Combining Variational Sampling and Metropolis–Hastings Sampling for Paraphrase Generation

by

Ali Hejazizo

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computing Science

University of Alberta

Abstract

Paraphrasing involves changing the expression of a sentence and rewording it to inform the same information as the original sentence and can occur at word-level, phrase-level, or sentence-level. Paraphrasing task has been attracting attention in recent years as several natural language processing (NLP) applications such as question answering, information extraction, information retrieval, and summarization benefit from the success of automatic paraphrase generation. Researchers have developed various paraphrase generation techniques, including knowledge-based approaches, supervised data-driven approaches, and unsupervised data-driven approaches. Knowledge-based approaches are labor intensive and do not generalize well and supervised approaches require massive parallel corpora of pairs of sentences and paraphrases.

In this work, we propose an unsupervised paraphrasing technique that works in word-level as well as phrase-level for sentence-level paraphrase generation. Existing work either samples directly from sentence space or from a variational latent space while our work combines them both. We show the drawbacks and difficulties of techniques that work at word-level only and propose a technique consisting of three word-level operations (word replacement, word deletion, and word insertion) and a novel phrase-level paraphrasing operation (phrase replacement). The three word-level operations sample directly from the sentence space while our phrase-level operation samples from the latent space of a variational autoencoder (VAE) trained on phrases.

We perform paraphrase generation iteratively with the objective of generating paraphrases that are 1) fluent, and 2) close in semantic information to the input sentence. We use Metropolis–Hastings (MH) algorithm, a Markov Chain Monte Carlo (MCMC) algorithm, to sample from sentence space and latent space. In each iteration, we randomly select a word/phrase and an operation to form a proposal and use MH to accept or reject the proposal and generate a paraphrase.

We show the effectiveness of our approach with a series of experiments. First, we train a VAE using Stanford Natural Language Inference (SNLI) dataset [8] and Quora dataset[1] for phrase replacement operation. Second, we evaluate our approach on the Quora dataset including 139k pairs of questions and paraphrases using iBLEU score as our main evaluation metric. The results show that our novel phrase replacement operation improves the quality of paraphrases when compared with techniques paraphrasing by direct word-level sampling only. We show our phrase-level operation can effectively edit multiple words at a time and generate high quality paraphrases. We also discuss the difficulties of evaluation with iBLEU score and VAE training.

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

To my parents,

For their endless love, support, and encouragement.

Acknowledgements

I would like to thank my supervisors, Dr. Lili Mou and Dr. Denilson Barbosa, for the invaluable insights they have provided throughout my M.Sc. as their student. I have been extremely lucky to have the chance of working with them who encourage me and care about my work.

I am further thankful to my parents and my sisters providing me with constant love and support throughout my years of study.

Last but not least, some special words of gratitude go to my friends, Zahra, Sina, Parnian, Nikoo, Nima, Shaghayegh, Shiva, Reyhaneh, Yousef, Yashar, Mahyar, and Masoud who have been my second family here and a major source of love, hope, encouragement, and happiness. Thanks for always being there for me.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

Paraphrasing is the act of changing the expression of a sentence to inform the same meaning in a different way. Automatic generation of paraphrases is an important task in Natural Language Processing (NLP) as it can benefit many applications such as question answering, information extraction, information retrieval, summarization, and data augmentation [33]. Paraphrase generation can occur at different levels [33]:

- Word-level: Lexical paraphrases occur in the word-level. This is also commonly known as synonyms. For example <cold, cool> and <happy, cheerful> are lexical paraphrases.

- Phrase-level: Phrases that share the same meaning are phrasal paraphrases such as <development area, area of growth>.

- Sentence-level: Sentential paraphrases refer to two sentences with the same semantic information such as <I completed the assignment, I finished the task>. In this example, the paraphrase is simply achieved by applying multiple lexical paraphrases. More complicated paraphrases can be constructed by changing the sentence syntax. For example <Can we ever store energy produced in lightning?, Is it possible to store the energy of lightning?> are two paraphrases that are made by more than simple word replacement.

In this work, we address the sentence-level paraphrase generation tasks where we use word-level and phrase-level techniques in our approach.

Paraphrasing can improve the performance of Question answering (QA) systems by providing different expressions of the same input information to increase the chance of finding the correct answer [15], [44]. This ensures that the system has correctly understood the user question. For example, all questions of <Who is the richest person in the world?, Who is the number 1 richest person?, Who has the highest net worth?, Who has the most money > are looking for the same answer while being very different syntactically. Dong et al., [15] shows the effectiveness of paraphrasing in QA task by proposing a neural scoring model to find the most likely text span that includes the answer to user question. The model input is augmented with paraphrases achieved from a combination of paraphrase generation techniques and evaluation results demonstrate the effectiveness of paraphrase generation augmentation. Soni et al., [51] address QA in electrical health records (EHRs) with deep variational autoencoders (VAEs) and an LSTM encoder/decoder architecture. The model improves the performance of QA systems and can be used to generate clinical paraphrases with high quality.

Paraphrasing can also help the evaluation process of several NLP applications such as document summarization and machine translation (MT) where the evaluation metric is the BLEU score. The BLEU score is an automatic, language independent, and inexpensive evaluation metric which measures how many of the words (and/or n-grams) in machine generated summary/translation appear in the human reference summary/translation [42]. However, it is impossible for a single reference sentence to capture all the variations which may result in a poor score for a high quality translation or summary. Multiple reference sentences is a solution but is expensive. An alternative solution is to use paraphrase generation to automatically generate multiple reference sentences from a single one [24]. Owczarzak et al., [41] present a method that generates domain-related paraphrases in word-level and phrase-level without requiring external knowledge resources using only the source and reference sentences and show that their paraphrasing method

Original	new harry potter book release date
Variant 1	the most recent harry potter book data of release
Variant 2	publish date of new harry potter book

Table 1.1: Query expansion with paraphrase generation

improves BLEU evaluation scores.

The final example is on how paraphrasing can address the term mismatch problem in information retrieval and information extraction systems. Paraphrasing can automatically generate different variants of a query as shown in table 1.1. A term that is missed in one variant may exist in the other which increases the chance of extracting the correct information. Culicover et al., [13] and Jones et al., [23] are among the earliest works that deal with paraphrase generation for keyword expansion in document text searching. Wasim et al., [57] use lexical paraphrases to generate related queries by ranking paraphrases with statistical information and selecting top n paraphrases. The results on a custom defined benchmark shows that paraphrasing performance is 20% better in mean average precision compared to a state-of-the-art retrieval technique which is based on pseudo relevance feedback. Beeferman et al., [6] propose a content-ignorant algorithm that clusters related queries and URLs for search engines.

In short, paraphrasing task is an important task by itself, and it also improves the performance of other areas of NLP.

1.2 Existing Techniques and Drawbacks

Researchers have developed various paraphrase generation techniques, including knowledge-based approaches, supervised data-driven approaches, and unsupervised data-driven approaches.

Knowledge-based paraphrasing systems can be categorized into approaches that uses dictionaries [17], [55], rules [18], or formal grammars [16], [19], [20], [36]. Knowledge-based paraphrasing is language dependent and labor intensive; it requires much time and effort for humans to design rules. Moreover, these methods may generate disfluent sentences, as manually specified rules may not be suitable for all sentences.

Supervised machine learning techniques require large human-written paraphrases that are labor intensive and expensive to obtain. Moreover, these annotated datasets are domain specific. For example, the Quora dataset [1] includes only questions. A supervised paraphrase generation system trained on the Quora dataset does not generalize well to a new domain.

Unsupervised paraphrase generation techniques do not require parallel corpora. There have been several unsupervised paraphrase generation studies. Techniques based on recurrent neural networks (RNNs) [53] generate text sequentially from left to right, but it is less controllable and makes it harder to apply constraints during generation. Natural language generation usually involves applying some constraints to control the output [39]. Miao et al., [39] paraphrases directly by sampling in the word space that is able to consider constraints such as keywords inclusion. However, it considers only one word at a time and cannot paraphrase in phrase or sentence level.

1.3 Contributions of the Thesis

In this work, we propose a technique for unsupervised paraphrase generation that combines word-level and phrase-level editing for sentence-level paraphrase generation.

Unlike previous paraphrasing techniques that either sample directly from the sentence space [39] or from the variational latent space [9], our technique combines both. We adopt the word-space sampler from [39] and train a phrase-level VAE for our latent space sampler. In total, we define four local edits for paraphrase generation:

- word replacement
- word deletion
- word insertion

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

- phrase replacement

We use the Metropolis-Hastings (MH) algorithm, a Markov chain Monte Carlo (MCMC) method, for sampling from sentence space and latent space. During sampling, we randomly select a local edit and the candidate word or phrase. We call the combination of the local edit and the candidate word/phrase a *proposal* which is in fact a transition from the original sentence to a candidate paraphrase. The MH algorithm either accepts or rejects the transition based on an *acceptance rate*. The acceptance rate is calculated by a heuristically defined objective, involving several constraints such as the fluency and word embedding similarity of the candidate paraphrase.

The advantage of our approach over VAE is that 1) VAE applies editing on the entire sentence while our approach can apply editing on a smaller chunk of a sentence with word-level and phrase-level operations, and 2) it allows controllability on the generated paraphrase. Compared with the word-level CGMH provided in [39] that only work in word-level, our approach allows larger edits as well through the phrase-level VAE.

To demonstrate the effectiveness of our approach, we evaluate paraphrasing performance on Quora dataset[2] which includes around 139k pairs of paraphrases. We evaluate the quality of the generated paraphrases with the BLEU score [42] which is an automatic evaluation metric and basically calculates the overlap between A generated paraphrase and the reference paraphrase in Quora dataset noted as BLEU_{ref}. It has been discussed that BLEU_{ref} is not an effective evaluation metric as simply copying the original sentence as a paraphrase results in a high BLEU_{ref} score. To address this issue, we also calculate BLEU score between the original sentence and the generated paraphrase $\text{BLEU}_{\text{orig}}$ and choose iBLEU score $(0.9 \times \text{BLEU}_{\text{ref}} - 0.1 \times \text{BLEU}_{\text{orig}})$ [52] which penalizes by similarity to the original sentence as our main evaluation metric.

The results, provided in Section 4.3, show that our approach improves the quality of the generated paraphrases by more than 2 points in iBLEU score.

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

In short, the main contributions of this thesis are:

- We propose a novel phrase-level VAE for paraphrase generation.
- We propose to combine word-level paraphrasing with phrase-level for a sentence-level paraphrasing.
- We empirically show the effectiveness of the hybrid model through a variety of experiments.

1.4 Thesis Structure

In this chapter, we have discussed the motivation behind our research study and proposed our combined system of discrete word-level and continuous phrase-level sentence editing as a possible solution to the challenging task of paraphrase generation. The remainder of this thesis is organized as follows: Chapter 2 provides background material to understand the process of training a VAE as well as an experiment to visualize the effectiveness of a variational latent space. We also provide a general overview of conventional and recent data driven approaches of paraphrasing. In chapter 3, we discuss our novel approach to paraphrasing task. We discuss the two components of word-level discrete sampling and phrase-level continuous sampling. We demonstrate how a phrase-level editing can address the drawbacks of word-level editing and provide our algorithm pseudo-code in details. Chapter 4 presents the datasets, evaluation metrics and the results of paraphrasing and autoencoding. We compare our technique with a word-level only technique and demonstrate how our novel phrase-level technique can improve the quality of paraphrases. We also provide an experiment on KL divergence loss in our VAE that shows its latent space is in fact variational. Finally, Chapter 5 provides a summary of our key contributions and findings and describes future directions.

