, from MSCOCO, does not perfectly preserve the original sentence meaning.

Worked Example Since the sketches $q_{1:D}$ are latent variables, interpretation is difficult. However, a detailed inspection of example output reveals some patterns.

Table 5.6 shows the model output for a single semantic input drawn from Paralex, across a range of different syntactic sketches. It shows that q_1 is primarily responsible for encoding the question type, with $q_1 = 13$ leading to 'what' questions and $q_1 = 2$ 'how' questions. q_2 and q_3 encode more fine grained details; for example, all outputs

Paralex	Where is the birthplace of woman pro golfer Dottie Pepper?
VAE	Where is the birthplace of Pepper pro golfer Dottie?
BTmpG	What is the birthplace of women pro golfer?
SOW/REAP	What is the birthplace for golfer?
Latent BoW	Where did the golfer golfer originate?
SEPARATOR	Where is the birthplace of Dottie?
CALYPSO	Where is Dottie Pepper from?
Mistral 7B	What is the origin place of Dottie pepper, renowned female golf Player?
QQP	What are the best ways to defrost lobster tails?
VAE	What are the best ways to defrost lobster tails?
BTmpG	How can I defrost my tails??
SOW/REAP	What is defrost?
Latent BoW	How do you something a something lobster?
SEPARATOR	What are some of the best ways to defrost chicken?
CALYPSO	How do you thaw frozen lobster tails?
Mistral 7B	How can I effectively thaw lobster tails?
MSCOCO	Set of toy animals sitting in front of a red wooden wagon.
VAE	Two stuffed animals sitting in front of a toy train.
BTmpG	A herd of sheep grazing in a field of grass.
SOW/REAP	A close up of a close up of a street
Latent BoW	A toy wagon with a toy horse and a toy wagon.
SEPARATOR	A toy model of a toy horse and buggy.
CALYPSO	A group of stuffed animals sitting next to a wooden cart.
Mistral 7B	A red wooden wagon is home to a collection of toy animals.

Table 5.5: Examples of generated paraphrases. CALYPSO is able to preserve the original meaning, while introducing significant syntactic variation.

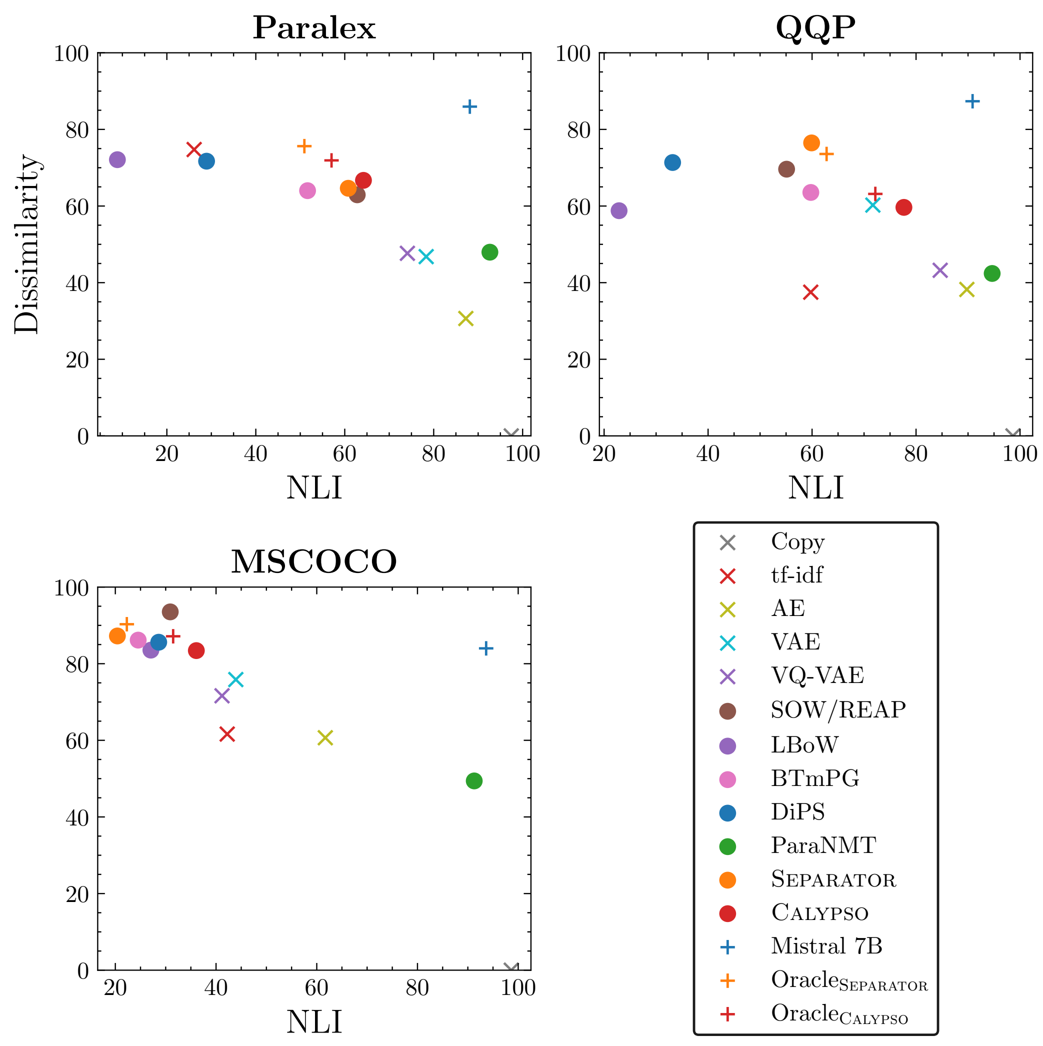


Figure 5.3: Dissimilarity (defined as 100 − Self-Bleu) against NLI scores for all models tested. The ideal model would be placed at the top-right of the plot. While Mistral 7B outperforms other systems, it was trained using many orders of magnitude more data and computational resources. Calypso offers the best balance between meaning preservation and dissimilarity of the non-LLM systems.

q_1	q_2	q_3	**Output**
Input			Two types of fats in body ?
0	3	6	What types of fats are in a body?
	13	7	What types of fats are there in body?
2	1	2	How many types of fats are there in the body?
	3	7	How many types of fats are there in a body?
5	3	6	What are the different types of fats in a body?
	5	7	What are the different types of fats in body?
	8	7	Types of fats are different from body fat?
		14	Two types of fats in body?
13	0	2	What are the different types of fats in the body?
		6	What are the different types of fats in a body?
	3	7	What are two types of fats in a body?
	5	7	What are the different types of fats in body?
		8	What are the different types of fats?
		14	What are the different types of fats in the body?

Table 5.6: Examples of model output, for a range of different sketches. The left hand side shows the sketch (i.e., the values of the codes $q_{1:D}$), with the corresponding model output on the right. q_1 primarily specifies the wh- word (e.g., outputs with $q_1 = 13$ are all 'what' questions), while q_2, q_3 correspond to more fine grained details, e.g., the outputs with $q_3 = 6$ all use the article 'a' when referring to 'body'.

Input	Two types of fat in body?
Exemplar	How many states are in the USA?
No sketch	What are the different types of fats in the body?
q_1	How many types of fats are there in the body?
q_1, q_2	How many fats does the body have?
q_1, q_2, q_3	How many fat are in the body?

Table 5.7: Model output for varying sketch granularities. When no sketch is used, the model defaults to the most common phrasing of the question. As more detail is included, the output converges towards the exemplar.

shown with $q_3 = 6$ use the indefinite article ‘a’.

We also examine how using increasingly granular sketches refines the syntactic template of the output. Table 5.7 shows the model output for a single semantic input, using varying granularities of sketch extracted from the exemplar. When no sketch is specified, the model defaults to a canonical phrasing of the question. When only q_1 is specified, the output becomes a ‘how many’ question, and when a full sketch is included, the output closely resembles the exemplar.

Generating Multiple Paraphrases We evaluate the ability of our system to generate multiple diverse paraphrases for a single input, and compare to the other comparison systems capable of producing more than one output. For both CALYPSO and SEPARATOR, we use beam search to sample from the sketch prediction network as in the top-1 case, and condition generation on the top-3 hypotheses predicted. For BTmPG, we use the paraphrases generated after 3, 6 and 10 rounds. For the VAE, we condition generation on 3 different samples from the encoding space. For Mistral 7B, we generate 3 samples with temperature 0.7. The results in Table 5.8 show that CALYPSO is able to generate multiple high quality paraphrases for a single input, with lower similarity between the candidates than other systems. In particular, CALYPSO achieves much better P-BLEU scores than Mistral 7B, indicating that despite the strong performance of LLMs, smaller task-specific models still have useful benefits.

5.5.3 Human Evaluation

Automatic evaluation is imperfect, and it is possible that the improvements in iBLEU scores for CALYPSO are an artifact of the metrics used. To determine whether CA-

	Paralex		QQP		MSCOCO	
Model	iBLEU ↑	P-BLEU ↓	iBLEU ↑	P-BLEU ↓	iBLEU ↑	P-BLEU ↓
VAE	20.49	67.62	11.52	64.71	17.22	55.66
BTmPG	15.50	89.20	9.13	82.02	13.20	80.38
Mistral 7B	7.82	77.21	4.47	74.11	4.58	64.54
SEPARATOR	21.67	62.98	13.63	**52.87**	13.77	57.79
CALYPSO	**22.75**	**40.48**	**17.49**	57.29	**18.39**	**41.29**

Table 5.8: Top-3 generation results. P-BLEU indicates the similarity between the different candidates, while iBLEU scores reported are the mean across the 3 candidates. CALYPSO is able to generate multiple high quality paraphrases with more diversity between them than comparison systems.

LYPSO indeed generates higher quality paraphrases, we performed a human evaluation. We elicited judgements from crowdworkers. They were shown a sentence and two paraphrases, each generated by a different system, and asked to select which one was preferred along three dimensions: the *dissimilarity* of the paraphrase compared to the original sentence; how well the paraphrase reflected the *meaning* of the original; and the *fluency* of the paraphrase. Annotators were recruited from the UK and USA via Amazon Mechanical Turk (AMT), and were compensated for their time above a living wage in those countries. We note that since this study, some doubt has been cast within the community on the quality of studies performed on AMT, and the human evaluations in Chapters 6 and 7 use Prolific for participant recruitment instead. However, a reproduction study of our claims found that they were reproducible (Arvan and Parde, 2024). A full Participant Information Sheet was provided, and the study was approved by an internal ethics committee. Annotators were asked to rate the outputs according to the following criteria:

- **Fluency** — Which system output is the most fluent and grammatical?
- **Meaning** — To what extent is the meaning expressed in the original sentence preserved in the rewritten version, with no additional information added?
- **Dissimilarity** — Does the rewritten version use different words or phrasing to the original? You should choose the system that uses the most different words or word order.

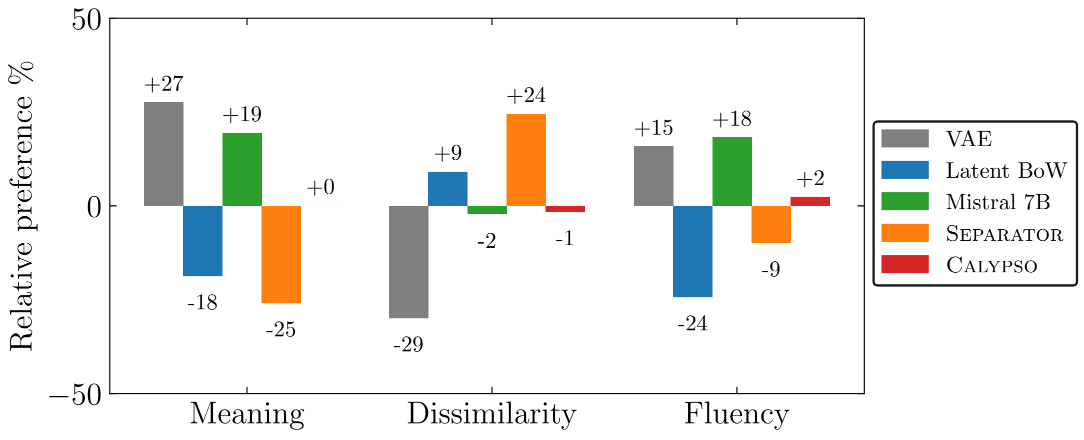


Figure 5.4: Results of our human evaluation. Although the VAE baseline is the best at preserving sentence meaning, it is the worst at introducing variation to the output. CALYPSO is more fluent and better at preserving meaning than both SEPARATOR and Latent BoW. CALYPSO generates outputs that are comparatively dissimilar compared to Mistral 7B, but the LLM is more fluent and better able to preserve the meaning of the input. Differences compared to CALYPSO are all significant (using a one-way ANOVA with post-hoc Tukey HSD test,p<0.05), except for Dissimilarity against Mistral 7B.

We evaluate a total of 300 sentences sampled equally from each of the three evaluation datasets, and collected 3 ratings for each sample. We assign each system a score of $+1$ when it was selected, -1 when the other system was selected, and took the mean over all samples (Louviere and Woodworth, 1990; Kiritchenko and Mohammad, 2017). Negative scores indicate that a system was selected less often than an alternative. We choose the five best performing models for our evaluation: CALYPSO, SEPARATOR, Latent BoW, VAE, and Mistral 7B.

Figure 5.4 shows that although the VAE baseline is the best at preserving question meaning, it is also the worst at introducing variation to the output. CALYPSO better preserves the original sentence meaning compared to the other non-LLM systems while introducing more diversity than the VAE, as well as generating much more fluent output. CALYPSO is more fluent and better at preserving meaning than both SEPARATOR and Latent BoW. CALYPSO generates outputs that are comparatively dissimilar compared to Mistral 7B, but the high parameter count and extensive pretraining of the LLM mean it is more fluent and better able to preserve the meaning of the input. Differences compared to CALYPSO are all significant (using a one-way ANOVA with post-hoc Tukey HSD test,p<0.05), except for Dissimilarity against Mistral 7B.

Variant	**Paralex**	**QQP**	**MSCOCO**
CALYPSO (oracle)	34.85	33.01	26.07
No initialisation scaling	−3.06	−2.48	−3.02
No hierarchy	−8.84	−12.72	−3.10
CALYPSO	24.93	18.42	19.04
No head dropout	−0.62	−0.74	−0.81
Post-hoc k-means	−3.30	−5.35	−2.83

Table 5.9: Changes in iBLEU score for a range of ablations from our full model. All components lead to an improvement in paraphrase quality across datasets.

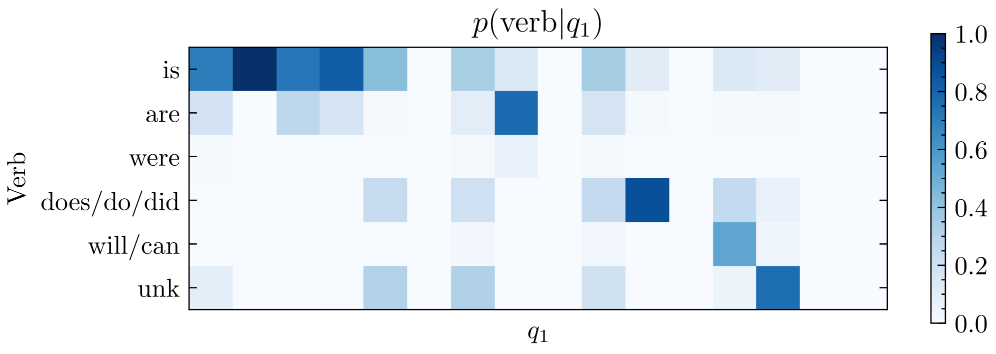
5.5.4 Ablations

To confirm that the hierarchical model allows for more expressive sketches, we perform two ablations. We compare to the full model using oracle sketches, so that code prediction performance is not a factor. We set the depth $D = 1$ and $K = 48$, giving equivalent total capacity to the full model ($D = 3, K = 16$) but without hierarchy. We also remove the initialisation scaling at lower depths, instead initialising all codebooks with the same scale. Table 5.9 shows that a non-hierarchical model with the same capacity is much less expressive, and that initialisation scaling leads to improved performance for all three datasets.

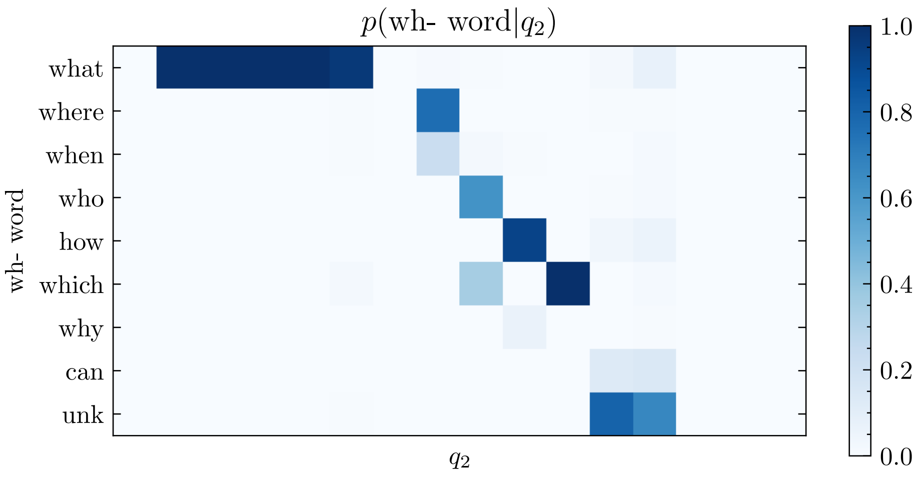
We also perform two ablations against the model using predicted sketches; we remove depth dropout, so that the model is always trained on a full encoding. We confirm that learning the codebooks jointly with the encoder/decoder leads to a stronger model, by first training a model with a continuous Gaussian bottleneck (instead of the CALYPSO); then, we recursively apply k-means clustering (Lloyd, 1982), with the clustering at each level taking place over the residual error from all levels so far, analogous to CALYPSO. The results of these ablations shown in Table 5.9 indicate that our approach leads to improvements across all datasets.

We define two features of sentences: (1) the presence of common auxiliary verbs that roughly indicate the tense of the sentence (present, future, etc.); and (2) the presence of different question or 'wh-' words[2]. We calculate the distributions of these features for each code q_d at different levels, with the results shown in Figure 5.5. Each column

[2]This analysis was performed for Paralex, which comprises entirely of questions.



(a) Distribution of verbs for each code within level 1.



(b) Distribution of wh- words for each code within level 2.

Figure 5.5: Plots showing the conditional distributions of two different sentence features, auxiliary verb and question type, for different values of the latent codes q_d. Each column represents the distribution over the feature for a specific code. The plots show that level 1 is a strong predictor of verb tense, and level 2 predicts question type, giving some insight into what syntactic features each level has learned to encode. We have reordered the columns of the plot to improve readability.

represents the distribution over the feature for a specific code. Figure 5.5a shows clear evidence that the sentences are (at least partly) clustered at the top level based on the verb used, while Figure 5.5b shows that level 2 encodes the question type.

5.6 Summary

In this chapter we present CALYPSO, a generative model of paraphrasing that uses a hierarchy of discrete latent variables as a rough syntactic sketch. CALYPSO applies HRQ-VAE to the task of paraphrase generation, representing the syntactic form of sentences as paths through a learned hierarchy, that can be predicted during testing.

The improvements in paraphrase generation performance for CALYPSO compared to SEPARATOR are a direct result of the more structured choice of representation. CALYPSO uses fewer parameters for the syntactic representations but more structure, which leads to richer representations that are also easier to predict during inference. They are still not truly interpretable, but the use of this weak structure allows us to inspect and interact with the representations in a way that would be extremely difficult with a continuous space. This adds support for Hypothesis I and Hypothesis II of this thesis, that weakly structured and discrete hierarchical representations can lead to improved performance on downstream tasks.

In the next chapter we investigate whether HRQ-VAE can also benefit more complex text-to-text generation tasks, and apply it to opinion summarisation. Opinion summarisation involves generating a textual summary that conveys the most frequent opinions from a large number of user reviews about a hotel, product or other entity. It is therefore a more challenging task than paraphrase generation, since models must be able to scale to large numbers of input reviews, and must generate multiple sentences at an appropriate level of abstraction. We will show that HRQ-VAE is a good fit for this problem, and can be used to generate high quality summaries.

Chapter 6

Opinion Summarisation with Hierarchical Sentence Representations

In the previous chapter, we showed that HRQ-VAE can be used to learn useful representations of the surface form of sentences, enabling us to more easily predict valid syntactic structures of generated paraphases at inference time. We now apply HRQ-VAE to a more complex task, *opinion summarisation*. Opinion summarisation involves aggregating opinions from multiple reviews about a product, hotel or other *entity* and generating an informative textual summary (Hu and Liu, 2004; Ganesan et al., 2010). We further assert that a good review aggregation system should identify frequent or common opinions, while abstracting away the details unique to a specific review. HRQ-VAE is a good candidate for meeting this joint requirement; the discretisation allows us to easily identify repeated opinions by simply counting them, and the hierarchy allows the model to encode high-level information (aspect, sentiment etc.) separately to specific details and phrasings.

In this chapter, we propose a method for unsupervised opinion summarisation that encodes sentences from customer reviews as paths through a learned hierarchical discrete latent space, then identifies common opinions based on the frequency of these paths. We are able to generate abstractive summaries by decoding the frequent encodings, as well as extractive summaries by selecting the sentences assigned to the same frequent encodings. The representation space is an instantiation of weak structure, since it is constrained to be tree structured but the meaning of each node is learned. Our method is attributable, because the model identifies sentences used to generate the summary as part of the summarisation process. It scales easily to many hundreds of input reviews, because aggregation is performed in the latent space rather than over long sequences of tokens. We also demonstrate that our approach enables a degree of control, generating aspect-specific summaries by restricting the model to parts of the encoding space that correspond to desired aspects (e.g., location or food). Automatic and human evaluation on two datasets from different domains demonstrates that our method generates summaries that are more informative than prior work and better grounded in the input reviews. The scalability and attributability of our method are direct results of the choice of discrete hierarchical representation, and so this chapter acts as a second piece of evidence in support of our hypotheses that weakly structured and discrete hierarchical representations are beneficial for text-to-text generation (Hypothesis I and Hypothesis II).

6.1 Introduction

Online review websites are a useful resource when choosing which hotel to visit or which product to buy, but it is impractical for a user to read hundreds of reviews. Following Ganesan et al. (2010), we define opinion summarisation, or *review aggregation*, as the task of generating a textual summary that reflects frequent or popular opinions expressed in a large number of reviews about an entity. Systems are *extractive* if they select sentences or spans from the input reviews to use as the summary, or *abstractive* if they generate novel output. We show idealised examples of both types of summary in Table 6.1. Review aggregation is challenging for a number of reasons. Firstly, it is difficult to acquire or create reference summaries, so models are almost always trained without access to gold standard references (Angelidis et al., 2021; Amplayo et al., 2021b, *inter alia.*). Secondly, popular entities may have hundreds of reviews, which can cause computational difficulties if the approach is not *scalable*. Finally, good summaries should be *abstractive* and not contain unnecessary detail, but should also not hallucinate false information. Ideally, a summarisation system should be *attributable*, offering some evidence to justify its output. Paraphrasing Rashkin et al. (2023), we say that a statement s is attributable to some evidence E, if a generic reader would agree that 'According to E, s is true'.

Previous work has either been exclusively extractive (which is inherently attributable and often scalable but leads to unnecessarily specific summaries) or exclusively abstractive (which often scales poorly and hallucinates, e.g., Bražinskas et al., 2020) . We propose a hybrid method, that produces abstractive summaries accompanied by references to input sentences which act as evidence for each output sentence, allowing

Review #1	Stayed for 4 nights at Arthur Frommer. The beds were supremely comfortable, and the ensuite, whilst compact, was modern. Location is terrific. It takes about 20 minutes to walk to the Oude Kerk and the same to walk to The Jewish Quarter or the Museum Quarter. Reception staff extremely helpful and professional, and speak pretty impeccable English.
Review #2	We stayed in this hotel for 3 nights. It is in a pretty and peaceful location, but within easy walking distance of restaurants and centre. The room was clean, the bed was reasonably comfortable.
Review #3	This is a great little hotel - it's not posh, it just does the essentials really well. The location is great - about 20 - 30 minutes walk to the Old Town and 10 minutes from Leidesplein or Rembrantsplein. Rooms are small but nicely decorated. The staff were always friendly and helpful. Would definitely stay again.
Extractive Summary	The beds were supremely comfortable. It is in a pretty and peaceful location, but within easy walking distance of restaurants. The staff were always friendly and helpful.
Abstractive Summary	The Mercure Amsterdam Arthur Frommer hotel offers comfortable beds, and stylishly decorated rooms in a quiet residential area. Its location is within easy walking distance of shops, restaurants, and public transportation. The staff were friendly and helpful, making for a pleasant stay overall.

Table 6.1: Idealised examples of extractive and abstractive summaries, based on 3 reviews of the Arthur Frommer hotel from SPACE. We highlight parts of the summaries and the spans in the reviews that support them. Extractive summaries copy representative spans verbatim from the reviews, and are therefore inherently attributable but likely to be less coherent. Abstractive summaries use novel language and are generally more fluent and coherent, but are not guaranteed to be attributable.

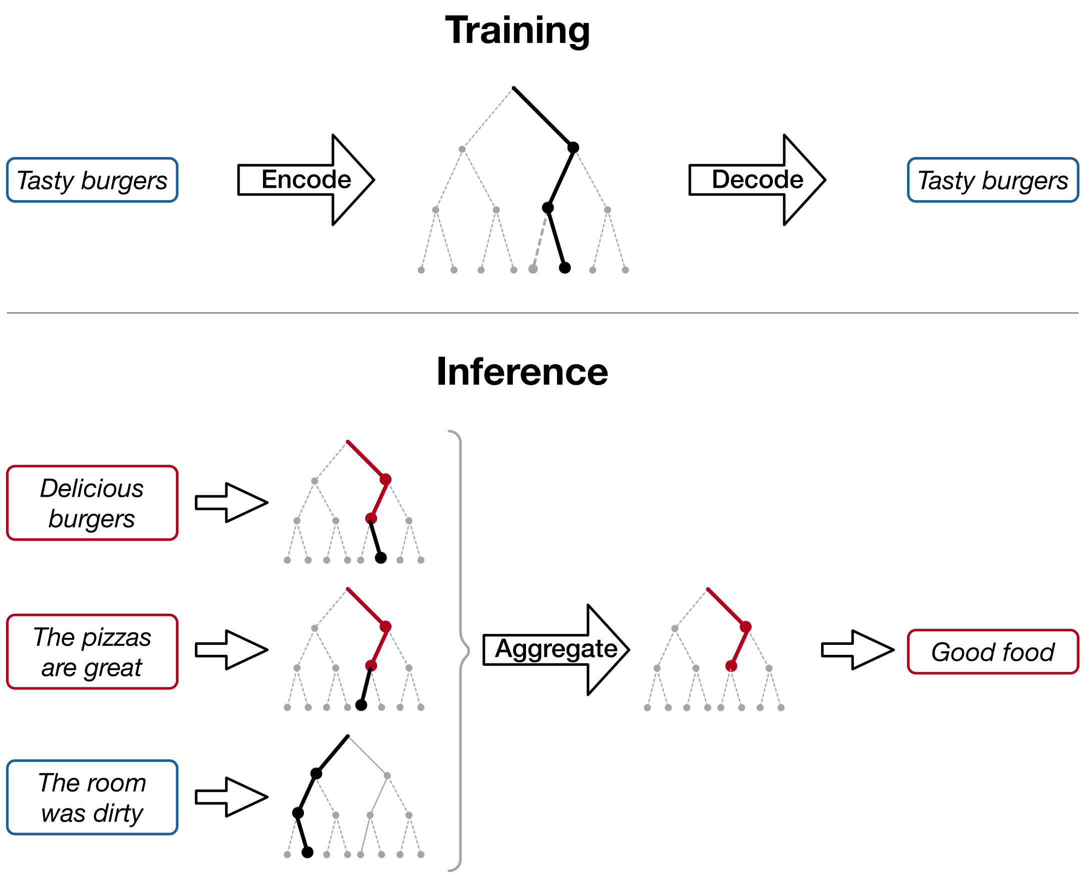


Figure 6.1: An idealised depiction of how HERCULES generates summaries of user reviews. HERCULES is trained to encode sentences from reviews as paths through a hierarchical discrete latent space (top). At inference time (bottom), we encode all sentences from the input reviews, and identify frequent paths or subpaths to use for the summary. The consensus opinion from the three example inputs is that the food is good, so the subpath shown in red is repeated; decoding it should result in an output like "Good food".

us to verify which parts of the input reviews were used to produce the output. Depicted in Figure 6.1, we first learn to encode natural language sentences from reviews as paths through a hierarchical discrete latent space. Then, given multiple review sentences about a specific entity, we identify common subpaths that are shared among many inputs, and decode them back to natural language, yielding the output summary. The sentences whose encodings contain the selected subpaths (shown in red in Figure 6.1) act as evidence for that generated sentence.

Our approach, HERCULES, is unsupervised and does not need reference summaries during training, instead relying on properties of the encoding space induced by the model. Since the aggregation process occurs in encoding space rather than over long sequences of tokens, HERCULES is highly scalable. Generated summaries are accompanied by supporting evidence from input reviews, making HERCULES attributable. It also offers a degree of controllability: we can generate summaries that focus on a specific aspect of an entity (e.g., location) or sentiment by restricting aggregation to subpaths that correlate with the desired property.

Our contributions are as follows:

- We propose a method for representing natural language sentences as paths through a hierarchical discrete latent space (Section 6.3).
- We exploit the properties of the learned hierarchy to identify common opinions from input reviews, and generate abstractive summaries alongside extractive *evidence sets* (Section 6.4).
- We conduct extensive experiments on two English datasets covering different domains, and show that our method outperforms previous state-of-the-art approaches, while offering the additional advantages of attributability and scalability (Sections 6.5 and 6.6).

6.2 Related Work

Supervised Methods Early work on opinion summarisation extracted reviewers' sentiment about specific features (Hu and Liu, 2004), presenting a quantitative aggregation of opinions. Beineke et al. (2004) propose the first method for generating *textual* summaries of opinions; they use the presence of hand crafted anchor words as inputs to logistic regression models, trained on examples of film reviews and 'highlights' sentences selected by editors to select relevant sentences.

Wang and Ling (2016) apply neural networks to the problem of summarizing multiple reviews, training an encoder-decoder model to generate summaries. However, their approach requires reference summaries during training, which are generally not available.

Cattan et al. (2023) propose using Key Point Analysis (KPA) to generate summaries. KPA first uses a model to extract a set of concise sentences or phrases, termed Key Points, from the input reviews. Then a second model maps each of the input sentences to its corresponding key points. Their approach requires labelled examples of key points to train both models, which may not be available in every domain.

Other approaches have sourced summaries to use as training data by scraping websites that contain professionally written reviews of products as well as bullet-point summaries (Bražinskas et al., 2021). These professional reviews are often created independently of the *customer* reviews, and so the resulting summaries do not reflect the distribution of users opinions.

Amplayo et al. (2021a,b) select 'central' reviews to use as proxy summaries, which may then be used as training data to fine-tune a pretrained language model. Hoever, they do not explicitly model the process of aggregating multiple diverse opinions into a single summary, and we argue that the reviews that are selected as proxt summaries are likely to contain specific details that are undesirable for summaries.

Iso et al. (2022) propose a method that highlights both common and contrastive opinions, generating summaries conditioned on reviews from two entities at the same time. Their model is trained using reviews as proxy summaries, in a similar manner to Amplayo et al. (2021b). At inference time, they penalise tokens that are likely to appear in the summaries for both entities, resulting in summaries that focus on the *differences* between the two entities. We draw on this idea when designing our agreggration function for selecting informative encodings; a useful summary should include information that not only appears repeatedly in input reviews, but also helps a user to distinguish one product from another.

Li and Chaturvedi (2024), published after this work, make attribution a core feature of their generated summaries. They propose a method that first extracts high level summary points (similar to Key Points) from reviews, using a model trained on annotated examples. Thet then identify supporting sentences using a score based on a combination of 'relatedness, specificity, popularity and diversity', which can act as a rationale and justifies why the summary point has been included.

Unsupervised Methods Erkan and Radev (2004) introduce an unsupervised summarisation method, which constructed a graph based on the overlap between sentences in a document, and identified salient sentences based on their centrality within that graph. While they did not explicitly consider opinion summarisation in their work, their method can nonetheless be used to summarise reviews.

Ganesan et al. (2010) propose the first *abstractive* opinion summarisation approach. They used a word graph to represent a set of input reviews, and searched for paths in the graph that are both highly redundant (therefore indicating frequent opinions) and also valid sentences.

Gerani et al. (2014) introduce an abstractive summarisation approach by using trees to model the discourse structure of each input review, then aggregating the trees from the multiple reviews into a graph that represents the opinions in the reviews. Finally, they generate textual summaries by mapping the graph to natural language with hand-crafted templates.

There has been previous work on aggregating user opinions in a learned latent space. Angelidis et al. (2021) and Basu Roy Chowdhury et al. (2022) train an autoencoder based on VQ-VAE to map review sentences to a discrete latent space. They then identify popular opinions by calculating the frequency of each of the discrete latent codes, and select the sentences that correspond to the top-k most frequent codes to use as an extractive summary.