Chapter 2

Background and Related Work

2.1 Autoencoders

Autoencoders are a type of neural networks that are used to achieve a compact representation of data in an efficient self-supervised manner. To this end, an autoencoder tries to learn a compressed knowledge representation from which the original input can be reconstructed. There are three main components in a traditional autoencoder: encoder, code, decoder which is shown in Figure 2.1. Internally code is a hidden layer $\boldsymbol{h}$ modeling the compact representation, encoder is a function $\boldsymbol{h} = f(\boldsymbol{x})$ that does the compression and decoder $\boldsymbol{r} = g(\boldsymbol{h})$ reconstructs the input.

During the learning process, the model tries to minimize a loss function that basically penalizes the model if output $g(f(x))$ is dissimilar from input x. To be specific, let $L(x^{(m)}, g(f(x^{(m)})))$ be the reconstruction loss for the m^{th} sample, then the total loss for training the autoencoder is:

$$J = \sum_{m=1}^{M} L(\boldsymbol{x}^{(\boldsymbol{m})}, g(f(\boldsymbol{x}^{(\boldsymbol{m})})) \tag{2.1}$$

2.1.1 Variational Autoencoders

Autoencoders ability to generate compact representation of the input is useful in many applications such as dimensionality reduction, data denoising, feature variation, etc. but not for generation. To have a model capable of generating new samples, we do not want to replicate the input but to sample and generate variations of the input.

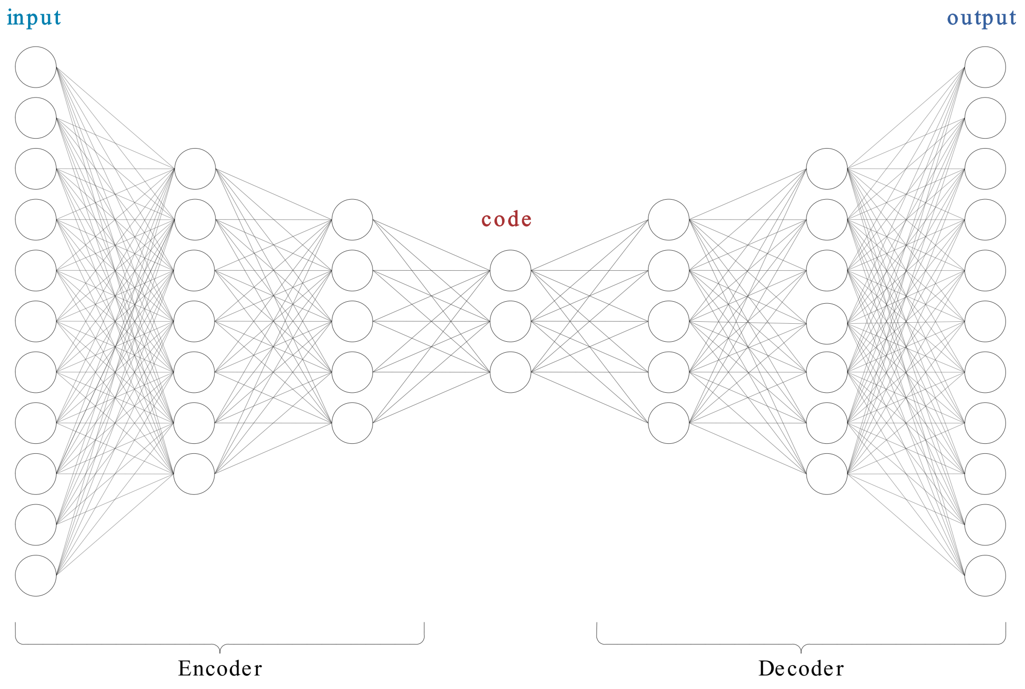

Figure 2.1: Autoencoder main components

To demonstrate the differences between traditional autoencoders and VAEs, we conducted an experiment on MNIST [25] dataset which includes around 70k images of digits.

First, a traditional autoencoder with a 2D latent space is trained and optimized on MNIST data. Considering only reconstruction loss, the encoded features in a traditional autoencoder may locate on a strange manifold as shown in Figure 2.2. The square images show the visualization of image reconstruction of a point in the specified region. For example, two points have been selected on Figure 2.2 where the reconstruction visualization shows numbers 6 and 1. The encoding visualization demonstrates separate clusters of images which makes the life of the decoder easier to decode them. However, sampling from the gaps (the square image with a question mark) results in an unrealistic image as no encoded vectors were coming from that region during the training process.

On the other hand, A VAE imposes probabilistic property to the encoding

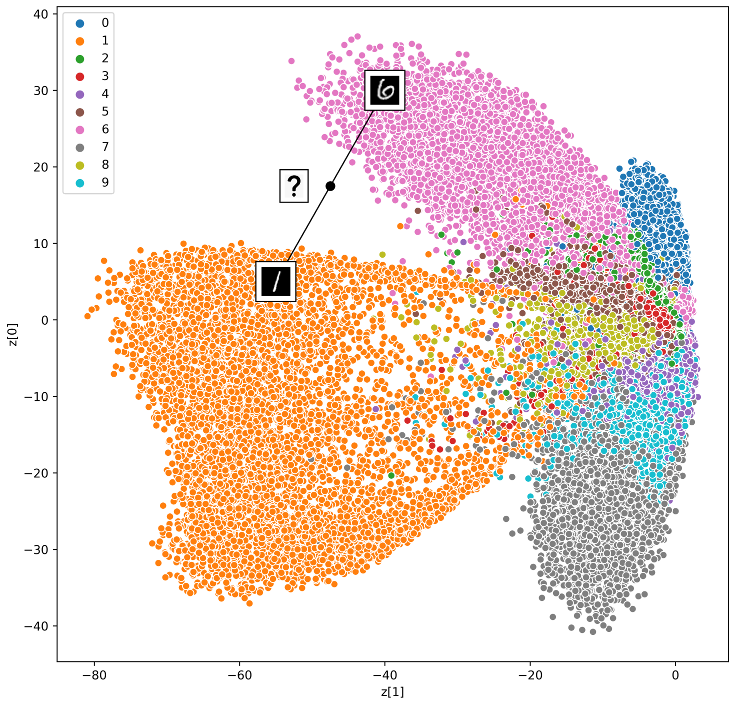

Figure 2.2: Latent space of the traditional autoencoder trained on MNIST optimized purely for reconstruction loss

network. The probabilistic property forces the encoder to generate a latent space that follows a unit Gaussian distribution. Therefore, their latent space is continuous which allows easy interpolation to generate new outputs.

VAEs achieve this property by computing two encoding vectors of size n in the latent space: 1) Mean vector (μ), and 2) Standard deviations vector (σ).

The two vectors of mean and standard deviation for the encoding vector consist of random variables where the i^{th} element is a random variable X_i that is sampled from a Gaussian distribution with σ_i and μ_i. A VAE model structure is shown in Figure 2.3.

Note that due to sampling, VAE results in a stochastic generation rather than deterministic. The mean vector shows the center of encoding and standard deviation indicates the area around it. Due to stochastic generation, on every single pass even with the same input, a different output is likely. Compared to a standard autoencoder, a decoder in a VAE is exposed to different variation of the same input. Therefore, VAE learns to decode not only a specific encoding in the latent space, but also ones with slight variations.

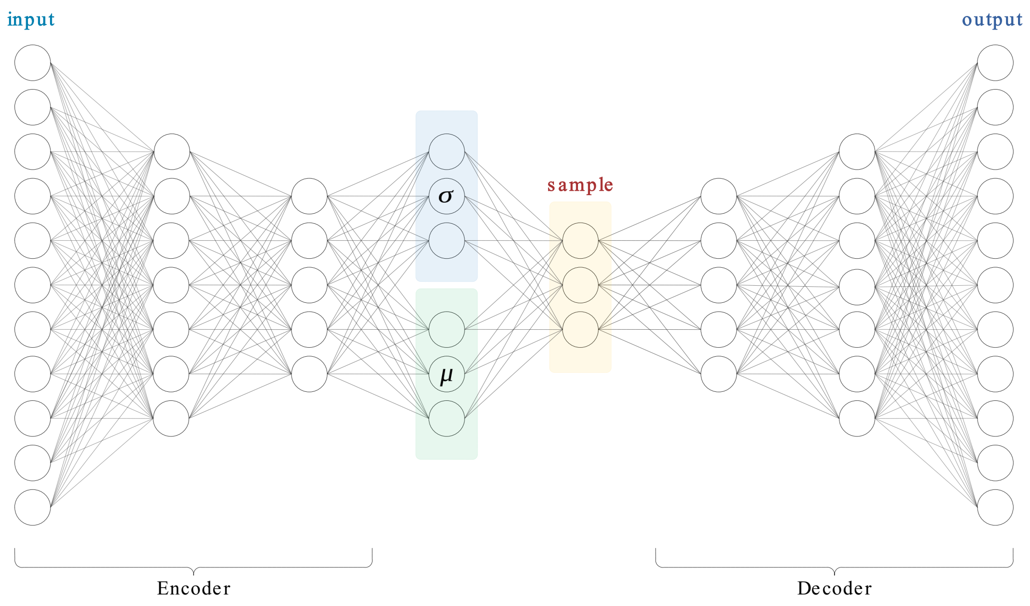

Figure 2.3: VAE model structure

Introducing σ and μ allows interpolation between local similar samples in the latent space. However, interpolating globally between dissimilar samples which are further from one another is also necessary. Otherwise, the encoder takes very different values for μ and σ to generate different classes as far as possible in separate clusters as shown in Figure 2.4. This helps the decoder to efficiently reconstruct the training data but is not plausible for interpolation. Ideally, classes should form a continuous distribution on a global scale while still being distinct as shown in Figure 2.5. This allows to smoothly interpolate between classes and generate new samples.

To force this, VAE adds an extra regularization term to the cost function that is Kullback-Leibler (KL) divergence between the true and approximate posterior distributions. KL divergence for two probability distribution p and q is defined as:

$$D_{KL}(p||q) = \sum_{i=1}^{N} p(x_i).(\log p(x_i)) - \log q(x_i)) \tag{2.2}$$

which is basically the expectation of the log difference between the two probabilities of p and q and measures how much two probability distributions diverge

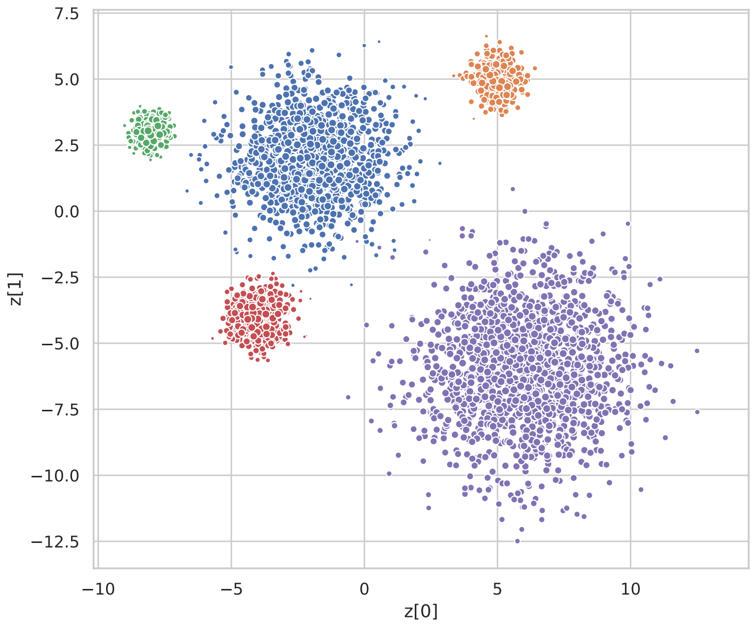

Figure 2.4: Locally optimized VAE latent space

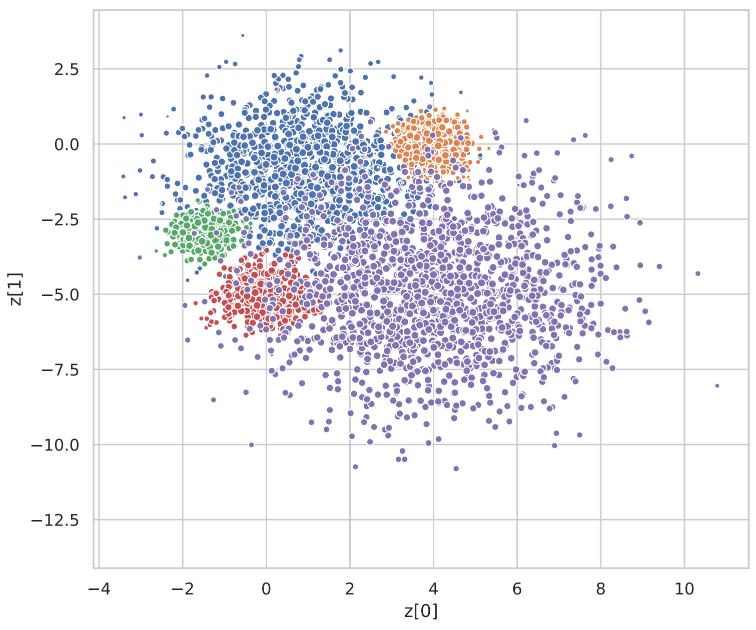

Figure 2.5: Globally optimized VAE latent space

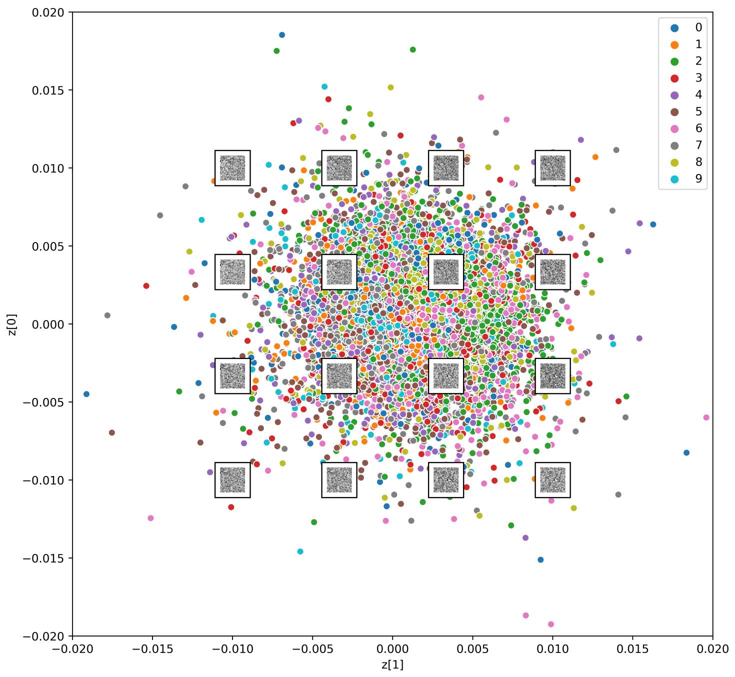

Figure 2.6: VAE latent space trained on MNIST that is optimized purely for KL loss

from each other.

Minimizing KL divergence in VAEs encourages the model to prefer μ and σ parameters that generate an encoding distribution close to the target distribution which is a standard normal Gaussian distribution. The model becomes penalized if it tries to cluster classes apart into specific regions.

Training a VAE on MNIST purely with KL loss results in an encoding latent space that is randomly placed at the origin. It is impossible for the decoder to decode anything meaningful as there really is not anything meaningful encoded. A visualization on VAE latent space trained purely with KL loss on MNIST dataset is shown in Figure 2.6. The square images are again reconstruction visualization of a point in the specified region which demonstrates that decoded images are fully distorted and noisy.

Optimizing VAE with both reconstruction and KL loss results in a latent space that maintains the similarity of nearby encodings by cluster-forming nature of reconstruction loss and packs different around the latent space origin by dense packing nature of the KL divergence loss. A VAE latent space

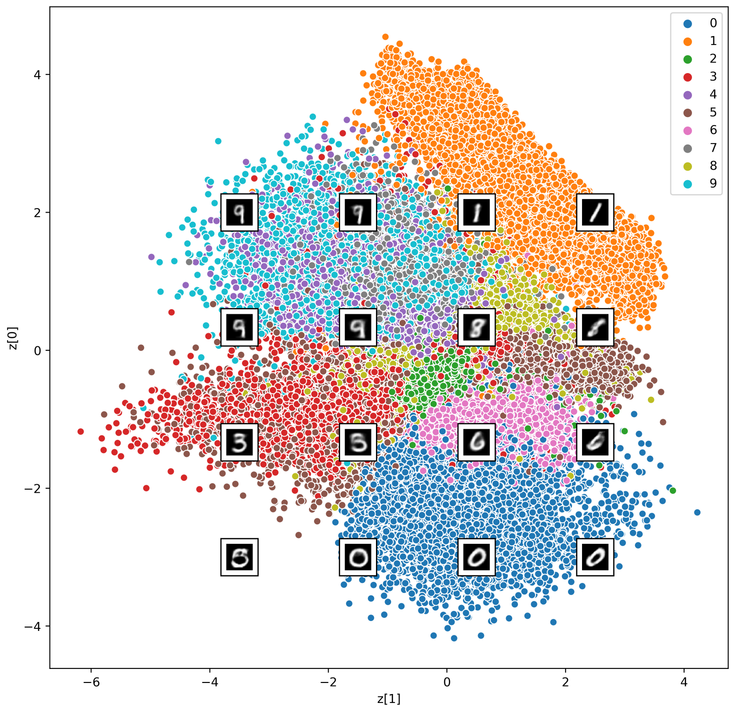

Figure 2.7: VAE latent space trained on MNIST that is optimized with both reconstruction and KL loss

trained on MNIST with both reconstruction and KL loss is demonstrated in Figure 2.7. Reconstruction visualization on Figure 2.7 shows successful image reconstruction with clear images of numbers. Comparing the axes with Figure 2.2 shows how KL loss successfully forces a dense packed latent space at the origin. Interpolation from this latent space results in meaningful outputs while there are no sudden gaps between distinct clusters.

Therefore, in this thesis, we use VAE to perform experiments on paraphrase generation and show its effectiveness in generating diverse while fluent paraphrases. There is relatively little prior work in the literature using VAE for sentence or paraphrase generation. In contrast to the unconditional sentence generation model proposed by [9], our VAE is conditioned on the original input sentence through fluency and word embedding similarity to generate a paraphrase. Gupta et al., [21] proposes a conditioned VAE on the intermediate representation of the input sentence to generate paraphrases. But their work is supervised and not directly comparable to our approach.

2.2 Paraphrase Generation

Paraphrase generation can be addressed with different approaches: knowledge-based techniques such as rule-based methods, supervised learning, and unsupervised learning. In the following sections, we describe these techniques with more emphasis on unsupervised data driven approaches.

2.2.1 Knowledge-based Paraphrase Generation Techniques

Knowledge-based methods rely on different types of resources such as rules, dictionaries, and formal grammars to propose a modification in the original sentence.

Predefined rules are either hand-written [37], [61] or automatically collected [5], [30], [60]. However, the coverage of these crafted rules and the complexity of the generated paraphrase has been shown not to be high enough, specifically in long complicated sentences [45]. Applying modifications on sentences with a thesaurus is simple but limited to synonyms substitution to generate paraphrases by word replacement in the original sentence. Bolshakov et al., [7] uses WordNet synonym data and performs paraphrase generation in two phases: 1) synonym candidates extraction, and 2) paraphrase validation. Given a sentence, it first searches for relevant synonyms in WordNet synonymy data to generate substitution candidates. In the second phase, the optimal synonym for each word is selected based upon the word context.

An interesting approach to identify similar words for synonym replacement is to use a dictionary such as Longman Dictionary of Contemporary English (LDOCE). Wallis et al., [55] uses the bag-of-words of word definitions as a measure of synonymy. A cosine similarity is then used to identify synonyms for word replacement in paraphrase generation.

However, these methods are usually labour intensive, expensive, and do not generalize well on unseen data. To address the problems of conventional paraphrasing techniques, many data-driven approaches have been proposed. These approaches can be categorized into two categories of supervised and unsupervised methods which are discussed in sections 2.2.2 and 2.2.3, respectively.

2.2.2 Supervised Paraphrase Generation

In recent years, data-driven machine learning techniques in particular deep learning with Seq2Seq encoding decoding architecture have become the prevailing adopted approach for paraphrase generation. Prakash et al., [43] uses a stacked RNN model based on residual learning. Wang et al., [56] introduces a simple semantic augmentation strategy that is then added to either an LSTM or a transformer and improves their capability in paraphrase generation. There is always the issue of out of vocabulary words. Cao et al., [11] proposes a Seq2Seq model allowing direct word copy from the source text that improves the quality and diversity of the generated paraphrases.

Recently, generative deep neural network models that are widely popular in computer vision have also been used for paraphrase generation. Gupta et al., [21] was one of the first attempts that proposes a supervised variant of VAE-LSTM where the encoder is fed with both input sentence and the reference paraphrase to generate a high quality latent space. Multiple paraphrases are then generated by sampling from the continuous latent space. Attention mechanism has recently become widely popular in NLP. Ma et al., [32] suggests that to generate paraphrases, existing Seq2Seq models memorize the patterns in sentences instead of capturing the words meaning. To tackle this problem, it proposes an encoder decoder model that uses attention mechanism instead of the simple linear soft-max operation. A paraphrase is then generated by querying distributed word representations.

Over recent years, Deep Reinforcement Learning (DRL) has gained tremendous popularity and success with a wide series of applications such as Atari games [40] and alphaZero [50]. Reinforcement learning turned to be extremely beneficial in the field of NLP. This is due to the fact that the loss function optimization in NLP is different from output accuracy measurement. In NLP, we tend to optimize the loss function by maximizing the likelihood estimation metric while other metrics such as BLEU and ROUGE [29] are used for output evaluation. We are not able to optimize the BLEU and ROUGE metrics directly as they are non-differentiable metrics. Li et al., [28] trains a DRL model consisting of a generator and an evaluator. The generator is a Seq2Seq model that generates a paraphrase given an input sentence and is trained with deep learning and fine-tuned by reinforcement learning. The evaluator, which can be trained with either supervised learning or inverse reinforcement learning, can determine if a sentence is a paraphrase for another one and calculates the reward for the generator.

Controllable paraphrase generation has also been addressed. Iyyer et al., [22] proposes a model to generate paraphrases with a desired syntactic form. The automated and human evaluations demonstrate that the model can generate high quality paraphrases that are comparable with the paraphrases generated by uncontrolled systems.

Given a massive dataset of pairs of paraphrases, supervised networks can generate high quality paraphrases and achieve state of the art results. However, it is extremely labour intensive and hard to collect such a corpus and in the end, these models do not generalize well to a new domain Therefore, unsupervised paraphrase generation has been investigated as a solution to the scarcity of labeled data.

2.2.3 Unsupervised Paraphrase Generation

Our focus in this thesis is on unsupervised paraphrase generation techniques that do not require a massive parallel dataset and still can generate high quality paraphrases.