Iso et al. (2021) use a continuous latent spaces to encode sentences from reviews, then identify representative encodings in that space by searching for a convex combination of encodings that maximizes the word overlap between input reviews and generated output. They generate a complete summary by decoding each of these encodings in turn, and show that this leads to more informative summaries than taking the simple average, but their approach scales poorly with large numbers of input reviews.

Bražinskas et al. (2020) model reviews as the textual realisation of a sample from a shared latent random variable which captures the common opinions across those reviews. The model is trained in an unsupervised manner to reconstruct a review from the latent variable and the other reviews in the set. At inference time, they generate summaries by decoding the mean of this latent variable for each set of reviews.

Large Language Models Louis and Maynez (2023) use a Natural Language Inference (NLI) model to construct 'silver standard' summaries to use as training data for fine-tuning a pretrained language model (T5, Raffel et al., 2020). However, their approach is computationally very expensive, calculating over 1B pairwise entailment relations between sentences in the training data, before fine-tuning a LLM. By contrast, HIRO uses a lightweight indexing encoder, combined with an off-the-shelf LLM that is prompted in a zero-shot manner.

Bhaskar et al. (2023) explore a range of pipelines that identify salient content with supervised clustering methods, and recursively summarise extracted groups of reviews with LLMs. They only consider aspect-specific summarisation, and their method requires a large number of calls to a LLM for each summary, but their results show that LLMs are a promising direction for opinion summarisation.

Since the work in this chapter was completed, there has been a focus on the distribution of sentiments within reviews and datasets of reviews. The majority of reviews tend to be positive, which can bias summarisation models towards generating overly positive summaries regardless of the specific inputs. This can result in poor generalisation – if a particular product has predominantly negative reviews, models may nonetheless generate a positive summary. Zhang et al. (2024b) propose using LLMs to generate synthetic reviews with a specific sentiment, then train a summarisation model on the resulting sentiment-balanced data, resulting in less biased summaries. Lei et al. (2024) use reinforcement learning, with a reward based on sentiment, to train a summarisation model where the distribution over output sentiments matches the distribution of sentiment in the input reviews.

6.3 Hierarchical Quantised Autoencoders

A good review aggregation system should identify frequent or common opinions, while abstracting away the details unique to a specific review. This joint requirement motivates our choice of a hierarchical discrete encoding: the discretisation allows us to easily identify repeated opinions by counting them, while the hierarchy allows the model to encode high-level information (aspect, sentiment etc.) separately to specific details and phrasings. It is not known *a priori* what an appropriate level of detail should be for the generated summary, and indeed this is likely to be contextual. Using a hierarchical encoding allows for a more adaptive approach. We therefore use HRQ-VAE, described in detail in Chapter 4, to represent sentences from reviews.

6.3.1 Probabilistic Model

Let $\mathbf{y}$ be a target sentence, represented as a sequence of tokens. We assume that the semantic content of $\mathbf{y}$ may be encoded as a sequence $q_{1:D}$ of discrete latent variables or *codes* $q_d \in \{1, \ldots, K\}$, with $1 \leq d \leq D$. Further, we assume that the $q_{1:D}$ are ordered *hierarchically*, such that q_1 represents high level information about the sentence (e.g., the aspect or overall sentiment) whereas q_D represents fine-grained information (e.g., the specific phrasing or choice of words used). The codes $q_{1:D}$ can be viewed as a single *path* through a hierarchy or tree as depicted in Figure 6.1, where each intermediate and leaf node in the tree corresponds to a sentence $\mathbf{y}$.

Recall from Chapter 4 that HRQ-VAE leads to the generative model shown in

Figure 4.2, which factorises as

$$p(\mathbf{y}) = \sum_{q_{1:D}} p(\mathbf{y}|q_{1:D}) \times \prod_{d=1}^{D} p(q_d), \tag{6.1}$$

with a posterior that factorises as

$$\phi(q_{1:D}|\mathbf{y}) = \phi(q_1|\mathbf{y}) \times \prod_{d=2}^{D} \phi(q_d|q_{<d}, \mathbf{y}). \tag{6.2}$$

As derived in Equation (4.7), the training objective is given by

$$\text{ELBO} = \mathbb{E}_{\phi}\big[-\log p(\mathbf{y}|q_{1:D})\big] + \beta_{KL} \sum_{d=1}^{D} \text{KL}\big[\phi(q_d|\mathbf{y}) \parallel p(q_d)\big] \tag{6.3}$$

where $q_d \sim \phi(q_d|\mathbf{y})$ and β_{KL} determines the weight of the KL term. We choose a uniform prior for $p(q_d)$.

6.3.2 Neural Parameterisation

The latent codes $q_{1:D}$ are discrete, but most neural methods operate in continuous space. We therefore need to define a mapping from the continuous output $\mathbf{z} \in \mathbb{R}^{\mathbb{D}}$ of an encoder network $\phi(\mathbf{z}|\mathbf{y})$ to the discrete codes $q_{1:D}$, and vice versa for a decoder $p(\mathbf{y}|\mathbf{z})$.

Encoder We use a Transformer architecture (Vaswani et al., 2017) with a multi-head pooling layer (Section 2.5.1) for the encoder $\phi(\mathbf{z}|\mathbf{x})$, which maps a sequence of tokens to a single dense vector. Then, we learn a codebook $\mathbf{C}_d \in \mathbb{R}^{K\times\mathbb{D}}$, which maps between discrete codes and continuous embedding $\mathbf{C}_d(q_d) \in \mathbb{R}^{\mathbb{D}}$.

Since the $q_{1:D}$ are intended to represent hierarchical information, the distribution over codes at each level is given by a softmax distribution with scores s_d given by the L2 distance from each of the codebook embeddings to the residual error between the input and the cumulative embedding from all previous levels,

$$s_d(q) = -\left(\left[\mathbf{z} - \sum_{d'=1}^{d-1} \mathbf{C}_{d'}(q_{d'})\right] - \mathbf{C}_d(q)\right)^2, \tag{6.4}$$

where $\mathbf{z}$ is the output of the encoder.

Decoder To generate an output sequence from a discrete path $q_{1:D}$, we first map it back to a continuous encoding $\mathbf{z}$ by performing the inverse of the decomposition process.

We take the sum of the embeddings of the codes q_d at each level,

$$\mathbf{z} = \sum_{d=1}^{D} \mathbf{C}_d(q_d). \tag{6.5}$$

The embeddings at each level can be viewed as refinements of the (cumulative) embedding so far, or alternatively as selecting the centroid of a subcluster within the current cluster. During inference, we set $q_d = \arg\max(s_d)$. We then use a Transformer (Vaswani et al., 2017) decoder $p(\mathbf{y}|\mathbf{z})$ to generate an output sequence from $\mathbf{z}$.

Importantly, it is not necessary to specify a path to the complete depth D; a *subpath* $q_{1:d}$ ($d < D$) still results in a valid embedding $\mathbf{z}$. We can therefore control the specificity of an encoding by varying its depth.

Note that so far, we have only described a method for mapping from sentences to codes, and vice versa. We describe how we exploit this mapping to identify frequent opinions in Section 6.4.

6.3.3 Training Setup

Recall that our goal is to learn a mapping from a sentence to a path through a learned hierarchy $q_{1:D}$, such that the top level codes represent abstract details about the sentence such as topic or sentiment, while the lower levels encode more fine-graned details about the sentence. This *semantic ordering* must be induced explicitly, and we now describe the training scheme used to achieve it.

We use the Gumbel reparameterisation (Jang et al., 2017; Maddison et al., 2017; Sønderby et al., 2017) to sample from the distribution over $q_{1:D}$. To encourage the model to explore the full codebook, we decay the Gumbel temperature τ according to the schedule in Equation (5.11). We approximate the expectation in Equation (6.3) by sampling from the training set and updating via backpropagation (Kingma and Welling, 2014).

Initialization Decay and Norm Loss Smaller perturbations in encoding space should result in more fine-grained changes in the information they encode. Therefore, we encourage *ordering* between the levels of hierarchy (such that lower levels encode more fine-grained information) by initialising the codebook with a decaying magnitude, such that deeper embeddings have a smaller norm than those higher in the hierarchy. Specifically, the norm of the embeddings at level d is weighted by a factor $(\alpha_{init})^{d-1}$. We also include an additional 'norm loss' $\mathcal{L}_{NL}$, described in Section 4.4, to encourage deeper embeddings to remain fine-grained during training,

$$\mathcal{L}_{NL} = \frac{\beta_{NL}}{D} \sum_{d=2}^{D} \left[\max\left(\gamma_{NL} \frac{||\mathbf{C}_d||_2}{||\mathbf{C}_{d-1}||_2}, 1\right) - 1 \right]^2,$$

where $\mathbf{C}_d$ is the codebook at level $d \in [1, \ldots, D]$, γ_{NL} determines the relative scale between levels and β_{NL} controls the strength of the loss. This ensures that the embeddings $\mathbf{C}_d$ are smaller than those at higher levels $\mathbf{C}_{<d}$, since this is otherwise not guaranteed. HRQ-VAE uses a very narrow bottleneck, and we found that models may exploit the available capacity of lower codebooks to improve the overall expressivity of the model, at the cost of the quality of the learned hierarchy.

Depth Dropout To encourage the hierarchy within the encoding space to correspond to hierarchical properties of the output, we truncate at each level during training with some probability p_{depth} (Hosking et al., 2022; Zeghidour et al., 2022). Instead of using the full depth of hierarchy to reconstruct the sentence embedding $\mathbf{z}$, the output of the quantiser is given by

$$\mathbf{z} = \sum_{d=1}^{D} \left(\mathbf{C}_d(q_d) \prod_{d'=1}^{d} \gamma_{d'} \right), \tag{6.6}$$

where

$$\gamma_h \sim \text{Bernoulli}(1 - p_{depth}). \tag{6.7}$$

This means that the model is sometimes trained to reconstruct the output based only on a *partial* encoding of the input, and should learn to cluster similar outputs together at each level in the hierarchy.

Denoising Objective To encourage the model to group sentences according to their meaning rather than their syntactic structure, we use a denoising objective as a form of distant supervision. We first split the reviews from the training corpus into sentences. Then, the model is trained to generate a target sentence from a different source sentence that has similar meaning but different surface form. For example, given the target sentence “We chose this hotel for price/location.”, a source might be “I chose this hotel for its price and location.”. The source sentences are retrieved automatically from other reviews in the training data using tf-idf (Jones, 1972) over bigrams; we select the top 5 most similar sentences for each target sentence with a minimum similarity of 0.6, and restrict to retrieving from reviews that have ratings equal to the target. We show

Source	*Target*
First the staff I met with were all VERY nice and helpful!	The staff which we met was very helpful and smiling.
Nice large room with plenty of storage space.	The room had plenty of storage and wardrobe space.
Highly Recommended Without Hesitation!	No complaints, highly recommended.
Easy Set Up!	This camera was very easy to set up.

Table 6.2: Examples of pairs of source and training sentences, used to train HERCULES. The pairs are automatically constructed using tf-idf to reduce the sensitivity of the model to the specific phrasing of a particular example and to encourage it to learn a space that is instead semantically structured.

some example pairs of source and target sentences in Table 6.2. Note that using tf-idf to identify related targets does not guarantee semantic equivalence; it is possible that two sentences may share a high degree of lexical overlap but express entirely different sentiments. In Chapter 7 we propose an extension to this approach that ensures stronger semantic consistency between source and target sentences.

6.4 Aggregating Reviews in Encoding Space

So far, we have described a method for mapping from a sentence $\mathbf{y}$ to a path $q_{1:D}$ and vice versa. We can now exploit the hierarchical property of the latent space to generate summaries.

Recall that the goal of review aggregation is to identify the majority or frequent opinions from a set of diverse inputs. This corresponds to identifying paths (or subpaths) in encoding space that are shared among many inputs. A simplified version of this process is depicted in the lower block of Figure 6.1; each sentence $\mathbf{y}^{(i)}$ in the input reviews is mapped to a path $q^{(i)}_{1:D}$ through the latent space. Summarizing these sentences is then reduced to the task of selecting a set of common subpaths, e.g., the subpath highlighted in red in Figure 6.1, which is shared between two out of three inputs.

Subpath Selection A simple approach would be to select the most frequent subpaths, but this would almost always result in high-level paths with $d = 1$ being selected (since every occurrence of a path $q_{1:d}$ entails an occurrence of all subpaths $q_{1:d'}, d' < d$).

In practice there is a tradeoff between frequency and specificity. Additionally, good summaries often exhibit structure; they generally include high-level comments, alongside more specific comments about details that particularly differentiate the current entity from others. Indeed, some datasets (e.g., AmaSum, Bražinskas et al., 2021, Section 6.5.1) were constructed by scraping overall 'verdicts' and specific 'pros and cons' from review websites. We therefore reflect this structure and propose both a 'generic' and 'specific' method for selecting subpaths. This is comparable to the notions of general and specific sentence discussed in Louis and Nenkova (2011).

To select *generic* subpaths, we construct a probability tree from the set of input sentence encodings, with the node weights set to the observed path frequency $p(q_{1:d})$. Then, we iteratively prune the tree, removing the lowest probability leaves until all leaf weights exceed a threshold, $\min\big(p(q_{1:d})\big) > 0.01$. Finally, we select the leaves with the top k weights to use for the summary. Empirically, this approach often selects paths with depth $d = 1$, but allows additional flexibility when a deeper subpath is particularly strongly represented.

Similar to Iso et al. (2022) we argue that the *specific* parts of the summary should also be comparative, highlighting details that are unique to the current entity. Thus, tf-idf (Jones, 1972) is a natural choice; we treat each path (and all its parent subpaths) as terms. We assign scores to each subpath $q_{1:d}$ proportional to its frequency within the current entity, and inversely proportional to the number of entities in which the subpath appears,

$$\text{score}(q_{1:d}) = \text{tf}(q_{1:d}) \times \log\big(\text{idf}(q_{1:d})\big). \tag{6.8}$$

Again, we select the subpaths with the top k scores to use for the summary.

The overall summary is the combination of the selected generic and specific subpaths. The abstractive natural language output is generated by passing the selected subpaths as inputs to the decoder.

Attribution Each sentence in the generated summary has an associated subpath. By identifying all inputs which share that subpath, we can construct an *evidence set* of sentences that act as an explanation or justification for the generated output. We show some examples of evidence sets in Table 6.15. We can also generate extractive summaries using these evidence sets, by selecting a single representative sentence from each set to use as the summary. We calculate the ROUGE-2 scores (Lin, 2004) between each pair of sentences in each evidence set, and use the sentence with highest similarity to all other sentences (i.e., the centroid) from each evidence set as the extractive summary.

Scalability Since the aggregation is performed in encoding space, our method scales linearly with the number of input sentences (compared to quadratic scaling for methods using LLMs that take a long sequence of all review sentences as input, e.g., Ouyang et al. (2022b)), and can therefore handle large numbers of input reviews. In fact, since we identify important opinions using a frequency-based method, our system does not perform well when the number of input reviews is small, since there is no strong signal as to which opinions are common.

Controlling the Output Given an aspect a (e.g., 'service') we source a set of keywords $\mathbb{K}_a$ (e.g., 'staff, friendly, unhelpful, concierge') associated with that aspect (Angelidis et al., 2021). We label each sentence in the training data with aspect a if it contains any of the associated keywords $\mathbb{K}_a$, then calculate the probability distribution over aspects for each encoding path, $p(a|q_{1:D})$. We can modify the scoring function in Equation (6.8), multiplying the subpath scores during aggregation by the corresponding likelihood of a desired aspect, thereby upweighting paths relevant to that aspect,

$$\text{score}_{asp}(q_{1:d}) = \text{tf}(q_{1:d}) \times \log\Big(\text{idf}(q_{1:d})\Big) \times p(a|q_{1:D}). \tag{6.9}$$

We can also control for the sentiment of the summary; for the case where reviews are accompanied by ratings, we can label each review sentence (and its subpath) with the rating r of the overall review, and reweight the subpath scores during aggregation by the likelihood of the desired rating $p(r|q_{1:D})$. In theory, we can use this approach to control for any property for which we have labelled sentences. We show some examples of controlled output in Tables 6.9 and 6.18.

6.5 Experimental Setup

6.5.1 Datasets

We perform experiments on two datasets from two different domains. **SPACE** (Angelidis et al., 2021) consists of hotel reviews from TripAdvisor, with 100 reviews per entity. It includes reference summaries constructed by human annotators, with multiple references for each entity. It also includes reference *aspect-specific* summaries, which we use to evaluate the controllability of HERCULES.

	SPACE	AmaSum
Entities	8350	7255
Reviews	303,357	533,972
Training pairs $(\mathbf{x}, \mathbf{x}_+)$	1,373,079	2,991,478

Table 6.3: Statistics for the training datasets.

	SPACE	AmaSum
Entities	25	200
Reviews per entity	100	560.4
Review length (words)	162.6	49.9
Ref. summaries (words)	82.0	80.1

Table 6.4: Statistics for the evaluation datasets.

AmaSum (Bražinskas et al., 2021) consists of reviews of Amazon products from a wide range of categories, with an average of 326 reviews per entity. The reference summaries were collected from professional review websites, and therefore are *not grounded* in the input reviews. The references in the original dataset are split into ‘verdict’, ‘pros’ and ‘cons’; we construct single summaries by concatenating these three. We filter the original dataset down to four common categories (Electronics, Shoes, Sports & Outdoors, Home & Kitchen), and evaluate on a subset of 50 entities, training separate models for each.

We trained and evaluated all systems in our experiments ourselves, using the same splits of each dataset. Summary statistics for both datasets are shown in Tables 6.3 and 6.4, and some examples of input reviews and associated reference summaries are given in Tables 6.5 to 6.7.

6.5.2 Comparison Systems

We present experiments for a range of baseline and comparison systems, both abstractive and extractive, and described in more detail in Section 6.2. For comparison, we construct extractive summaries using HERCULES by selecting the centroid from each evidence set based on ROUGE-2 F1 score.

As a lower bound, we select a **random review** from the inputs and use it as the summary. We also select the **centroid** of the set of reviews, by calculating the

Review	We stayed at the Navona on the first week of our honeymoon (feb 27th to march 2nd). It was VERY clean, perfectly situated (within walking distance of most sites) AND the staff was SO FRIENDLY. We loved our stay there. I'd highly recommend it, even though the morning coffee is instant (which should be a crime in Italy!)
Review	If you're familiar with European two-stars, you'll know what to expect here. The rooms and hotel were clean and comfortable, and breakfast was tasty. I thought the hotel was hard to find (and I never did quite figure out the surrounding streets) but the general location was good. The Piazza Navona is touristy but great fun. The beds were very small, and I think a large person would have difficulty with them. And it is indeed very noisy - I finally wore earplugs to block out the street noise.
Review	Three of us stayed at the Hotel Navona and shared a room. It had two twins pushed together and a separate cot-like bed. We all found the room to be more than large enough and very comfortable. I slept on the cot-like bed and slept very well. Shower worked well. Our room was on a courtyard with a large, heavy oak door. You cannot beat the atmosphere and location. The staff is very helpful and friendly. Location, Location, Location!
Summary	The staff was very friendly and helpful. The rooms were quiet and spotless, but were not the biggest. The good size bed and the daily service is good too. Breakfast is simple but nice & filling. The location of the hotel was superb, as well!

Table 6.5: An example of three input reviews (selected at random) for the Hotel Navona and their associated reference summary from SPACE.

Rooms	The rooms are a comfortable size, very clean and quiet, and offer hot showers, large flat screen TVs, and wifi
Location	This charming hotel is located within steps of all the major sights.
Service	The hotel's staff was very friendly and accommodating.
Food	The breakfast, for both quality and service, received very mixed reviews. The breakfast is mostly considered either poor or just fine. So was the service for the breakfast. Complaints included a stiff staff, bad coffee, few choices, and packaged hard rolls for bread.

Table 6.6: Examples of reference aspect-specific summaries for the Hotel Navona, from SPACE.

ROUGE-2 F1 score between each pair of reviews for an entity and selecting the review with highest similarity to all other reviews. We include an extractive **oracle** as an upper bound, by selecting the input sentence with highest ROUGE-2 similarity to each reference sentence.

Lexrank (Erkan and Radev, 2004) is an unsupervised extractive method using graph-based centrality scoring of sentences.

QT (Angelidis et al., 2021) uses vector quantisation to map sentences to a discrete encoding space, then generates extractive summaries by selecting representative sentences from clusters.

SemAE (Basu Roy Chowdhury et al., 2022) is an extractive method that extends QT, relaxing the discretisation and encoding sentences as mixtures of learned embeddings.

CopyCat (Bražinskas et al., 2020) is an abstractive approach that models sentences as observations of latent variables representing entity opinions.

Mistral 7B Instruct v0.2 and **Llama 2 7B/13B** Chat are open-weight LLMs that were released after the work in this chapter was performed. We nonetheless include them for consistency with future chapters. All three LLMs were prompted zero-shot with the prompt in Appendix C.1, and sampled with temperature 0.7. We report the mean and standard deviation scores based on 3 samples.

BiMeanVAE and **COOP** (Iso et al., 2021) are abstractive methods that encode full reviews as continuous latent vectors, and take the average (BiMeanVAE) or an optimised combination (COOP) of review encodings.

Finally, for aspect specific abstractive summarisation we compare to **AceSum**

Review	In my opinion Roku streamers are the best on the market. I have the Roku express on 3 TV's. If your TV has a USB and HDMI port everything connects neat and tidy. The cables and power supply (if needed) are included
Review	What I love about it ... you can extend it down from a high TV with a long HDMI cord. What drives me nutty ... I have to tell it to turn on closed captioning EVERY TIME and for EVERY EPISODE on Hulu. I don't have that same issue on the AmazonFire, for example. This is inexpensive and totally does what I need it to do. Just not quite as gracefully as I'd hoped (you really have to have the remote pointed right at it!)
Review	Purchased as a replacement to an ageing Roku2. Bad choice. The Roku Express is not a quality product. It doesn't not respond to the controller, constantly lags / buffers and requires frequent restarts after becoming unresponsive. This is not an issue with my Internet or Wifi as this box sits 10 feet from by router and streaming from any other device does not suffer from the same quality issues.
Summary	Powered by TV so you can use it anywhere. Can be controlled through the Roku app. Good WiFi connectivity with an HDMI port that provides extra options. Easy, fast setup and good selection of streaming stations. Has to be physically attached to the TV. Some units unpredictably drop WiFi

Table 6.7: An example of three input reviews (selected at random for the Roku TV stick and their associated reference summary from AmaSum.

(Amplayo et al., 2021a). AceSum uses multi-instance learning to induce a synthetic dataset of review/summary pairs with associated aspect labels, which is then used to train an abstractive summarisation model.

Most of the abstractive methods are not scalable and have upper limits on the number of input reviews. CopyCat and the LLMs have a maximum input sequence length, while COOP exhaustively searches over combinations of input reviews. We use 8 randomly selected reviews as input to CopyCat, Mistral, Llama 2 and COOP.

6.5.3 Automatic Metrics

We use ROUGE F1 (Lin, 2004, R-2/R-L in Table 6.8 and Table 6.9) to compare generated summaries to the references, calculated using the 'jackknifing' method for multiple references as implemented for the GEM benchmark (Gehrmann et al., 2021).

We also evaluate the extent to which the generated summaries are entailed by both the reference summaries and the input reviews using SummaC (Laban et al., 2022), reported as SC_{refs} and SC_{in} respectively. SummaC segments input reviews into sentence units and aggregates NLI scores between pairs of sentences to measure the strength of entailment between the source reviews and generated summary. SC_{in} is the only *reference free* metric we use, and directly measures how well the generated summaries are supported by the input reviews. Since the references for AmaSum were constructed independently from the input reviews, we consider SC_{in} to be our primary metric for AmaSum.

6.5.4 Model Configuration

We use a Transformer architecture (Vaswani et al., 2017) for our encoder $\phi(\mathbf{z}|\mathbf{x})$ and decoder $p(\mathbf{y}|\mathbf{z})$. Token embeddings were initialised from BERT (Devlin et al., 2019)[1]. We set the codebook size $K = 12$, with the number of levels $D = 12$, based on development set performance. Other hyperparameters are given in Appendix C.1. Our code and dataset splits are available at `https://github.com/tomhosking/hercules`.

For SPACE, we generate summaries using 5 generic and 5 specific paths (Section 6.4). For AmaSum, which was constructed from a single verdict sentence followed by more specific pros and cons, we use 1 generic path and 13 specific paths.

[1]We experimented with using BERT as the encoder but found no significant improvement, since the discrete encoding is the main bottleneck in the model.

	System	SPACE				AmaSum			
		R-2 ↑	R-L ↑	SC$_{\text{ins}}$ ↑	SC$_{\text{refs}}$ ↑	R-2 ↑	R-L ↑	SC$_{\text{ins}}$ ↑	SC$_{\text{refs}}$ ↑
Extractive	Rand. Review	6.2	17.1	51.5	26.0	1.0	9.5	59.2	22.4
	k-means	9.5	19.8	72.7	31.0	2.3	12.0	73.9	23.3
	LexRank	5.9	16.4	54.4	22.6	2.7	12.2	67.2	23.5
	QT	10.3	21.5	**93.8**	41.2	1.5	11.4	66.2	22.4
	SemAE	11.1	23.5	65.9	27.9	1.6	11.3	57.2	21.8
	HERCULES$_{\text{ext}}$	**13.2**	**24.4**	89.0	**44.0**	**3.0**	**12.5**	**84.0**	**24.4**
Abstractive	CopyCat	12.1	22.9	77.2	37.3	1.5	11.2	63.1	23.0
	Mistral 7B	5.3$_{\pm 0.1}$	19.6$_{\pm 0.4}$	52.5$_{\pm 1.5}$	24.6$_{\pm 0.5}$	1.9$_{\pm 0.0}$	12.6$_{\pm 0.0}$	50.6$_{\pm 0.2}$	22.0$_{\pm 0.0}$
	Llama 2 7B	4.4$_{\pm 0.2}$	17.6$_{\pm 0.1}$	58.3$_{\pm 1.0}$	24.8$_{\pm 0.3}$	1.5$_{\pm 0.0}$	11.5$_{\pm 0.0}$	58.6$_{\pm 0.1}$	22.6$_{\pm 0.0}$
	Llama 2 13B	5.5$_{\pm 0.3}$	18.7$_{\pm 0.2}$	54.9$_{\pm 1.3}$	24.5$_{\pm 0.2}$	1.6$_{\pm 0.1}$	12.0$_{\pm 0.1}$	51.7$_{\pm 0.3}$	22.1$_{\pm 0.0}$
	BiMeanVAE	13.7	27.1	75.1	36.2	2.0	12.5	52.5	21.8
	COOP	14.2	**27.2**	77.5	39.4	**2.8**	**14.1**	58.3	22.5
	HERCULES$_{\text{abs}}$	**14.8**	**27.2**	**95.1**	**60.2**	2.0	11.8	**82.7**	**25.2**
	(References)	74.4	76.7	61.3	91.3	83.4	85.3	66.2	86.4
	(Oracle)	45.0	53.3	69.3	69.6	14.4	26.0	76.3	26.4

Table 6.8: Results for automatic evaluation of summary generation. R-2 and R-L represent ROUGE-2/L F1 scores. SC$_{\text{refs}}$ and SC$_{\text{in}}$ indicate degree of entailment (measured using SummaC) of generated summaries against reference summaries and input reviews respectively. The best scores for each system type (extractive and abstractive) are bolded. Overall, both variants of HERCULES outperform comparison systems. In particular, summaries generated by HERCULES score highest on SC$_{\text{in}}$, indicating that they most strongly represent the information contained in the input reviews.

	SPACE$_{\text{asp}}$			
System	R-2 ↑	R-L ↑	SC$_{\text{refs}}$ ↑	SC$_{\text{in}}$ ↑
QT$_{\text{asp}}$	10.24	22.64	33.05	**77.32**
AceSum$_{\text{ext}}$	**12.10**	**27.15**	**38.04**	67.48
HERCULES$_{\text{ext}}$	7.93	19.96	26.12	66.64
AceSum	**12.65**	**29.08**	**35.95**	**70.76**
HERCULES$_{\text{abs}}$	10.04	25.35	32.63	70.52
References	–	–	92.86	64.64

Table 6.9: ROUGE scores for controllable summarisation, compared to the aspect-specific summaries in SPACE. Although not specifically designed for aspect-specific summarisation, HERCULES is nonetheless able to generate useful summaries about a specified aspect.

6.6 Results

Automatic Evaluation The results in Table 6.8 show that HERCULES outperforms previous approaches on both datasets. On SPACE, HERCULES$_{\text{abs}}$ achieves the highest ROUGE scores by some distance, and performs very well in terms of the faithfulness metric SC_{ins}.

On AmaSum, HERCULES$_{\text{ext}}$ achieves higher ROUGE scores than HERCULES$_{\text{abs}}$; since the abstractive summaries are generated solely from the encodings, the decoder can sometimes mix up product types with similar descriptions (e.g., headphones and speakers) and is penalised accordingly.

ROUGE scores are very low for all systems on both datasets. The references for SPACE were created by annotators in an extractive manner, and are therefore somewhat unnatural. The references for AmaSum were created independently of the input reviews and may therefore not reflect the balance of opinions conveyed by the reviews. We therefore consider SC$_{\text{in}}$ to be the most informative metric; both variants of HERCULES achieve the highest scores. Surprisingly, a number of the systems achieve SC$_{\text{in}}$ scores higher than the references, indicating that they are generating summaries that are more strongly entailed by the inputs than the gold standard. In Chapter 7 we identify a failure mode of SummaC that accounts for these surprisingly high scores. Some systems that model the summary as a single sequence, like Mistral and COOP, achieve high ROUGE-L scores because they generate very fluent output, but are less informative and less grounded in the input reviews according to SC_{in}.

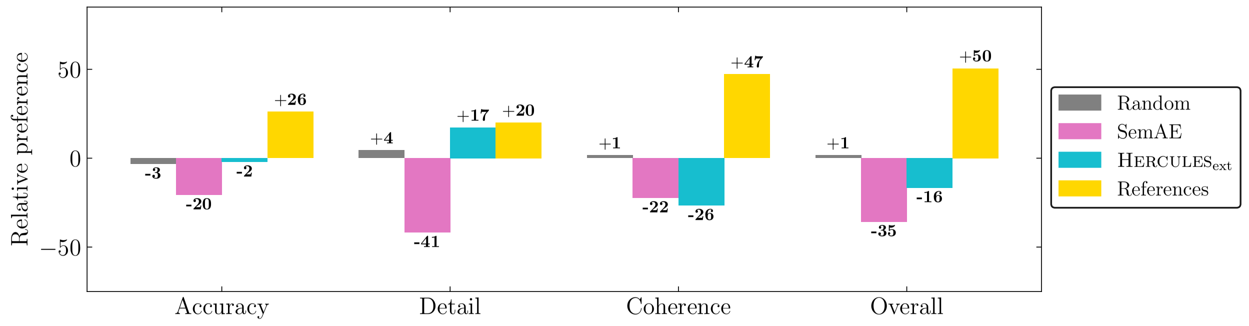
Controllable Generation Although HERCULES was not explicitly designed for controllability, our use of the hierarchical encoding nonetheless enables a degree of control over the output sumamry. We therefore report the results of aspect-specific summarisation on SPACE in Table 6.9 averaged across 'rooms', 'location', 'cleanliness', 'building', 'service' and 'food', with some example output shown in Table 6.17. Despite not being specifically trained or designed to generate aspect-specific summaries, $\text{HERCULES}_{\text{abs}}$ achieves reasonable scores across the range of metrics, and achieves comparable SC_{in} scores to AceSum even though AceSum is trained in a supervised manner while HERCULES is unsupervised. We conclude that HERCULES allows us to control the output of the model and generate summaries which focus on a specific aspect.

Human Evaluation We conduct a human evaluation to evaluate whether HERCULES generates summaries that accurately reflect the opinions in the input reviews. We recruit crowdworkers through Prolific, since AMT is no longer considered reliable by the community. We show participants a set of 50 reviews, followed by two generated summaries. Using an interface based on Potato (Pei et al., 2022), we solicit pairwise preferences along three dimensions, as well as an overall preference:

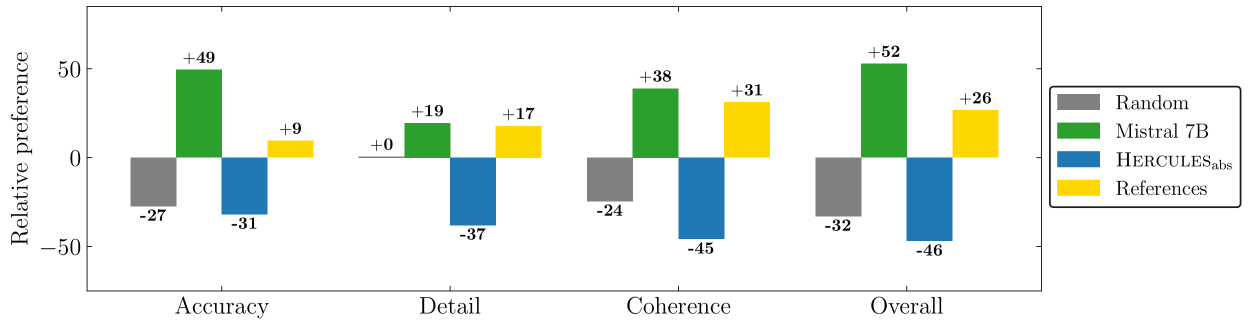
- **Accuracy** — Which summary accurately reflects the balance of opinion in the reviews?
- **Detail** — Which summary includes more specific details?
- **Coherence** — Which summary is easy to read and avoids contradictions?
- **Overall** — Which summary do you think is better, overall?