Machine translation (MT) for paraphrase generation as an unsupervised method emerged to use non-parallel corpora in two languages [10], [34]. The idea is that feeding the same input sentence to multiple machine translator generates sentences that are syntactically different but semantically similar. Neural machine translation (NMT) later emerged as a dominant approach [1]. Back-translation has also been used where two machine translation models are trained. A single sentence is first translated to a different language and the result is translated back again [34], [59]. However, this is in contrast with humans. Monolingual people can still paraphrase sentences. Roy et al.,

[48] investigates the necessity of data in two languages for paraphrasing and proposes a residual variant of vector-quantized VAE for paraphrase generation using only a monolingual corpus. The study shows that monolingual models can achieve better results compared to bilingual ones. Nevertheless, their work is mainly about paraphrase identification and data augmentation rather than paraphrasing itself and is not directly comparable to our technique.

Siddique et al., [49] is the first work that proposes an unsupervised paraphrasing technique with DRL and achieves state-of-the-art results. Their work formulates the problem as training a DRL policy to maximize the reward which is the quality of the generated paraphrases. To avoid starting from a random policy, their work uses a pre-trained VAE as a warm-start and uses VAE's output to transition according to its policy.

Miao et al., [39] proposes a constrained sentence generation technique using Metropolis-Hastings sampling (CGMH) [38] that has applications in sentence generation with keywords, unsupervised paraphrase generation, and unsupervised sentence error correction. The proposed work defines local operations namely word replacement, deletion, and insertion and directly samples from the sentence space. Liu et al., [31] proposes unsupervised paraphrasing with simulated annealing (UPSA) which follows CGMH operations, but addresses the paraphrasing problem as a stochastic searching problem and introduces some modifications such as: 1) it modifies the searching objective by adding expression diversity 2) introduces a copy mechanism to recover the deleted words that are rare and have low language-model probability.

In this thesis, we adopted the word level editions in [39]. The main differences between our work, CGMH, and UPSA is that we introduce a phrase level operation that adds sampling from a continuous latent space while benefiting from direct sampling from the sentence space. We also have a different strategy in searching for the best paraphrase among the generated paraphrases which is discussed in details in Section 4.4.1.

Chapter 3

Our Approach

Unsupervised paraphrase generation techniques work either in variational latent [9] or sentence space [39]. In this work, we propose a novel paraphrase generation technique that employs the benefits of both.

3.1 Overview

In our approach we have two components: 1) Sentence space sampler, and 2) VAE latent space sampler.

We perform paraphrase generation in an iterative manner where in each iteration, we apply a local operation on a randomly selected word or phrase with either the sentence space or VAE latent space sampler. The local operations are deletion, insertion, word replacement as in [39], as well as our novel phrase replacement with variational latent-space sampling. The selected local operation and random word or phrase form a proposal for transition from the input sentence to several candidate paraphrases. We define a heuristic scoring function that evaluates the quality of a candidate paraphrase. This involves a few key aspects of paraphrasing such as fluency and semantic preservation. Finally, a proposal may be rejected according to an acceptance ratio that is calculated from a pre-specified stationary distribution that is discussed in more detail in Section 3.2.

For the proposed phrase replacement with latent-space sampling, we first perform phrase detection which is discussed in subsection 3.3.2 and select a phrase randomly. To generate candidates for phrase replacement proposal, we train a VAE that allows to sample from the latent-code and generate new sentences that are semantically close to the input sentence. Sampling from the latent space results in several candidates that, similar to sampling from word space, will be accepted or rejected based on the quality evaluation.

3.2 Sampling with Metropolis-Hastings

The Metropolis-Hastings (MH) algorithm [38] is a Markov-Chain Monte Carlo (MCMC) [12] stochastic sampler from a user-defined distribution $p(\cdot)$, which is also known as the *stationary distribution*. In our paraphrase generation task, the distribution is $p(\mathrm{y}|\mathrm{x})$ modeling probability of a paraphrase y given x.

The MH algorithm iteratively performs local edits modifying a current candidate paraphrase y to a new one y′ based on a *proposal distribution*. Then, an *acceptance rate* is computed to either accept or reject the proposal. Theoretical results show that MH yields an unbiased sample when the number of edits is large enough.

3.2.1 Proposal Distribution

We design a set of word-level and phrase-level proposals that are randomly performed in each step with probabilities 0.25 for each word-level generator and 0.25 for phrase level modification. We set these probabilities empirically, and they do not matter much because the proposal distribution will be corrected in an *ad hoc* way to achieve the stationary distribution.

Word-level proposals include word-level deletion, insertion, replacement, which we also follow from [39]. In each step a word is selected randomly and an operation is performed. Operations have equal probability which is [0.25, 0.25, 0.25] for $[p_{\text{insert}}, p_{\text{delete}}, p_{\text{replace}}]$.

With word replacement, a new paraphrase is achieved by replacing a selected word with another. Assume the current sentence (x) has n words $x = [w_1, ..., w_{m-1}, w_m, w_{m+1}, ..., w_n]$. The condition probability to replace the m_{th} word with a candidate word (w^c) is as follows:

$$g_{\text{replace}}(y'|x) = \pi(w_m^* = w^c|x_{-m}) = \frac{\pi(w_1, ..., w_{m-1}, w^c, w_{m+1}, ..., w_n)}{\sum_{w \in \nu} \pi(w_1, ..., w_{m-1}, w, w_{m+1}, ..., w_n)} \tag{3.1}$$

where ν is the set of all vocabulary words and $g_{replace}(y'|x)$ is the probability of transition to y' as the target sentence with word replacement action. However, computation of $\pi(w_1, ..., w_{m-1}, w, w_{m+1}, ..., w_n)$ for all $w^c \in \nu$ is expensive as we have to compute $\pi(w_1, ..., w_{m-1}, w^c, w_{m+1}, ..., w_n)$ for each sentence candidate separately. Therefore, we pre-select the words with a backward and forward language model and only consider words that have a high fluency score:

$$Q(w^c) = \min(\pi(w_1, ..., w_{m-1}, w_m^* = w^c), \pi(w_m^* = w^c, w_{m+1}, ..., w_n)) \tag{3.2}$$

We then sample a word for replacement following the the conditional probability of filtered words by Equation 3.1.

Insertion is similar to replacement. We first insert a special token (<PHD>) in the selected position and then perform a replacement action by selecting a word to replace with <PHD> token. Therefore, g_{insert} is similar to Equation 3.1. Similar to previous proposals, word deletion is where we randomly select a word and simply delete it. Assuming $x = [w_1, ..., w_{m-1}, w_m, w_{m+1}, ..., w_n]$ for deleting word w_m, g_{delete} is equal to 1 if $y' = [w_1, ..., w_{m-1}, w_{m+1}, ..., w_n]$. Similarly it is 0 for any other sentence.

By introducing deletion and insertion operations, the ergodicity of Markov Chain is guaranteed. This is true because it is possible to reach from a sentence to any other sentence by first deleting all words in the source sentence and then inserting words of the target sentence one by one.

However, word-level operations are limited to the modification of one word a time. There are cases where one has to apply a series of editing operations to generate a coherent and meaningful paraphrase. While the intermediate sentences may become worse in terms of quality, the final sentence has a high chance to become accepted. For example, a transition from a long sentence such as "*how come people on quora ask questions here, when they can get them on google and why is quora just a question/answer site, will it expand?*" to "*why do people ask questions on quora that can easily be answered by google?*" requires many successful word deletion operations which is unlikely. After a few word deletions, the resulting sentence is very incoherent and has a very low chance of acceptance.

To address the aforementioned issue, we propose phrase-level modification in addition to word-level edits in [39] where we can directly propose to substitute a phrase by sampling in a latent space. Details are described in Section 3.3.

3.2.2 Stationary Distribution

In our framework, paraphrase generation task can be viewed as sampling from a desired distribution $p(y)$ where sentences are fluent and semantically close to the input sentence x. The desired distribution is also known as the stationary distribution of the Markov chain.

We follow [39] and define the (unnormalized) stationary distribution as:

$$\tilde{p}(y|x) \propto p(x) \cdot \underbrace{\mathcal{X}_c^0(x) ... \mathcal{X}_c^n(x)}_{\text{constraints}} \tag{3.3}$$

where $p(x)$ is the likelihood of a sentence in general and $\mathcal{X}_c^0(x) ... \mathcal{X}_c^n(x)$ are basically the requirements for the generated candidate sentence. Each constraint is implemented as a scoring function and a candidate sentence with a higher score has a higher acceptance chance. Constraints can be categorized into hard and soft. Hard constraints are binary indicators where the output is 1 if the constraint is satisfied and 0 otherwise. For example, paraphrases usually share the same keywords which can be applied as a hard constraint. Soft constraints are "*smoothed*" indicator functions that generate a score between 0 and 1 indicating the degree of satisfaction. For example, word embedding similarity between the original sentence and a candidate paraphrase is a soft constraint. The higher the similarity, the higher the score.

Therefore, the stationary distribution for paraphrasing can be defined as

$$\tilde{p}(y|x) = p_{\text{LM}}(y) \cdot \mathcal{X}_{\text{match}}(y|x) \tag{3.4}$$

where $p_{\text{LM}}(y)$ indicates the fluency of y ($f_{\text{fluency}}(y)$) which is a probability score given by a language model and $\mathcal{X}_{\text{match}}(y|x)$ is a score indicating the closeness of the original sentence and a paraphrase in meaning (f_{semantic}) in terms of constraints. We have used several constraints namely keyword matching, word embedding similarity, and skip-thoughts similarity [39].

We use RAKE [47] to extract keywords and put a hard constraint of keyword matching on generated paraphrases. If the generated paraphrase include the keywords then $\mathcal{X}_{\text{match}}(y|x) = 1$, and 0 otherwise. Word embedding similarity is a soft constraint. For every word in the input sentence, we find the closest word in the paraphrase. Then the word embedding score is either the minimum or the average of cosine similarities computed for all the words in the input sentence. Finally, skip thought learns fixed length representation of sentences and is a soft constraint computed between the original sentence and its paraphrase as a matching score.

3.2.3 Acceptance Rate

Since both proposal and stationary distribution could be arbitrary in the MH algorithm, there is no guarantee that we would obtain an unbiased sample of the stationary distribution if we just follow the proposal distribution.

Thus, the MH algorithm also computes an acceptance rate to reject a few proposals with a certain probability. The acceptance rate for word or phrase replacement, insertion, and deletion is computed by:

$$P^*_{\text{replace}}(y'|x) = \frac{p_{\text{replace}} \cdot g_{\text{replace}}(x|y').\pi(y')}{p_{\text{replace}} \cdot g_{\text{replace}}(y'|x).\pi(y')} \approx \frac{\pi(w_m|x_{-m}).\pi(y')}{\pi(w'_m|x_{-m}).\pi(x)} = 1 \tag{3.5}$$

$$P^*_{\text{insert}}(y'|x) = \frac{p_{\text{delete}} \cdot g_{\text{delete}}(x|y').\pi(y')}{p_{\text{insert}} \cdot g_{\text{insert}}(y'|x).\pi(y')} \tag{3.6}$$

$$P^*_{\text{delete}}(y'|x) = \frac{p_{\text{insert}} \cdot g_{\text{insert}}(x|y').\pi(y')}{p_{\text{delete}} \cdot g_{\text{delete}}(y'|x).\pi(y')} \tag{3.7}$$

Note that deletion and insertion operations are inverse operations to each other. Therefore, 3.6 and 3.7 are reciprocal. When with probability P^*, we accept the proposal, i.e., the current candidate becomes y' and with the probability of $1 - P^*$, we reject the proposal, and the current candidate remains y. The procedure of propose-and-accept is repeated until convergence [39].

Candidates with high scores have a high acceptance rate and vice versa. However, a candidate with a low score still has a chance to be accepted. This mechanism helps to avoid local minima. In theory, if the number of edits is large, the eventual sample will be an unbiased estimate of the stationary distribution.

3.3 Latent-Space Sampling for Phrase Edits

The main contribution of this thesis is introducing phrase-level edit into edit-based unsupervised paraphrase generation. In this section, we discuss the motivation behind our contribution and the process of phrase-level paraphrasing with VAE latent-space sampling.