The full instructions are reproduced in Appendix C.2, and a screenshot of the annotation interface is shown in Appendix C.3. We gather annotations for 10 entities each from the SPACE and AmaSum test sets, with 3 annotations for each. We compare each variant (extractive and abstractive) of HERCULES to the corresponding best performing systems, SemAE and Mistral 7B.

The results in Figure 6.2 show that the extractive variant of HERCULES produces summaries that are considered to be more accurate and detailed than prior systems, although the reference summaries are still preferred. However, the abstractive variant of HERCULES is significantly outperformed by Mistral 7B (using a one-way ANOVA with



(a) Extractive systems



(b) Abstractive systems

Figure 6.2: Results of our human evaluation, for both extractive and abstractive systems. Crowdworkers were asked for pairwise preferences between generated summaries in terms of their Accuracy, Detail, and Coherence & Fluency, as well as an overall preference. Higher scores are better. HERCULES$_{ext}$ generates significantly more detailed summaries than the comparison extractive system, SemAE (using a one-way ANOVA with post-hoc Tukey HSD test,$p<0.05$). However, the reference summaries and LLM (Mistral 7B) are significantly preferred to HERCULES$_{abs}$ along all dimensions.

post-hoc Tukey HSD test, $p<0.05$) along all dimensions. Indeed, the 'random review' baseline performs comparably to the abstractive variant. We note that the training data for Mistral 7B is not publicly available, and it is possible that it was trained on the reviews and summaries in our datasets. A manual inspection of the HERCULES$_{\text{abs}}$ outputs showed that although the generated summaries accurately reflect the balance of opinions in the input reviews, they also tend to be highly generic and with poor discourse structure compared to Mistral and real human-written reviews.

Overall, our results indicate that HERCULES successfully identifies clusters of sentences that are indicative of the overall opinions in the input reviews, leading to a strong extractive system. However, our model uses a comparatively weak decoder that must be trained from scratch, and must generate summary sentences based solely upon the embedding of the selected subpaths. This highly compressed bottleneck limits the ability of HERCULES to generate rich and detailed outputs, and we propose an alternative approach that combines the hierarchical cluster selection method with a strong LLM decoder in Chapter 7.

6.7 Analysis

Attribution Since our approach is attributable and produces evidence sets alongside each abstractive summary sentence, we can evaluate the degree to which the generated sentences are supported by the evidence they cite. Inspired by the approach used in Malon (2023), we use an NLI model (ALBERT, trained on VitC; Lan et al., 2020; Schuster et al., 2021) to determine how many sentences in each cluster either entail or are entailed by the corresponding sentence in the output summary, and take the mean. Considering both forward and backward entailment in this manner accounts for the different levels of granularity between the inputs and summary (Zhang et al., 2024a); input reviews are likely to be more specific than summary sentences, but concise summary sentences are likely to contain multiple assertions, e.g. "The food was good *and* the rooms were clean". HERCULES is the only system that includes evidence alongside generated summaries;[2] the percentage of summary sentences generated by HERCULES$_{\text{abs}}$ that have support from at least *one* sentence in the evidence (partial support) is 85.4%, and from at least *half* the sentences in the evidence (majority support) is 27.6%.

[2]Although extractive systems are inherently attributable, they generally only provide a single source for each summary statement.

	SPACE			AmaSum		
Ablation	R-2 ↑	R-L ↑	SC_{in} ↑	R-2 ↑	R-L ↑	SC_{in} ↑
HERCULES$_{abs}$	14.76	27.22	92.04	2.05	11.77	82.72
No norm loss	-1.32	-1.02	-1.86	-0.01	-0.11	+1.08
No denoising	-1.99	-2.85	-5.34	-0.17	-0.23	-7.75
Generic only	-0.82	-0.59	+3.28	-0.66	-0.70	-14.77
Specific only	-1.49	-3.41	-11.24	-1.15	-2.18	-9.91
VAE + k-means	-2.77	-3.71	-34.14	-1.12	-2.54	-1.70

Table 6.10: Changes in key metrics for a range of ablations of the HERCULES$_{abs}$ model. Removing the components tested leads to a drop in performance.

Ablations To evaluate to the contribution of each component towards the overall performance, we perform a range of ablation studies. Table 6.10 shows the changes in key metrics for models trained without the norm loss and without the denoising objective. We also evaluate summaries generated using only the generic and specific subpath selection methods, rather than a combination of both. Finally, we evaluate the importance of learning the clusters at the same time as the model, rather than post-hoc: we train a model with the same training data and hyperparameters as HERCULES but a *continuous* encoding; use k-means clustering over sentence encodings to identify a set of centroids for each entity; and finally generate a summary by passing the centroids to the decoder. The centroids extracted from a continuous VAE using k-means may not necessarily correspond to a valid sentence, leading to poor quality output. The results show that all components lead to improved summary quality, although different components are more important for different datasets.

Encoding Space Figure 6.3 shows a t-SNE (van der Maaten and Hinton, 2008) plot of the embeddings of all review sentences for a single entity from SPACE, with the summary subpaths overlaid on top in blue. We include a more detailed view of two summary subpaths (left and right panels), showing the increasing level of detail as more levels are specified. We also highlight sample input sentences from the evidence set (circled in red), demonstrating how the generated output can be attributed to input sentences conveying similar opinions.

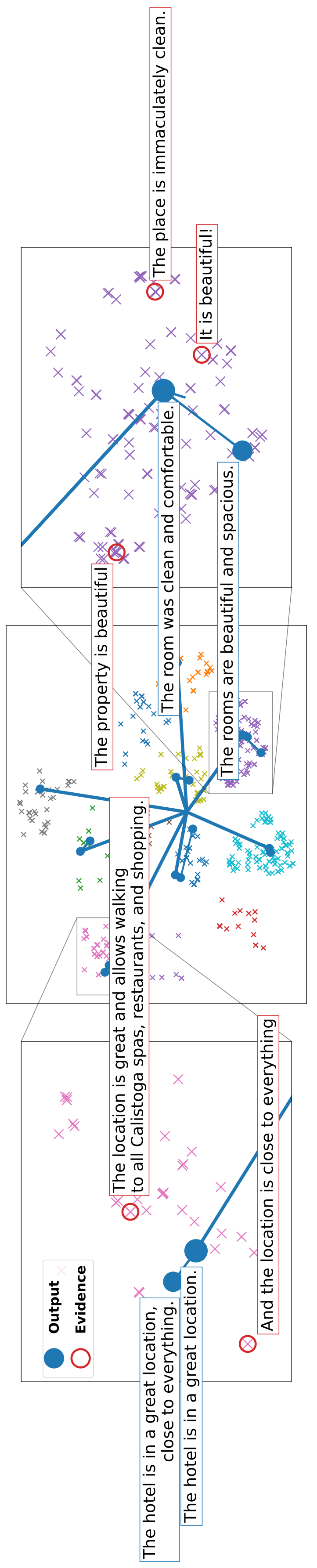


Figure 6.3: A t-SNE (van der Maaten and Hinton, 2008) plot of the embeddings of all review sentences from a single entity from SPACE, where the colour of the points represents the top level code q_1. The summary subpaths are overlaid in blue, alongside output from different hierarchy depths. A selection of evidential inputs are circled in red.

System	**Output**
Reference	The staff were very friendly, spoke fluent English, and helped with our local transportation needs and restaurant recommendations. The entire hotel was very clean, and the rooms and bathrooms were cleaned every day. The room was of good size for Paris and included a balcony. The bathroom was good sized, fully equipped, and private. Breakfast was continental and perfectly adequate. The location is good.
SemAE	We were able to walk to all the tourists sights, except Montmarte and the Eiffel Tower. The hotel staff was very friendly and helpful during our stay. The staff is very friendly and helpful and the breakfast is a perfect start to the day. The rooms was sufficent in size, the bed and pillows were very comfortable. The ladies in the breakfast area were very friendly and courteous. The bathroom was clean.
Mistral 7B	The St. Christophe Hotel is a charming, cozy establishment in a lively Parisian neighborhood. Rooms are small but clean and elegantly furnished, with helpful and pleasant staff. The location is great, with easy access to metro stations and nearby restaurants. Free Wi-Fi is available in the lobby. Some rooms face a noisy street, so request a quieter room if necessary. Overall, a great value for Paris and a lovely place to stay.
HERCULES$_{\text{ext}}$	The room was very small. The staff is very friendly and helpful. It is walking distance to the highlights of the Latin quarter but a few blocks away from the college crowd (a good thing). The rooms were clean. The breakfast was sparse in choices. The location was great, being close to the place Monge Metro station. Breakfast was served in the basement. The bathroom was clean. They spoke English. The cafe across the street was yummy.
HERCULES$_{\text{abs}}$	The room was clean and comfortable. The staff was very friendly and helpful. Walking distance to everything. Breakfast was good. The hotel is in a great location, just a few minutes walk from the train station. Breakfast was fine. The room and bathroom were very clean. The staff spoke English and were very helpful. There is also a small restaurant on the ground floor.

Table 6.11: Examples of generated summaries for the St Christophe Hotel, from SPACE. HERCULES output summaries convey useful information without being overly specific or verbose.

System	**Output**
Reference	The staff were helpful and friendly, especially in the spa! The hotel is consistently clean and well kept and The rooms are luxurious. The rooms have also been upgraded with new white comforters and new ruffle dusters. The hotel also features kitchens so that we can prepare breakfast and eat out in the beautiful garden and lunch at the picnic area so that we can barbeque, but the food in wine country is also wonderful! The location of the hotel is within walking distance to all the shops and restaurants. The grounds are beautiful with the heated indoor/outdoor pools .
SemAE	The resort is within walking distance of the market and many wonderful restaurants. We love to go down after dinner and hang out in the pool, it is delightful. We had a kitchen to cook our own meals, and spent alot of time in the 3 mineral pools. We will definitely come back to Roman Spa for the a stay and spa treatments, and enjoy the lovely environment at the motel.
Mistral 7B	The Roman Spa Hot Springs Resort in Calistoga is a premier destination for relaxation and rejuvenation. With breathtaking grounds, beautiful mineral pools, and elegant, spacious rooms, this resort offers a peaceful retreat from daily life. The central location provides easy access to dining, shopping, and local attractions. Guests rave about the friendly staff, comfortable accommodation, and the therapeutic spa services. The mineral pools are a highlight, offering a relaxing experience in and out throughout the day.
HERCULES$_{\text{ext}}$	The staff was friendly and helpful. The pools well maintained. The rooms are clean and spacious. Thank you Roman Spa for being a great place to stay. It seems that the hotel is improving. The pools and hot tubs are fantastic. The grounds are very well maintained, with beautiful flowers, and waterfalls all throughout. Again, no service seems to be the mantra at the Roman Spa. The grounds were nice. The location is great and allows walking to all Calistoga spas, restaurants, and shopping.
HERCULES$_{\text{abs}}$	The staff was very friendly and helpful. the pool was great. The rooms were clean and comfortable. We had a great time. The hotel is in a great location. the pool and hot tub were great. The grounds are beautiful and well maintained. service was impeccable. The grounds are very nice and well maintained. The hotel is in a great location, close to everything.

Table 6.12: Examples of generated summaries for the Roman Spa Hot Springs, from SPACE.

System	**Output**
Reference	This uniquely shaped vacuum is ideal for sucking debris out of corners and tight spaces. V-shaped design drives debris in front of the vacuum for easier clean-up. Swivel head makes it easy to maneuver around furniture. Power cord extends 20 feet. Great for picking up debris in tight spots with no attachments needed. Runs quietly. Users say this vacuum is prone to clogging and requires routine maintenance
SemAE	Easy to do but taking off the foot, which is where our clogs happen, takes a lot of pushing/maneuvering (I hold the stick and the white tab, then push the foot of the vacuum with my feet). The rubber edge that catches the dirt makes it a bit clumsy to move around but, all in all, this Bissell is helpful. For a fraction of the cost, this does a better job.
Mistral 7B	This lightweight vacuum receives praise for its excellent suction ability, effectively picking up dirt and fine dust. Its long cord and ease of use make it a hit with users, who also appreciate its performance on hard floors such as Pergo and tile. However, some caution against using it on carpet and note that pet hair may clog the bottom cleaner. Overall, it's a satisfying purchase at a reasonable price, especially for those dealing with pet hair. However, some users have reported issues with the vacuum not picking up debris and the bottom cleaner detaching frequently.
HERCULES$_{\text{ext}}$	Works great on my Handscraped, hardwood floors. This vacuum is not made for carpet, but works well on any hard flat surface. Lightweight and easy to maneuver. There are three areas that trap dirt and need to be cleaned in order to empty the vacuum. We bought this little vacuum for our new hardwood floors, and it's the best thing ever. My only complaint is the cord is pretty short. I love this vacuum!! This vacuum is amazing. Love this little vacuum. The suction is great! It picks up pet hair and dirt as advertised. And it picks up dust!! I have 4 cats and 2 dogs.
HERCULES$_{\text{abs}}$	The suction power is great. No scrubbing necessary. This little vacuum is amazing. Easy to maneuver. Great for hard floors. No more dust Bunnies! My dog sheds so much. This vacuum is amazing! Does not stay in place. Excellent customer service. The cord is too short. Love this vacuum! Great for pet hair.

Table 6.13: Examples of generated summaries for the Bissell PowerEdge Vacuum from AmaSum.

System	**Output**
Reference	A gaming-specific external hard drive designed for the whole range of Xbox consoles. Two terabyte options (2TB and 4TB) offer plenty of space for installing games, apps, and files. Can be used with multiple Xbox consoles. The noise level of the hard drive is louder than most other options
SemAE	It had about 1.8 TB for use after installation and the updates. It stop working after 2 years of having it. Never had a problem after a few months of ownership. I will be purchasing another one of these hard drives to replace the one that stopped working. It does not need to be purchased separately. Failed after 2.5 years of use for an xbox. Bought this for my husband.
Mistral 7B	These reviews express satisfaction with the product's fast performance, timely delivery, compatibility with Xbox, and ease of use. Some customers were particularly pleased with its design and their children's enjoyment of it. A few mentioned replacing a previous failed drive. Overall, the product received positive feedback.
HERCULES$_{\text{ext}}$	Very noticeably speeds up loading times for gaming on my Xbox. Tons of storage now for my Xbox one. No issues at all. All Seagate though. Used for Xbox one. It ' s just been sitting on my TV stand connected to me Xbox one X. Plug and play. So we ended up buying an external drive. Super easy to install. I have about 50 games installed on this hard drive and still have 75% of space left. Love this Ssd! Plenty of space for extra games. Stopped working after 2 years. This hard drive is awesome.
HERCULES$_{\text{abs}}$	Stopped working in less than a year. Easy to install and use. Works great with my Macbook pro. Plug and play. Plenty of room. This one does. Fast load times. Great for gaming. This hard drive is very fast. No SD card reader. This external hard drive is great. This Ssd is fast. Bought this for my bedroom. Great price and fast shipping.

Table 6.14: Examples of generated summaries for the Seagate 2TB Game Drive from AmaSum.

Output	Breakfast was good.
Evidence	Breakfast was very good for us Breakfast offers a variety of things to eat. The buffet breakfast is varied and satisfying The buffet breakfast was all fresh food with a good choice Breakfast was good.
Output	Great camera for the price.
Evidence	I like the camera. Overall a great camera at a good price. I like the range of the lens. Great camera. This is a good camera for the money.

Table 6.15: Examples of evidence sets produced by HERCULES. Each output sentence generated by the model is attached to a set of input sentences that share the same subpath. Although all sentences selected as evidence share the same broad topic and sentiment, they do not all directly entail the output, highlighting the difficulty with using NLI models to automatically evaluate attribution.

Example Output Tables 6.11 to 6.14 show example summaries generated by HERCULES. They cover a wide range of aspects (rooms, service, location, food, etc.), conveying useful information without being overly specific or verbose. However, the content of the summaries is also fairly generic; the sentences tend to be quite short and lack detail. By contrast, the references and Mistral 7B summaries include more specific details and have a more clear overall structure.

Table 6.15 shows examples of evidence sets, illustrating how HERCULES is able to generate output that retains key information from the inputs, while discarding unnecessary detail. They also demonstrate show how output statements can have majority (top) and partial (bottom) support. Although only two out of five sentences in the bottom directly support the generated output "Great camera for the price", we note that all members of the evidence set are aligned in terms of sentiment. Measuring attributability via entailment is an overly restrictive approach and likely to result in a conservative estimate of how well the evidence supports the summaries.

Table 6.16 shows a breakdown of generated output at different granularities. Given the input sentence, we show the output of the decoder with subpaths of varying granu-

Input	The staff was very helpful; the free breakfast was the best we had on this trip.
Output $(d = 1)$	Breakfast was good.
Output $(d = 2)$	The continental breakfast was a joke.
Output $(d = 3)$	The breakfast was one of the best I have ever had.
Output $(d = 4)$	The breakfast was one of the best I've had in a hotel.
Cluster $(d = 3)$	Continental breakfast was the BEST so far on our trip!!! The staff was very helpful; the free breakfast was the best we had on this trip. The Cafe has the among the best breakfast and lunch in Vegas (closed for dinner).

Table 6.16: An example of how our model encodes sentences at different granularities. As more levels are used, the output increasingly converges towards the meaning expressed by the input. We also show other input sentences that are assigned the same subpath (of depth = 3); despite very different phrasing, they convey a common opinion.

larities, demonstrating how subpaths of increasing depth lead to more detailed output. The output for $d = 2$ conveys the opposite sentiment to the other outputs; since the hierarchy was trained on real review sentences, it is strongly biased towards generating plausible output and may therefore hallucinate by specifying a greater degree of detail than is specified by a given encoding. It may be that the inputs that get mapped to this path up to level $d = 1$ and 2 are broadly balanced in terms of sentiment, but the decoder is unable to reflect this ambiguity.

Table 6.17 shows examples of aspect-specific summaries generated by HERCULES$_{\text{abs}}$, for the same entity. Each generated summary focusses primarily on the desired aspect, although not perfectly.

Table 6.18 shows an example of generated summaries with sentiment control; by constraining generation to paths that are associated with particular ratings, we can generate summaries that are skewed towards positive or negative sentiment.

Failure Modes and Limitations HERCULES is trained to reconstruct a target sentence from a source retrieved using tf-idf, but tf-idf is not sensitive to negation and does not distinguish between syntax and semantics. We observe that the model sometimes

Aspect	**Output**
Rooms	The room was very small. We had a room facing the street. The room was dark and dingy. The room and bathroom were very clean.
Food	The coffee was undrinkable. The breakfast was a bit disappointing. There is also a small restaurant on the ground floor. Breakfast is served in the basement.
Location	The hotel is in a great location, just a few minutes walk from the train station. The hotel is very basic. There is also a small restaurant on the ground floor. The location is very convenient.

Table 6.17: Aspect-specific summaries from HERCULES$_{\text{abs}}$ convey information specific to the desired topic.

Rating = 1 (bad)	Then it stopped working. It died in less than a year. Do not buy this machine. It didn't even last a year. Bought this in January 2017.
Rating = 5 (good)	This is a great fan. It's very quiet. I love this fan. The light is bright. This is a very nice remote.
Rating = 1 (bad)	The carpet was stained and dirty. The room was filthy. The bathroom was disgusting. The staff was unfriendly and unhelpful. Avoid this hotel at all costs.
Rating = 5 (good)	The hotel is very close to the airport. The shuttle service was great. The pool and hot tub were great. The food was delicious. The view from our room was breathtaking.

Table 6.18: Examples of sentiment-controlled summaries generated by HERCULES, from SPACE and AmaSum.

clusters sentences with superficially similar surface forms but different meanings. For example, “The breakfast buffet was very good” and “The breakfast buffet was not very good either” are assigned to the same path by our model.

The model is trained to generate output sentences based solely on the latent encoding: this is required to ensure that the model learns a useful encoding space. However, it also makes the model susceptible to some types of hallucination. Sentences about similar topics are likely to be assigned to the same paths, so the model may generate output that mentions a different entity of similar type (e.g., headphones instead of speakers).

Since our approach identifies common opinions based on frequency of sentence encodings, we require a relatively large number of input sentences. We were not able to experiment with other popular datasets like Amazon (He and McAuley, 2016), Yelp (Chu and Liu, 2019) or Rotten Tomatoes (Wang and Ling, 2016) since these datasets only include a small number (usually 8) of input reviews.

Our method has the potential to be applied to other types of summarisation or tasks involving aggregation. However, the requirement for high redundancy in the input makes this challenging. For example, news summarisation generally involves a much smaller number of documents than opinion summarisation, as well as a wider domain of topics that much be encoded, and would require a cleaner organisation of information within the hierarchy.

The abstractive summaries are generated solely based on the latent encoding, and our model does not include a copy mechanism or attend to the original inputs when decoding. It therefore does not always generalize well to new domains. However, this limitation is mitigated by not requiring any labelled data during training: HERCULES can easily be retrained on a new domain.

Generating output based only on latent encodings means that the model is also susceptible to hallucinating, since the output is less directly linked to the inputs. However, unlike other methods, HERCULES provides evidence sets alongside the generated summaries, making it easier to check whether the output is faithful.

Finally, HERCULES generates summary sentences independently, leading to summaries that are less coherent than approaches that model the summary as a single sequence.

6.8 Summary

In this chapter, we propose HERCULES, a method for aggregating user reviews into textual summaries by identifying frequent opinions in a discrete latent space. Our approach generates summaries that are more informative than comparison systems, while also scaling to large numbers of input reviews and providing evidence to justify its output.

HERCULES demonstrates that HRQ-VAE can be successfully used for other text-to-text generation tasks than paraphrase generation. By using a discrete hierarchical representation for sentences from reviews, we are able to identify popular opinions by comparing the occurrence frequencies of different parts of the hierarchy. This chapter acts as a third piece of evidence in support of our hypothesis that it is beneficial to choose weakly structured representations (Hypothesis I). We used a denoising autoencoder objective to encourage the model to group the representations to semantically related sentences together in the hierarchy, supporting our hypothesis that distant supervision may be used to assign meaning to a structured representation (Hypothesis III). The method is also attributable and scales easily to large numbers of input reviews, properties which are direct results of the choice of representation and support our hypothesis that discrete and hierarchical representations help make text-to-test problems feasible (Hypothesis II).

However, the method is not without its drawbacks. The discrete bottleneck must group together topically related sentences so that we can identify popular opinions, but it must also contain sufficiently detailed information about the input sentence that the decoder is able to generate a meaningful reconstruction. These objectives are in opposition with each other, and so the model is prone to generating overly generic output, and often generates output with incorrect locations or product names. Our human evaluation showed that there is significant room for improvement. Our results also show that, although they use orders of magnitude more parameters and so incur significant training and inference costs, general purpose LLMs can outperform specialised models. In the next chapter, we investigate whether we can achieve the best of both worlds by combining the tractability of weakly structured representations with the fluency of LLMs.

Chapter 7

The LLM Era: Opinion Summarisation with Hierarchical Indexing

In Chapter 6 we showed how HRQ-VAE can be used to perform opinion summarisation, with the discrete hierarchical representation enabling us to generate summaries that are attributable and scale to large numbers of input reviews. However, the discrete bottleneck used for HERCULES must perform two functions: it must group together topically related sentences so that we can identify popular opinions; and, it must also contain sufficiently detailed information about the input sentence that the decoder is able to generate a meaningful reconstruction. These objectives are in opposition with each other, and so HERCULES is prone to generating overly generic output and often hallucinates.

Throughout the thesis so far, we have found that LLMs are able to generate highly fluent output, albeit at the cost of large parameter counts, expensive training and inference, and limited context window lengths. In this chapter, we seek to combine the strengths of generalised LLMs with specialised models that use structured representations, by performing *content selection* in a discrete hierarchical space and using a LLM for the 'last mile' generation task.

We propose a method for unsupervised abstractive opinion summarisation, that combines the attributability and scalability of extractive approaches with the coherence and fluency of Large Language Models (LLMs). Our method, HIRO, learns an index structure that maps sentences to a path through a semantically organised discrete hierarchy. At inference time, we populate the index and use it to identify and retrieve clusters of sentences containing popular opinions from input reviews. Then, we use a pretrained LLM to generate a readable summary that is grounded in these extracted evidential clusters. Compared to previous chapters which used a denoising autoencoder objective, we use a contrastive training objective to explicitly assign the desired meaning to the encoding space, supporting Hypothesis III. The modularity of our approach allows us to evaluate its efficacy at each stage. We show that HIRO learns an encoding space that is more semantically structured than prior work (Hypothesis I), and generates summaries that are more representative of the opinions in the input reviews. The discrete hierarchical nature of the index leads directly to the scalability and attributability of the approach, supporting Hypothesis II. Human evaluation confirms that HIRO generates significantly more coherent, detailed and accurate summaries.

7.1 Introduction

Online review websites are a useful resource when choosing which hotel to visit or which product to buy, but it is impractical for a user to read hundreds of reviews. Automatic opinion summarisation aims to aggregate a large and diverse set of customer reviews about a particular *entity* into a single, easy to read summary. A good summary should accurately reflect the balance of opinions in the input reviews, highlighting the most common or *popular* opinions, while omitting unnecessary details. A useful summary should also help *compare* between competing options, and include points that differentiate the current entity from others.

Some prior work on *abstractive* opinion summarisation has almost exclusively either required costly supervision (Bražinskas et al., 2021; Cattan et al., 2023) or has assumed that the number of input reviews is limited (Coavoux et al., 2019; Bražinskas et al., 2020; Amplayo et al., 2021a,b; Iso et al., 2021). This defeats the point of a summary: a user could feasibly read 8 reviews in a reasonable period of time. A good summarisation system should be *scalable*, since popular products online may receive thousands of reviews. It should also be *attributable*, offering some evidence to justify its output. Paraphrasing Rashkin et al. (2023), we say that a statement s is attributable to some evidence E, if a generic reader would agree that 'According to E, s is true'. Finally, it should generate summaries that are *coherent* and *faithful* to the input reviews.

Large Language Models (LLMs) have been shown to generate highly fluent summaries in the news domain (Bhaskar et al., 2023) but are a flawed solution because current instruction-tuned models are not attributable, and because their context windows limit the number of reviews they are able to base their summaries on. Models with long context windows have been proposed (Beltagy et al., 2020; Gu et al., 2022) but these are not currently instruction-tuned, and it has been shown that LLMs are biased toward information at the start and end of the input (Liu et al., 2023).

Our approach, **H**ierarchical **I**ndexing for **R**etrieval-Augmented **O**pinion Summarization (HIRO), identifies informative sentences using hierarchical indexing and then passes the selected sentences as input to a LLM, similar to retrieval-augmented generation (RAG, Lewis et al., 2020). By separating the steps of content selection and generation, we can combine the attributability and scalability of the discrete representation with the strong generative abilities of LLMs, leading both to a higher quality index and to more informative and coherent output summaries.

HIRO consists of three modules, allowing for increased control, flexibility and interpretability. The **Hierarchical Indexer** is an encoder that maps sentences from reviews to paths through a hierarchical discrete latent space. The **Retriever** uses the index to identify clusters of sentences for each entity that contain popular and informative opinions. These sentence clusters are passed to a **Generator**, a pretrained LLM, that generates coherent summaries that are grounded in the retrieved sentences.

Our contributions are as follows:

- We propose a method for learning an encoder that maps sentences to a path through a semantically structured discrete hierarchy.
- We show how to exploit this discrete hierarchy at inference time to identify clusters of related and prevalent sentences from input reviews.
- We introduce an automatic metric that measures whether generated summaries reflect the input reviews, while penalizing overly generic statements.
- Through extensive experiments on two English datasets from different product domains, we demonstrate that passing these retrieved sentences in a zero-shot manner to a pretrained LLM generates summaries that better reflect the distribution of opinions within the input reviews. Human evaluation shows that summaries generated by HIRO are significantly more coherent and accurate than prior work, and are preferred by annotators.

Our code and dataset splits are available at `https://github.com/tomhosking/hiro`.

7.2 Related Work

We refer to Section 6.2 for a description of previous work on opinion summarisation. We describe here some additional work relevant to the specific approaches used in this chapter.

Content Selection The idea of first selecting relevant parts of an input before generating output has been well studied (Kedzie et al., 2018; Puduppully et al., 2019; Amplayo et al., 2021b; Narayan et al., 2023, *inter alia*), and has been shown to be very effective when used in conjunction with LLMs in the form of retrieval-augmented generation (RAG, Lewis et al., 2020). Xu et al. (2023) find that retrieval augmentation is beneficial even when using models that can accept long inputs. Wang et al. (2023) show that including an additional filtering or selection step to RAG is better than naively passing all retrieved documents as input.

Evaluation of Summaries Automatic evaluation of generated summaries is extremely challenging. Prior work has shown that ROUGE (Lin, 2004) scores correlate poorly with human assessments of summary quality (Callison-Burch et al., 2006; Tay et al., 2019; Fabbri et al., 2021; Shen and Wan, 2023; Clark et al., 2023; Aharoni et al., 2023). Some datasets are created automatically, with references that are not directly based on the input reviews (Bražinskas et al., 2021). Modern summarisation system outputs are now often preferred to human-written references (Goyal et al., 2022; Bhaskar et al., 2023).

SummaC (Laban et al., 2022) is a *reference-free* metric that uses an NLI model to evaluate the degree of support between a summary and the input documents, but it overly rewards trivial statements; using the obvious statement "The hotel was a building" as a summary for every entity achieves a near-perfect SummaC score of 99.9% on SPACE, a dataset of hotel reviews (Angelidis et al., 2021).

Malon (2023) propose a metric that uses a NLI model to evaluate *prevalence*, i.e. how many input reviews contain supporting evidence for each sentence in a summary, and explicitly penalizes trivial or redundant output. However, we found it has a similar failure mode to SummaC, with the statement "The rooms are clean and comfortable" achieving a prevalence score of 72% on SPACE. We propose a modification to prevalence in Section 7.6.2 that penalizes overly generic summaries.

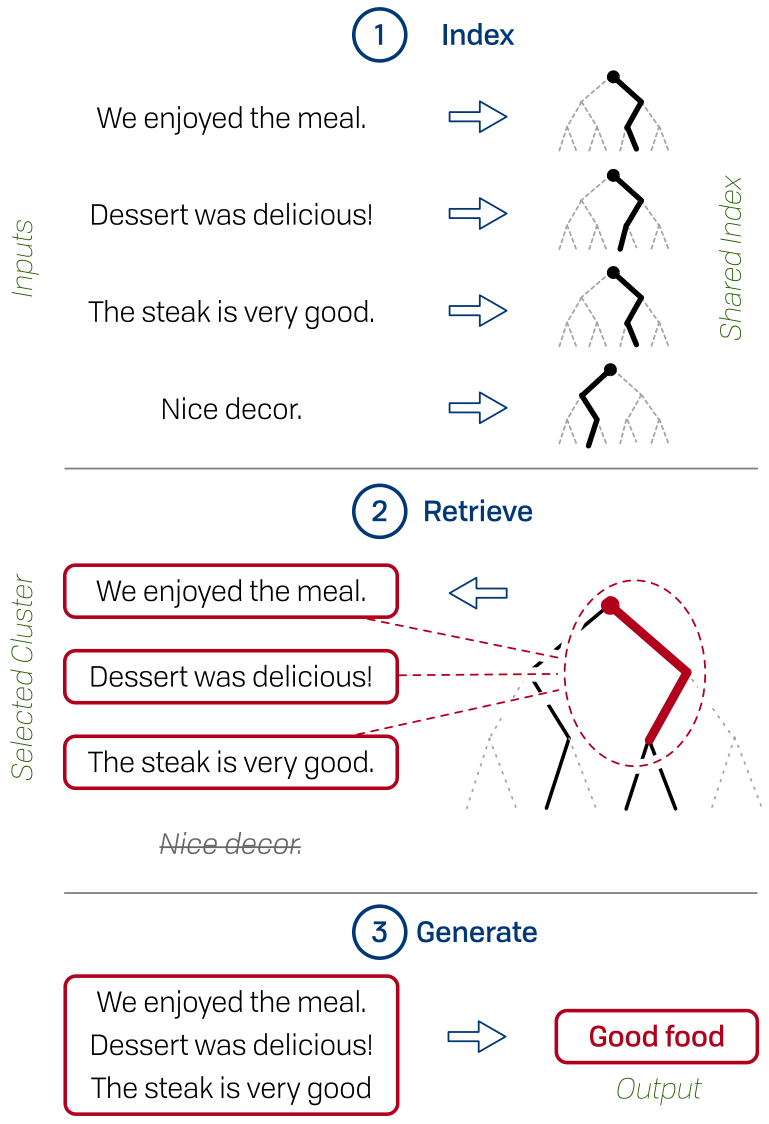


Figure 7.1: HIRO uses three modules to generate summaries of customer reviews. First, we use our encoder to **index** all sentences from input review into a learned hierarchy. Then we identify paths within this index that occur frequently, and **retrieve** the corresponding clusters of sentences. Finally, we pass these clusters to an LLM to **generate** an output summary.

7.3 Overview

Let $\mathcal{R}_e$ be a set of reviews about an entity $e \in \mathcal{E}$, where each review $\mathbf{R} \in \mathcal{R}_e$ is composed of a number of sentences $\mathbf{x}$. The goal is to generate a textual summary that includes the most informative opinions from $\mathcal{R}_e$, while abstracting away the details specific to any one review.

HIRO generates summaries following a modular approach, depicted in Figure 7.1. We learn an index structure that maps each sentence $\mathbf{x}$ from the input reviews to a path $q_{1:D}$ through a discrete hierarchy. We choose a hierarchical representation so that sentences are grouped at a useful level of abstraction; the upper levels of the hierarchy should partition sentences by topic, while the lowest levels should group together sentences with equivalent meaning.

At inference time, we encode all sentences from the reviews, then identify the paths or *subpaths* $q_{1:d}$ within the index that are particularly *popular*, and retrieve the corresponding sentences. This 'retrieval' process is query-free, instead relying on properties of the hierarchical index to determine the frequency of different opinions. By indexing sentences hierarchically according to their semantics, we can easily identify opinions or topics that occur frequently by simply counting their occurrence in the index.

Finally, we generate a summary by passing the selected clusters of sentences as input to a LLM. This '*retrieval-augmented*' usage of LLMs allows us to benefit from the fluency and coherence of LLMs, while retaining the attributability and scalability of extractive opinion summarisation methods.

We now detail the three modules that comprise HIRO, evaluating each of them in turn to confirm their efficacy compared to previous methods.

7.4 Learning a Hierarchical Index Structure

Our goal is to learn an encoding space where sentences with similar meanings are grouped together. The space should be *discretised* so that frequent opinions can be easily identified by counting the membership of each part of the index, and it should be *hierarchical* so that opinions may be aggregated at an appropriate level of granularity, rather than by details or phrasings specific to a particular review. Finally, the encoding space should be structured semantically, to enable accurate aggregation of opinions; sentences with equivalent meaning should clearly be indexed to the same point in the hierarchy, while sentences that are topically related but not equivalent should be grouped together at a higher level.

We base our encoder on HRQ-VAE Chapter 4. HRQ-VAE uses an encoder-decoder architecture with a discrete hierarchical bottleneck to generate summaries. It is trained as a denoising autoencoder, and therefore needs to learn a representation that is both compressed enough to enable aggregation, but also expressive enough for the decoder to be able to generate meaningful output. These factors are in direct competition, with the compressed bottleneck leading to output that is generic and contains hallucinations. By contrast, HIRO uses an external LLM as the 'decoder', allowing us to focus solely on learning a representation that is useful for identifying informative opinions.

7.4.1 Method

The HIRO encoder module maps a single sentence $\mathbf{x}$ to a path $q_{1:D}$ through a discrete hierarchy, using the residual vector quantisation technique introduced in Chapter 4 (Chen et al., 2010; Zeghidour et al., 2022; Hosking et al., 2023b).

First, we use a Transformer encoder followed by attention pooling (Vaswani et al., 2017; Liu and Lapata, 2019) to map a sequence of tokens $\mathbf{x}$ to a single dense embedding $\mathbf{z} \in \mathbb{R}^{\mathbb{D}}$. Then, we decompose $\mathbf{z}$ into a path through a latent discrete hierarchy $q_{1:D}$, where $q_d \in \{1, \dots, K\}$ are discrete 'codes' at each level d. Briefly, we induce a distribution over codes at each level $p(q_d)$, parameterised by a softmax with scores s_d given by the Euclidean distance from learned codebook embeddings to the residual error between the input and the cumulative embedding from all previous levels,

$$s_d(q) = -\left(\left[\mathbf{z} - \sum_{d'=1}^{d-1} \mathbf{C}_{d'}(q_{d'})\right] - \mathbf{C}_d(q)\right)^2, \tag{7.1}$$

where $\mathbf{C}_d \in \mathbb{R}^{K\times\mathbb{D}}$ is a codebook which maps each discrete code to a continuous embedding $\mathbf{C}_d(q_d) \in \mathbb{R}^{\mathbb{D}}$. During training, we use the Gumbel reparameterisation (Jang et al., 2017; Maddison et al., 2017; Sønderby et al., 2017) to sample from the distribution $p(q_d)$. During inference, we set $q_d = \arg\max s_d$.

Since our goal is to learn a representation where semantically similar sentences are grouped together, we use a training objective that explicitly induces this arrangement in encoding space. We train the encoder with a contrastive learning objective, bringing representations of semantically similar sentences (i.e., positive pairs) together, and pushing dissimilar ones apart.

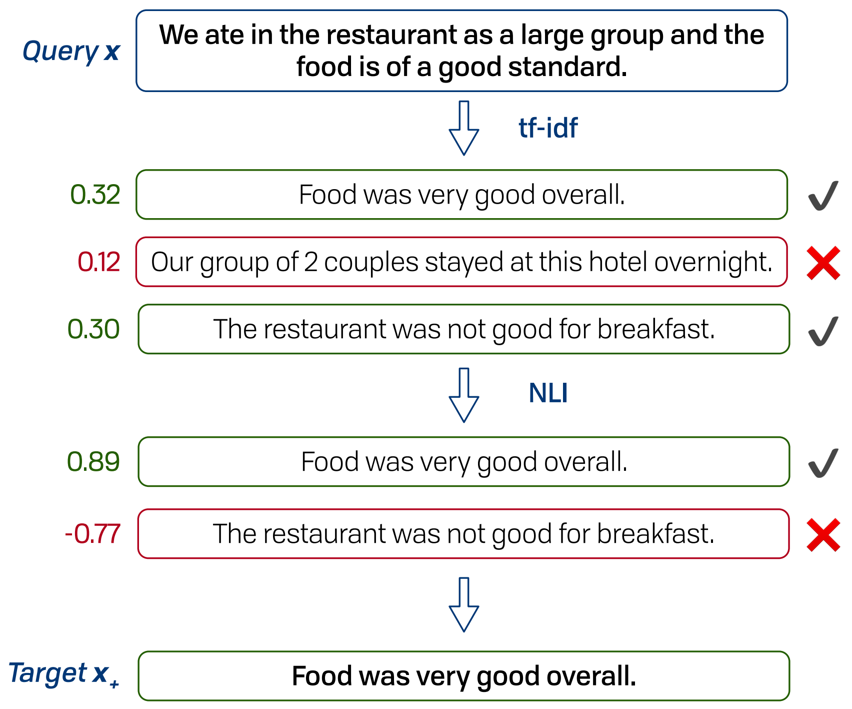

Query x
We ate in the restaurant as a large group and the food is of a good standard.
tf-idf
0.32
Food was very good overall.
✓
0.12
Our group of 2 couples stayed at this hotel overnight.
✗
0.30
The restaurant was not good for breakfast.
✓
NLI
0.89
Food was very good overall.
✓
-0.77
The restaurant was not good for breakfast.
✗
Target x+
Food was very good overall.

Figure 7.2: An example of the process for constructing the positive pairs used to train our model. Given a query sentence, we first use tf-idf to identify possible candidates from the training data, keeping only those sentences with similarity over a specified threshold. Then we check for entailment using an NLI model, and use any sentences labelled as 'entailed' as positive targets.

For each sentence in the training data, we construct positive pairs of semantically related sentences $\mathbf{x}, \mathbf{x}_+$ as follows: given a random 'query' sentence $\mathbf{x}$ from the training data, we identify possible candidate 'targets' based on tf-idf similarity; then, we check whether each candidate is entailed by the query using an NLI model (DeBERTa v3, trained on Debiased NLI; He et al., 2021; Wu et al., 2022), and use the sentences which are labelled as entailed as positive targets $\mathbf{x}_+$. An example is shown in Figure 7.2. We do not use any 'hard' negatives; instead, during training we calculate the pairwise tf-idf similarity between all samples in a batch, and include only those samples with similarity below a threshold as negatives $\mathbf{X}_-$. We set this threshold to 0.3 based on validation set performance. This prevents 'false negatives' being wrongly forced apart, and allows sentences that are topically related but not strictly equivalent to remain in the same high-level grouping within the index.

We found that it was crucial to include sufficient exploration in the process of constructing positive pairs. The candidate sentences retrieved using tf-idf should be sufficiently similar that we are likely to find ones that are entailed, but not so similar that they only have minor lexical differences.

We use a modified version of the InfoNCE training objective (van den Oord et al., 2018) designed to bring the representations of a query $\mathbf{x}$ closer to those of a positive target $\mathbf{x}_+$, while pushing them apart from negative samples $\mathbf{X}_-$,

$$\begin{aligned}\mathcal{L} =& - \rho(\mathbf{x}, \mathbf{x}_+) \log f, \qquad (7.2)\\ f =& \frac{\exp\big(s(\mathbf{x}, \mathbf{x}_+)\big)}{\exp\big(s(\mathbf{x}, \mathbf{x}_+)\big) + \frac{\omega}{|\mathbf{X}_-|} \sum\limits_{\mathbf{x}_- \in \mathbf{X}_-} \exp\big(s(\mathbf{x}, \mathbf{x}_-)\big)},\end{aligned}$$

where $\rho(\mathbf{x}, \mathbf{x}_+)$ is the tf-idf similarity between $\mathbf{x}$ and $\mathbf{x}_+$ that weights the *confidence* of the positive pairs, inspired by MarginMSE (Hofstätter et al., 2021), and ω is a constant that controls the strength of the negative examples. The similarity function $s(\cdot, \cdot)$ is given by the mean dot product between the embedding of all subpaths $q_{1:d}$ for $d \leq D$,

$$s(\mathbf{x}, \mathbf{x}') = \frac{1}{D} \sum_{d=1}^{D} \max\big(\mathbf{C}(q_{1:d})^T \mathbf{C}(q'_{1:d}), 0\big), \qquad (7.3)$$

where $\mathbf{C}(q_{1:d}) = \sum_d \mathbf{C}_d(q_d)$ is the full embedding of path $q_{1:d}$. Intuitively, this brings together the representations of the positive pairs at each level in the hierarchy, while penalizing any overlap with the representations of negative examples.

The similarity is floored at zero, to prevent the model from being able to ‘game’ the loss by pushing negative examples further and further apart. Although the positive pairs are selected based on a directional entailment label, we do not exploit this directionality in our training objective.

We employ the techniques proposed in Chapter 4 to induce a hierarchical encoding space proposed, including depth dropout, initialization decay, and norm loss. We additionally include the entropy of the distribution over codes, $-\sum_{d,q_d} \log\big(p(q_d)\big)$, as an additional term in the objective, to encourage exploration of the latent space during training. Although HIRO is not a true generative model, this term is analagous to the KL term found in the ELBO.

7.4.2 Evaluation

We now evaluate whether the combination of discrete hierarchical encoding and contrastive objective leads to a more semantically distributed representation than previous methods.

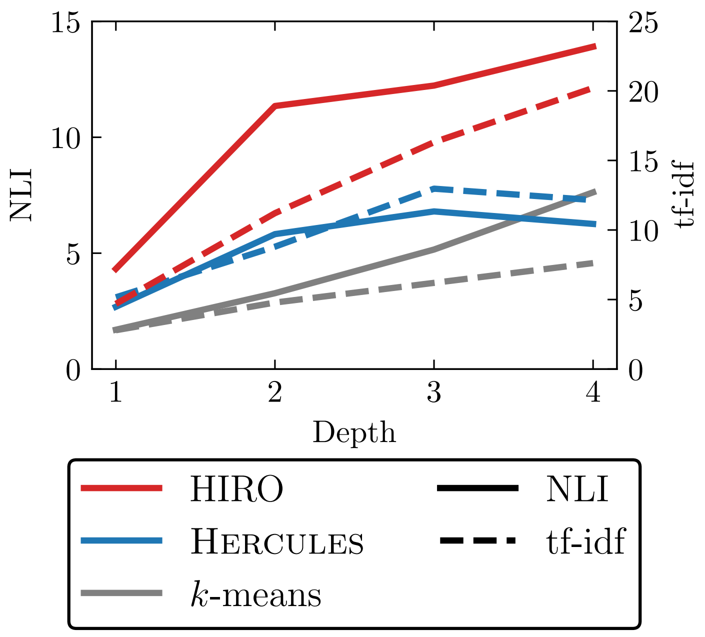


Figure 7.3: Cluster quality by depth, as measured by the difference between cluster purity and colocation for the SPACE test set, according to NLI (solid line) and tf-idf (dashed line) similarity measures. HIRO learns a higher quality encoding space than comparison methods.

Experimental Setup We experiment using SPACE (Angelidis et al., 2021), which consists of hotel reviews from TripAdvisor with 100 reviews per entity, as well as reference summaries created by annotators. We encode all the review sentences from the SPACE test set, then calculate both the *purity* (the mean intra-cluster similarity) and *colocation* (the mean inter-cluster similarity) for the clusters of sentences assigned to each subpath $q_{1:d}$ for $d \leq 4$. Finally, we take the difference between the purity and colocation as an overall measure of the quality of the representation space.

We compare to HERCULES (Chapter 6), which trains a hierarchical encoder jointly with a decoder, and to a baseline using recursive k-means over embeddings from a pretrained embeddings model (MiniLM, Reimers and Gurevych, 2019). We apply k-means to the embeddings, then calculate the residual errors between cluster centroids and embeddings, apply k-means to those errors, and repeat. All methods use the same number of clusters at each level ($k = 12$).

Model Configuration We use a 6 layer Transformer, with token embeddings initialised from BERT base (Devlin et al., 2019).[1] We set the codebook size $K = 12$, with the number of levels $D = 12$, based on validation set performance. Other hyperparameters are given in Appendix D.1.

HIRO learns a higher quality encoding space Figure 7.3 shows the overall quality (using both NLI and tf-idf measures of similarity), indicating that HIRO learns a more semantically distributed space at all levels of depth than comparison approaches, according to both similarity measures. The separation between clusters increases with depth, confirming that the encoder learns a semantic hierarchy. We believe these results indicate that our method could potentially be used for more general purpose document retrieval (similar to Li et al., 2023), which is beyond the scope of this paper.

7.5 Retrieving Popular Opinions

Recall that a good summary should include opinions that occur repeatedly in the input reviews, as well as opinions that differentiate the current entity from others. In Section 7.4 we showed how the HIRO encoder maps a single sentence $\mathbf{x}$ to a path $q_{1:D}$. We now exploit the discrete, hierarchical nature of the representation space to index a large number of review sentences, then identify informative sentences to use to generate a summary. We hypothesize that our content selection method leads to clusters that better represent the balance of opinions in the input reviews.

7.5.1 Method

For each review $\mathbf{R} \in \mathcal{R}_e$ about an entity $e \in \mathcal{E}$, we separately encode each sentence within the review to its path $q_{1:D}$, giving an indexed review $\mathbf{Q}(\mathbf{R})$.

Our content selection method identifies the parts of the hierarchy that are particularly popular for each entity, and extracts the corresponding clusters of sentences. This process is query-free; instead, we assign each subpath in the hierarchy $q_{1:d}$ a score based on its popularity within the indexed input reviews, and 'retrieve' all sentences that were mapped to the k highest-scoring subpaths.

Our scoring function is inspired by tf-idf (Jones, 1972), which is designed to measure the importance of a particular term with respect to a set of baseline documents. We

[1]We experimented with using BERT as the encoder but found no significant improvement, since the discrete encoding is the main bottleneck in the model.

define the *term popularity* $\mathrm{tp}(q_{1:d}, e)$ of a path as the fraction of indexed reviews for entity e which contain the subpath $q_{1:d}$,

$$\mathrm{tp}(q_{1:d}, e) = \frac{1}{|\mathcal{R}_e|} \sum_{\mathbf{R} \in \mathcal{R}_e} \mathbb{I}\big(q_{1:d} \in \mathbf{Q}(\mathbf{R})\big), \tag{7.4}$$

where $\mathbb{I}$ is the indicator function. We define the *inverse baseline popularity* ibp as the reciprocal of the mean term popularity across *all entities* $\mathcal{E}$,

$$\mathrm{ibp}(q_{1:d}) = \left(\frac{\alpha + \sum_{e \in \mathcal{E}} \mathrm{tp}(q_{1:d}, e)}{\alpha + |\mathcal{E}|} \right)^{-1}, \tag{7.5}$$

where the smoothing constant α allows us to balance between absolute and comparative popularity. The overall score is then

$$\mathrm{score}(q_{1:d}, e) = \mathrm{tp}(q_{1:d}, e) \times \mathrm{ibp}(q_{1:d}). \tag{7.6}$$

Intuitively, the score represents the relative popularity within the current entity of a path $q_{1:d}$ compared to all entities in the dataset.

Our scoring scheme conveniently accounts for the fact that short paths are more common;[2] short paths will also have a higher baseline popularity, leading to an overall score that is effectively normalised for the depth of the subpath.

After retrieving the clusters of sentences corresponding to the top-k highest scoring subpaths, we filter out sentences with very low lexical similarity to other cluster members, and combine any clusters of sentences with high lexical overlap.

7.5.2 Evaluation

We evaluate whether our method successfully selects clusters of sentences containing informative opinions, using reviews from SPACE, as in Section 7.4.2. First, we measure the overlap between retrieved clusters and *oracle clusters* constructed by selecting sentences with high overlap to the reference summaries,[3] using the Adjusted Rand Index (ARI, Hubert and Arabie, 1985). Second, we evaluate the average *prevalence* (Malon, 2023) of sentences in retrieved clusters, which uses a NLI model (ALBERT, trained on VitC; Lan et al., 2020; Schuster et al., 2021) to evaluate how many input reviews support each retrieved sentence.

We compare HIRO to the clusters extracted by HERCULES (Hosking et al., 2023b), and the oracle upper bound. As a baseline, we apply k-means to MiniLM embeddings

[2]The presence of a path $q_{1:D}$ for a review also implies the presence of all subpaths $q_{1:d}, d < D$.

[3]SPACE includes multiple reference summaries for each entity; we randomly select one when determining the oracle clusters.

System	**Prevalence**	**ARI**
k-means	38.1	0.59
HERCULES	32.3	0.50
HIRO	**46.5**	**0.69**
(Oracle)	48.1	0.73

Table 7.1: Evaluation of our cluster selection method, compared to a range of baseline approaches. HIRO selects clusters of sentences that more closely match the references and contain more prevalent opinions.

(Reimers and Gurevych, 2019), then extract the 25 sentences whose embeddings are closest to each centroid. For HIRO, we select the top $k = 8$ subpaths for each entity, and set the smoothing constant $\alpha = 6$.

HIRO selects higher prevalence sentences Table 7.1 confirms that HIRO retrieves clusters that more closely match the oracle clusters, and contain opinions that are more prevalent in the input reviews compared to prior work. The oracle ARI score is less than 1 because some sentences appear multiple times in different clusters.

7.6 Generating Coherent Summaries

Given the retrieved clusters of sentences for each entity, we want to generate a coherent and fluent textual summary. LLMs are well suited to this constrained rewriting task, and we leverage the zero-shot abilities of instruction-tuned models to map clusters of sentences to a readable summary.

7.6.1 Method

We generate a summary from the retrieved clusters in three ways, with varying trade-offs between coherence and attributability, depicted in Figure 7.4.

HIRO$_{\text{ext}}$ We generate *extractive* summaries by selecting the centroid of each cluster; we compute all pairwise ROUGE-2 scores between sentences in each cluster, and choose the sentence with highest average similarity to other cluster members. This approach is inherently attributable, since each summary sentence is extracted verbatim from a review.

HIRO$_{\text{sent}}$ We generate summaries one sentence at a time by passing the contents of a single cluster as input to an instruction-tuned LLM with a simple prompt that requests a single sentence that summarizes the main points. This leads to more fluent output that is likely to be attributable, since each output sentence has an associated cluster of evidential sentences used to generate it.

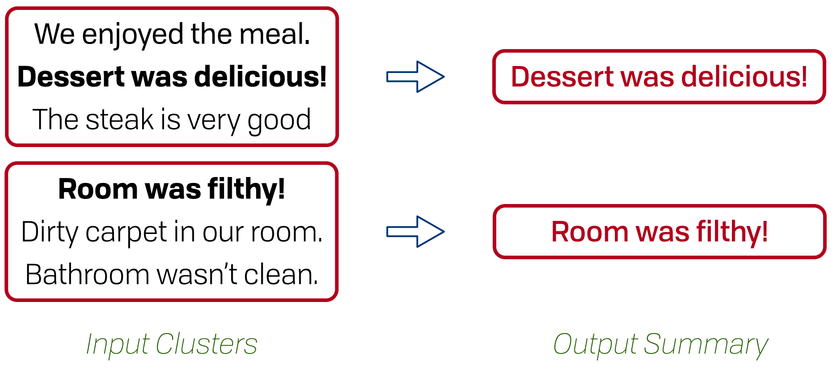
HIRO$_{\text{doc}}$ We generate summaries as a single document, by passing the sentences from all retrieved clusters for an entity to the LLM in one go. This gives summaries that are more coherent and less redundant, since the LLM has control over the whole summary. However, it is not possible to identify which cluster was used to generate each part of the summary, and therefore more difficult to determine the attributability of the output.

The ideal balance between coherence and the granularity of associated evidence is likely to vary by application.

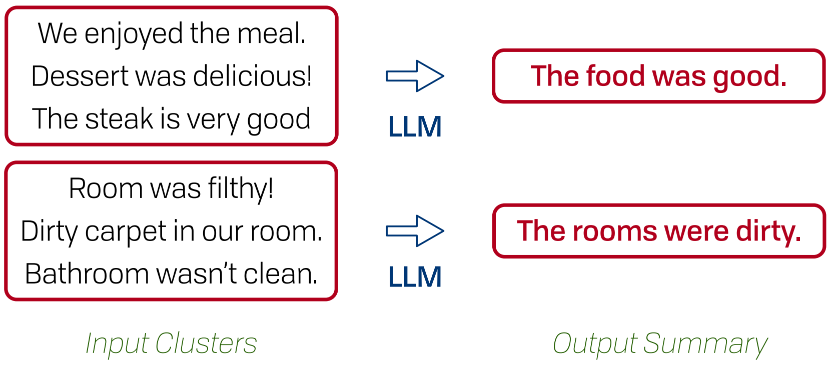
Experimental Setup For the variants that require an LLM, we use Mistral 7B Instruct v0.2 to generate summaries from retrieved clusters. Mistral 7B was chosen based on its qualitative performance during model development, but we compare using alternative models in Section 7.6.4. The LLM is queried in a zero-shot manner, and the prompts used are given in Appendix D.2. We sample with a temperature of 0.7, and report the mean and standard deviation scores based on 3 samples. We were unable to fine tune or few-shot prompt the LLM since no training summaries are available.

7.6.2 Automatic Evaluation

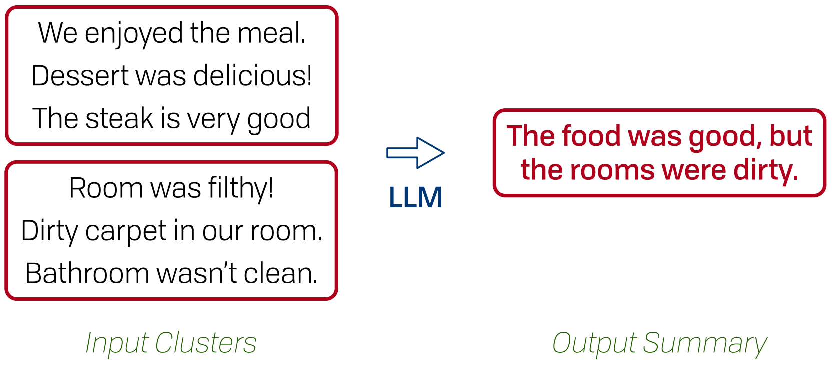
Human evaluation is the gold standard (Section 7.6.3), but automatic metrics remain useful for model development. ROUGE scores are no longer reliable (Callison-Burch et al., 2006; Tay et al., 2019; Fabbri et al., 2021) and SummaC overly rewards generic summaries (Section 7.2), but we nonetheless report them for consistency with prior work. Malon (2023) propose a *prevalence* metric, that uses an NLI model to determine how many input reviews contain supporting evidence for each sentence in the summary, but this suffers from a failure mode that overly rewards generic statements. A good



(a) HIRO$_{\text{ext}}$



(b) HIRO$_{\text{sent}}$



(c) HIRO$_{\text{doc}}$

Figure 7.4: Diagrams depicting the three variants of HIRO: extractive, sentence-wise, and document-level.

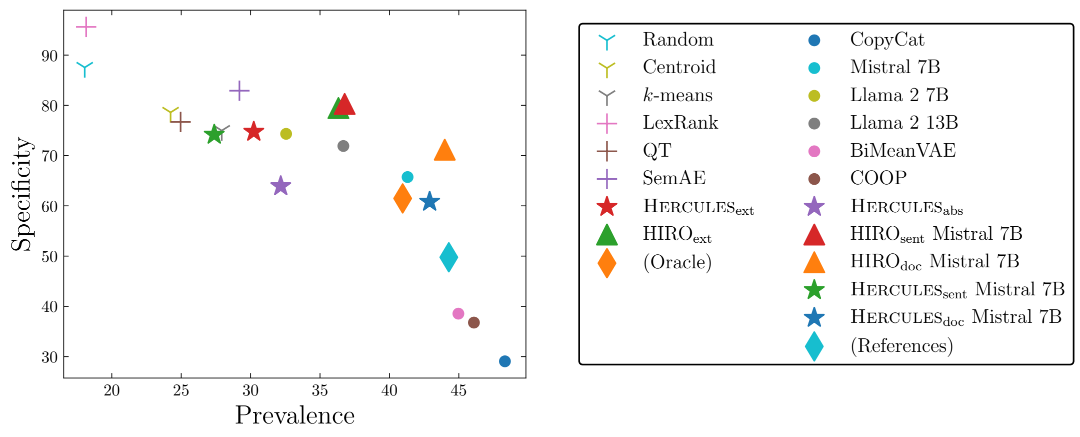


Figure 7.5: A plot showing Specificity against Prevalence for all models evaluated, for Space. The ideal model would fall in the top-right corner of the plot. There is a clear tradeoff between the two - CopyCat generates summaries with high Prevalence scores, but low Specificity, and vice-versa for LexRank. In general, extractive systems (marked with an X) are more specific but less prevalent than abstractive systems (marked with a disc), since the summaries comprise complete sentences from input reviews. All three variants of HIRO offer the best tradeoff between the competing factors.

summary should include common opinions, but should also help a user to differentiate between multiple entities.

To counteract this problem, we propose a modified version of prevalence, that explicitly penalizes generic summaries. First, we define the *genericness* of a summary as the average number of summaries from *other entities* that support each sentence in a given summary, as measured by an NLI model (ALBERT, trained on VitC; Lan et al., 2020; Schuster et al., 2021). Then, we define the Specificity Adjusted Prevalence score (SAP) as

$$\texttt{SAP} = \texttt{prevalence} - \alpha\, \texttt{genericness}, \tag{7.7}$$

where α is a constant that controls the balance between absolute prevalence and specificity. In practice, the ideal summary is unlikely to be entirely unique and a user may want to allow some degree of overlap between generated summaries. We arbitrarily set $\alpha = 0.5$, but found that the overall ranking of models did not materially change for values of α between 0.3 and 0.7. We also plot Specificity against Prevalence in Figure 7.5, with HIRO clearly exhibiting the best tradeoff between the two factors.

Datasets We evaluate summary generation using SPACE (Section 7.4.2), which includes multiple reference summaries created by human annotators for each entity. We

		SPACE						
	System	R-2 ↑	R-L ↑	Prev. ↑	Gen. ↓	SAP ↑	SC$_{\text{ins}}$ ↑	SC$_{\text{refs}}$ ↑
Extractive	Rand. Review	6.2	17.1	18.0	12.5	11.8	51.5	26.0
	k-means	9.5	19.8	27.9	25.0	15.4	72.7	31.0
	LexRank	5.9	16.4	18.2	**4.4**	16.0	54.4	22.6
	QT	10.3	21.5	25.0	23.3	13.3	**93.8**	41.2
	SemAE	11.1	23.5	29.2	17.1	20.7	65.9	27.9
	HERCULES$_{\text{ext}}$	**13.2**	**24.4**	30.2	25.2	17.6	89.0	**44.0**
	HIRO$_{\text{ext}}$	11.7	22.1	**36.3**	20.5	**26.1**	82.1	37.4
Abstractive	CopyCat	12.1	22.9	**48.3**	70.9	12.9	77.2	37.3
	Zero-shot Mistral 7B	5.3$_{\pm 0.1}$	19.6$_{\pm 0.4}$	41.3$_{\pm 1.3}$	34.3$_{\pm 3.3}$	24.2$_{\pm 0.8}$	52.5$_{\pm 1.5}$	24.6$_{\pm 0.5}$
	BiMeanVAE	13.7	27.1	45.0	61.4	14.2	75.1	36.2
	COOP	14.2	**27.2**	46.1	63.2	14.5	77.5	39.4
	HERCULES$_{\text{abs}}$	**14.8**	**27.2**	32.2	36.1	14.1	**95.1**	**60.2**
	HIRO$_{\text{sent}}$ + Mistral 7B	4.5$_{\pm 0.1}$	18.2$_{\pm 0.0}$	36.4$_{\pm 0.9}$	**20.2**$_{\pm 0.4}$	26.3$_{\pm 1.0}$	57.1$_{\pm 0.3}$	24.7$_{\pm 0.1}$
	HIRO$_{\text{doc}}$ + Mistral 7B	7.0$_{\pm 0.2}$	20.5$_{\pm 0.3}$	44.0$_{\pm 3.0}$	28.8$_{\pm 2.1}$	**29.6**$_{\pm 2.1}$	55.0$_{\pm 0.5}$	24.2$_{\pm 0.6}$
	(References)	74.4	76.7	44.3	50.2	19.2	61.3	91.3
	(Oracle)	45.0	53.3	41.0	38.5	21.7	69.3	69.6

Table 7.2: Results for automatic evaluation of summary generation, on the SPACE test set. R-2 and R-L represent ROUGE-2/L F1 scores, and SC$_{\text{ins}}$ and SC$_{\text{refs}}$ refer to the SummaC scores against the input reviews and reference summaries respectively. Prev. refers to Prevalence, Gen. refers to Genericness, and SAP refers to Specificity-Adjusted Prevalence. We consider SAP to be our primary metric. The best scores for extractive and abstractive systems are shown in bold. Results for systems involving LLMs are based on 3 samples, with the mean and standard deviation shown. HIRO generates summaries with the best balance between prevalent opinions and specificity.

		AmaSum						
	System	R-2 ↑	R-L ↑	Prev. ↑	Gen. ↓	SAP ↑	SC_{ins} ↑	SC_{refs} ↑
Extractive	Rand. Review	1.0	9.5	16.3	8.0	12.3	59.2	22.4
	k-means	2.3	12.0	14.9	11.4	9.1	73.9	23.3
	LexRank	2.7	12.2	9.0	**3.0**	7.5	67.2	23.5
	QT	1.5	11.4	10.9	7.3	7.3	66.2	22.4
	SemAE	1.6	11.3	8.7	4.1	6.7	57.2	21.8
	$\text{HERCULES}_{\text{ext}}$	**3.0**	12.5	9.5	6.7	6.2	**84.0**	24.4
	HIRO_{ext}	2.7	**12.6**	**19.4**	9.5	**14.6**	83.5	**24.7**
Abstractive	CopyCat	1.5	11.2	15.8	21.0	5.3	63.1	23.0
	Zero-shot Mistral 7B	1.9±0.0	12.6±0.0	17.3±0.2	17.6±0.4	8.5±0.2	50.6±0.2	22.0±0.0
	BiMeanVAE	2.0	12.5	14.7	24.1	2.7	52.5	21.8
	COOP	2.8	14.1	**18.8**	30.3	3.7	58.3	22.5
	$\text{HERCULES}_{\text{abs}}$	2.0	11.8	8.5	9.2	3.9	**82.7**	**25.2**
	$\text{HIRO}_{\text{sent}}$ + Mistral 7B	3.5±0.0	14.1±0.1	14.6±0.3	**6.9**±0.1	**11.2**±0.3	53.8±0.3	22.7±0.0
	HIRO_{doc} + Mistral 7B	**4.0**±0.0	**15.1**±0.1	15.3±0.1	9.4±0.3	10.6±0.1	46.8±0.5	21.9±0.0
	(References)	83.4	85.3	9.3	7.0	5.8	66.2	86.4
	(Oracle)	14.4	26.0	12.3	9.0	7.8	76.3	26.4

Table 7.3: Results for automatic evaluation of summary generation, on the AmaSum test set. R-2 and R-L represent ROUGE-2/L F1 scores, and SC_{ins} and SC_{refs} refer to the SummaC scores against the input reviews and reference summaries respectively. Prev. refers to Prevalence, Gen. refers to Genericness, and SAP refers to Specificity-Adjusted Prevalence. We consider SAP to be our primary metric. The best scores for extractive and abstractive systems are shown in bold. Results for systems involving LLMs are based on 3 samples, with the mean and standard deviation shown. HIRO generates summaries with the best balance between prevalent opinions and specificity.

also include **AmaSum** (Bražinskas et al., 2021), to evaluate summary generation on a wide range of categories of Amazon products, with an average of 560 reviews per entity. The reference summaries were collected from professional review websites, and therefore are not necessarily grounded in the input reviews. We use the same splits, based on four product categories, as released by Hosking et al. (2023b). We found that it was not necessary to train HIRO on a single product domain, and so we report results for a single HIRO model trained on all four product categories.