3.3.1 Drawback of Word-Level Edits

Word-level editing is specifically powerful for synonyms replacement but is limited. There are cases that to achieve a high quality paraphrase, multiple words should be replaced at once. Replacement of only one word at a time may worsen the sentence quality, thereby reducing the proposal probability of becoming accepted. Consider a paragraphing process where three words, w_1, w_2, w_3, have to be deleted and two words ,w_4, w_5, inserted to generate a high quality paraphrase. During the total five steps of paraphrasing process, every intermediate node may not be a fluent sentence and thus the acceptance rate is low. This means that it is unlikely to have such paraphrase generation by word-level edits in the MH framework. For example, paraphrasing from "*How to learn a computer language like Java?*" to "*How to learn Java programming language?*" requires Transition from "*a computer language like Java*" to "*Java programming language*". We need to delete the word Java at the end of the sentence and then insert it before computer in the next step. The intermediate step of deletion results in "*How to learn a computer language like*" which is not a coherent sentence and will have a low chance of being accepted.

Our observation is that if we can directly perform phrase-level editing, for example, directly delete w_1, w_2 and w_3 and insert w_4 and w_5 afterward, we only need to evaluate the acceptance rate on the final sentence once, which will yield a higher chance for large edits in paraphrasing. Phrase replacement is capable to apply this transition in one step by sampling from latent-space for a phrase like "*a computer language like Java*". The generated candidates likely include "*Java programming language*" as a candidate paraphrase which results in a fluent sentence with a high probability of being accepted.

Therefore, we propose a phrase-level paraphrasing method by making use of an external parser for phrase detection. Once a phrase is selected, we sample from a VAE latent space for phrase-level paraphrasing as a proposal in MH sampling.

3.3.2 Phrase Detection

A phrase is a group of words that express a particular meaning. For example, in the sentence "*J.K Rowling is publishing a new children's book online*", "*a new children s' book*" is a more meaningful unit than "*is publishing a new children*" even though both have 5 tokens. [1]

To perform meaningful phrase-level paraphrasing, we first detect phrases by constituency parsing with the CoreNLP toolkit [35]. Figure 3.1 shows an example of the constituency tree on the sentence "*J.K. Rowling is publishing a new children's book online.*" where each node is known as a constituent of a sentence, such as "*J.K Rowling*" being a noun phrase (NP) and "*publishing a new children's book online*" being a verb phrase (VP). We treat an intermediate node in the parse tree as a phrase if the node has more than a certain number of leaf words. This allows us to perform paraphrasing at different granularities of the phrases like "*publishing a new children's book online*" and "*a new children's book online*".

It should be noted that to generate a candidate in each iteration of paraphrasing, we perform phrase detection on the most recently generated para-

[1] "*children's*" is tokenized into two parts: "*children*" and "*'s*".

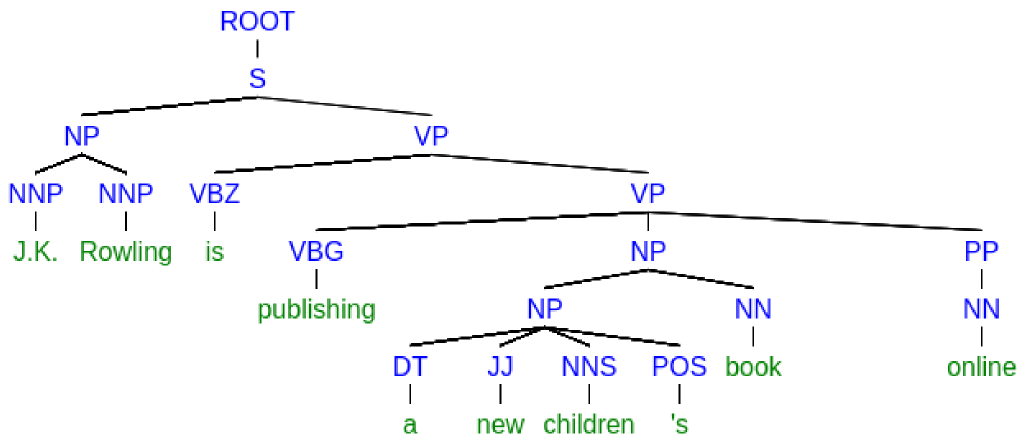

Figure 3.1: Constituency parse tree of a sentence achieved with CoreNLP toolkit.

phrase.

3.3.3 Autoencoding with VAE

Variational autoencoders are different than traditional autoencoders by imposing a prior distribution $p(z)$ on the latent variable z [27]. Typically the prior distribution is set to a standard normal distribution $\mathcal{N}(0, \mathbf{I})$. Given an input sentence y, VAE encodes the data in a latent variable z and the decoder reconstructs y from z. For a generative model that is parametrized by θ, $p_\theta(\mathbf{Z}, \mathbf{Y}) = p_\theta(\mathbf{Z})p_\theta(\mathbf{Y}|\mathbf{Z})$ and a dataset $\mathcal{D} = \{y^{(n)}\}_{n=1}^{N}$, the likelihood of a data point is as follows

$$\begin{aligned} logp_\theta(y^{(n)}) &\geq \mathbb{E}_{z\sim q_\phi(z|y^{(n)})}\left[log\left\{\frac{p_\theta(y^{(n)}, z)}{q_\phi(z|y^{(n)})}\right\}\right] && (3.8)\\ &= \mathbb{E}_{z\sim q_\phi(z|y^{(n)})}\left[logp_\theta(y^{(n)}|z)\right] - KL\left(q_\phi(z|y^{(n)})||p(z)\right) && (3.9)\\ &\triangleq \mathcal{L}^{(n)}(\theta, \phi) && (3.10)\end{aligned}$$

where $q_\phi(z|y)$ and $p_\theta(y|z)$, parametrized by ϕ and θ, are modeled as neural networks. The training objective of VAE is to minimize the lower bound of the likelihood $\mathcal{L}(\theta, \phi)$:

$$J^{(n)} = J_{\text{rec}}(\theta, \phi, y^n) + KL\left(q_\phi(z|y^{(n)})||p(z)\right) \quad (3.11)$$

The first term, called the reconstruction loss, maximizes how well phrases are expected to be decoded from the latent variable z. We also want the encoder posterior $q_\phi(z|y^n)$ to be similar to the prior distribution ($p(z)$). The second term, which can be thought of as a regularization term, is the KL-divergence between z's posterior and prior distributions.

3.3.4 Candidate Phrase Generation

To generate candidates for phrase replacement, we make use of a probabilistic latent space model. We first train an autoencoder on phrases extracted from training datasets and feed in a detected phrase by the parser, encode it, and sample a new point in the neighborhood as a candidate paraphrase. Multiple candidates are achieved by drawing multiple samples from the neighborhood where each candidate is likely have a different expression while preserving the input sentence semantic information. It should be noted that the phrase detection and selection is performed in each iteration.

We sample a latent code by the posterior $z_* \; p(z|x_*)$. Our posterior sampling enables us to encode the information on the input paraphrase, whereas varying the expression by the learned distribution. Compared with adding random noise to a deterministic autoencoder, the VAE is able to learn an adaptive noise to each data sample, so that noise may vary based on the uncertainty

3.4 Objective Function

Our paraphrasing technique maximizes an objective function $f(x)$ that considers different aspects of a generated candidate paraphrase that were previously discussed in details in section 3.2.2. The objective function can be defined as to maximize:

$$f(x) = f_{\text{fluency}}(y') \cdot f_{\text{semantic}}(y', x) \tag{3.12}$$

where $f_{\text{fluency}}(y')$ is given by a language model and $f_{\text{semantic}}(y', x)$ is a matching score that represents semantic preservation with respect to the original sentence.

3.5 Summary of Our Algorithm

In summary, we propose an algorithm that combines a sentence space sampler with a VAE latent space sampler. We present three word level operations as well as a novel phrase level operation. Paraphrase generation happens in an iterative manner where in each iteration, one operation along with one random word or phrase forms a proposal. A proposal is either accepted or rejected with a probability score related to the quality of the generated paraphrase and some required constraints. If a proposal becomes accepted, a new paraphrase is generated.

Procedure 1 Paraphrase Generation

```
Input: Original Sentence, $x$, and number of paraphrases to generate, $N$
Output: Generated Paraphrase, $x_\tau$
  $y \leftarrow x$
  for $t \in \{1, ..., N\}$ do
    Randomly select a modification operation from word replacement, word deletion, word insertion, and phrase replacement
    if modification is a word level operation then
      Randomly select a word
      Generate a candidate, $y'$, with discrete word space sampling
    else
      Extract phrases in the input sentence by constituency parsing and randomly select a phrase
      Generate a candidate, $y'$, by sampling from VAE continuous latent space
    end if
    Compute the acceptance probability ($p_{\text{accept}}$) by Equations 3.5, 3.6, and 3.7.
    With probability $p_{\text{accept}}$, $y \leftarrow y'$
  end for
  return $x_\tau$ s.t. $\tau = \text{argmax}_{\tau \in \{1,...,N\}} f(x_\tau)$ where $f(x_\tau)$ is defined by Equation 3.12
```

Chapter 4

Experiments and Results

4.1 Datasets and Experimental Setup

For our experiments training and evaluation, we used two standard benchmark datasets: 1) Stanford Natural Language Inference (SNLI) dataset [8], and 2) Quora dataset[1].

We use both SNLI and Quora datasets for training and evaluation of autoencoding sentences. For paraphrase generation experiments we only use Quora dataset for training and evaluation which contains pairs of paraphrase sentences.

4.1.1 Quora Dataset

Following the previous work on paraphrase generation task [21], [28], [43], we use the standard benchmark Quora dataset, containing 149k pairs of paraphrase sentences and 260k non-paraphrase sentences. We used the 149k pairs of paraphrases as ground truth for our training and evaluation.

We follow the standard split with 3k and 30k samples for validation and testing, respectively. The rest of the dataset is used for training the autoencoding and paraphrase generation models. Unfortunately, the exact split data from the previous works is not available and we performed our own data split, and our results are statistically comparable to previous work. It should also be noted that the sentences in the Quora dataset are questions that are different from the sentences in SNLI. Some samples are shown in Table 4.1.

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

Sentences
Do you believe there is life after death?
What Game of Thrones villain would be the most likely to give you mercy?
What are some examples of products that can be made from crude oil?
How do I read and find my YouTube comments?
How can I be a good geologist?
Paraphrases
Is it true that there is life after death?
What Game of Thrones villain would you most like to be at the mercy of?
What are some of the products made from crude oil?
How can I see all my Youtube comments?
What should I do to be a great geologist?

Table 4.1: Sample from Quora corpus.

Text
A smiling costumed woman is holding an umbrella.
A soccer game with multiple males playing.
An older and younger man smiling.
A man inspects the uniform of a figure in some East Asian country.
A black race car starts up in front of a crowd of people.

Table 4.2: Sample from SNLI corpus.

4.1.2 SNLI Dataset

The pairs of paraphrase sentences in Quora dataset are required for paraphrase generation evaluation. However, to train an autoencoder, we do not need labels and it is easy to collect unlabeled data. Therefore, for the purpose of improvement in the quality of the VAE latent-space representation, we also use the SNLI dataset for training VAE.

The SNLI dataset is a massive corpus of text that contains 570k comparatively simple sentences of premise and hypothesis pairs. Sentences are manually annotated with Amazon Mechanical Turk with the labels entailment, contradiction, and neutral. It is created through an image captioning task and is widely used for sentence representation modeling. Some samples from the dataset are shown in Table 4.2.

LSTM Hidden Dimension	100d, single layer
Word Embeddings	300d
Latent Dimension	100d
Epochs	20
Learning Rate	Fixed rate of 0.001
Batch Size	128
Max Sequence Length	20
Vocab Size	30000

Table 4.3: Experimental settings for autoencoding.