Comparison Systems We select a **random review** from the inputs as a lower bound. We include an extractive **oracle** as an upper bound, by selecting the input sentence with highest ROUGE-2 similarity to each reference sentence.[4] For a **k-means** baseline, we run k-means on MiniLM sentence embeddings (Reimers and Gurevych, 2019), then extract the nearest sentence to the cluster centroids. We set $k = 8$ to match the average sentence length of the reference summaries.

Lexrank (Erkan and Radev, 2004) is an unsupervised extractive method using graph-based centrality scoring of sentences.

QT (Angelidis et al., 2021) uses vector quantisation to map sentences to a discrete encoding space, then generates extractive summaries by selecting representative sentences from clusters.

SemAE (Basu Roy Chowdhury et al., 2022) is an extractive method that extends QT, relaxing the discretisation and encoding sentences as mixtures of learned embeddings.

CopyCat (Bražinskas et al., 2020) is an abstractive approach that models sentences as observations of latent variables representing entity opinions, trained in a 'leave one out' manner.

BiMeanVAE and **COOP** (Iso et al., 2021) are abstractive methods that encode full reviews as continuous latent vectors using an autoencoder, and take the average (BiMeanVAE) or an optimised combination (COOP) of review encodings.

We compare to a recent open weight instruction-tuned **LLM**, specifically Mistral 7B Instruct v0.2 (Jiang et al., 2023). Since no summaries to use as training data are available, the LLM was prompted zero-shot as per Appendix D.2, and sampled with temperature 0.7. We report the mean and standard deviation scores based on 3 samples. We additionally report results on the Llama 2 family of models in Section 7.6.4.

Most of the abstractive methods are not scalable and have upper limits on the number of input reviews. CopyCat and the LLMs have a maximum input sequence length, while

[4]When multiple references are available, we select one at random.

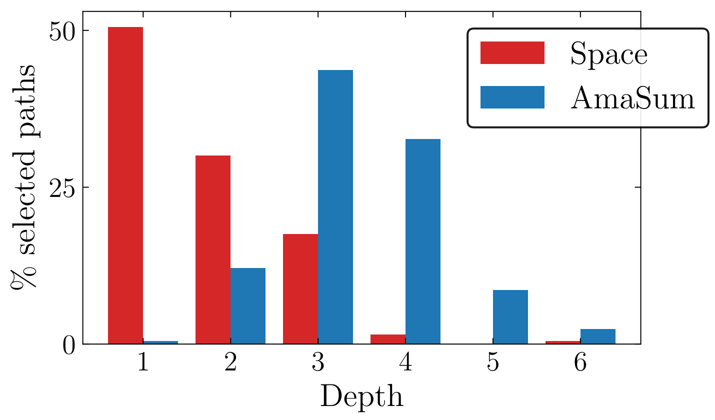


Figure 7.6: Distribution of selected subpaths by depth. A significant fraction of the extracted clusters come from paths deeper than the top level.

COOP exhaustively searches over combinations of input reviews. We use 8 randomly selected reviews as input to CopyCat, COOP, Llama 2, and Mistral 7B.

HIRO gives the best balance between prevalence and specificity The results in Tables 7.2 and 7.3 show that different systems have different strengths and weaknesses. The ideal balance between tradeoffs will likely depend on the exact use case of a system. However, we note that HIRO achieves the highest SAP scores across both datasets, indicating that it generates summaries with the best balance between absolute prevalence and specificity. While CopyCat and COOP achieve the highest prevalence scores on SPACE and AmaSum respectively, they also display some of the highest genericness; qualitatively, the outputs are very similar to each other, with few specific details. We give example outputs in Tables 7.5 and 7.6, and an example of selected subpaths, sentence clusters and corresponding HIRO outputs in Table 7.7.

References are not the upper bound While the oracle summaries score highly in terms of specificity-adjusted prevalence, some systems (including HIRO) outperform them. This indicates the difficulty with constructing reference summaries for entities with large numbers of reviews; it is not feasible for human annotators to reliably summarize such a large amount of information.

HIRO is more faithful to selected evidence To evaluate how faithful the generated summaries are to the retrieved sentence clusters or *evidence sets*, we use an NLI model to determine how many sentences in each cluster either entail or are entailed by the corresponding sentence in the output summary, and take the mean. Considering both

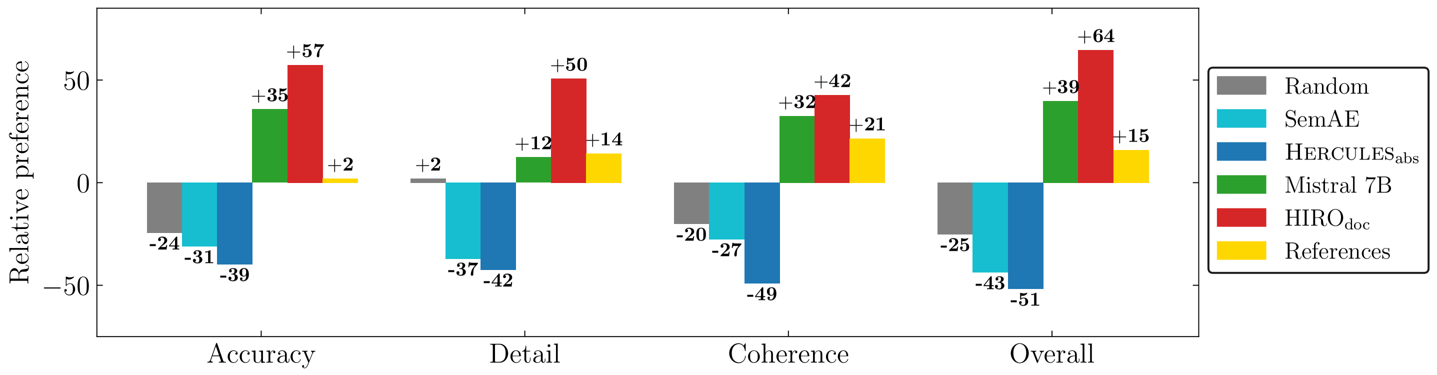


Figure 7.7: Results of our human evaluation. Crowdworkers were asked for pairwise preferences between generated summaries in terms of their Accuracy, Detail, and Coherence & Fluency, as well as an overall preference. HIRO generates more coherent and detailed summaries that better represent the opinions in the input reviews than comparison systems, and are preferred by human annotators.

System	**% Partial**	**% Majority**
HERCULES$_{\text{abs}}$	85.4	27.6
HIRO$_{\text{sent}}$	81.6	21.6
HIRO$_{\text{doc}}$	**91.8**	**28.4**

Table 7.4: Results for automatic evaluation of the evidence supplied by attributable systems, showing the percentage of summary sentences that have support from at least one sentence in the evidence set (partial support) and from at least half the sentences in the evidence (majority support). HIRO generates summaries that have strong partial support from the associated evidence sets, with improved majority support compared to HERCULES.

forward and backward entailment in this manner accounts for the different levels of granularity between the inputs and summary (Zhang et al., 2024a); input reviews are likely to be more specific than summary sentences, but concise summary sentences are likely to contain multiple assertions, e.g. "The food was good *and* the rooms were clean". HIRO$_{\text{doc}}$ does not align the evidence sets with each generated sentence, so we calculate the maximum support score over all sets for each summary sentence. Most abstractive systems are not attributable, and so we only compare with HERCULES. Table 7.4 shows the percentage of summary sentences that have support from at least *one* sentence in the evidence (partial support) and from at least *half* the sentences in the evidence (majority support). HIRO$_{\text{doc}}$ generates summaries that have strong partial support from the associated evidence sets, with improved majority support compared to HERCULES despite also being significantly more detailed.

HIRO makes use of the hierarchy We confirm that HIRO exploits the hierarchical nature of the representation space. Figure 7.6 shows the distribution of selected subpath depths for both datasets, indicating that HIRO takes advantage of the hierarchy and selects clusters deeper than the top level. This is particularly true for AmaSum, where there is a wider range of product types, causing the selection process to search deeper in the tree.

7.6.3 Human Evaluation

In Section 7.6.2 we showed that previous automatic metrics have failure modes, and we assume that our proposed metric SAP itself is not a perfect measure of summary quality. Although human evaluation is not without its own biases (Hosking et al., 2023a), we conduct a human evaluation to verify that HIRO generates summaries that are coherent and accurately reflect the opinions in the input reviews. We recruit crowdworkers through Prolific, selected to be L1 English speakers from the US or UK with a minimum of 100 previously approved studies, compensated above the UK living wage at 12GBP/hr. Participants were allowed to rate at most 5 samples. We show annotators a set of 50 reviews (chosen based on pilot studies to balance annotator load with reliability), followed by two generated summaries. We solicit pairwise preferences (Louviere and Woodworth, 1990) along three dimensions, as well as an overall preference:

- **Accuracy** — Which summary accurately reflects the balance of opinion in the reviews?
- **Detail** — Which summary includes more specific details?
- **Coherence** — Which summary is easy to read and avoids contradictions?
- **Overall** — Which summary do you think is better, overall?

The full instructions are reproduced in Appendix C.2. Ties (i.e., 'no difference') were allowed. We gather annotations for 10 entities each from the SPACE and AmaSum test sets, with 3 annotations for each pairwise combination of system outputs, leading to a total of 900 pairwise ratings. The study was approved by the School of Informatics ethics committee, ref. #491139.

We compare HIRO$_{\text{doc}}$ to the top performing extractive and abstractive systems from Tables 7.2 and 7.3, SemAE and HERCULES$_{\text{abs}}$. HIRO$_{\text{doc}}$ uses Mistral 7B as the generator, so we also compare to Mistral 7B *without* HIRO (i.e., prompted directly with reviews). Finally, we include a random review as a lower bound, and the references as an upper bound.

Humans prefer summaries generated by HIRO The results in Figure 7.7 show that HIRO$_{\text{doc}}$ produces summaries that outperform comparison systems across all dimensions, producing summaries that coherently and accurately represent the opinions in

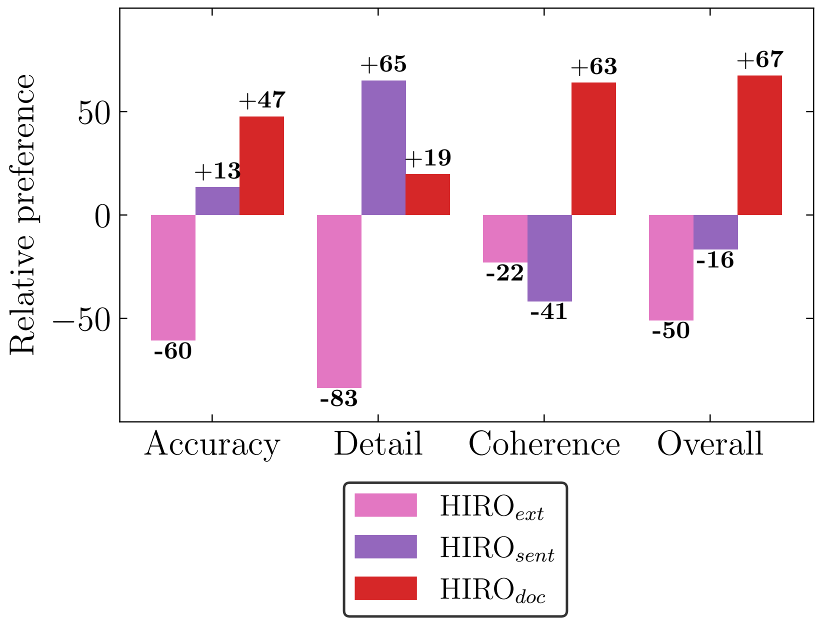


Figure 7.8: Results of a human evaluation comparing the three variants of HIRO: extractive, sentence-wise and document. Overall, annotators prefer the coherence of the document approach, but the sentence-wise variant generates more detailed summaries that are also more attributable. The preferred tradeoff between coherence and attribution will vary depending on the application.

the input reviews. Differences between HIRO$_{\text{doc}}$ and other systems are significant in all cases (using a one-way ANOVA with post-hoc Tukey HSD test, $p < 0.05$), except for coherence versus Mistral 7B. Both Mistral 7B and HIRO$_{\text{doc}}$ outperform the reference summaries, supporting findings from prior work (Bhaskar et al., 2023; Hosking et al., 2023a).

Comparing HIRO variants We run an additional human evaluation to evaluate the tradeoff between quality and attribution between the three HIRO variants, collecting annotations for 5 entities from each dataset from 3 annotators each. Figure 7.8 shows that annotators prefer the output of HIRO$_{\text{doc}}$, although the sentence-wise variant generates more detailed summaries. The preferred tradeoff between attribution and coherence will likely vary depending on the application.

7.6.4 Analysis

Ablations Tables 7.8 and 7.9 compare HIRO to zero-shot prompting, for a range of different LLM sizes from the Llama 2 Chat family of LLMs (Touvron et al., 2023).

System	**Output**
Reference	The staff were very welcoming and were always happy to help you with whatever was needed. The comfortably-sized rooms were very nice and clean, and the bathrooms were exceptionally clean and well equipped. Breakfast was continental and included in the price, offering baguettes, croissants, cereal, yogurts, etc.
CopyCat	We stayed at this hotel for 3 nights. The hotel was very clean and the staff very helpful. The location is perfect for exploring the city and the metro station is a short walk away. We will be back!
Hercules$_{abs}$	The room was clean and comfortable. The staff was very friendly and helpful. walking distance to everything. The room was clean and comfortable. breakfast was good. The hotel is in a great location, just a few minutes walk from the train station. breakfast was fine. The room and bathroom were very clean. The staff spoke English and were very helpful. There is also a small restaurant on the ground floor.
Mistral 7B	The St. Christophe Hotel in Paris offers small, clean, and cozy rooms in a lively neighborhood near the Place Contrascarpe and Rue Mouffetard. The staff are helpful and friendly, and the location is convenient with easy access to metro stations and nearby restaurants. The hotel lacks air conditioning, so it may not be ideal during warm weather. Free Wi-Fi is available in the lobby. Overall, the St. Christophe Hotel is a charming and elegant choice for travelers to Paris.
HIRO$_{ext}$	Nearby, and very close to 2 Metro stations. The hotel is 1 block to the Metro (line 7) between the North and South exits of the "place Monge" stop. The bathroom was clean and functional. The staff were friendly and helpful at all times. Breakfast was Plentiful-Cold cereal, great coffee, yogurt and a wonderful basket of Croissants, breads and Pastries. The shower excellent.
HIRO$_{sent}$	The Hotel Saint Christophe is conveniently located near multiple metro stations. The Hotel Saint Christophe is centrally located near the Metro and restaurants in the Latin Quarter. The Hotel Saint Christophe offers clean, functional bathrooms in small rooms. The Hotel Saint Christophe's friendly and helpful staff provided excellent service. Breakfast offers cereal, yogurt, pastries, and hot beverages. Guests praised the Hotel Saint Christophe for its excellent showers.
HIRO$_{doc}$	The Hotel Saint Christophe is centrally located in the Latin Quarter, close to multiple Metro stations and attractions such as Notre Dame and Jardin Luxembourg. Rooms are clean and adequate, though small, and breakfast offers a variety of options. The staff are friendly and helpful, making for a pleasant stay. The hotel is conveniently located near shops, restaurants, and markets. The bathrooms are clean and modern, though some may be smaller than expected for Paris.

Table 7.5: Examples of generated summaries for 'Hotel Saint Christophe' in Paris, from SPACE. HIRO$_{\text{doc}}$ generates concise, informative and coherent summaries. Mistral 7B prompted directly with reviews also generates coherent summaries, but is limited to a small sample of reviews which do not necessarily reflect the overall balance of opinions. Previous abstractive systems, CopyCat, COOP and Hercules$_{\text{abs}}$, generate generic summaries that lack detail.

System	**Output**
Reference	These headphones are your best option if you're looking for noise-cancelling earbuds, but they are pricey. Noise-cancelling earbuds. Available in black or white. Stay Hear tips keep earbuds in your ears while maximizing comfort. Impressive sound quality. Carrying case included. These are expensive for earbuds, so you should only buy them if you're going to use them often
CopyCat	So far so good. The only issue I have is that the volume is not as loud as my old one, but other than that I'm happy with this purchase.
Hercules$_{abs}$	Love these Earbuds. These are great for listening to music. Noise cancelling works well. Long battery life. These are my favorite Headphones. The sound quality is very clear. The sound quality is great and the noise cancellation is good. Got this for my wife for Christmas. The sound quality is great and the Mic is clear. The Bass is a bit lacking. The ear pads are soft and comfortable. These Headphones are the best ever. The Headphones are very comfortable.
Mistral 7B	These earbuds have received positive reviews from users who travel frequently, find them a game changer, and have even convinced friends to buy them. Some users have purchased them for their children who are also satisfied. The earbuds have received five-star ratings from both the buyers and their sons. One user's wife loves them and uses them when the user snores at night. However, some users have noted that the noise-cancelling feature is not as effective as over-ear headphones.
HIRO$_{ext}$	The noise cancelling is fantastic. These are the best Headphones I've ever owned. Well worth the money! The batteries last a long time and charge quickly. I have the over the ears pair and also these in the ears.
HIRO$_{sent}$	The Bose QuietComfort 20 headphones offer exceptional noise cancellation and comfort. These Bose QuietComfort 20 headphones are exceptional, providing excellent noise cancellation and sound quality. These headphones are worth the investment due to their exceptional noise-cancelling capabilities. Battery lasts long for noise cancellation, charging quickly. Some users report over 10 hours usage. Some prefer Bose QuietComfort 20 earbuds for flights and noisy environments, others find them dizzying.
HIRO$_{doc}$	The Bose QuietComfort 20 Acoustic Noise Cancelling Headphones are widely praised for their excellent noise cancellation capability, making them a popular choice for frequent flyers and those working in noisy environments. They are also praised for their comfort, sound quality, and long battery life. However, some customers have expressed concerns about the non-replaceable battery and the price. Overall, these headphones are considered a worthwhile investment for their impressive noise cancellation and sound quality.

Table 7.6: Examples of generated summaries for the 'Bose QuietComfort 20' headphones, from AmaSum. Mistral 7B refers to the opinions of a single user, which is not appropriate for a summary of thousands of reviews.

$q_{1:d}$	(6,2,10)
Evidence	The pool area was very nice. The staff was very Friendly and helpful. the pool area was very enjoyable The pool area is very nice. The pool area is fantastic The pool area was nice the pool area is very attractive The staff was so friendly and helpful. The pool was nice The staff was friendly and accommodating. The pool was very nice and the lobby inviting the staff was welcoming and helpful The hotel staff was very friendly and accommodating. Staff was friendly and helpful. The pool area is very nice and large with several water features. The staff was really great and helpful
HIRO$_{sent}$	The pool area and friendly staff make this hotel a enjoyable stay.
$q_{1:d}$	(9,)
Evidence	I was extremely disappointed in the rooms Only one mirror in the room. The rooms are loud. the rooms are awful the room was small and shabby we were a little disappointed because the room was a lot smaller than we expected Extremely disappointed in the room, although the help was very nice as was the outdoor area. i suspect we were in the older part of the hotel with a double room. Since the parking is directly under the rooms, it was Very loud espcially from 12-3 am.
HIRO$_{sent}$	Rooms were small, loud, and in need of renovation with poor housekeeping.
HIRO$_{doc}$	The Fairfield Inn and Suites Key West received positive reviews for its friendly and helpful staff, attractive and nice pool area, and free parking. However, some guests were disappointed with the small and shabby rooms, lack of storage space, and noise from parking and adjacent rooms. The continental breakfast was also mentioned as a plus.

Table 7.7: Examples of selected subpaths $q_{1:d}$, the corresponding evidential clusters, the resulting HIRO$_{\text{sent}}$ output sentences, and the overall HIRO$_{\text{doc}}$ summary for the 'Fairfield Inn', from SPACE. We show only two out of five input clusters, and a subset of all evidence sentences, due to space constraints.

Increased parameter count does lead to improved SAP scores for the zero-shot approach on SPACE, but the largest 70B model scores worst on AmaSum. For all choices of LLM, HIRO leads to summaries with a better balance between prevalence and specificity than zero-shot prompting; HIRO is likely to lead to improvements for any choice of instruction-tuned LLM.

We also compare using HIRO to identify clusters against a k-means baseline, and HERCULES (Chapter 6). For both datasets, using clusters selected with HIRO leads to summaries that are much less generic, while remaining comparatively prevalent. This confirms the results in Section 7.5.2, indicating that HIRO selects more informative clusters of sentences than the comparison methods.

Qualitative Analysis Table 7.10 shows output from Mistral 7B and HIRO as well as a reference summary, for The Grand Hotel Maryland from SPACE. While all generated summaries are fluent and convincing, Mistral 7B makes reference to the opinion of a single user, which should not appear in a summary. It also describes positive praise about a named member of staff, but a manual analysis shows that only 3 out of 5 reviews mentioning that individual are positive; the true sentiment is much more mixed than the summary indicates. These extrapolations highlight the limitation of models that can only accept a limited number of reviews as input.

The examples of selected subpaths, sentence clusters and corresponding HIRO outputs in Table 7.7 demonstrate the difficulty of evaluating attribution. The final cluster contains sentences that are all topically related, indicating that HIRO has learned a successful clustering. While the sentences and corresponding HIRO$_{\text{sent}}$ output are all broadly negative, it is not straightforward to determine whether the sentence “Only one mirror in the room” counts as direct evidence towards the output “Rooms were small, loud, and in need of renovation [...]”. This partly explains the relatively low majority support scores in Table 7.4; some of the selected evidence may be consistent in topic and sentiment, but not directly entail the resulting output.

As well as collecting pairwise preferences in our human evaluation, we allowed annotators to leave qualitative comments. Table 7.11 shows all non-trivial comments from annotators for pairs including HIRO, with anonymised labels replaced by the true model names. The majority of comments are positive towards HIRO, highlighting improved levels of detail and a better balance of the input reviews. However, some annotators note that HIRO summaries may be less natural or too conservative.

			Space				
	Clusters	**LLM**	R-2 ↑	R-L ↑	Prev. ↑	Gen. ↓	SAP ↑
Llama 2 7B	Zero-shot	Llama 2 7B	4.4	17.6	32.6	25.7	19.7
	HIRO$_{\text{sent}}$	Llama 2 7B	5.2	16.9	36.2	**20.0**	26.2
	HIRO$_{\text{doc}}$	Llama 2 7B	**6.9**	**19.9**	**48.5**	29.9	**33.5**
Llama 2 13B	Zero-shot	Llama 2 13B	6.3	19.1	36.7	28.1	22.6
	HIRO$_{\text{sent}}$	Llama 2 13B	6.6	18.8	35.0	**23.7**	23.1
	HIRO$_{\text{doc}}$	Llama 2 13B	**8.5**	**21.7**	**46.2**	27.0	**32.7**
Llama 2 70B	Zero-shot	Llama 2 70B	5.6	19.4	**48.9**	37.2	30.3
	HIRO$_{\text{sent}}$	Llama 2 70B	5.8	18.5	41.1	**19.6**	31.3
	HIRO$_{\text{doc}}$	Llama 2 70B	**8.3**	**21.3**	48.5	30.0	**33.5**
Sent-wise	k-means	Mistral 7B	**4.5**	17.1	30.1	30.8	14.7
	Hercules	Mistral 7B	3.9	17.0	27.4	25.8	14.5
	HIRO$_{\text{sent}}$	Mistral 7B	**4.5**	**18.2**	**36.4**	**20.2**	**26.3**
Doc-wise	k-means	Mistral 7B	6.4	20.5	40.2	36.3	22.0
	Hercules	Mistral 7B	**7.6**	**21.0**	42.9	39.2	23.3
	HIRO$_{\text{doc}}$	Mistral 7B	7.0	20.5	**44.0**	**28.8**	**29.6**

Table 7.8: Automatic evaluations comparing HIRO to zero-shot summarisation, using a range of different LLMs, for Space. We show the mean scores based on 3 samples, and best scores within each comparison are bolded. In all cases, HIRO improves on the zero-shot approach. We also compare HIRO to other cluster selection methods, finding that HIRO leads to summaries with a better balance between prevalance and specificity.

			AmaSum				
	Clusters	**LLM**	R-2 ↑	R-L ↑	Prev. ↑	Gen. ↓	SAP ↑
Llama 2 7B	Zero-shot	Llama 2 7B	1.5	11.5	**17.7**	17.6	8.9
	HIRO$_{\text{sent}}$	Llama 2 7B	3.6	14.0	14.6	**6.3**	**11.5**
	HIRO$_{\text{doc}}$	Llama 2 7B	**4.0**	**15.2**	16.0	12.1	10.0
Llama 2 13B	Zero-shot	Llama 2 13B	2.3	13.1	18.1	14.3	10.9
	HIRO$_{\text{sent}}$	Llama 2 13B	3.8	14.3	13.5	**6.2**	10.4
	HIRO$_{\text{doc}}$	Llama 2 13B	**4.3**	**15.7**	**19.0**	9.9	**14.1**
Llama 2 70B	Zero-shot	Llama 2 70B	2.1	12.5	16.8	22.4	5.6
	HIRO$_{\text{sent}}$	Llama 2 70B	3.6	14.5	15.1	**3.7**	**13.2**
	HIRO$_{\text{doc}}$	Llama 2 70B	**4.4**	**16.0**	**17.8**	9.9	12.9
Sent-wise	k-means	Mistral 7B	3.2	13.3	12.9	12.4	6.7
	HERCULES	Mistral 7B	3.3	11.8	8.5	8.7	4.1
	HIRO$_{\text{sent}}$	Mistral 7B	**3.5**	**14.1**	**14.6**	**6.9**	**11.2**
Doc-wise	k-means	Mistral 7B	3.7	14.5	**16.2**	13.9	9.2
	HERCULES	Mistral 7B	3.8	14.7	13.8	11.6	8.0
	HIRO$_{\text{doc}}$	Mistral 7B	**4.0**	**15.1**	15.3	**9.4**	**10.6**

Table 7.9: Automatic evaluations comparing HIRO to zero-shot summarisation, using a range of different LLMs, for AmaSum. We show the mean scores based on 3 samples, and best scores within each comparison are bolded. In all cases, HIRO improves on the zero-shot approach. We also compare HIRO to other cluster selection methods, finding that HIRO leads to summaries with a better balance between prevalance and specificity.

System	**Output**
Reference	The staff were very friendly, courteous and helpful. The hotel and room was clean. The rooms are very spacious with a nice oceanfront view and a gigantic balcony! Pool bar is great and food is good, too. The hotel has a perfect location. It was very nicely located with a 24 hours cafe downstairs, 2 pools , 2 bars and some beach stores to shop in.
Mistral 7B	The Grand Hotel in Ocean City, Maryland, consistently receives positive reviews for its exceptional customer service, cleanliness, and convenient location on the boardwalk. Rooms offer great views, and amenities include a spa, restaurant, lounge, indoor pool, game room, and exercise room. Guests appreciate the family-friendly atmosphere and the ease of having everything they need within the hotel. Some reviewers praise Nina Railean for her helpfulness and accommodating nature. However, one review warns against staying during Memorial Day weekend due to loud parties.
HIRO$_{sent}$	All rooms offer excellent ocean views at this hotel. The Grand Hotel & Spa offers direct ocean views and easy beach access. The Grand Hotel & Spa offers clean and spacious rooms. No problems with accommodations or access despite rain and floor issues. The Grand Hotel & Spa offers beautiful views and a comfortable stay with excellent service. The Grand Hotel & Spa exceeded expectations. Rooms need updating peeling wallpaper, large, dark interiors.
HIRO$_{doc}$	The Grand Hotel & Spa offers stunning ocean views from most rooms, with some having a direct view of the boardwalk and beach. The hotel's design ensures that guests can enjoy the ocean despite room location. The beach is clean, and the location is convenient for easy access. Rooms are described as clean and large, but some note a need for improvement in terms of lighting and room condition. Overall, guests had positive experiences and appreciated the ocean access.

Table 7.10: Generated summaries for The Grand Hotel Maryland, from Space, comparing HIRO to zero-shot Mistral 7B. While all generated summaries are fluent and convincing, Mistral 7B makes reference to the opinion of a single user, and positive sentiment about a member of staff that is not supported by the full set of reviews. These extrapolations highlight the problem with models that can only accept a limited number of reviews as input.

HIRO provides more specific detail about user frustrations and the product itself. Mistral 7B is quite generic.
Mistral 7B mentions that desk staff were unfriendly, but this is not substantiated by the reviews, the majority of which are overwhelmingly positive. It's also a contradiction since earlier it says the staff were friendly.
Only HIRO references the high failure rate.
HIRO seems more accurate in its appraisal of the staff; the reviews were mixed. However Mistral 7B is slightly more informative, particularly in relation to location and proximity to amenities. I feel like HIRO sits on the fence quite a lot, whereas Mistral 7B attempts to summarise better.
I think HIRO is the better written and reflects a broader view on the reviews but References isn't bad it does cover PS4 and gaming which alot of reviews were using it for.
SemAE is incoherent. HIRO does an excellent job of summarising everything, balancing all perspectives.
Can tell HIRO is AI generated
HIRO is by far the better written and has a little more detail and reflects the reviews more than $\text{Hercules}_{\text{abs}}$

Table 7.11: Comments from annotators for all pairs involving HIRO. Anonymised system labels have been replaced with the system names. The majority of comments are positive towards HIRO, although annotators comment that HIRO summaries may be less natural, or too conservative.

7.7 Citation-Enabled LLMs

Since this work was completed, open-weight 'citation-enabled' LLMs have become available. These are instruction tuned models that have been trained to optionally include references back to their inputs when generating, primarily for RAG applications where multiple documents may be included as part of the input. While these models are not as interpretable as HIRO, since there is no guarantee that a citation will be used or will refer to the correct input, they do offer an additional way to generate summaries whose attribution can be verified.

Table 7.12 shows some example outputs for the Hotel Navona, from SPACE, using a modified prompt with a range of citation-enabled LLMs, as well as Mistral 7B for comparison. This modified prompt includes a numeric label for each cluster extracted by HIRO, and instructs the LLM to include citations as part of its output. While Mistral 7B does correctly include citations, it also reverts to a very linear summary structure that summarizes each cluster in turn. The Llama 3 models and Command R+ generate fluent summaries that additionally reference which cluster each statement is based on. A comprehensive evaluation is beyond the scope of this chapter, but initial examination of the output indicates that the citations are generally correct and the summaries are high quality. Future work could consider a more comprehensive evaluation of a variant of HIRO based on citation-enabled LLMS.

7.8 Limitations

Since our approach identifies common opinions based on frequency of sentence encodings, we require a relatively large number of input sentences. We were not able to experiment with other popular datasets like Amazon (He and McAuley, 2016), Yelp (Chu and Liu, 2019) or Rotten Tomatoes (Wang and Ling, 2016) since these datasets only include a small number (usually 8) of input reviews.

HIRO can be trained without needing reference summaries, but the encoding space has a fixed capacity and is unlikely to generalize well to unseen domains. We welcome future work on extending the encoder or learning a domain-agnostic encoding space.

Although our experiments were limited to English, HIRO does not make any strong assumptions about the language of the reviews that it summarizes. However, it does make use of entailment models and tf-idf to select positive pairs for training, as well as LLMs for summary generation. Its application to other languages is therefore restricted to languages for which these resources are available.

7.9 Summary

In this chapter, we propose HIRO, a modular method for unsupervised opinion summarisation that uses a hierarchical index over sentences to select clusters of prevalent opinions. Our approach leverages pretrained Large Language Models to generate coherent and fluent summaries that are attributable and accurately reflect the popularity of opinions in the input reviews. Extensive experiments show that, as well as generating higher quality summaries, HIRO learns a more semantically distributed representation than competitive baselines.

While we limit our experiments to opinion summarisation, we believe that HIRO demonstrates the efficacy of combining LLMs with more specialised models based on structured representations, and the approach could be usefully applied to a wide range of other retrieval-augmented generation tasks. We welcome future work on integrating discrete and hierarchical encodings more deeply into large language models.