4.1.3 Setup

Autoencoding

While our proposed model uses VAE for phrase-level paraphrasing, we also compared VAE with Wasserstein autoencoders (WAE) [54]. VAEs require that posterior distribution $q(z|x)$ to be close to prior distribution $p(z)$ for every input sequence of words x. However, WAE imposes a different regularization on posterior distribution through the aggregated posterior of z, i.e., $q(z) = \sum_x q(z|x)p_D(x)$. It is much easier to train a WAE as it does not require training tricks such as KL annealing and word dropout [3].

We used Adam optimizer [26] for all autoencoding models with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. Compared to WAE, it is notoriously difficult to train a VAE in the RNN settings and usually involves KL annealing and word dropout. In our experiments, we followed the training practices in [2] that uses KL annealing and adapted *peaking annealing* that anneals λ_{VAE} with a sigmoid function and stops annealing $\lambda.KL$ when it reaches its peak value. Word dropout starts with no dropout, increasing up to a maximum value of 0.5 with a rate of 0.05 in each epoch.

Experimental settings (for both VAE and WAE) are mentioned in Table 4.3 in details.

Paraphrase Generation

For unsupervised paraphrase generation, we require a language model to evaluate the likelihood of a question after each edition. We trained a language model using only the questions in the training set. Our language model is a simple two-layer LSTM with a hidden size of 300 and a vocabulary size of the top 30k most frequent words.

4.2 Evaluation Measures

BLEU score [42] against the ground truth is widely adapted as an evaluation metric in text generation tasks.

For autoencoding evaluation, we use the BLEU score to evaluate reconstruction performance by autoencoders. The fluency of the generated sentence is measured with the loss value of the OpenAI GPT model proposed in [46]. To evaluate if the distribution of generated sentences by autoencoders is close to the original sentence, we report the unigram-KL **UniKL** which evaluates how close is the distribution of the generated sentences to the training corpus, entropy of the word distribution, as well as the average length of the generated sentences.

For paraphrase generation, the BLEU score against the ground truth alone could not measure the diversity between the paraphrase and the original sentence. Miao et al., [39] demonstrates that simply copying the input sentence as a paraphrase results in a high BLEU score. Therefore, we follow [39] and also include the BLEU score against the original sentence. We also include iBLUE score [52] which is a variant of BLEU score penalized by similarity with the original sentence.

4.3 Overall Results

4.3.1 Phrase Autoencoding

The evaluation performance of autoencoders in sentence generation is shown in Table 4.4. Both WAE and VAE include penalties in addition to the reconstruction loss. WAE has a higher BLEU score in reconstruction when trained with appropriate hyperparameters. The UniKL is also lower for WAE which means that the generated sentences are closer to the original sentences dis-

	BLEU$^{\uparrow}$	**Fluency** (OpenAI GPT Loss$^{\downarrow}$)	**UniKL**$^{\downarrow}$	**Entropy**	**AvgLen**
Corpus	-		-	$\rightarrow$ 6.514	$\rightarrow$ 6.87
WAE$_{\lambda_{\mathrm{KL}}=0}$	**78.97**	11.13	**0.029**	**6.441**	**6.70**
WAE$_{\lambda_{\mathrm{KL}}=0.01}$	71.52	11.21	0.034	6.415	6.53
WAE$_{\lambda_{\mathrm{KL}}=0.1}$	55.55	11.10	0.037	6.410	6.49
WAE$_{\lambda_{\mathrm{KL}}=1.0}$	49.84	11.43	0.040	6.404	6.42
VAE (KL-annealed)	40.78	**10.97**	0.053	6.346	6.23

Table 4.4: Comparison of different phrase-level autoencoders. $^{\uparrow}/^{\downarrow}$ means the higher/lower, the better. It is preferable that the generated sentences be close to corpus statistics in terms of **Entropy** and **AvgLen** (indicated by $\rightarrow$).

tribution. In terms of entropy and average length of the sentences, WAE is slightly better by generating sentences similar to the corpus in terms of statistics. Overall, the WAE performs better in phrase reconstruction as expected, but in paraphrase generation, we like the generated sentences to be different than the original input which makes VAE a better choice. A perfect reconstruction is impossible in VAE as each input's posterior is encoded to be close to the prior. The fluency of the generated sentences is better in VAE. This means that the generated sentences in our VAE are more fluent than the ones in WAE. Fluency of generated sentences reduces the rejection probability in MH sampling during paraphrasing and results in better paraphrases. A comparison of paraphrasing performance is presented in the following section.

4.3.2 Paraphrasing

Table 4.5 presents the evaluation performance on test data. The results of supervised methods with a variety of training sample sizes are also included from [39] to demonstrate the effect of parallel data size in supervised setting. Unfortunately, some of the supervised methods consider BLEU$_{\text{orig}}$ during evaluation, and therefore iBLEU score is missing.

We propose two variants of our PhraseVAE paraphrase generation system. In one variant we do not include the word edition operations and in another, we have all four operations of word replacement, deletion, and insertion as well as the phrase level replacement operation. The results show that our phrase replacement operation can achieve a performance close to the CGMH system that paraphrases in word level only. This is due to the fact that a phrase

	Model	iBLEU$^{\uparrow}$	BLEU$_{\text{ref}}^{\uparrow}$	BLEU$_{\text{orig}}^{\downarrow}$
	Origin Sentence	-	30.49	100.0
Supervised	VAE-SVG (100k)	-	22.50	-
	VAE-SVG-eq (100k)	-	**22.90**	-
	VAE-SVG (50k)	-	17.10	-
	VAE-SVG-eq (50k)	-	17.40	-
Supervised	Seq2seq (100k)	**17.13**	22.79	33.83
	Seq2seq (50k)	15.4	20.18	27.59
	Seq2seq (20k)	12.85	16.77	**22.44**
Unsupervised	VAE	5.6	9.25	**27.23**
	CGMH	11.29	17.97	48.82
	PhraseVAE+MH (no word edition)	11.25	19.64	64.23
	PhraseVAE+MH	**13.31**	20.22	48.86
	PhraseWAE+MH ($\lambda_{KL} = 0.1$)	12.59	**20.30**	56.70

Table 4.5: Paraphrase generation performance on the Quora dataset. Performance resutls for supervised methods are quoted from [39]. $^{\uparrow}/^{\downarrow}$ means the higher/lower, the better.

edition including multiple words can result in a word edition. For example replacement of *"Donald Trump presidency"* with *"Donald Trump election"* is equal to a word edition where *"presidency"* is replaced with *"election"*. Similarly, a replacement with phrases *"Donald Trump 2016 presidency"* and *"Trump presidency"* is equal to a word insertion and deletion, respectively.

As expected, VAE performance in paraphrasing is slightly better than WAE. Best performance for WAE achieved when λ_{KL} is set to 0.1 that had the best fluency score in reconstruction as mentioned in Table 4.4.

4.4 Analysis

In this section, we discuss the paraphrasing performance in detail and provide examples. We also discuss the evaluation and VAE training difficulties.

4.4.1 Best Paraphrase Search

Many paraphrases are generated by iterative phrase and word level editions. To find the best paraphrase sentence among many, [39] selects the first generated paraphrase that has a BLEU$_{\text{ref}}$ score lower than 55 to make sure a significant difference between the generated paraphrase and the original sen-

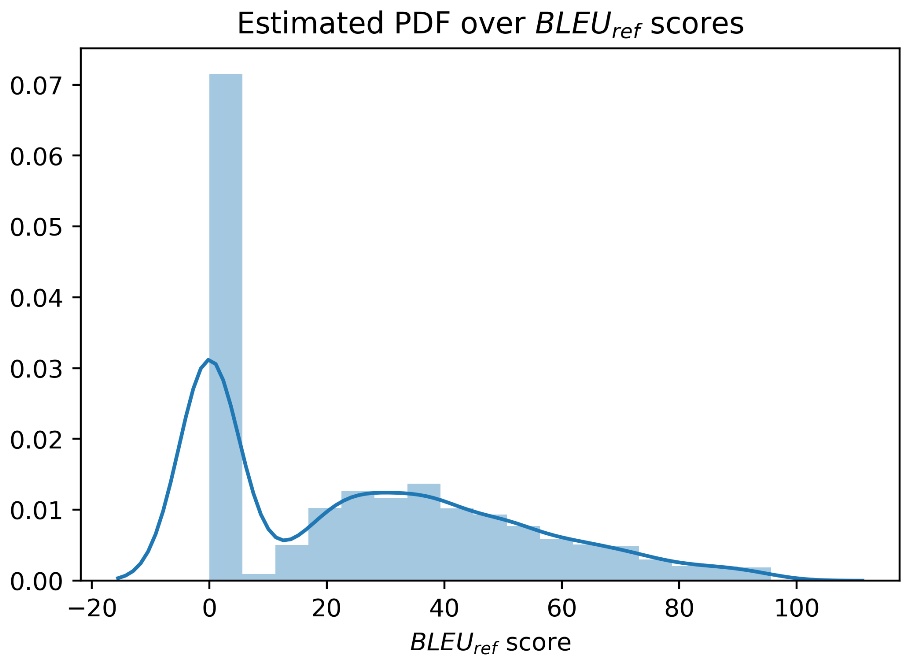

Figure 4.1: $\mathrm{BLEU}_{\mathrm{ref}}$ distribution on Quora validation sentences.

tence. We find this incomplete by studying the $\mathrm{BLEU}_{\mathrm{ref}}$ scores distribution which is shown for Quora validation sentences in Figure 4.1. Around 15% of the paraphrases in the validation set have a $\mathrm{BLEU}_{\mathrm{ref}}$ higher than 55. If we follow the [39] approach in final paraphrase selection, we are losing a relatively large chunk of sentences that can be easily paraphrased with fewer editions compared to the other sentences.

To overcome this problem, we use BERT [14] language model as a semantic search engine to rank the generated paraphrases. We first encode the original question as a vector and compute the dot product between the original question and the encoded vector of each generated paraphrase. Finally, we select the generated paraphrase with the highest score as the final paraphrase. To avoid copying the exact input, we only consider the paraphrases that have a $\mathrm{BLEU}_{\mathrm{ref}}$ lower than 90. This value is a lot higher than the threshold set by [39] and only 1% of ground truth paraphrases in the validation set have a $\mathrm{BLEU}_{\mathrm{ref}}$ higher than 90.

4.4.2 Paraphrasing Quality

Table 4.4 demonstrates that our method outperforms CGMH with higher BLEU_{ref} and iBLEU scores. We generally have a higher $\text{BLEU}_{\text{orig}}$ score. This is because phrase level edition compared to word level editions applies changes on a bigger chunk of a sentence in each step. This makes it harder to become accepted, thereby lower level of acceptance and higher BLEU_{ref}.

Table 4.6 and Table 4.7 show some paraphrases generated by our method using VAE and WAE as autoencoding methods. We can see that the generated paraphrases have high quality and are close to the original sentence semantically while different syntactically.

4.4.3 Paraphrasing Examples and Process

Table 4.8 and 4.9 provides a real example of the paraphrase generation process with our system and paraphrasing system of [39], respectively. In the case of our system, first a word replacement operation is performed following by a phrase level. To replace a phrase, the constituency parsing tree of the original sentence, *what are your top 5 movies for 2016?*, is generated which is shown in Figure 4.2 and one random phrase is selected. Here *are the top 5 movies for 2016* is selected randomly. Sampling from our VAE latent space results in the following candidate phrases:

- *"are your top ten movies of 2016"*
- *"are your top 10 favourite movies"*
- *"are your top ten favorite horror movies"*
- *"are the best movie of 2016"*
- *"are your top 5 favorite bollywood movies"*

from which *"are the best movies of 2016"* is successfully accepted by MH, resulting in *"what are the best movies of 2016"* as the final paraphrase. The provided example shows how phrase replacement effectively reduces the number of editions during the process of paraphrase generation.