	LLM	Output
Standard prompt	HIRO$_{\text{doc}}$ + Mistral 7B	The Hotel Navona is centrally located in Rome, providing easy access to major attractions. The breakfast was described as average with small, basic rooms, some of which were noisy and dirty. Staff were consistently described as friendly and helpful, making up for any drawbacks. The hotel is in an old palazzo and retains much of its original charm. Some rooms feature 14th century paintings on the ceilings. Overall, the hotel's prime location and friendly staff make up for its minor drawbacks.
Citation prompt	Mistral 7B	Group [1] and [3] primarily focus on the hotel's breakfast and room conditions, with complaints about the quality and size, as well as noise and cleanliness issues. [1] also mentions the location being average. Group [2] highlights the hotel's excellent location, which is central and convenient for accessing major Rome attractions. Group [5] emphasizes the friendly and helpful staff, with some mentioning specific staff members who went above and beyond. Group [6] consists of various comments about room assignments, including being fortunate to find the hotel, having rooms with historical features, and booking methods. Group [7] focuses on the hotel's cleanliness and basic accommodations, with some mentioning the small size and clean bathrooms. Group [8] touches on the hotel's history and cleanliness, with one comment mentioning a poor smell on the first floor.
	Llama 3 8B Instruct	The hotel's location is a major strength, with many reviewers praising its central location and walking distance to major attractions (Group 2). However, some rooms are small and dark (Group 3), and the breakfast is unimpressive (Group 1). The staff is generally friendly and helpful (Group 4), and the hotel is clean and well-maintained (Group 7). The hotel's history and architecture are also notable, with some rooms featuring 14th-century paintings and frescoes (Group 6).
	Cohere Cmd R+ (104B)	Reviewers praised the hotel's excellent location [2], with many attractions within walking distance. Opinions varied on room size and cleanliness [3], but the staff was friendly and helpful [5]. Breakfast was basic [1], and the hotel featured historical details [6,8]. Overall, the hotel was clean and air-conditioned [7].

Table 7.12: Examples of summaries generated using HIRO clusters with a prompt that asks the model to include citations. Mistral 7B generates a summary one cluster at a time, which is overly long. The other two LLMs, which were released since this work, successfully cite which input cluster they have used to generate parts of the output.

For example, the output embeddings of each layer in a Transformer-based model could be quantized and reconstructed before being passed to the next layer.

This chapter acts as the final piece of evidence in support of our core hypothesis; the discrete hierarchical encoding space directly enables attributable and scalable opinion summarisation (Hypothesis I and Hypothesis II). Unlike previous chapters that used a denoising autoencoder objective, HIRO uses a contrastive training objective to learn an ordered encoding space, supporting Hypothesis III. Building on Chapter 6, we show that the use of structured representations is compatible with LLMs. Using each approach on the subtasks for which they are best suited — compressed structured representations for scalable content selection, LLMs for fluent language modelling — allows us to achieve higher performance than a single model for the full task.

Chapter 8

Conclusions and Future Work

In this thesis, we argue that weakly structured representations are beneficial for text-to-text generation models. We consider two tasks with natural invariances: paraphrase generation, where the goal is to generate an output sentence with the same meaning but different surface form as an input sentence; and opinion summarisation, where a model should aggregate opinions about a hotel or product based on their meaning, ignoring the specific choice of phrasing.

In Chapter 3 we introduce SEPARATOR, a paraphrase generation method that uses an encoder-decoder model to map an input sentence into a latent encoding space, and then back to an output paraphrase. A principled information bottleneck and a careful choice of training scheme (Section 3.3.2) lead to an encoding space that separately represent the meaning and surface form of an input sentence, with Vector Quantisation (VQ-VAE, van den Oord et al., 2017) used to learn a discrete representation of the surface form. This separation enables us to paraphrase the input sentence, varying the surface form of the output by directly manipulating the syntactic encoding of the input and keeping the semantic encoding constant. Extensive experiments and a human evaluation show that SEPARATOR is able to generate paraphrases with a better tradeoff between semantic preservation and syntactic novelty compared to previous methods.

In Chapter 4 we address the lack of a tractable factorisation of the joint distribution over codes learned by VQ-VAE, and introduce Hierarchical Residual Quantisation (HRQ-VAE), a method for learning hierarchical discrete latent representations of input data by recursively quantising the residual error between an input embedding and its discretised approximation. We introduce the theoretical foundations of the approach and give practical details for stable training of models that use HRQ-VAE. We report validation experiments on MNIST, a dataset of handwritten digit images, and show that

HRQ-VAE learns more informative representations than VQ-VAE.

In Chapter 5 we combine the factorised representation spaces from Chapter 3 with HRQ-VAE and describe CALYPSO, a method for generating paraphrases that learns hierarchical representations of the syntactic structure of input sentences. This hierarchical encoding is easier to predict at test time than the less structured representation used by SEPARATOR, leading to a higher quality of generated paraphrases.

In Chapter 6 we apply HRQ-VAE to unsupervised opinion summarisation and introduce HERCULES, a method that encodes sentences from customer reviews into a hierarchical discrete latent space, then identifies common opinions based on the frequency of their encodings. Our approach is attributable and scales to large numbers of input reviews, while also generating abstrative summaries that are more informative than prior work. HERCULES exploits the discrete and hierarchical properties of the learned representation to aggregate opinions from reviews about hotels and Amazon products.

In Chapter 7 we combine the scalability and attributability of structured representations with the fluency of Large Language Models(LLMs). We introduce HIRO, an unsupervised opinion summarisation method that uses a hierarchical discrete latent space based on HRQ-VAE to identify clusters of sentences from reviews that contain popular opinions. Passing these retrieved clusters to a LMM leads to fluent and coherent summaries. HIRO demonstrates how methods based on weakly structured representations are compatible with powerful (but unstructured) LLMs.

8.1 Findings

Core Hypotheses Recall our hypothesis that "weakly structured representations are beneficial for text-to-text generation" (Hypothesis I). In Chapters 3 and 5 we chose a representation structure which encodes the meaning and form of an input sentence separately, a natural choice given the task definition. This choice of representation leads directly to an ability to generate paraphrases that more closely matched the original meaning of the input, while introducing more syntactic diversity than prior work. Furthermore, in Chapter 5 we used HRQ-VAE (Chapter 4) to learn a hierarchical representation of the surface form, allowing us to represent (and therefore predict more accurately at inference time) the high level syntactic structure of a sentence separately from more fine-grained details. In Chapters 6 and 7 we applied HRQ-VAE to opinion summarisation, learning an embedding space that is semantically structured and that allows us to efficiently aggregate opinions from input reviews. Overall, we believe the experiments in this thesis act as substantial evidence in support of Hypothesis I; the performance improvements in each case are a direct result of choosing a representation that is weakly structured.

Secondly, we hypothesised that "discrete and hierarchical representations can be used to make text-to-text generation problems feasible" (Hypothesis II). In Chapter 3 we used a discrete representation for the *syntactic structure* of sentences, and this discreteness directly enabled us to easily predict alternative surface forms when generating paraphrases. Predicting a high dimensional dense vector to use as the target surface form is an extremely difficult regression problem, but using a discrete representation reduces it to a simple classification task. In Chapter 5 we further simplified the prediction problem by using a hierarchical structure to represent the syntactic structure, allowing us to first predict a high-level surface form before gradually refining the level of detail, and leading to improved quality when generating paraphrases. In Chapters 6 and 7 we used a discrete hierarchical structure to represent the meaning of sentences from product and hotel reviews. The discreteness allowed us to identify which opinions occur frequently by simply counting them, an operation that would be much more challenging in a continuous space. Furthermore, the hierarchical nature of the representation directly enabled us to perform this aggregation at an appropriate level of abstraction. Overall, our experiments support Hypothesis II.

Finally, we hypothesised that "given a structured representation, some degree of supervision is required to assign meaning to the structure" (Hypothesis III). In Chapters 3 and 5 we used a denoising objective to assign meaning to the parts of the representation, whereby the model must reconstruct a target sentence from one input with the correct meaning but different surface form, and another input with the correct surface form but different meaning. In Chapters 6 and 7, where we want the levels of the hierarchical representation to correspond to a hierarchy of meaning, we automatically constructed pairs of sentences with similar meanings but different surface forms, and trained the model to place these pairs close together in the hierarchy. In both cases, we found that these sources of distant supervision lead to structured representations where the parts of the structure have an associated meaning, in support of Hypothesis III. Removing these sources of supervision led to catastrophic failures in performance.

Additional Themes Over the course of the thesis, we made some additional observations that fall into two main themes.

Firstly, evaluation of text-to-text generation is extremely challenging (Celikyilmaz et al., 2020, inter alia)[1]. There is rarely a single correct output for text-to-text problems, and determining whether a generated output meets the desired criteria is a complex problem. For both tasks considered in this thesis, further complexity is introduced by the existence of multiple competing desiderata: a good paraphrase should both preserve meaning *and* introduce syntactic diversity; a good summary of reviews should balance accurately representing the prevalance of opinions *and* provide some details that distinguish the current hotel or product from the others. In both cases, the two factors are in opposition. For example, it is trivial to generate an output sentence with the same meaning as a given sentence, by simply copying it. This approach would maximise meaning preservation, but is useless in practice. Evaluation of text-to-text generation systems therefore often involves some form of decision about a trade-off between competing factors that may be context dependent. Whether it is more important that paraphrases convey precisely the same meaning or introduce a wide range of diversity will depend on how the system is being used.

Secondly, while the experiments in the thesis support Hypothesis I and show that weak structure is beneficial, the benefits do come at a cost of complexity. The models presented in this thesis require additional resources to train (e.g., syntactic chunkers and NLI models) and required significant effort to tune the hyperparameters and achieve stable training. Although our methods could in principle be applied to languages other than English, these resources may not always be available. Rich Sutton's Bitter Lesson (Sutton, 2019) argues that computation will always win over complexity, a theory that is currently being borne out in the form of LLMs. LLMs require comparatively little complexity in terms of design, instead scaling parameter counts and training data to achieve strong performance. We hope that there remains significant value in combining both approaches, with models like HIRO (Chapter 7) providing a possible direction for combining the efficiency and interpretability of structured models with the fluency and flexibility of large-scale LLMs.

8.2 Future Work

HRQ-VAE for Information Retrieval Information Retrieval (IR) involves retrieving a set of documents that are relevant to an input query from a large collection of can-

[1]Seraphina Goldfarb-Tarrant tried to warn me about this in personal correspondence in 2019, before the start of this thesis. She was correct.

didates. Currently, a popular approach is to learn a pair of encoder models that map queries and documents to a shared embedding space, with queries mapped to similar points in space as the documents that are relevant to it (Karpukhin et al., 2020, inter alia). Then given a query at inference time, fast nearest-neighbour methods (e.g., FAISS, Douze et al., 2024) are used to efficiently find the nearest document embeddings to the query embedding. These nearest-neighbour methods generally use some form of quantization to discretise the embeddings and allow for fast lookups. In Chapters 4 and 7 we showed that learning the quantization concurrently with the encoder was beneficial compared to post-hoc clustering (k-means). We believe that HRQ-VAE could be used effectively in IR to learn a embedding space where the discretisation is semantically meaningful rather than post-hoc and arbitrary, leading to improved performance when retrieving documents.

HRQ-VAE uses the full dimensionality of the encoding space at each level in the hierarchy, but it would be beneficial in an IR context for the dimensionality of the space to increase at deeper levels. This would allow for faster comparison of query and document embeddings at the top levels, with increasingly granularity at deeper levels where fewer comparisons would need to be performed. This would require a modification to the way in which the discrete codes are mapped to a continuous embedding (and vice-versa), but should in principle be possible.

Adaptive Hierarchies The HRQ-VAE models introduced in this thesis use a fixed codebook size and number of levels. These are hyperparameters that must be tuned for the specific task and dataset, limiting the ease of applying the technique to a wide range of problems. Ideally, the capacity of the hierarchy would be adaptive in some way; the model should only use as much of the tree as it needs. For example, if a particular input sample is well specified by the centroid of an intermediate cluster at level $d < D$, then the residual error should already be small and there should be no benefit in further refining the approximated encoding. Future work could apply careful regularisation techniques to lower the dependence on the codebook size and depth hyperparameters and reduce any redundancy in the learned encoding space. For example, Chen and Fuge (2024) propose including a penalty term to the training objective that minimises the volume occupied by an embedding space, resulting in an encoder that only uses as many dimensions of the embedding space as it needs.

LLMs as Decoders In Chapter 7 we presented a method that combined the fluency and coherence of LLMs with the scalability and interpretability of structured models. Future work could expand on this approach for a wider range of tasks, and use LLMs primarily as decoders with other models performing any required reasoning or pre-processing of the input. We note that there is already some work in this direction: Wang et al. (2023) and Xu et al. (2024) propose using auxiliary lightweight compression or filtering models to filter out irrelevant context and highlight important information in retrieved documents, before using a LLM to do the final generation. We believe there is potential for more complex pre-processing models, that could be explicity designed to perform reasoning or long-context understanding with structured representation spaces, leaving LLMs as specialised language generators.

Structured Representations for LLMs An alternative approach to combining pre-training and structure would be to introduce the structure inside the pretrained model directly. Current LLMs based on Transformers represent text as a sequence of dense vectors, relying on the attention mechanism in the model to capture any structural interactions. However, these dense representations are unlikely to be optimal; for example, Sherborne et al. (2023) found that the encoding space in a supposedly multilingual pretrained model was separable by language, implying poor crosslingual generalisation. For tasks involving long inputs, sequences of dense vectors become increasingly unwieldy for a model to handle, with content potentially getting 'lost in the middle' (Liu et al., 2023). Introducing weak structure to the architectures of future LLMs offers a potential solution. We know that long documents (e.g., books) have an implicit hierarchical structure, with information grouped by sentence, paragraph, chapter and so on, but models do not currently exploit this knowledge. Combining our understanding of the structures and invariances of text-to-text generation problems with the efficacy of large-scale training from data offers a rich seam of opportunity for the future.

Appendix A

Separator

A.1 Hyperparameters

The hyperparameters are shown in Table A.1 and were selected by manual tuning, based on a combination of: (a) validation encoding separation, (b) validation BLEU scores using oracle exemplars, and (c) validation iBLEU scores using predicted syntactic codes.

A.2 Reproducibility Notes

All experiments were run on a single Nvidia RTX 2080 Ti GPU. Training time for SEPARATOR was approximately 2 days on Paralex, and 1 day for QQP. SEPARATOR contains a total of 69,139,744 trainable parameters.

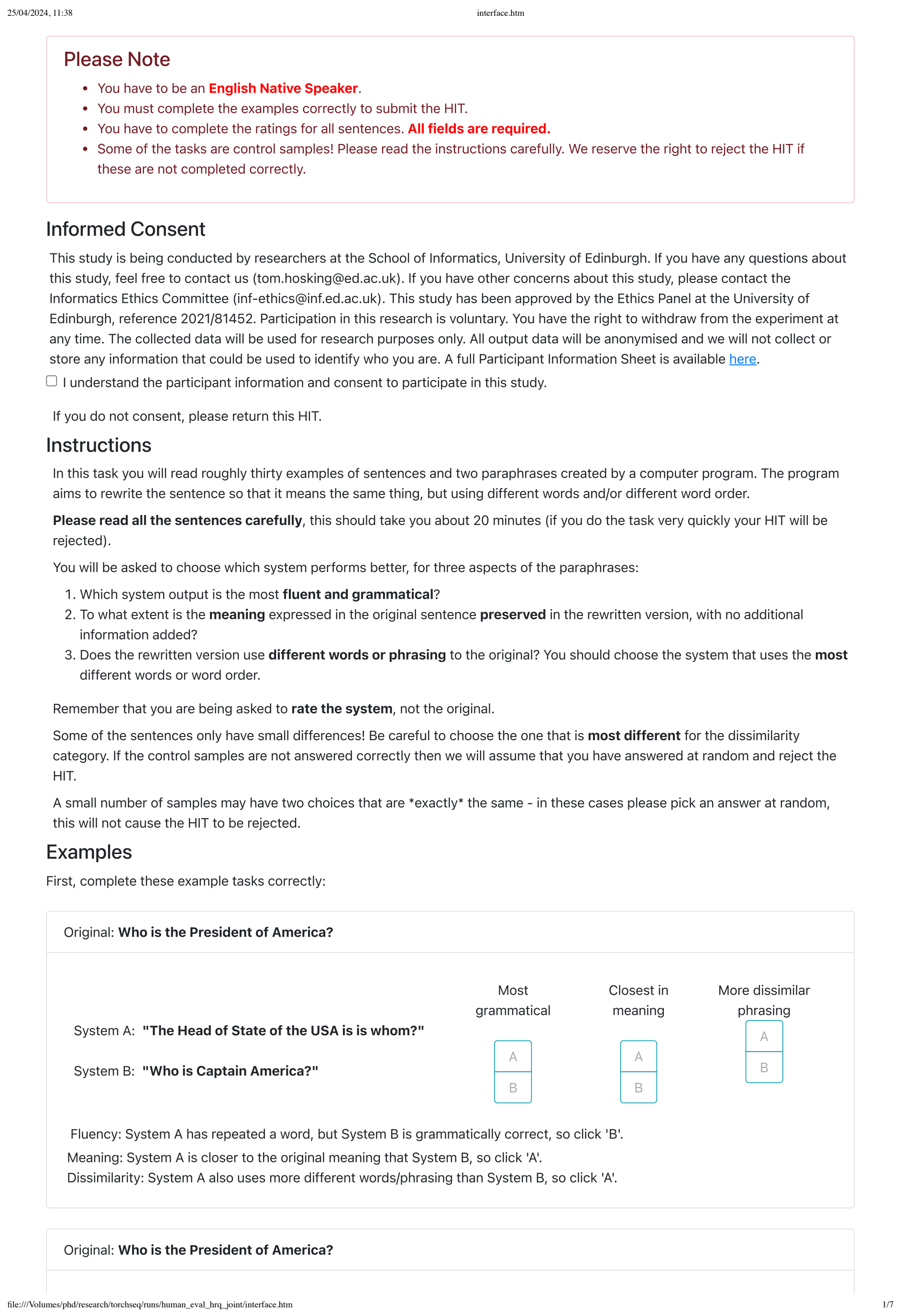
```
Here is a sentence:
"{sentence}"
Please give a single paraphrase of this sentence:
```

Prompt A.1: Paraphrasing prompt for Mistral 7B

Parameter	**Value**
Embedding dimension D	768
Encoder layers	5
Decoder layers	5
Feedforward dimension	2048
Transformer heads	8
Semantic/syntactic heads H_{sem}, H_{syn}	6/2
Quantiser heads $\tilde{H}_{syn}$	4
Quantiser codebook size K	256
Optimizer	Adam (Kingma and Ba, 2015)
Learning rate	0.005
Batch size	64
Token dropout	0.2 (Xie et al., 2017)
Decoder	Beam search
Beam width	4
Commitment weight λ	0.25
Code classifier	
Num. hidden layers	2
Hidden layer size	2712

Table A.1: Hyperparameter values used for our experiments.

A.3 Annotation Interface


25/04/2024, 11:38 interface.htm

Please Note

- You have to be an **English Native Speaker**.
- You must complete the examples correctly to submit the HIT.
- You have to complete the ratings for all sentences. **All fields are required.**
- Some of the tasks are control samples! Please read the instructions carefully. We reserve the right to reject the HIT if these are not completed correctly.

Informed Consent

This study is being conducted by researchers at the School of Informatics, University of Edinburgh. If you have any questions about this study, feel free to contact us (tom.hosking@ed.ac.uk). If you have other concerns about this study, please contact the Informatics Ethics Committee (inf-ethics@inf.ed.ac.uk). This study has been approved by the Ethics Panel at the University of Edinburgh, reference 2021/81452. Participation in this research is voluntary. You have the right to withdraw from the experiment at any time. The collected data will be used for research purposes only. All output data will be anonymised and we will not collect or store any information that could be used to identify who you are. A full Participant Information Sheet is available here.

☐ I understand the participant information and consent to participate in this study.

If you do not consent, please return this HIT.

Instructions

In this task you will read roughly thirty examples of sentences and two paraphrases created by a computer program. The program aims to rewrite the sentence so that it means the same thing, but using different words and/or different word order.

Please read all the sentences carefully, this should take you about 20 minutes (if you do the task very quickly your HIT will be rejected).

You will be asked to choose which system performs better, for three aspects of the paraphrases:

1. Which system output is the most **fluent and grammatical**?
2. To what extent is the **meaning** expressed in the original sentence **preserved** in the rewritten version, with no additional information added?
3. Does the rewritten version use **different words or phrasing** to the original? You should choose the system that uses the **most** different words or word order.

Remember that you are being asked to **rate the system**, not the original.

Some of the sentences only have small differences! Be careful to choose the one that is **most different** for the dissimilarity category. If the control samples are not answered correctly then we will assume that you have answered at random and reject the HIT.

A small number of samples may have two choices that are *exactly* the same - in these cases please pick an answer at random, this will not cause the HIT to be rejected.

Examples

First, complete these example tasks correctly:

Original: **Who is the President of America?**

System A: **"The Head of State of the USA is is whom?"**

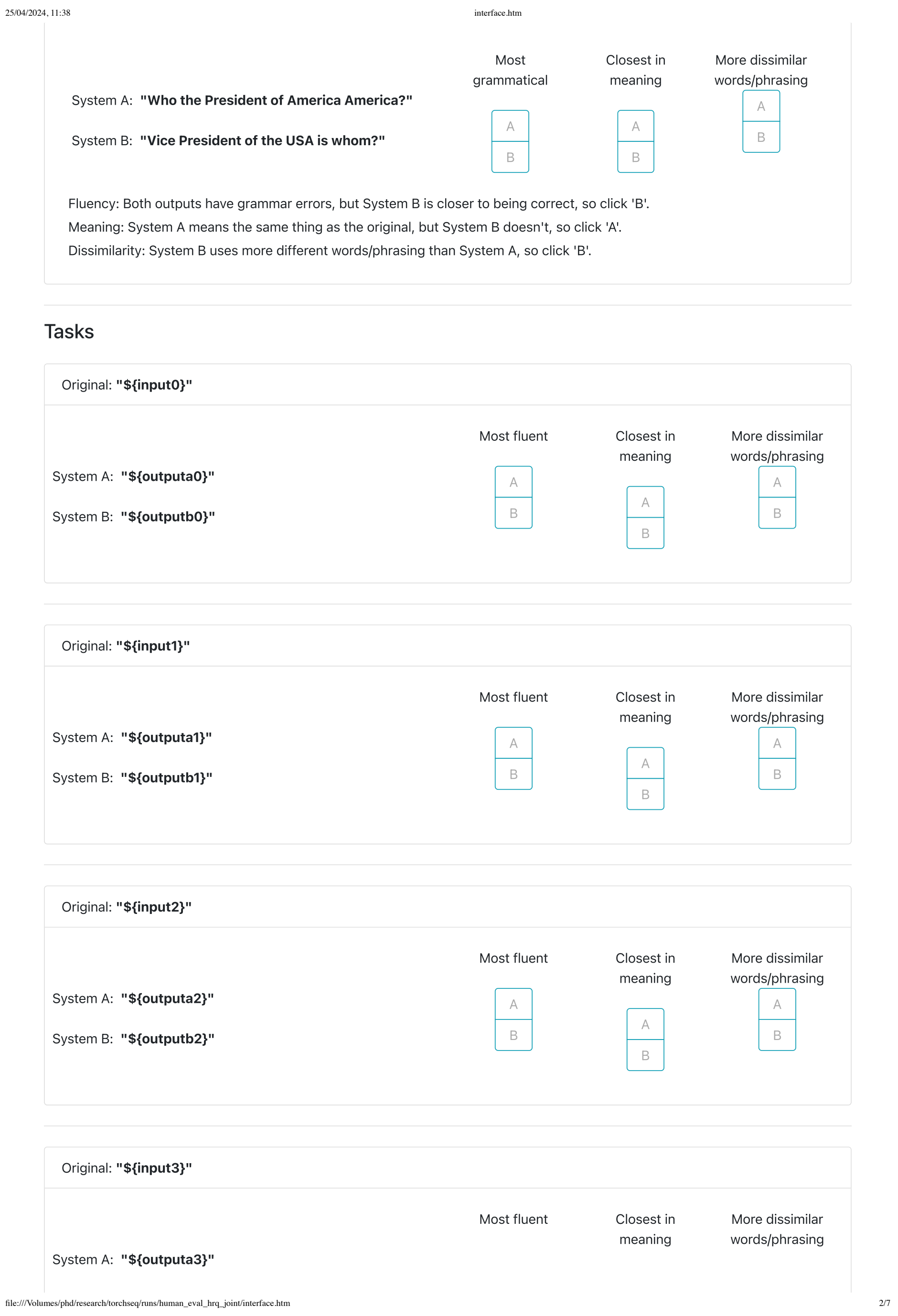
System B: **"Who is Captain America?"**

Most grammatical	Closest in meaning	More dissimilar phrasing
A	A	A
B	B	B

Fluency: System A has repeated a word, but System B is grammatically correct, so click 'B'.
Meaning: System A is closer to the original meaning that System B, so click 'A'.
Dissimilarity: System A also uses more different words/phrasing than System B, so click 'A'.

Original: **Who is the President of America?**

file:///Volumes/phd/research/torchseq/runs/human_eval_hrq_joint/interface.htm 1/7



25/04/2024, 11:38 interface.htm

System A: **"Who the President of America America?"**

System B: **"Vice President of the USA is whom?"**

Fluency: Both outputs have grammar errors, but System B is closer to being correct, so click 'B'.
Meaning: System A means the same thing as the original, but System B doesn't, so click 'A'.
Dissimilarity: System B uses more different words/phrasing than System A, so click 'B'.

Tasks

Original: **"${input0}"**

System A: **"${outputa0}"**

System B: **"${outputb0}"**

Original: **"${input1}"**

System A: **"${outputa1}"**

System B: **"${outputb1}"**

Original: **"${input2}"**

System A: **"${outputa2}"**

System B: **"${outputb2}"**

Original: **"${input3}"**

System A: **"${outputa3}"**

file:///Volumes/phd/research/torchseq/runs/human_eval_hrq_joint/interface.htm 2/7

(continues for remainder of batch)

Appendix B

Calypso

B.1 Hyperparameters

The hyperparameters given in Table B.1 were selected by manual tuning, based on a combination of: (a) validation iBLEU scores with depth masking, (b) validation BLEU scores using oracle sketches, and (c) validation iBLEU scores using predicted syntactic codes.

The Gumbel temperature τ is decayed during training as a function of the step t, according to the following equation:

$$\tau(t) = \max(2 - \frac{2}{1 + e^{t/10000}}, 0.5). \tag{B.1}$$

Intuitively, this smoothly decays τ from an initial value of 2, with a half-life of 10k steps, to a minimum value of 0.5.

Parameter	**Value**
Encoder/decoder	
Embedding dimension D	768
Encoder layers	5
Decoder layers	5
Feedforward dimension	2048
Transformer heads	8
Semantic/syntactic dim	192/594
Depth D	3
Codebook size K	16
Optimizer	Adam (Kingma and Ba, 2015)
Learning rate	0.01
Batch size	64
Token dropout	0.2 (Xie et al., 2017)
Decoder	Beam search
Beam width	4
Code predictor	
Num. hidden layers	2
Hidden layer size	3072

Table B.1: Hyperparameter values used for our experiments.

Appendix C

Hercules

C.1 Replication details

Models were trained on a single A100 GPU, with training taking roughly 24 hours for SPACE and 6 hours for each AmaSum domain.

The prompt used for LLM summarisation was as follows:

```
Review:
[...] (x8)
Write a summary in 70 words or less:
```

Prompt C.1: Baseline LLMs

Table C.1 show the hyperparameters used for our experiments. The Gumbel temperature was decayed from τ_0 to τ_{min} according to

$$\tau = \max\big(\tau_0 \times \exp(-\frac{t}{\gamma_{temp}}), \tau_{min}\big), \tag{C.1}$$

in line with Jang et al. (2017).

We used the default settings for SummaC (Laban et al., 2022) as given on the project GitHub, using the SummaCConv variant trained on VitaminC (Schuster et al., 2021) and mean aggregation.

C.2 Annotation Instructions

Instructions

In this task you will be presented with some reviews of a product/hotel, followed by two summaries produced by different automatic systems. Your task is to rate the system summaries based on the criteria listed below.

Parameter	**Value**
Embedding dim. D	768
Encoder layers	5
Decoder layers	5
Feedforward dim.	2048
Transformer heads	8
Depth D	12
Codebook size K	12
Optimizer	Adam (Kingma and Ba, 2015)
Learning rate	5e-4
Batch size	200
Token dropout	0.2 (Xie et al., 2017)
Decoder	Beam search
Beam width	4
α_{init}	0.5
τ_0	1.0
τ_{min}	0.5
γ_{temp}	33333
β_{KL}	0.0025
β_{NL}	0.05
γ_{NL}	1.5

Table C.1: Hyperparameter values used for our experiments.

First, please skim read through the reviews, to try to get an overall idea of what opinions are mentioned frequently. Then, read the system summaries carefully, and rate each one according to the criteria.

Please read the criteria descriptions and system summaries carefully, and whenever is necessary re-read the summaries or reviews. You might want to use your browser's search function to help find parts of reviews that are relevant.

Criteria

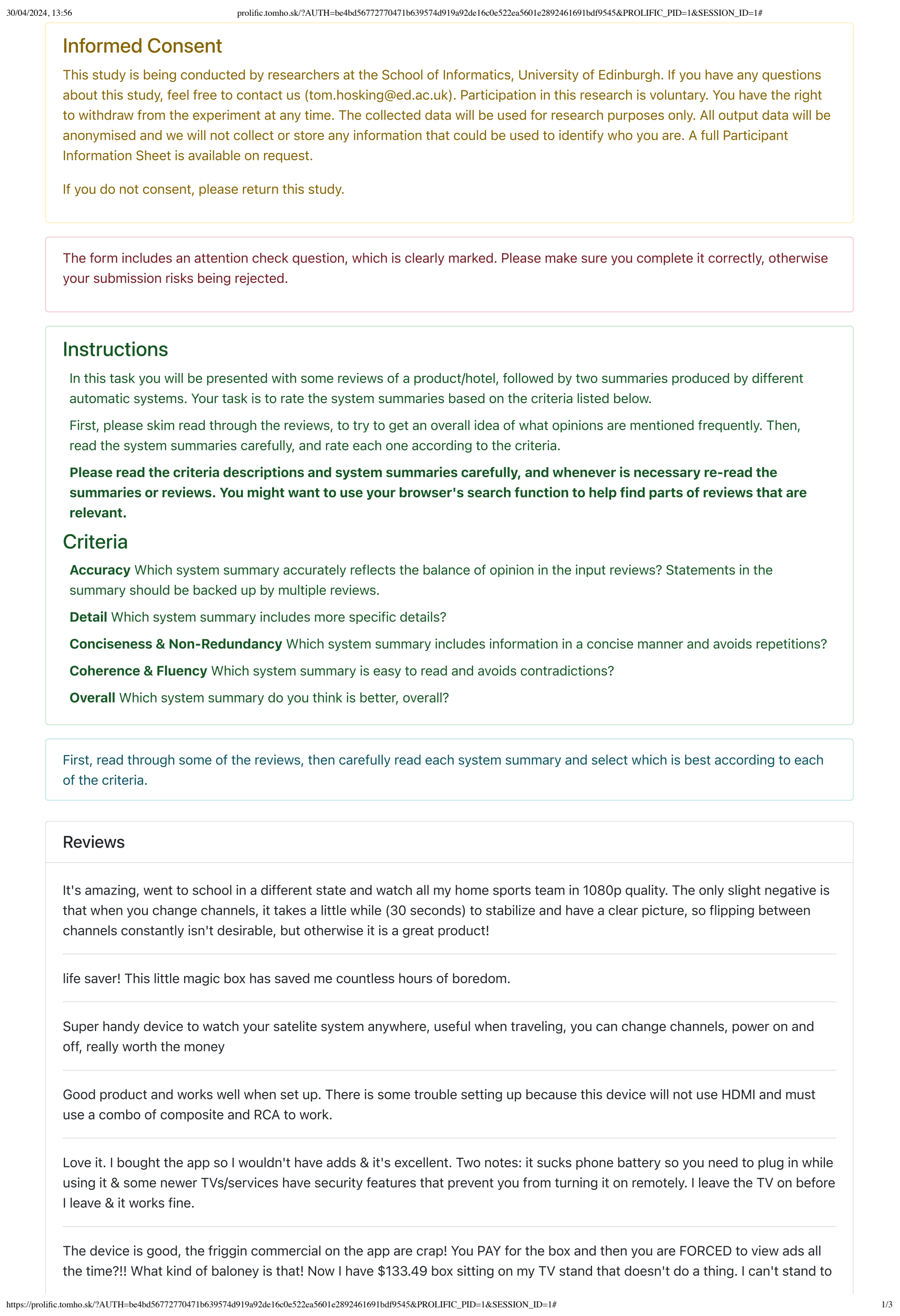
Accuracy – Which system summary accurately reflects the balance of opinion in the input reviews? Statements in the summary should be backed up by multiple reviews.

Detail – Which system summary includes more specific details?

Coherence & Fluency – Which system summary is easy to read and avoids contradictions?

Overall – Which system summary do you think is better, overall?

C.3 Annotation Interface



30/04/2024, 13:56 prolific.tomho.sk/?AUTH=be4bd56772770471b639574d919a92de16c0e522ea5601e2892461691bdf9545&PROLIFIC_PID=1&SESSION_ID=1#

Informed Consent

This study is being conducted by researchers at the School of Informatics, University of Edinburgh. If you have any questions about this study, feel free to contact us (tom.hosking@ed.ac.uk). Participation in this research is voluntary. You have the right to withdraw from the experiment at any time. The collected data will be used for research purposes only. All output data will be anonymised and we will not collect or store any information that could be used to identify who you are. A full Participant Information Sheet is available on request.

If you do not consent, please return this study.

The form includes an attention check question, which is clearly marked. Please make sure you complete it correctly, otherwise your submission risks being rejected.

Instructions

In this task you will be presented with some reviews of a product/hotel, followed by two summaries produced by different automatic systems. Your task is to rate the system summaries based on the criteria listed below.

First, please skim read through the reviews, to try to get an overall idea of what opinions are mentioned frequently. Then, read the system summaries carefully, and rate each one according to the criteria.

Please read the criteria descriptions and system summaries carefully, and whenever is necessary re-read the summaries or reviews. You might want to use your browser's search function to help find parts of reviews that are relevant.

Criteria

Accuracy Which system summary accurately reflects the balance of opinion in the input reviews? Statements in the summary should be backed up by multiple reviews.

Detail Which system summary includes more specific details?

Conciseness & Non-Redundancy Which system summary includes information in a concise manner and avoids repetitions?

Coherence & Fluency Which system summary is easy to read and avoids contradictions?

Overall Which system summary do you think is better, overall?

First, read through some of the reviews, then carefully read each system summary and select which is best according to each of the criteria.

Reviews

It's amazing, went to school in a different state and watch all my home sports team in 1080p quality. The only slight negative is that when you change channels, it takes a little while (30 seconds) to stabilize and have a clear picture, so flipping between channels constantly isn't desirable, but otherwise it is a great product!

life saver! This little magic box has saved me countless hours of boredom.

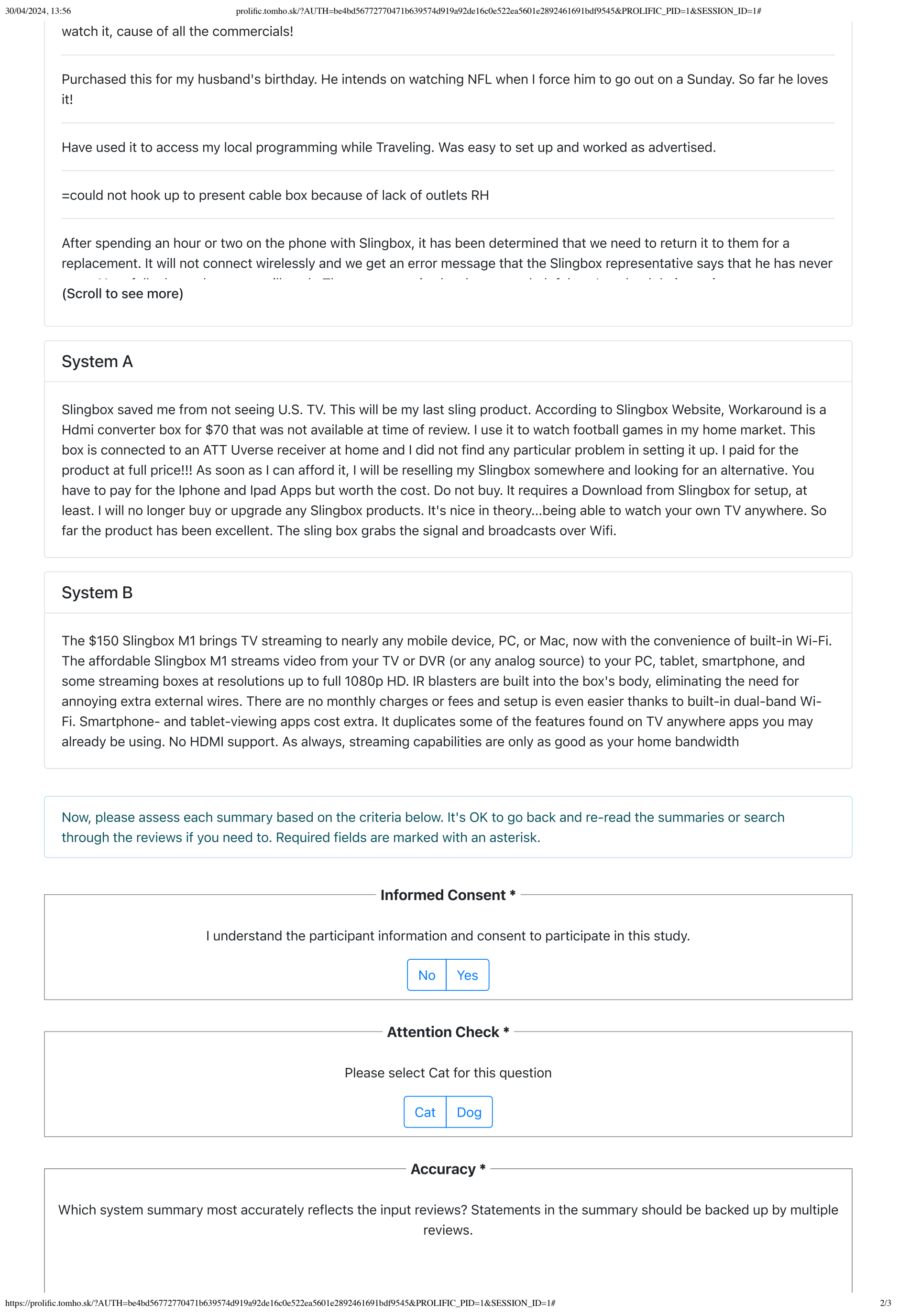
Super handy device to watch your satelite system anywhere, useful when traveling, you can change channels, power on and off, really worth the money

Good product and works well when set up. There is some trouble setting up because this device will not use HDMI and must use a combo of composite and RCA to work.

Love it. I bought the app so I wouldn't have adds & it's excellent. Two notes: it sucks phone battery so you need to plug in while using it & some newer TVs/services have security features that prevent you from turning it on remotely. I leave the TV on before I leave & it works fine.

The device is good, the friggin commercial on the app are crap! You PAY for the box and then you are FORCED to view ads all the time?!! What kind of baloney is that! Now I have $133.49 box sitting on my TV stand that doesn't do a thing. I can't stand to

https://prolific.tomho.sk/?AUTH=be4bd56772770471b639574d919a92de16c0e522ea5601e2892461691bdf9545&PROLIFIC_PID=1&SESSION_ID=1# 1/3



30/04/2024, 13:56 prolific.tomho.sk/?AUTH=be4bd56772770471b639574d919a92de16c0e522ea5601e2892461691bdf9545&PROLIFIC_PID=1&SESSION_ID=1#

watch it, cause of all the commercials!

Purchased this for my husband's birthday. He intends on watching NFL when I force him to go out on a Sunday. So far he loves it!

Have used it to access my local programming while Traveling. Was easy to set up and worked as advertised.

=could not hook up to present cable box because of lack of outlets RH

After spending an hour or two on the phone with Slingbox, it has been determined that we need to return it to them for a replacement. It will not connect wirelessly and we get an error message that the Slingbox representative says that he has never

(Scroll to see more)

System A

Slingbox saved me from not seeing U.S. TV. This will be my last sling product. According to Slingbox Website, Workaround is a Hdmi converter box for $70 that was not available at time of review. I use it to watch football games in my home market. This box is connected to an ATT Uverse receiver at home and I did not find any particular problem in setting it up. I paid for the product at full price!!! As soon as I can afford it, I will be reselling my Slingbox somewhere and looking for an alternative. You have to pay for the Iphone and Ipad Apps but worth the cost. Do not buy. It requires a Download from Slingbox for setup, at least. I will no longer buy or upgrade any Slingbox products. It's nice in theory...being able to watch your own TV anywhere. So far the product has been excellent. The sling box grabs the signal and broadcasts over Wifi.

System B

The $150 Slingbox M1 brings TV streaming to nearly any mobile device, PC, or Mac, now with the convenience of built-in Wi-Fi. The affordable Slingbox M1 streams video from your TV or DVR (or any analog source) to your PC, tablet, smartphone, and some streaming boxes at resolutions up to full 1080p HD. IR blasters are built into the box's body, eliminating the need for annoying extra external wires. There are no monthly charges or fees and setup is even easier thanks to built-in dual-band Wi-Fi. Smartphone- and tablet-viewing apps cost extra. It duplicates some of the features found on TV anywhere apps you may already be using. No HDMI support. As always, streaming capabilities are only as good as your home bandwidth

Now, please assess each summary based on the criteria below. It's OK to go back and re-read the summaries or search through the reviews if you need to. Required fields are marked with an asterisk.

Informed Consent *

I understand the participant information and consent to participate in this study.

No Yes

Attention Check *

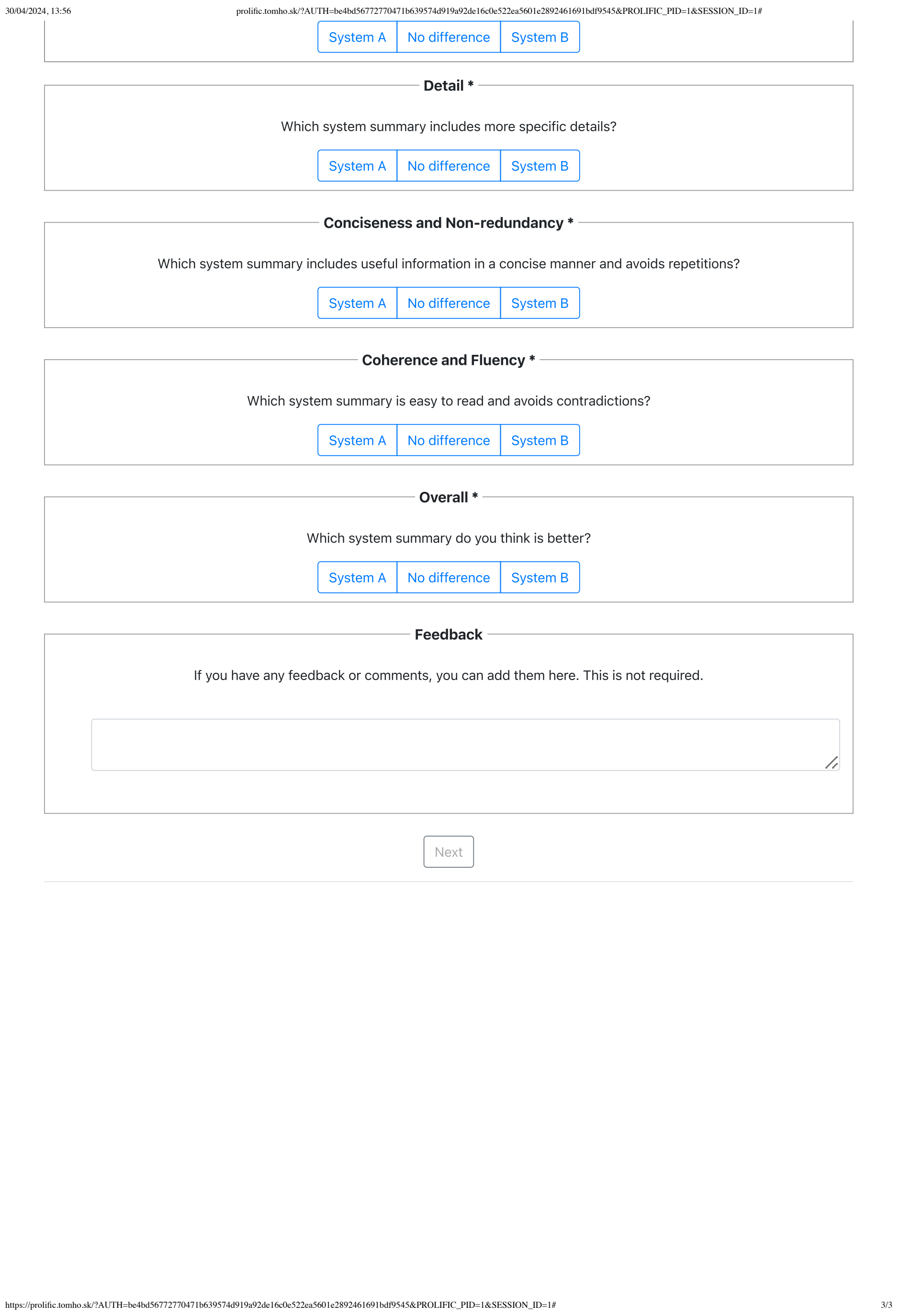
Please select Cat for this question

Cat Dog

Accuracy *

Which system summary most accurately reflects the input reviews? Statements in the summary should be backed up by multiple reviews.

https://prolific.tomho.sk/?AUTH=be4bd56772770471b639574d919a92de16c0e522ea5601e2892461691bdf9545&PROLIFIC_PID=1&SESSION_ID=1# 2/3



30/04/2024, 13:56 prolific.tomho.sk/?AUTH=be4bd56772770471b639574d919a92de16c0e522ea5601e2892461691bdf9545&PROLIFIC_PID=1&SESSION_ID=1#

System A | No difference | System B

Detail *

Which system summary includes more specific details?

System A | No difference | System B

Conciseness and Non-redundancy *

Which system summary includes useful information in a concise manner and avoids repetitions?

System A | No difference | System B

Coherence and Fluency *

Which system summary is easy to read and avoids contradictions?

System A | No difference | System B

Overall *

Which system summary do you think is better?

System A | No difference | System B

Feedback

If you have any feedback or comments, you can add them here. This is not required.

Next

https://prolific.tomho.sk/?AUTH=be4bd56772770471b639574d919a92de16c0e522ea5601e2892461691bdf9545&PROLIFIC_PID=1&SESSION_ID=1# 3/3

C.4 Breakdown of Results

We report the automatic evaluation scores broken down by AmaSum domains in Table C.2 and Table C.3, and the human evaluation results broken down by dataset in Appendix C.4.

	System	Electronics		Home/Kitchen		Shoes		Sports/Outdoors	
		R-2 ↑	R-L ↑	R-2 ↑	R-L ↑	R-2 ↑	R-L ↑	R-2 ↑	R-L ↑
Extractive	Random	0.95	9.51	1.09	9.78	0.76	8.41	1.29	10.17
	Centroid	1.78	11.53	2.50	12.47	1.63	9.43	2.07	11.41
	LexRank	2.47	12.18	3.22	12.84	1.96	10.51	3.00	13.29
	QT	1.55	10.95	1.79	12.15	1.23	11.13	1.46	11.43
	SemAE	1.32	10.97	2.37	12.95	1.32	9.64	1.32	11.48
	HERCULES$_{ext}$	3.29	12.48	3.19	12.89	2.67	11.75	3.00	12.93
Abstractive	CopyCat	1.46	11.92	2.11	11.86	0.98	9.00	1.46	12.07
	InstructGPT	2.83	13.89	2.99	14.39	2.23	12.52	2.80	13.76
	BiMeanVAE	2.32	12.42	2.32	12.93	1.46	11.80	2.05	12.81
	COOP	3.46	14.56	2.66	14.22	2.78	13.39	2.28	14.31
	HERCULES$_{abs}$	2.46	12.57	2.22	11.53	1.80	11.77	1.72	11.21

Table C.2: Results for ROUGE scores with respect to references on AmaSum, broken down by product category.

	System	Electronics		Home/Kitchen		Shoes		Sports/Outdoors	
		SC$_{refs}$ ↑	SC$_{in}$ ↑	SC$_{refs}$ ↑	SC$_{in}$ ↑	SC$_{refs}$ ↑	SC$_{in}$ ↑	SC$_{refs}$ ↑	SC$_{in}$ ↑
Extractive	Random	22.55	57.27	22.87	57.87	22.20	62.80	22.03	58.72
	Centroid	23.71	62.81	23.66	66.18	22.80	67.18	23.71	62.33
	LexRank	24.04	68.36	22.49	57.91	25.22	84.44	22.19	58.11
	QT	22.32	59.32	22.80	65.18	22.38	73.91	22.19	66.43
	SemAE	22.07	55.64	21.81	53.13	21.82	63.59	21.61	56.39
	HERCULES$_{ext}$	25.36	82.79	24.45	81.36	22.92	86.39	24.79	85.66
Abstractive	CopyCat	24.45	64.23	22.02	53.10	22.42	69.36	23.06	65.36
	InstructGPT	22.42	47.50	21.55	44.20	22.10	47.57	21.40	43.24
	BiMeanVAE	21.88	45.48	21.86	50.81	21.59	58.56	21.79	55.29
	COOP	22.95	53.23	22.74	63.97	22.14	60.76	22.19	55.45
	HERCULES$_{abs}$	25.55	79.49	25.25	82.15	24.59	85.09	25.53	84.16
	(References)	87.69	63.72	87.11	65.12	85.49	69.86	86.73	67.58

Table C.3: Results for automatic faithfulness metrics on AmaSum, broken down by product category.

		SPACE			**AmaSum**		
	System	Info ↑	Cohe ↑	Conc ↑	Info ↑	Cohe ↑	Conc ↑
Extractive	Random	-5.33	-2.67	-4.67	-14.04	3.07	-1.75
	LexRank	-27.33	-39.33	-44.67	7.05	-5.29	-4.41
	QT	-8.67	-10.00	-4.67	-7.42	-0.87	5.68
	HERCULES$_{\text{ext}}$	1.33	-2.00	0.00	-1.54	-1.98	5.05
	(Gold)	40.00	54.00	54.00	20.00	6.30	-5.75
Abstractive	Random	-18.67	-2.67	-11.33	-22.67	-6.22	-1.78
	InstructGPT	-10.67	5.33	18.00	11.11	2.22	0.89
	COOP	-12.00	-24.67	-20.00	-4.00	-0.89	0.89
	HERCULES$_{\text{abs}}$	4.67	-16.00	-18.67	-1.78	-8.44	-5.33
	(References)	36.67	38.00	32.00	21.67	16.67	6.67

Table C.4: Breakdown of human evaluation results by dataset.

Appendix D

HIRO

D.1 Hyperparameters

Parameter	**Value**
Embedding dim. $\mathbb{D}$	768
Encoder layers	5
Feedforward dim.	2048
Transformer heads	8
Depth D	12
Codebook size K	12
Optimizer	Adam (Kingma and Ba, 2015)
Learning rate	1e-4
Batch size	384
ω	150
α_{init}	0.5
τ_0	1.0
τ_{min}	0.5
γ_{temp}	33333
β_{KL}	0.0025
β_{NL}	0.05
γ_{NL}	1.5
top-k subpaths	8
$\mathrm{tp-ibp}$ smoothing α	6 (SPACE), 3 (AmaSum)

Table D.1: Hyperparameter values used for our experiments.

D.2 LLM prompts

```
Review:
[...] (x8)
Write a summary in 70 words or less:
```

Prompt D.1: Baseline LLMs

```
Here is a list of sentences taken from reviews of the {entity name}:

[...]

In no more than 10 words, write a single concise sentence that includes the
     main point:
```

Prompt D.2: HIRO$_{\text{sent}}$

```
Here is a list of sentences taken from reviews of the {entity name}:

[...]

In no more than 60 words, write a concise summary that includes the main
    points:
```

Prompt D.3: HIRO$_{\text{doc}}$

Bibliography

Aharoni, R., Narayan, S., Maynez, J., Herzig, J., Clark, E., and Lapata, M. (2023). Multilingual summarization with factual consistency evaluation. In Rogers, A., Boyd-Graber, J., and Okazaki, N., editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 3562–3591, Toronto, Canada. Association for Computational Linguistics.

Akbik, A., Blythe, D., and Vollgraf, R. (2018). Contextual string embeddings for sequence labeling. In Bender, E. M., Derczynski, L., and Isabelle, P., editors, *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Amplayo, R. K., Angelidis, S., and Lapata, M. (2021a). Aspect-controllable opinion summarization. In Moens, M.-F., Huang, X., Specia, L., and Yih, S. W.-t., editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6578–6593, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Amplayo, R. K., Angelidis, S., and Lapata, M. (2021b). Unsupervised opinion summarization with content planning. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 12489–12497. AAAI Press.

Angelidis, S., Amplayo, R. K., Suhara, Y., Wang, X., and Lapata, M. (2021). Extractive opinion summarization in quantized transformer spaces. *Transactions of the Association for Computational Linguistics*, 9:277–293.

Arvan, M. and Parde, N. (2024). ReproHum #0712-01: Human evaluation reproduction report for "hierarchical sketch induction for paraphrase generation". In Balloccu, S., Belz, A., Huidrom, R., Reiter, E., Sedoc, J., and Thomson, C., editors, *Proceedings of the Fourth Workshop on Human Evaluation of NLP Systems (HumEval) @ LREC-COLING 2024*, pages 210–220, Torino, Italia. ELRA and ICCL.

Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In Bengio, Y. and LeCun, Y., editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Banarescu, L., Bonial, C., Cai, S., Georgescu, M., Griffitt, K., Hermjakob, U., Knight, K., Koehn, P., Palmer, M., and Schneider, N. (2013). Abstract Meaning Representation for sembanking. In Pareja-Lora, A., Liakata, M., and Dipper, S., editors, *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria. Association for Computational Linguistics.

Banko, M., Mittal, V. O., and Witbrock, M. J. (2000). Headline generation based on statistical translation. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 318–325, Hong Kong. Association for Computational Linguistics.

Bannard, C. and Callison-Burch, C. (2005). Paraphrasing with bilingual parallel corpora. In Knight, K., Ng, H. T., and Oflazer, K., editors, *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 597–604, Ann Arbor, Michigan. Association for Computational Linguistics.

Bao, Y., Zhou, H., Huang, S., Li, L., Mou, L., Vechtomova, O., Dai, X.-y., and Chen, J. (2019). Generating sentences from disentangled syntactic and semantic spaces. In Korhonen, A., Traum, D., and Màrquez, L., editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6008–6019, Florence, Italy. Association for Computational Linguistics.

Barzilay, R. and McKeown, K. R. (2001). Extracting paraphrases from a parallel corpus. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*, pages 50–57, Toulouse, France. Association for Computational Linguistics.

Basu Roy Chowdhury, S., Zhao, C., and Chaturvedi, S. (2022). Unsupervised extractive opinion summarization using sparse coding. In Muresan, S., Nakov, P., and Villavicencio, A., editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1209–1225, Dublin, Ireland. Association for Computational Linguistics.

Beineke, P., Hastie, T., and Vaithyanathan, S. (2004). The sentimental factor: Improving review classification via human-provided information. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 263–270, Barcelona, Spain.

Beltagy, I., Peters, M. E., and Cohan, A. (2020). Longformer: The long-document transformer. *arXiv:2004.05150*.

Bengio, Y., Léonard, N., and Courville, A. C. (2013). Estimating or propagating gradients through stochastic neurons for conditional computation. *CoRR*, abs/1308.3432.

Bhaskar, A., Fabbri, A., and Durrett, G. (2023). Prompted opinion summarization with GPT-3.5. In Rogers, A., Boyd-Graber, J., and Okazaki, N., editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 9282–9300, Toronto, Canada. Association for Computational Linguistics.

Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. (2017). Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877.

Bowman, S. R., Vilnis, L., Vinyals, O., Dai, A., Jozefowicz, R., and Bengio, S. (2016). Generating sentences from a continuous space. In Riezler, S. and Goldberg, Y., editors, *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*, pages 10–21, Berlin, Germany. Association for Computational Linguistics.

Bowman, S. R., Vilnis, L., Vinyals, O., Dai, A. M., Józefowicz, R., and Bengio, S. (2015). Generating sentences from a continuous space. *CoRR*, abs/1511.06349.

Bražinskas, A., Lapata, M., and Titov, I. (2020). Unsupervised opinion summarization as copycat-review generation. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5151–5169, Online. Association for Computational Linguistics.

Bražinskas, A., Lapata, M., and Titov, I. (2021). Learning opinion summarizers by selecting informative reviews. In Moens, M.-F., Huang, X., Specia, L., and Yih, S. W.-t., editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9424–9442, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Bronstein, M. M., Bruna, J., Cohen, T., and Veličković, P. (2021). Geometric deep learning: Grids, groups, graphs, geodesics, and gauges.

Brown, P. F., Della Pietra, S. A., Della Pietra, V. J., and Mercer, R. L. (1993). The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language models are few-shot learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Callison-Burch, C., Osborne, M., and Koehn, P. (2006). Re-evaluating the role of Bleu in machine translation research. In McCarthy, D. and Wintner, S., editors, *11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 249–256, Trento, Italy. Association for Computational Linguistics.

Cao, Y. and Wan, X. (2020). DivGAN: Towards diverse paraphrase generation via diversified generative adversarial network. In Cohn, T., He, Y., and Liu, Y., editors, *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2411–2421, Online. Association for Computational Linguistics.

Carnie, A., Siddiqi, D., and Sato, Y. (2014). *The Routledge Handbook of Syntax*. Routledge Handbooks in Linguistics. Taylor & Francis.

Cattan, A., Eden, L., Kantor, Y., and Bar-Haim, R. (2023). From key points to key point hierarchy: Structured and expressive opinion summarization. In Rogers, A., Boyd-Graber, J., and Okazaki, N., editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 912–928, Toronto, Canada. Association for Computational Linguistics.

Celikyilmaz, A., Clark, E., and Gao, J. (2020). Evaluation of text generation: A survey. *CoRR*, abs/2006.14799.

Chadebec, C., Vincent, L., and Allassonniere, S. (2022). Pythae: Unifying generative autoencoders in python - a benchmarking use case. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A., editors, *Advances in Neural Information Processing Systems*, volume 35, pages 21575–21589. Curran Associates, Inc.

Chen, M., Tang, Q., Wiseman, S., and Gimpel, K. (2019a). Controllable paraphrase generation with a syntactic exemplar. In Korhonen, A., Traum, D., and Màrquez, L., editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5972–5984, Florence, Italy. Association for Computational Linguistics.

Chen, M., Tang, Q., Wiseman, S., and Gimpel, K. (2019b). A multi-task approach for disentangling syntax and semantics in sentence representations. In Burstein, J., Doran, C., and Solorio, T., editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2453–2464, Minneapolis, Minnesota. Association for Computational Linguistics.

Chen, Q. and Fuge, M. (2024). Compressing latent space via least volume. In *The Twelfth International Conference on Learning Representations*.

Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., and Abbeel, P. (2016). Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

Chen, Y., Tao, G., and Wang, C. (2010). Approximate nearest neighbor search by residual vector quantization. *Sensors (Basel, Switzerland)*, 10:11259–73.

Cho, K., van Merriënboer, B., Bahdanau, D., and Bengio, Y. (2014a). On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar. Association for Computational Linguistics.

Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014b). Learning phrase representations using RNN encoder–decoder for statistical machine translation. In Moschitti, A., Pang, B., and Daelemans, W., editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Chopra, S., Hadsell, R., and LeCun, Y. (2005). Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546 vol. 1.

Chu, E. and Liu, P. J. (2019). Meansum: A neural model for unsupervised multi-document abstractive summarization. In Chaudhuri, K. and Salakhutdinov, R., editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 1223–1232. PMLR.

Church, A. (1941). *The Calculi of Lambda Conversion. (AM-6).* Princeton University Press.

Clark, E., Rijhwani, S., Gehrmann, S., Maynez, J., Aharoni, R., Nikolaev, V., Sellam, T., Siddhant, A., Das, D., and Parikh, A. (2023). SEAHORSE: A multilingual, multifaceted dataset for summarization evaluation. In Bouamor, H., Pino, J., and Bali, K., editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9397–9413, Singapore. Association for Computational Linguistics.

Coavoux, M., Elsahar, H., and Gallé, M. (2019). Unsupervised aspect-based multi-document abstractive summarization. In Wang, L., Cheung, J. C. K., Carenini, G., and Liu, F., editors, *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 42–47, Hong Kong, China. Association for Computational Linguistics.

Coster, W. and Kauchak, D. (2011). Simple English Wikipedia: A new text simplification task. In Lin, D., Matsumoto, Y., and Mihalcea, R., editors, *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 665–669, Portland, Oregon, USA. Association for Computational Linguistics.

de Marneffe, M.-C. and Manning, C. D. (2008). The Stanford typed dependencies representation. In Bos, J., Briscoe, E., Cahill, A., Carroll, J., Clark, S., Copestake, A., Flickinger, D., van Genabith, J., Hockenmaier, J., Joshi, A., Kaplan, R., King, T. H., Kuebler, S., Lin, D., Lønning, J. T., Manning, C., Miyao, Y., Nivre, J., Oepen, S., Sagae, K., Xue, N., and Zhang, Y., editors, *Coling 2008: Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation*, pages 1–8, Manchester, UK. Coling 2008 Organizing Committee.

Deutsch, D., Bedrax-Weiss, T., and Roth, D. (2021). Towards question-answering as an automatic metric for evaluating the content quality of a summary. *Transactions of the Association for Computational Linguistics*, 9:774–789.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In Burstein, J., Doran, C., and Solorio, T., editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Dolan, B., Quirk, C., and Brockett, C. (2004). Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 350–356, Geneva, Switzerland. COLING.

Dong, C., Li, Y., Gong, H., Chen, M., Li, J., Shen, Y., and Yang, M. (2022). A survey of natural language generation. *ACM Comput. Surv.*, 55(8).

Dong, L., Mallinson, J., Reddy, S., and Lapata, M. (2017). Learning to paraphrase for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 875–886, Copenhagen, Denmark. Association for Computational Linguistics.

Douze, M., Guzhva, A., Deng, C., Johnson, J., Szilvasy, G., Mazaré, P., Lomeli, M., Hosseini, L., and Jégou, H. (2024). The faiss library. *CoRR*, abs/2401.08281.

Du, X., Shao, J., and Cardie, C. (2017). Learning to ask: Neural question generation for reading comprehension. In Barzilay, R. and Kan, M.-Y., editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1342–1352, Vancouver, Canada. Association for Computational Linguistics.

Erkan, G. and Radev, D. R. (2004). LexRank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22:457–479.

Fabbri, A., Wu, C.-S., Liu, W., and Xiong, C. (2022). QAFactEval: Improved QA-based factual consistency evaluation for summarization. In Carpuat, M., de Marneffe, M.-C., and Meza Ruiz, I. V., editors, *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2587–2601, Seattle, United States. Association for Computational Linguistics.

Fabbri, A. R., Kryściński, W., McCann, B., Xiong, C., Socher, R., and Radev, D. (2021). SummEval: Re-evaluating Summarization Evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409.

Fader, A., Zettlemoyer, L., and Etzioni, O. (2013). Paraphrase-driven learning for open question answering. In Schuetze, H., Fung, P., and Poesio, M., editors, *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1608–1618, Sofia, Bulgaria. Association for Computational Linguistics.

Fan, A., Jernite, Y., Perez, E., Grangier, D., Weston, J., and Auli, M. (2019). ELI5: Long form question answering. In Korhonen, A., Traum, D., and Màrquez, L., editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3558–3567, Florence, Italy. Association for Computational Linguistics.

Freitag, M., Rei, R., Mathur, N., Lo, C.-k., Stewart, C., Avramidis, E., Kocmi, T., Foster, G., Lavie, A., and Martins, A. F. T. (2022). Results of WMT22 metrics shared task: Stop using BLEU – neural metrics are better and more robust. In Koehn, P., Barrault, L., Bojar, O., Bougares, F., Chatterjee, R., Costa-jussà, M. R., Federmann, C., Fishel, M., Fraser, A., Freitag, M., Graham, Y., Grundkiewicz, R., Guzman, P., Haddow, B., Huck, M., Jimeno Yepes, A., Kocmi, T., Martins, A., Morishita, M., Monz, C., Nagata, M., Nakazawa, T., Negri, M., Névéol, A., Neves, M., Popel, M., Turchi, M., and Zampieri, M., editors, *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 46–68, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Friederici, A. D. (2011). The brain basis of language processing: From structure to function. *Physiological Reviews*, 91(4):1357–1392. PMID: 22013214.

Fu, Y., Feng, Y., and Cunningham, J. P. (2019). Paraphrase generation with latent bag of words. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 32, pages 13645–13656. Curran Associates, Inc.

Ganesan, K., Zhai, C., and Han, J. (2010). Opinosis: A graph based approach to abstractive summarization of highly redundant opinions. In Huang, C.-R. and Jurafsky, D., editors, *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 340–348, Beijing, China. Coling 2010 Organizing Committee.

Ganitkevitch, J., Van Durme, B., and Callison-Burch, C. (2013). PPDB: The paraphrase database. In Vanderwende, L., Daumé III, H., and Kirchhoff, K., editors, *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 758–764, Atlanta, Georgia. Association for Computational Linguistics.

Gao, T., Yao, X., and Chen, D. (2021). SimCSE: Simple contrastive learning of sentence embeddings. In Moens, M.-F., Huang, X., Specia, L., and Yih, S. W.-t., editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Garcez, A. D. and Lamb, L. C. (2023). Neurosymbolic ai: the 3rd wave. *Artificial Intelligence Review*, 56(11):12387–12406. This version of the article has been accepted for publication, after peer review and is subject to Springer Nature's AM terms of use, but is not the Version of Record and does not reflect post-acceptance improvements, or any corrections. The Version of Record will be available online at: http://link.springer.com/journal/10462.