#	Type	Examples
1	Original	what are your top 5 movies for 2016?
	Reference	what are the best movies of 2016?
	Generated	what are some best movies of 2016?
2	Original	what is ground?
	Reference	what is a ground?
	Generated	what is a ground?
3	Original	how did donald trump got elected when there are so many people against him?
	Reference	how did donald trump win the election?
	Generated	how did donald trump win the presidency?
4	Original	which is the best smartphone i can buy under rs.6000?
	Reference	which is the best phone to buy under rs.6000?
	Generated	which is the best phone to buy under 15000
5	Original	how do i really make money online?
	Reference	how can i earn money online?
	Generated	how can i make money online
6	Original	how do i get my english better?
	Reference	how can i improve my english pronunciation?
	Generated	how can i improve my english?
7	Original	how do I improve my communication skills?
	Reference	how can I improve my communication effectively?
	Generated	how can I improve my communication skill?
8	Original	what is the best harry potter movie and why? is it also your favorite? why or why not?
	Reference	which is the best harry potter movie?
	Generated	which is the best harry potter movie?
9	Original	how come people on quora ask questions here, when they can get them on google and why is quora just a question/answer site, will it expand?
	Reference	why do people ask questions on quora that can easily be answered by google?
	Generated	why do people ask questions on quora that can easily be answered by google?
8	Original	what are the best online short courses in digital marketing?
	Reference	which is the best digital marketing course?
	Generated	which is the best digital marketing course?

Table 4.6: Paraphrase generation examples generated by VAE.

#	Type	Examples
1	Original	will there really be any war between india and pakistan over the uri attack? what will be its effects?
	Reference	will there be a nuclear war between india and pakistan?
	Generated	will there be a nuclear war between india and pakistan?
2	Original	which is the best course for digital marketing?
	Reference	what are the best online courses for digital marketing?
	Generated	what are the best online courses for digital marketing?
3	Original	why did trump win the election?
	Reference	why did donald trump win the election?
	Generated	how did donald trump win the presidency?
4	Original	how can i earn money using my quora profile?
	Reference	how can i earn money through quora?
	Generated	how can i earn money through youtube?
5	Original	who will win upcoming usa election?
	Reference	who will win the us election?
	Generated	who will win the us election?
6	Original	what are the most famous caves in the chhattisgarh?
	Reference	which are the famous caves in chhattisgarh?
	Generated	what are the famous caves in chhattisgarh?
7	Original	who would win an all out war between pakistan and india if no other country got involved?
	Reference	who will win in a war between india and pakistan?
	Generated	who would win in a war between india and pakistan?
8	Original	how can i get rid of anxiety?
	Reference	how do i get rid of my anxiety?
	Generated	how do i get rid of my acne
9	Original	how do i recover emails that i deleted forever in gmail?
	Reference	how do i recover deleted emails in my gmail account?
	Generated	how do i recover deleted emails in my gmail password?
10	Original	what are the easiest ways for me to make money?
	Reference	what are the ways to earn money?
	Generated	what are the ways to earn money?
10	Original	what causes obesity?
	Reference	what are the causes of obesity?
	Generated	what are the causes obesity?

Table 4.7: Paraphrase generation examples generated by WAE.

Step	State (Sentence)	Proposal
Origin	what are your top 5 movies for 2016?	**replace word** *your* with *the*
1	what are the top 5 movies for 2016?	**replace phrase** *are the top 5 movies for 2016* with *are the best movies of 2016*
output	what are the best movies of 2016 ?	-

Table 4.8: Paraphrase generation process with phrase-level and word-level.

Step	State (Sentence)	Proposal
Origin	what are your top 5 movies for 2016?	**replace word** *top* with *best*
1	what are your best 5 movies for 2016?	**replace word** *5* with *comic*
2	what are your best comic movies for 2016?	**replace word** *your* with *the*
3	what are the best comic movies for 2016?	**delete word** *comic*
output	what are the best movies of 2016?	-

Table 4.9: Paraphrase generation process in word-level.

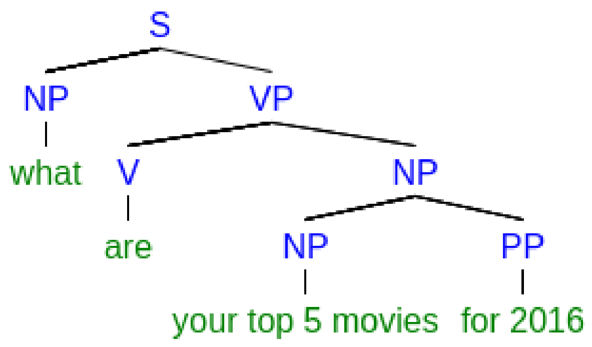

Figure 4.2: Constituency parsing tree of an example sentence used for phrase replacement operation of our system.

Phrase replacement becomes more crucial in long sentences where it is necessary to delete a large chunk of a sentence and make it shorter. Removing words usually results in a sentence that is no longer coherent or close semantically to the original sentence. Therefore, there is a high chance that a word deletion becomes rejected. For example Table 4.10 shows an example of a real paraphrase generation process where a long sentence is paraphrased in only one iteration to a shorter sentence with phrase replacement action. Here the entire sentence is treated as a phrase and in step 1, phrase-replacement action paraphrased it successfully to a short sentence. The output is then followed by some word-level replacement to generate the final paraphrase. To generate such a paraphrase with word-level editions only, the "*and why? is it also your favorite?*" part consisting of seven words should be deleted in seven successful word-deletion actions which are very unlikely. Another example where this problem is solved with phrase level replacement is provided in Table 4.11.

Overall, there are three types of generated paraphrases:

1. Paraphrases that are fluent and semantically related. Below is a real example generated by our system:
   - Original: "What is the best way to lose weight fast?"
   - Paraphrase: "What is the best way to lose weight fast?"
2. Paraphrases that are fluent, but semantic has changed. Below is a real example generated by our system:
   - Original: "What is the best thing you had done in your life until now?"
   - Paraphrase: "What is the most embarrassing moment in your life?"
3. Totally distorted sentences which are very rare.

4.4.4 Evaluation Difficulties

Unfortunately, the BLEU score does not consider meaning. Studying the generated paraphrases, we have observed that there are several cases that the

Step	State (Sentence)	Proposal
Origin	what is the best harry potter movie and why? is it also your favorite?	**replace phrase** (The entire sentence is replaced with the next sentence.)
1	what is the best harry potter film	**replace word** *film* with *book*
2	what is the best harry potter book?	**replace word** *what* with *which*
3	which is the best harry potter book?	**replace word** *book* with *movie*
Output	which is the best harry potter movie?	-

Table 4.10: Paraphrase generation of a long sentence with phrase-level.

Step	State (Sentence)	Proposal
Origin	how did donald trump got elected when there are so many people against him?	**replace phrase** (The entire sentence is replaced with the next sentence.)
1	how did donald trump win the 2016	**replace word** *2016* with *presidency*
Output	how did donald trump win the presidency	-

Table 4.11: Paraphrase generation of a long sentence with phrase-level.

generated paraphrase is semantically similar to the reference paraphrase but expressed with different words which result in an unfair penalty for the system performance. For example, the generated paraphrases number 1 and 3 in Table 4.6 are valid paraphrases both semantically and syntactically but different from the reference paraphrase. This example shows that including a single paraphrase in paraphrase generation datasets as a reference is not enough. A single sentence can have multiple valid paraphrases. As a solution, one can provide multiple ground truth paraphrases for each sentence. However, it results in a higher cost for data collection.

Another problem is the presence of extra information in the reference paraphrase which is not predictable from the original sentence. For instance generated paraphrase number 6 in Table 4.6 is a totally valid paraphrase including all the information in the original sentence. However, the reference sentence is different by introducing extra information through the use of the word ”pronunciation”.

4.4.5 Training Difficulties

As mentioned, two losses are involved in training a VAE: 1) reconstruction loss and 2) KL divergence loss between the posterior and prior of the latent space. Unlike images, it is extremely hard to train a VAE in NLP domain [27] as the KL term tends to vanish to zero which results in an ineffective latent space. To tackle this problem we follow the tricks from previous works which involve KL annealing and word dropout [9].

We show that our VAE latent space is in fact variational by conducting an experiment to compare the KL loss with and without applying the KL annealing. KL loss is believed to demonstrate that the latent space is in fact variational. Without KL divergence loss, the latent variable z is completely ignored and VAE memorizes each input as a single latent point [58]. Figure 4.3 demonstrates the KL loss where it is weighted by different values of λ_{VAE}.

Fortunately, WAEs networks do not have training difficulties of VAEs and still retain probabilistic properties. There is no need for KL annealing or word dropout in training a WAE and it is robust to hyperparameters.

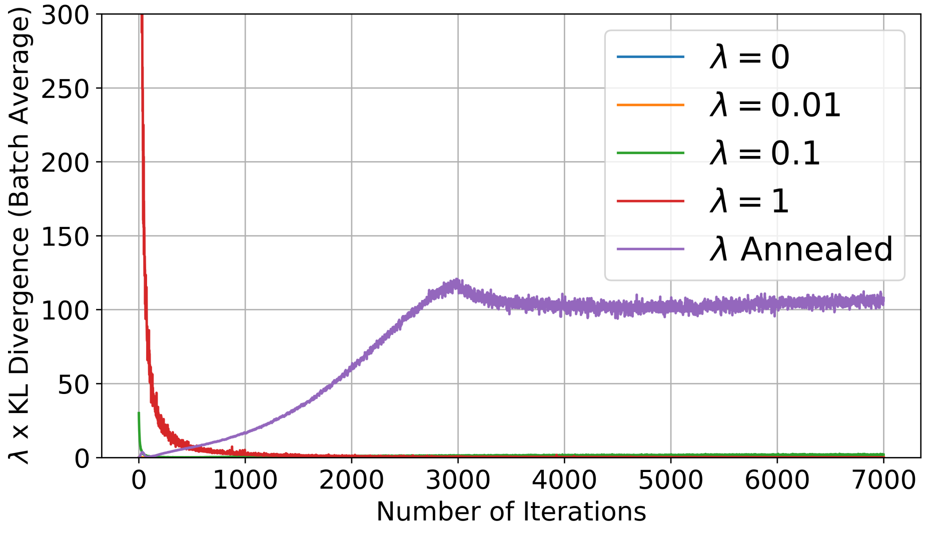

Figure 4.3: VAE KL loss $(\lambda \cdot KL)$ learning curve for different values of λ and the training trick where λ is annealed.

Chapter 5

Conclusion and Future Work

5.1 Summary of Contributions

Our main objective of this thesis is to develop a system that can generate high quality paraphrases with no supervision. To this end, our system employs an unsupervised approach that works in word level and phrase level. We proposed a novel approach of phrase level paraphrasing through autoencoding that helps with multiple words replacement. We used VAE for autoencoding the phrases and our experiments demonstrated that our novel phrase level operation on top of the word level operations can improve the quality of generated paraphrases. The main contributions and findings of this thesis are summarized in the following:

- We studied the drawbacks of word level modification for paraphrase generation and proposed a paraphrasing system that combines discrete word level sampling with a continuous phrase level sampling. Therefore, our system, while benefiting from word level modification, is able to address the existing issues in techniques that operate at word level only.

- We showed that paraphrasing with only word level modifications does not perform well for long sentences where multiple word modifications are required to generate the ground truth paraphrase. As mentioned before, the intermediate sentences are not fluent which results in a high rejection probability.