Gehrmann, S., Adewumi, T., Aggarwal, K., Ammanamanchi, P. S., Aremu, A., Bosselut, A., Chandu, K. R., Clinciu, M.-A., Das, D., Dhole, K., Du, W., Durmus, E., Dušek, O., Emezue, C. C., Gangal, V., Garbacea, C., Hashimoto, T., Hou, Y., Jernite, Y., Jhamtani, H., Ji, Y., Jolly, S., Kale, M., Kumar, D., Ladhak, F., Madaan, A., Maddela, M., Mahajan, K., Mahamood, S., Majumder, B. P., Martins, P. H., McMillan-Major, A., Mille, S., van Miltenburg, E., Nadeem, M., Narayan, S., Nikolaev, V., Niyongabo Rubungo, A., Osei, S., Parikh, A., Perez-Beltrachini, L., Rao, N. R., Raunak, V., Rodriguez, J. D., Santhanam, S., Sedoc, J., Sellam, T., Shaikh, S., Shimorina, A., Sobrevilla Cabezudo, M. A., Strobelt, H., Subramani, N., Xu, W., Yang, D., Yerukola, A., and Zhou, J. (2021). The GEM benchmark: Natural language generation, its evaluation and metrics. In Bosselut, A., Durmus, E., Gangal, V. P., Gehrmann, S., Jernite, Y., Perez-Beltrachini, L., Shaikh, S., and Xu, W., editors, *Proceedings of the 1st Workshop on Natural Language Generation, Evaluation, and Metrics (GEM 2021)*, pages 96–120, Online. Association for Computational Linguistics.

Gerani, S., Mehdad, Y., Carenini, G., Ng, R. T., and Nejat, B. (2014). Abstractive summarization of product reviews using discourse structure. In Moschitti, A., Pang, B., and Daelemans, W., editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1602–1613, Doha, Qatar. Association for Computational Linguistics.

Gersho, A. and Gray, R. (1991). *Vector Quantization and Signal Compression*. The Springer International Series in Engineering and Computer Science. Springer US.

Glorot, X., Bordes, A., and Bengio, Y. (2011). Deep sparse rectifier neural networks. In Gordon, G., Dunson, D., and Dudík, M., editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 315–323, Fort Lauderdale, FL, USA. PMLR.

Golchin, S. and Surdeanu, M. (2024). Time travel in LLMs: Tracing data contamination in large language models. In *The Twelfth International Conference on Learning Representations*.

Goyal, T. and Durrett, G. (2020). Neural syntactic preordering for controlled paraphrase generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 238–252, Online. Association for Computational Linguistics.

Goyal, T., Li, J. J., and Durrett, G. (2022). News summarization and evaluation in the era of GPT-3. *arXiv preprint*.

Gray, R. (1984). Vector quantization. *IEEE ASSP Magazine*, 1(2):4–29.

Gu, A., Goel, K., and Ré, C. (2022). Efficiently modeling long sequences with structured state spaces. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Guo, Z., Huang, Z., Zhu, K. Q., Chen, G., Zhang, K., Chen, B., and Huang, F. (2021). Automatically paraphrasing via sentence reconstruction and round-trip translation. In Zhou, Z.-H., editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 3815–3821. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Gutmann, M. and Hyvärinen, A. (2010). Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In Teh, Y. W. and Titterington, M., editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 297–304, Chia Laguna Resort, Sardinia, Italy. PMLR.

Hamilton, K., Nayak, A., Božić, B., and Longo, L. (2022). Is neuro-symbolic AI meeting its promises in natural language processing? A structured review. *Semantic Web*, Preprint(Preprint):1–42. Publisher: IOS Press.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society.

He, P., Gao, J., and Chen, W. (2021). Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing.

He, R. and McAuley, J. (2016). Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the 25th International Conference on World Wide Web*, WWW '16, page 507–517, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Heilman, M. and Smith, N. A. (2010). Good question! statistical ranking for question generation. In Kaplan, R., Burstein, J., Harper, M., and Penn, G., editors, *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 609–617, Los Angeles, California. Association for Computational Linguistics.

Hinton, G. E. and Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507.

Hirschman, L., Light, M., Breck, E., and Burger, J. D. (1999). Deep read: A reading comprehension system. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 325–332, College Park, Maryland, USA. Association for Computational Linguistics.

Hochreiter, S. and Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Hofstätter, S., Althammer, S., Schröder, M., Sertkan, M., and Hanbury, A. (2021). Improving efficient neural ranking models with cross-architecture knowledge distillation.

Hornik, K., Stinchcombe, M., and White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366.

Hosking, T., Blunsom, P., and Bartolo, M. (2023a). Human feedback is not gold standard.

Hosking, T. and Lapata, M. (2021). Factorising meaning and form for intent-preserving paraphrasing. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1405–1418, Online. Association for Computational Linguistics.

Hosking, T. and Riedel, S. (2019). Evaluating rewards for question generation models. In Burstein, J., Doran, C., and Solorio, T., editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2278–2283, Minneapolis, Minnesota. Association for Computational Linguistics.

Hosking, T., Tang, H., and Lapata, M. (2022). Hierarchical sketch induction for paraphrase generation. In Muresan, S., Nakov, P., and Villavicencio, A., editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2489–2501, Dublin, Ireland. Association for Computational Linguistics.

Hosking, T., Tang, H., and Lapata, M. (2023b). Attributable and scalable opinion summarization. In Rogers, A., Boyd-Graber, J., and Okazaki, N., editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8488–8505, Toronto, Canada. Association for Computational Linguistics.

Hosking, T., Tang, H., and Lapata, M. (2024). Hierarchical indexing for retrieval-augmented opinion summarization. *Transactions of the Association for Computational Linguistics*, 12:1533–1555.

Hu, J. E., Rudinger, R., Post, M., and Durme, B. V. (2019). Parabank: Monolingual bitext generation and sentential paraphrasing via lexically-constrained neural machine translation. *CoRR*, abs/1901.03644.

Hu, M. and Liu, B. (2004). Mining and summarizing customer reviews. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, page 168–177, New York, NY, USA. Association for Computing Machinery.

Huang, K.-H. and Chang, K.-W. (2021a). Generating syntactically controlled paraphrases without using annotated parallel pairs. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1022–1033, Online. Association for Computational Linguistics.

Huang, K.-H. and Chang, K.-W. (2021b). Generating syntactically controlled paraphrases without using annotated parallel pairs. In Merlo, P., Tiedemann, J., and Tsarfaty, R., editors, *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1022–1033, Online. Association for Computational Linguistics.

Huang, M., Mao, Z., Chen, Z., and Zhang, Y. (2023). Towards accurate image coding: Improved autoregressive image generation with dynamic vector quantization. *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 22596–22605.

Hubert, L. and Arabie, P. (1985). Comparing partitions. *Journal of Classification*, 2(1):193–218.

Huck, G. and Goldsmith, J. (1996). *Ideology and Linguistic Theory: Noam Chomsky and the Deep Structure Debates*. Routledge history of linguistic thought series. Routledge.

Iso, H., Wang, X., Angelidis, S., and Suhara, Y. (2022). Comparative opinion summarization via collaborative decoding. In Muresan, S., Nakov, P., and Villavicencio, A., editors, *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3307–3324, Dublin, Ireland. Association for Computational Linguistics.

Iso, H., Wang, X., Suhara, Y., Angelidis, S., and Tan, W.-C. (2021). Convex Aggregation for Opinion Summarization. In Moens, M.-F., Huang, X., Specia, L., and Yih, S. W.-t., editors, *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3885–3903, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Iyyer, M., Wieting, J., Gimpel, K., and Zettlemoyer, L. (2018). Adversarial example generation with syntactically controlled paraphrase networks. In Walker, M., Ji, H., and Stent, A., editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1875–1885, New Orleans, Louisiana. Association for Computational Linguistics.

Izacard, G., Caron, M., Hosseini, L., Riedel, S., Bojanowski, P., Joulin, A., and Grave, E. (2022). Unsupervised dense information retrieval with contrastive learning. *Transactions on Machine Learning Research*.

Jang, E., Gu, S., and Poole, B. (2017). Categorical reparameterization with Gumbel-softmax. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., de las Casas, D., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., Lavaud, L. R., Lachaux, M.-A., Stock, P., Scao, T. L., Lavril, T., Wang, T., Lacroix, T., and Sayed, W. E. (2023). Mistral 7b.

Jones, K. S. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28:11–21.

Juang, B.-H. and Gray, A. (1982). Multiple stage vector quantization for speech coding. In *ICASSP '82. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 7, pages 597–600.

Jégou, H., Douze, M., and Schmid, C. (2011). Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):117–128.

Kaiser, L., Roy, A., Vaswani, A., Parmar, N., Bengio, S., Uszkoreit, J., and Shazeer, N. (2018). Fast decoding in sequence models using discrete latent variables. *CoRR*, abs/1803.03382.

Kalchbrenner, N. and Blunsom, P. (2013). Recurrent continuous translation models. In Yarowsky, D., Baldwin, T., Korhonen, A., Livescu, K., and Bethard, S., editors, *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1700–1709, Seattle, Washington, USA. Association for Computational Linguistics.

Karpukhin, V., Oguz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D., and Yih, W.-t. (2020). Dense passage retrieval for open-domain question answering. In Webber, B., Cohn, T., He, Y., and Liu, Y., editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Kedzie, C., McKeown, K., and Daumé III, H. (2018). Content selection in deep learning models of summarization. In Riloff, E., Chiang, D., Hockenmaier, J., and Tsujii, J., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1818–1828, Brussels, Belgium. Association for Computational Linguistics.

Kingma, D. P. and Ba, J. (2015). Adam: A method for stochastic optimization. In Bengio, Y. and LeCun, Y., editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Kingma, D. P. and Welling, M. (2014). Auto-encoding variational bayes. In Bengio, Y. and LeCun, Y., editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Kiritchenko, S. and Mohammad, S. (2017). Best-worst scaling more reliable than rating scales: A case study on sentiment intensity annotation. In Barzilay, R. and Kan, M.-Y., editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 465–470, Vancouver, Canada. Association for Computational Linguistics.

Knight, W. (2023). Openai's ceo says the age of giant ai models is already over.

Koehn, P. (2004). Statistical significance tests for machine translation evaluation. In Lin, D. and Wu, D., editors, *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona, Spain. Association for Computational Linguistics.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: Open source toolkit for statistical machine translation. In Ananiadou, S., editor, *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Kumar, A., Ahuja, K., Vadapalli, R., and Talukdar, P. (2020). Syntax-guided controlled generation of paraphrases. *Transactions of the Association for Computational Linguistics*, 8(0):330–345.

Kumar, A., Bhattamishra, S., Bhandari, M., and Talukdar, P. (2019). Submodular optimization-based diverse paraphrasing and its effectiveness in data augmentation. In Burstein, J., Doran, C., and Solorio, T., editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3609–3619, Minneapolis, Minnesota. Association for Computational Linguistics.

Laban, P., Schnabel, T., Bennett, P. N., and Hearst, M. A. (2022). SummaC: Re-visiting NLI-based models for inconsistency detection in summarization. *Transactions of the Association for Computational Linguistics*, 10:163–177.

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., and Soricut, R. (2020). ALBERT: A lite BERT for self-supervised learning of language representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020.* OpenReview.net.

Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Lee, D., Kim, C., Kim, S., Cho, M., and Han, W.-S. (2022). Autoregressive image generation using residual quantization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11523–11532.

Lei, Y., Song, K., Cho, S., Wang, X., Huang, R., and Yu, D. (2024). Polarity calibration for opinion summarization.

Lenc, K. and Vedaldi, A. (2016). Learning covariant feature detectors. In Hua, G. and Jégou, H., editors, *Computer Vision – ECCV 2016 Workshops*, pages 100–117, Cham. Springer International Publishing.

Levelt, W. J. M. (1993). *Speaking: From Intention to Articulation.* The MIT Press.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. (2019). Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., Riedel, S., and Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA. Curran Associates Inc.

Li, H., Ai, Q., Zhan, J., Mao, J., Liu, Y., Liu, Z., and Cao, Z. (2023). Constructing tree-based index for efficient and effective dense retrieval. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '23, page 131–140, New York, NY, USA. Association for Computing Machinery.

Li, H. and Chaturvedi, S. (2024). Rationale-based opinion summarization.

Li, Z., Jiang, X., Shang, L., and Liu, Q. (2019). Decomposable neural paraphrase generation. In Korhonen, A., Traum, D., and Màrquez, L., editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3403–3414, Florence, Italy. Association for Computational Linguistics.

Liévin, V., Dittadi, A., Maaløe, L., and Winther, O. (2019). Towards hierarchical discrete variational autoencoders. In *Second Symposium on Advances in Approximate Bayesian Inference*.

Lin, C.-Y. (2004). ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Lin, T.-Y., Maire, M., Belongie, S. J., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *ECCV*.

Lin, Z. and Wan, X. (2021). Pushing paraphrase away from original sentence: A multi-round paraphrase generation approach. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1548–1557, Online. Association for Computational Linguistics.

Liu, N. F., Lin, K., Hewitt, J., Paranjape, A., Bevilacqua, M., Petroni, F., and Liang, P. (2023). Lost in the middle: How language models use long contexts.

Liu, Y. and Lapata, M. (2019). Hierarchical transformers for multi-document summarization. In Korhonen, A., Traum, D., and Màrquez, L., editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5070–5081, Florence, Italy. Association for Computational Linguistics.

Lloyd, S. (1982). Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–137.

Locatello, F., Bauer, S., Lučić, M., Rätsch, G., Gelly, S., Schölkopf, B., and Bachem, O. F. (2019). Challenging common assumptions in the unsupervised learning of disentangled representations. In *International Conference on Machine Learning*. Best Paper Award.

Louis, A. and Maynez, J. (2023). OpineSum: Entailment-based self-training for abstractive opinion summarization. In Rogers, A., Boyd-Graber, J., and Okazaki, N., editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 10774–10790, Toronto, Canada. Association for Computational Linguistics.

Louis, A. and Nenkova, A. (2011). Automatic identification of general and specific sentences by leveraging discourse annotations. In Wang, H. and Yarowsky, D., editors, *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 605–613, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Louviere, J. J. and Woodworth, G. G. (1990). Best worst scaling: A model for largest difference judgments [working paper]. *Faculty of Business*.

Lui, M. and Baldwin, T. (2012). langid.py: An off-the-shelf language identification tool. In Zhang, M., editor, *Proceedings of the ACL 2012 System Demonstrations*, pages 25–30, Jeju Island, Korea. Association for Computational Linguistics.

Luo, H., Liu, Y., Liu, P., and Liu, X. (2023). Vector-quantized prompt learning for paraphrase generation. In Bouamor, H., Pino, J., and Bali, K., editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 13389–13398, Singapore. Association for Computational Linguistics.

Maddison, C. J., Mnih, A., and Teh, Y. W. (2017). The concrete distribution: A continuous relaxation of discrete random variables. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Madnani, N. and Dorr, B. J. (2010). Generating phrasal and sentential paraphrases: A survey of data-driven methods. *Computational Linguistics*, 36(3):341–387.

Mahon, L. and Lapata, M. (2024). A modular approach for multimodal summarization of tv shows.

Mallinson, J., Sennrich, R., and Lapata, M. (2017). Paraphrasing revisited with neural machine translation. In Lapata, M., Blunsom, P., and Koller, A., editors, *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 881–893, Valencia, Spain. Association for Computational Linguistics.

Malon, C. (2023). Automatically evaluating opinion prevalence in opinion summarization. In *The 6th Workshop on e-Commerce and NLP (KDD 2023)*.

Martin, R. C., Crowther, J. E., Knight, M., Tamborello II, F. P., and Yang, C.-L. (2010). Planning in sentence production: Evidence for the phrase as a default planning scope. *Cognition*, 116(2):177–192.

Mathieu, E., Lan, C. L., Maddison, C. J., Tomioka, R., and Teh, Y. W. (2019). Continuous hierarchical representations with Poincaré variational auto-encoders. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R., editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 12544–12555.

Meng, Y., Ao, X., He, Q., Sun, X., Han, Q., Wu, F., fan, C., and Li, J. (2021). Conrpg: Paraphrase generation using contexts as regularizer.

Mikolov, T., Corrado, G., Chen, K., and Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. *Proceedings of the International Conference on Learning Representations (ICLR 2013)*, pages 1–12.

Min, S., Krishna, K., Lyu, X., Lewis, M., Yih, W.-t., Koh, P., Iyyer, M., Zettlemoyer, L., and Hajishirzi, H. (2023). FActScore: Fine-grained atomic evaluation of factual precision in long form text generation. In Bouamor, H., Pino, J., and Bali, K., editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12076–12100, Singapore. Association for Computational Linguistics.

Mooney, R. (2014). Acl 2014 workshop on semantic parsing. `https://www.cs.utexas.edu/~mooney/cramming.html` [Accessed: (15th May 2024)].

Murphy, K. P. (2023). *Probabilistic Machine Learning: Advanced Topics*. MIT Press.

Narayan, S., Maynez, J., Amplayo, R. K., Ganchev, K., Louis, A., Huot, F., Sandholm, A., Das, D., and Lapata, M. (2023). Conditional generation with a question-answering blueprint. *Transactions of the Association for Computational Linguistics*, 11:974–996.

Ng, H. T., Teo, L. H., and Kwan, J. L. P. (2000). A machine learning approach to answering questions for reading comprehension tests. In *2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 124–132, Hong Kong, China. Association for Computational Linguistics.

Nivre, J., de Marneffe, M.-C., Ginter, F., Hajič, J., Manning, C. D., Pyysalo, S., Schuster, S., Tyers, F., and Zeman, D. (2020). Universal Dependencies v2: An evergrowing multilingual treebank collection. In Calzolari, N., Béchet, F., Blache, P., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4034–4043, Marseille, France. European Language Resources Association.

Novikova, J., Dušek, O., and Rieser, V. (2018). RankME: Reliable human ratings for natural language generation. In Walker, M., Ji, H., and Stent, A., editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 72–78, New Orleans, Louisiana. Association for Computational Linguistics.

Opper, M., Prokhorov, V., and N, S. (2023). StrAE: Autoencoding for pre-trained embeddings using explicit structure. In Bouamor, H., Pino, J., and Bali, K., editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7544–7560, Singapore. Association for Computational Linguistics.

Oren, Y., Meister, N., Chatterji, N. S., Ladhak, F., and Hashimoto, T. (2024). Proving test set contamination in black-box language models. In *The Twelfth International Conference on Learning Representations*.

Ormazabal, A., Artetxe, M., Soroa, A., Labaka, G., and Agirre, E. (2022). Principled paraphrase generation with parallel corpora. In Muresan, S., Nakov, P., and Villavicencio, A., editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1621–1638, Dublin, Ireland. Association for Computational Linguistics.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P. F., Leike, J., and Lowe, R. (2022a). Training language models to follow instructions with human feedback. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A., editors, *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P. F., Leike, J., and Lowe, R. (2022b). Training language models to follow instructions with human feedback. In *NeurIPS*.

Pan, X., Wang, M., Wu, L., and Li, L. (2021). Contrastive learning for many-to-many multilingual neural machine translation. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 244–258, Online. Association for Computational Linguistics.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In Isabelle, P., Charniak, E., and Lin, D., editors, *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Pearson, K. (1901). Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Pei, J., Ananthasubramaniam, A., Wang, X., Zhou, N., Dedeloudis, A., Sargent, J., and Jurgens, D. (2022). Potato: The portable text annotation tool. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. In Walker, M., Ji, H., and Stent, A., editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Post, M. (2018). A call for clarity in reporting BLEU scores. In Bojar, O., Chatterjee, R., Federmann, C., Fishel, M., Graham, Y., Haddow, B., Huck, M., Yepes, A. J., Koehn, P., Monz, C., Negri, M., Névéol, A., Neves, M., Post, M., Specia, L., Turchi, M., and Verspoor, K., editors, *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Puduppully, R., Dong, L., and Lapata, M. (2019). Data-to-text generation with content selection and planning. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6908–6915. AAAI Press.

Radford, A. and Narasimhan, K. (2018). Improving language understanding by generative pre-training.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language models are unsupervised multitask learners.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Rajpurkar, P., Zhang, J., Lopyrev, K., and Liang, P. (2016). SQuAD: 100,000+ questions for machine comprehension of text. In Su, J., Duh, K., and Carreras, X., editors, *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Rashkin, H., Nikolaev, V., Lamm, M., Aroyo, L., Collins, M., Das, D., Petrov, S., Tomar, G. S., Turc, I., and Reitter, D. (2023). Measuring Attribution in Natural Language Generation Models. *Computational Linguistics*, 49(4):777–840.

Razavi, A., van den Oord, A., and Vinyals, O. (2019). Generating diverse high-fidelity images with VQ-VAE-2. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R., editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing*

Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada, pages 14837–14847.

Reimers, N. and Gurevych, I. (2019). Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In Inui, K., Jiang, J., Ng, V., and Wan, X., editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Roberts, M., Thakur, H., Herlihy, C., White, C., and Dooley, S. (2024). To the cutoff... and beyond? a longitudinal perspective on LLM data contamination. In *The Twelfth International Conference on Learning Representations*.

Roy, A. and Grangier, D. (2019). Unsupervised paraphrasing without translation. In Korhonen, A., Traum, D., and Màrquez, L., editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6033–6039, Florence, Italy. Association for Computational Linguistics.

Roy, A., Vaswani, A., Neelakantan, A., and Parmar, N. (2018). Theory and experiments on vector quantized autoencoders. *CoRR*, abs/1805.11063.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323:533–536.

Rush, A. M., Chopra, S., and Weston, J. (2015). A neural attention model for abstractive sentence summarization. In Màrquez, L., Callison-Burch, C., and Su, J., editors, *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389, Lisbon, Portugal. Association for Computational Linguistics.

Russell, S. and Norvig, P. (2010). *Artificial Intelligence: A Modern Approach.* Prentice Hall, 3 edition.

Russin, J., Jo, J., O'Reilly, R., and Bengio, Y. (2020). Compositional generalization by factorizing alignment and translation. In Rijhwani, S., Liu, J., Wang, Y., and Dror, R., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 313–327, Online. Association for Computational Linguistics.

Schroff, F., Kalenichenko, D., and Philbin, J. (2015). Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Schuster, T., Fisch, A., and Barzilay, R. (2021). Get your vitamin C! robust fact verification with contrastive evidence. In Toutanova, K., Rumshisky, A., Zettlemoyer, L., Hakkani-Tur, D., Beltagy, I., Bethard, S., Cotterell, R., Chakraborty, T., and Zhou, Y., editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 624–643, Online. Association for Computational Linguistics.

See, A., Liu, P. J., and Manning, C. D. (2017). Get to the point: Summarization with pointer-generator networks. In Barzilay, R. and Kan, M.-Y., editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Sennrich, R., Haddow, B., and Birch, A. (2016). Neural machine translation of rare words with subword units. In Erk, K. and Smith, N. A., editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Seo, M. J., Kembhavi, A., Farhadi, A., and Hajishirzi, H. (2017). Bidirectional attention flow for machine comprehension. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Shah, D., Lei, T., Moschitti, A., Romeo, S., and Nakov, P. (2018). Adversarial domain adaptation for duplicate question detection. In Riloff, E., Chiang, D., Hockenmaier, J., and Tsujii, J., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1056–1063, Brussels, Belgium. Association for Computational Linguistics.

Sharma, M., Tong, M., Korbak, T., Duvenaud, D., Askell, A., Bowman, S. R., DURMUS, E., Hatfield-Dodds, Z., Johnston, S. R., Kravec, S. M., Maxwell, T., McCandlish, S., Ndousse, K., Rausch, O., Schiefer, N., Yan, D., Zhang, M., and Perez, E. (2024). Towards understanding sycophancy in language models. In *The Twelfth International Conference on Learning Representations*.

Shen, Y. and Wan, X. (2023). Opinsummeval: Revisiting automated evaluation for opinion summarization.

Sherborne, T., Hosking, T., and Lapata, M. (2023). Optimal Transport Posterior Alignment for Cross-lingual Semantic Parsing. *Transactions of the Association for Computational Linguistics*, 11:1432–1450.

Shu, R., Nakayama, H., and Cho, K. (2019). Generating diverse translations with sentence codes. In Korhonen, A., Traum, D., and Màrquez, L., editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1823–1827, Florence, Italy. Association for Computational Linguistics.

Sønderby, C. K., Poole, B., and Mnih, A. (2017). Continuous relaxation training of discrete latent variable image models. In *Beysian DeepLearning workshop, NIPS*, volume 201.

Song, H. O., Xiang, Y., Jegelka, S., and Savarese, S. (2016). Deep metric learning via lifted structured feature embedding. In *Computer Vision and Pattern Recognition (CVPR)*.

Sun, H. and Zhou, M. (2012). Joint learning of a dual SMT system for paraphrase generation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 38–42, Jeju Island, Korea. Association for Computational Linguistics.

Sundermeyer, M., Schl, R., and Ney, H. (2012). LSTM Neural Networks for Language Modeling. *Proc. Interspeech*, pages 194–197.

Surís, D., Liu, R., and Vondrick, C. (2021). Learning the predictability of the future. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12602–12612.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3104–3112.

Sutton, R. (2019). The bitter lesson. `http://www.incompleteideas.net/IncIdeas/BitterLesson.html` [Accessed: (28th May 2024)].

Takida, Y., Shibuya, T., Liao, W., Lai, C., Ohmura, J., Uesaka, T., Murata, N., Takahashi, S., Kumakura, T., and Mitsufuji, Y. (2022). SQ-VAE: variational bayes on discrete representation with self-annealed stochastic quantization. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvári, C., Niu, G., and Sabato, S., editors, *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 20987–21012. PMLR.

Tay, W., Joshi, A., Zhang, X., Karimi, S., and Wan, S. (2019). Red-faced ROUGE: Examining the suitability of ROUGE for opinion summary evaluation. In Mistica, M., Piccardi, M., and MacKinlay, A., editors, *Proceedings of the 17th Annual Workshop of the Australasian Language Technology Association*, pages 52–60, Sydney, Australia. Australasian Language Technology Association.

Thomson, C., Reiter, E., and Belz, A. (2024). Common Flaws in Running Human Evaluation Experiments in NLP. *Computational Linguistics*, pages 1–10.

Thoppilan, R., Freitas, D. D., Hall, J., Shazeer, N., Kulshreshtha, A., Cheng, H.-T., Jin, A., Bos, T., Baker, L., Du, Y., Li, Y., Lee, H., Zheng, H. S., Ghafouri, A., Menegali, M., Huang, Y., Krikun, M., Lepikhin, D., Qin, J., Chen, D., Xu, Y., Chen, Z., Roberts, A., Bosma, M., Zhao, V., Zhou, Y., Chang, C.-C., Krivokon, I., Rusch, W., Pickett, M., Srinivasan, P., Man, L., Meier-Hellstern, K., Morris, M. R., Doshi, T., Santos, R. D., Duke, T., Soraker, J., Zevenbergen, B., Prabhakaran, V., Diaz, M., Hutchinson, B., Olson, K., Molina, A., Hoffman-John, E., Lee, J., Aroyo, L., Rajakumar, R., Butryna, A., Lamm, M., Kuzmina, V., Fenton, J., Cohen, A., Bernstein, R., Kurzweil, R., Aguera-Arcas, B., Cui, C., Croak, M., Chi, E., and Le, Q. (2022). Lamda: Language models for dialog applications.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D., Blecher, L., Ferrer, C. C., Chen, M., Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., Fuller, B., Gao, C., Goswami, V., Goyal, N., Hartshorn, A., Hosseini, S., Hou, R., Inan, H., Kardas, M., Kerkez, V., Khabsa, M., Kloumann, I., Korenev, A., Koura, P. S., Lachaux, M.-A., Lavril, T., Lee, J., Liskovich, D., Lu, Y., Mao, Y., Martinet, X., Mihaylov, T., Mishra, P., Molybog, I., Nie, Y., Poulton, A., Reizenstein, J., Rungta, R., Saladi, K., Schelten, A., Silva, R., Smith, E. M., Subramanian, R., Tan, X. E., Tang, B., Taylor, R., Williams, A., Kuan, J. X., Xu, P., Yan, Z., Zarov, I., Zhang, Y., Fan, A., Kambadur, M., Narang, S., Rodriguez, A., Stojnic, R., Edunov, S., and Scialom, T. (2023). Llama 2: Open foundation and fine-tuned chat models.

Vahdat, A. and Kautz, J. (2020). NVAE: A deep hierarchical variational autoencoder. In *Neural Information Processing Systems (NeurIPS)*.

van den Oord, A., Li, Y., and Vinyals, O. (2018). Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748.

van den Oord, A., Vinyals, O., and Kavukcuoglu, K. (2017). Neural discrete representation learning. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 6306–6315.

van der Maaten, L. and Hinton, G. E. (2008). Visualizing high-dimensional data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605.

Vasuki, A. and Vanathi, P. T. (2006). A review of vector quantization techniques. *IEEE Potentials*, 25:39–47.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Vendrov, I., Kiros, R., Fidler, S., and Urtasun, R. (2016). Order-embeddings of images and language. In Bengio, Y. and LeCun, Y., editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Wang, L. and Ling, W. (2016). Neural network-based abstract generation for opinions and arguments. In Knight, K., Nenkova, A., and Rambow, O., editors, *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 47–57, San Diego, California. Association for Computational Linguistics.

Wang, Z., Araki, J., Jiang, Z., Parvez, M. R., and Neubig, G. (2023). Learning to filter context for retrieval-augmented generation.

Watson, L. N. and Gkatzia, D. (2024). ReproHum #0712-01: Reproducing human evaluation of meaning preservation in paraphrase generation. In Balloccu, S., Belz, A., Huidrom, R., Reiter, E., Sedoc, J., and Thomson, C., editors, *Proceedings of the Fourth Workshop on Human Evaluation of NLP Systems (HumEval) @ LREC-COLING 2024*, pages 221–228, Torino, Italia. ELRA and ICCL.

Weng, L. (2018). From autoencoder to beta-vae. *lilianweng.github.io*.

Wieting, J. and Gimpel, K. (2018). ParaNMT-50M: Pushing the limits of paraphrastic sentence embeddings with millions of machine translations. In Gurevych, I. and Miyao, Y., editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 451–462, Melbourne, Australia. Association for Computational Linguistics.

Wieting, J., Neubig, G., and Berg-Kirkpatrick, T. (2020). A bilingual generative transformer for semantic sentence embedding. In Webber, B., Cohn, T., He, Y., and Liu, Y., editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1581–1594, Online. Association for Computational Linguistics.

Willetts, M., Miscouridou, X., Roberts, S., and Holmes, C. (2021). Relaxed-responsibility hierarchical discrete vaes.

Williams, W., Ringer, S., Ash, T., MacLeod, D., Dougherty, J., and Hughes, J. (2020). Hierarchical quantized autoencoders. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 4524–4535. Curran Associates, Inc.

Wong, D., Juang, B., and Gray, A. (1981). Recent developments in vector quantization for speech processing. In *ICASSP '81. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 6, pages 1–4.

Wu, Y., Gardner, M., Stenetorp, P., and Dasigi, P. (2022). Generating data to mitigate spurious correlations in natural language inference datasets. In Muresan, S., Nakov, P., and Villavicencio, A., editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2660–2676, Dublin, Ireland. Association for Computational Linguistics.

Xie, Z., Wang, S. I., Li, J., Lévy, D., Nie, A., Jurafsky, D., and Ng, A. Y. (2017). Data noising as smoothing in neural network language models. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Xu, F., Shi, W., and Choi, E. (2024). RECOMP: Improving retrieval-augmented LMs with context compression and selective augmentation. In *The Twelfth International Conference on Learning Representations*.

Xu, P., Ping, W., Wu, X., McAfee, L., Zhu, C., Liu, Z., Subramanian, S., Bakhturina, E., Shoeybi, M., and Catanzaro, B. (2023). Retrieval meets long context large language models.

Xue, M., Liu, D., Lei, W., Fu, J., Lan, J., Li, M., Yang, B., Xie, J., Zhang, Y., Peng, D., and Lv, J. (2023). Unifying discrete and continuous representations for unsupervised paraphrase generation. In Bouamor, H., Pino, J., and Bali, K., editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13805–13822, Singapore. Association for Computational Linguistics.

Yang, H., Lam, W., and Li, P. (2021). Contrastive representation learning for exemplar-guided paraphrase generation.

Zeghidour, N., Luebs, A., Omran, A., Skoglund, J., and Tagliasacchi, M. (2022). Soundstream: An end-to-end neural audio codec. *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, 30:495–507.

Zhang, H., Xu, Y., and Perez-Beltrachini, L. (2024a). Fine-grained natural language inference based faithfulness evaluation for diverse summarisation tasks. In Graham, Y. and Purver, M., editors, *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1701–1722, St. Julian's, Malta. Association for Computational Linguistics.

Zhang, Y., Li, P., Lai, Y., Zhou, D., and He, Y. (2024b). Large, small or both: A novel data augmentation framework based on language models for debiasing opinion summarization.

Ziegler, D. M., Stiennon, N., Wu, J., Brown, T. B., Radford, A., Amodei, D., Christiano, P., and Irving, G. (2020). Fine-tuning language models from human preferences.

Łańcucki, A., Chorowski, J., Sanchez, G., Marxer, R., Chen, N., Dolfing, H. J., Khurana, S., Alumäe, T., and Laurent, A. (2020). Robust training of vector quantized bottleneck models. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7.