- Through the sample paraphrases generated by our system, we showed that there are several issues with the BLEU evaluation metric. We provided examples of generated paraphrases that our paraphrase and the ground truth paraphrase are semantically equal but expressed syntactically different, resulting in a heavy penalty by BLEU metric. This is due to the fact that BLEU metric does not consider sentence meaning.

- We studied the similarity distribution between the original sentences and the ground truth paraphrases. Our study showed that many of the existing paraphrases in the Quora dataset are actually very close to the original sentence in terms of the BLEU score. Therefore, to continue paraphrasing until a significant literal difference is made can result in a poor score. To address this problem, we suggested using a language model to rank the generated paraphrases based on semantic score and select the top one as the final paraphrase.

5.2 Limitations and Future Perspectives

The current approach for phrase level sampling from a continuous latent space does not consider separating semantic information from syntactic information. An interesting direction for our phrase level operation is to sample from a disentangled syntactic and semantic (DSS) space. A DSS-VAE is proposed by [4] that uses two separate latent space variables z_{sem} and z_{syn} to model the semantic and syntactic information of a sentence, respectively. This is useful as a semantically similar phrase can be generated by fixing z_{sem} and sampling in z_{syn} space. As future work, we consider experiments with DSS-VAE to generate better candidates for phrase level replacement.

References

[1] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[2] H. Bahuleyan, L. Mou, O. Vechtomova, and P. Poupart, "Variational attention for sequence-to-sequence models," *arXiv preprint arXiv:1712.08207*, 2017.

[3] H. Bahuleyan, L. Mou, H. Zhou, and O. Vechtomova, "Stochastic wasserstein autoencoder for probabilistic sentence generation," *arXiv preprint arXiv:1806.08462*, 2018.

[4] Y. Bao, H. Zhou, S. Huang, L. Li, L. Mou, O. Vechtomova, X. Dai, and J. Chen, "Generating sentences from disentangled syntactic and semantic spaces," *arXiv preprint arXiv:1907.05789*, 2019.

[5] R. Barzilay and L. Lee, "Learning to paraphrase: An unsupervised approach using multiple-sequence alignment," *arXiv preprint cs/0304006*, 2003.

[6] D. Beeferman and A. Berger, "Agglomerative clustering of a search engine query log," in *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2000, pp. 407–416.

[7] I. A. Bolshakov and A. Gelbukh, "Synonymous paraphrasing using wordnet and internet," in *International Conference on Application of Natural Language to Information Systems*, Springer, 2004, pp. 312–323.

[8] S. R. Bowman, G. Angeli, C. Potts, and C. D. Manning, "A large annotated corpus for learning natural language inference," *arXiv preprint arXiv:1508.05326*, 2015.

[9] S. R. Bowman, L. Vilnis, O. Vinyals, A. M. Dai, R. Jozefowicz, and S. Bengio, "Generating sentences from a continuous space," *arXiv preprint arXiv:1511.06349*, 2015.

[10] C. Callison-Burch, "Paraphrasing and translation," PhD thesis, University of Edinburgh Edinburgh, 2007.

[11] Z. Cao, C. Luo, W. Li, and S. Li, "Joint copying and restricted generation for paraphrase," *arXiv preprint arXiv:1611.09235*, 2016.

[12] S. Chib and E. Greenberg, "Understanding the metropolis-hastings algorithm," *The american statistician*, vol. 49, no. 4, pp. 327–335, 1995.

[13] P. W. Culicover, "Paraphrase generation and information retrieval from stored text.," *Mech. Translat. & Comp. Linguistics*, vol. 11, no. 3-4, pp. 78–88, 1968.

[14] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[15] L. Dong, J. Mallinson, S. Reddy, and M. Lapata, "Learning to paraphrase for question answering," *arXiv preprint arXiv:1708.06022*, 2017.

[16] M. Dras, "A meta-level grammar: Redefining synchronous tag for translation and paraphrase," in *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, Association for Computational Linguistics, 1999, pp. 80–87.

[17] A. Fujita, K. Furihata, K. Inui, Y. Matsumoto, and K. Takeuchi, "Paraphrasing of japanese light-verb constructions based on lexical conceptual structure," in *Proceedings of the Workshop on Multiword Expressions: Integrating Processing*, Association for Computational Linguistics, 2004, pp. 9–16.

[18] A. Fujita, S. Kato, N. Kato, and S. Sato, "A compositional approach toward dynamic phrasal thesaurus," in *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, Association for Computational Linguistics, 2007, pp. 151–158.

[19] C. Gardent, M. Amoia, and E. Jacquey, "Paraphrastic grammars," 2004.

[20] C. Gardent and E. Kow, "Generating and selecting grammatical paraphrases," in *Proceedings of the Tenth European Workshop on Natural Language Generation (ENLG-05)*, 2005.

[21] A. Gupta, A. Agarwal, P. Singh, and P. Rai, "A deep generative framework for paraphrase generation," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[22] M. Iyyer, J. Wieting, K. Gimpel, and L. Zettlemoyer, "Adversarial example generation with syntactically controlled paraphrase networks," *arXiv preprint arXiv:1804.06059*, 2018.

[23] K. S. Jones and J. I. Tait, "Automatic search term variant generation," *Journal of documentation*, 1984.

[24] D. Kauchak and R. Barzilay, "Paraphrasing for automatic evaluation," in *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, Association for Computational Linguistics, 2006, pp. 455–462.

[25] D. Keysers, "Comparison and combination of state-of-the-art techniques for handwritten character recognition: Topping the mnist benchmark," *arXiv preprint arXiv:0710.2231*, 2007.

[26] D. P. Kingma and J. Ba, "Adam (2014), a method for stochastic optimization," in *Proceedings of the 3rd International Conference on Learning Representations (ICLR), arXiv preprint arXiv*, vol. 1412, 2015.

[27] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[28] Z. Li, X. Jiang, L. Shang, and H. Li, "Paraphrase generation with deep reinforcement learning," *arXiv preprint arXiv:1711.00279*, 2017.

[29] C.-Y. Lin, "Rouge: A package for automatic evaluation of summaries," in *Text summarization branches out*, 2004, pp. 74–81.

[30] D. Lin and P. Pantel, "Discovery of inference rules for question-answering," *Natural Language Engineering*, vol. 7, no. 4, pp. 343–360, 2001.

[31] X. Liu, L. Mou, F. Meng, H. Zhou, J. Zhou, and S. Song, "Unsupervised paraphrasing by simulated annealing," *arXiv preprint arXiv:1909.03588*, 2019.

[32] S. Ma, X. Sun, W. Li, S. Li, W. Li, and X. Ren, "Query and output: Generating words by querying distributed word representations for paraphrase generation," *arXiv preprint arXiv:1803.01465*, 2018.

[33] N. Madnani and B. J. Dorr, "Generating phrasal and sentential paraphrases: A survey of data-driven methods," *Computational Linguistics*, vol. 36, no. 3, pp. 341–387, 2010.

[34] J. Mallinson, R. Sennrich, and M. Lapata, "Paraphrasing revisited with neural machine translation," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, 2017, pp. 881–893.

[35] C. D. Manning, M. Surdeanu, J. Bauer, J. R. Finkel, S. Bethard, and D. McClosky, "The stanford corenlp natural language processing toolkit," in *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, 2014, pp. 55–60.

[36] K. R. McKeown, "Paraphrasing using given and new information in a question-answer system," in *Proceedings of the 17th annual meeting on Association for Computational Linguistics*, Association for Computational Linguistics, 1979, pp. 67–72.

[37] ——, "Paraphrasing using given and new information in a question-answer system," *Technical Reports (CIS)*, p. 723, 1980.

[38] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *The journal of chemical physics*, vol. 21, no. 6, pp. 1087–1092, 1953.

[39] N. Miao, H. Zhou, L. Mou, R. Yan, and L. Li, “Cgmh: Constrained sentence generation by metropolis-hastings sampling,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 6834–6842.

[40] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, “Playing atari with deep reinforcement learning,” *arXiv preprint arXiv:1312.5602*, 2013.

[41] K. Owczarzak, D. Groves, J. Van Genabith, and A. Way, “Contextual bitext-derived paraphrases in automatic mt evaluation,” in *Proceedings of the Workshop on Statistical Machine Translation*, Association for Computational Linguistics, 2006, pp. 86–93.

[42] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, “Bleu: A method for automatic evaluation of machine translation,” in *Proceedings of the 40th annual meeting on association for computational linguistics*, Association for Computational Linguistics, 2002, pp. 311–318.

[43] A. Prakash, S. A. Hasan, K. Lee, V. Datla, A. Qadir, J. Liu, and O. Farri, “Neural paraphrase generation with stacked residual lstm networks,” *arXiv preprint arXiv:1610.03098*, 2016.

[44] H. Qin, “Question paraphrase generation for question answering system,” Master’s thesis, University of Waterloo, 2015.

[45] C. Quirk, C. Brockett, and W. B. Dolan, “Monolingual machine translation for paraphrase generation,” in *Proceedings of the 2004 conference on empirical methods in natural language processing*, 2004, pp. 142–149.

[46] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, *Improving language understanding by generative pre-training*, 2018.

[47] S. Rose, D. Engel, N. Cramer, and W. Cowley, “Automatic keyword extraction from individual documents,” *Text mining: applications and theory*, vol. 1, pp. 1–20, 2010.

[48] A. Roy and D. Grangier, “Unsupervised paraphrasing without translation,” *arXiv preprint arXiv:1905.12752*, 2019.

[49] A. Siddique, S. Oymak, and V. Hristidis, “Unsupervised paraphrasing via deep reinforcement learning,” in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 1800–1809.

[50] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, *et al.*, “Mastering the game of go without human knowledge,” *nature*, vol. 550, no. 7676, pp. 354–359, 2017.

[51] S. Soni and K. Roberts, “A paraphrase generation system for ehr question answering,” in *Proceedings of the 18th BioNLP Workshop and Shared Task*, 2019, pp. 20–29.

[52] H. Sun and M. Zhou, "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)*, 2012, pp. 38–42.

[53] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[54] I. Tolstikhin, O. Bousquet, S. Gelly, and B. Schoelkopf, "Wasserstein auto-encoders," *arXiv preprint arXiv:1711.01558*, 2017.

[55] P. Wallis, "Information retrieval based on paraphrase," in *Proceedings of PACLING Conference*, Citeseer, 1993.

[56] S. Wang, R. Gupta, N. Chang, and J. Baldridge, "A task in a suit and a tie: Paraphrase generation with semantic augmentation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 7176–7183.

[57] M. Wasim, M. N. Asim, M. U. Ghani, Z. U. Rehman, S. Rho, and I. Mehmood, "Lexical paraphrasing and pseudo relevance feedback for biomedical document retrieval," *Multimedia Tools and Applications*, vol. 78, no. 21, pp. 29 681–29 712, 2019.

[58] Z. Yang, Z. Hu, R. Salakhutdinov, and T. Berg-Kirkpatrick, "Improved variational autoencoders for text modeling using dilated convolutions," *arXiv preprint arXiv:1702.08139*, 2017.

[59] A. W. Yu, D. Dohan, M.-T. Luong, R. Zhao, K. Chen, M. Norouzi, and Q. V. Le, "Qanet: Combining local convolution with global self-attention for reading comprehension," *arXiv preprint arXiv:1804.09541*, 2018.

[60] S. Zhao, H. Wang, T. Liu, and S. Li, "Pivot approach for extracting paraphrase patterns from bilingual corpora," in *Proceedings of ACL-08: HLT*, 2008, pp. 780–788.

[61] C. Zong, Y. Zhang, K. Yamamoto, M. Sakamoto, and S. Shirai, "Approach to spoken chinese paraphrasing based on feature extraction.," in *NLPRS*, Citeseer, 2001, pp. 551–556.