A Deep Generative Framework for Paraphrase Generation

Ankush Gupta and **Arvind Agarwal**
{ankushgupta,arvagarw}@in.ibm.com
IBM Research Labs
New Delhi, India

Prawaan Singh and **Piyush Rai**
prawaan@iitk.ac.in and piyush@cse.iitk.ac.in
Indian Institute of Technology,
Kanpur, India

Abstract

Paraphrase generation is an important problem in NLP, especially in question answering, information retrieval, information extraction, conversation systems, to name a few. In this paper, we address the problem of generating paraphrases automatically. Our proposed method is based on a combination of deep generative models (VAE) with sequence-to-sequence models (LSTM) to generate paraphrases, given an input sentence. Traditional VAEs when combined with recurrent neural networks can generate free text but they are not suitable for paraphrase generation for a given sentence. We address this problem by conditioning the both, encoder and decoder sides of VAE, on the original sentence, so that it can generate the given sentence's paraphrases. Unlike most existing models, our model is simple, modular and can generate multiple paraphrases, for a given sentence. Quantitative evaluation of the proposed method on a benchmark paraphrase dataset demonstrates its efficacy, and its performance improvement over the state-of-the-art methods by a significant margin, whereas qualitative human evaluation indicate that the generated paraphrases are well-formed, grammatically correct, and are relevant to the input sentence. Furthermore, we evaluate our method on a newly released question paraphrase dataset, and establish a new baseline for future research.

Introduction

Paraphrase generation is an important problem in many NLP applications such as question answering, information retrieval, information extraction, and summarization. QA systems are often susceptible to the way questions are asked; in fact, for knowledge-based (KB) QA systems, question paraphrasing is crucial for bridging the gap between questions asked by users and knowledge based assertions (Fader, Zettlemoyer, and Etzioni 2014; Yin et al. 2015). Similarly paraphrasing finds applications in information retrieval by generating query variants, and in machine translation or summarization by generating variants for automatic evaluation. In addition to being directly useful in QA systems, paraphrase generation is also important for generating training data for various learning tasks, such as question type classification, paraphrase detection, etc., that are useful in other applications. Question type classification has application in conversation systems, while paraphrase detection is an important problem for translation, summarization, social QA (finding closest question to FAQs/already asked question) (Figueroa and Neumann 2013). Due to the nature and complexity of the task, all of these problems suffer from lack of training data, a problem that can readily benefit from the paraphrase generation task.

Despite the importance of the paraphrase generation problem, there has been relatively little prior work in the literature, though much larger amount of work exists on paraphrase detection problem. Traditionally, paraphrase generation has been addressed using rule-based approaches (McKeown 1983a; Zhao et al. 2009), primarily due to the inherent difficulty of the underlying natural language generation problem. However, recent advances in deep learning, in particular generative models (Bowman et al. 2015; Chung et al. 2015), have led to powerful, data-driven approaches to text generation.

In this paper, we present a deep generative framework for automatically generating paraphrases, given a sentence. Our framework combines the power of sequence-to-sequence models, specifically the long short-term memory (LSTM) (Hochreiter and Schmidhuber 1997), and deep generative models, specifically the variational autoencoder (VAE) (Kingma and Welling 2013; Rezende, Mohamed, and Wierstra 2014), to develop a novel, end-to-end deep learning architecture for the task of paraphrase generation.

In contrast to the recent usage of VAE for sentence generation (Bowman et al. 2015), a key differentiating aspect of our proposed VAE based architecture is that it needs to generate paraphrases, *given* an original sentence as input. That is, the generated paraphrased version of the sentence should capture the essence of the original sentence. Therefore, unconditional sentence generation models, such as (Bowman et al. 2015), are not suited for this task. To address this limitation, we present a mechanism to *condition* our VAE model on the original sentence for which we wish to generate the paraphrases. In the past, conditional generative models (Sohn, Lee, and Yan 2015; Kingma et al. 2014) have been applied in computer vision to generate images conditioned on the given class label. Unlike these methods where number of classes are finite, and do not require any intermediate representation, our method conditions both the sides (i.e. encoder and decoder) of VAE on the intermediate representation of the input question obtained through LSTM.

One potential approach to solve the paraphrase generation problem could be to use existing sequence-to-sequence models (Sutskever, Vinyals, and Le 2014), in fact, one variation of sequence-to-sequence model using stacked residual LSTM (Prakash et al. 2016) is the current state of the art for this task. However, most of the existing models for this task including stacked residual LSTM, despite having sophisticated model architectures, lack a principled generative framework. In contrast, our deep generative model enjoys a simple, modular architecture, and can generate not just a single but *multiple*, semantically sensible, paraphrases for any given sentence.

It is worth noting that existing models such as sequence-to-sequence models, when applied using beam search, are not able to produce multiple paraphrases in a principled way. Although one can choose top k variations from the ranked results returned by beam-search, k_{th} variation will be qualitatively worse (by the nature of beam-search) than the first variation. This is in contrast to the proposed method where all variations will be of relatively better quality since they are the *top* beam-search result, generated based on different z sampled from a latent space. We compare our framework with various sophisticated sequence-to-sequence models including the state-of-the-art stacked residual model (Prakash et al. 2016) for paraphrase generation, and show its efficacy on benchmark datasets, on which it outperforms the state-of-the-art by significant margins. Due to the importance of the paraphrase generation task in QA system, we perform a comprehensive evaluation of our proposed model on the recently released Quora questions dataset[1], and demonstrates its effectiveness for the task of question paraphrase generation through both quantitative metrics, as well as qualitative analysis. Human evaluation indicate that the paraphrases generated by our system are well-formed, and grammatically correct for the most part, and are able to capture new concepts related to the input sentence.

Methodology

Our framework uses a variational autoencoder (VAE) as a generative model for paraphrase generation. In contrast to the standard VAE, however, we additionally condition the encoder and decoder modules of the VAE on the *original* sentence. This enables us to generate paraphrase(s) specific to an input sentence at test time. In this section, we first provide a brief overview of VAE, and then describe our framework.

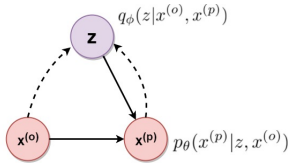

Figure 1: A macro-view of our model: the paraphrase generation model is also conditioned on the original sentence

[1]https://data.quora.com/First-Quora-Dataset-Release-Question-Pairs

Variational Autoencoder (VAE)

The VAE (Kingma and Welling 2014; Rezende, Mohamed, and Wierstra 2014) is a deep generative latent variable model that allows learning rich, nonlinear representations for high-dimensional inputs. The VAE does so by learning a latent representation or "code" $\boldsymbol{z} \in \mathbb{R}^K$ for an input $\boldsymbol{x} \in \mathbb{R}^D$ such that the original input $\boldsymbol{x}$ can be well reconstructed from the latent code $\boldsymbol{z}$. In contrast to the standard autoencoder (Goodfellow, Bengio, and Courville 2016) which learns, for any input $\boldsymbol{x}$, a *deterministic* latent code $\boldsymbol{z}$ via a deterministic encoder function q_ϕ, the VAE encoder is actually a *posterior distribution* $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ (also known as the *recognition model*) over the latent code $\boldsymbol{z}$. The posterior $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ is usually assumed to be a Gaussian distribution $\mathcal{N}(\mu(\boldsymbol{x}), \text{diag}(\sigma^2(\boldsymbol{x})))$, and the parameters $\phi = \{\mu(\boldsymbol{x}), \sigma^2(\boldsymbol{x})\}$ are nonlinear transformations of the input $\boldsymbol{x}$ and are the outputs of feedforward neural networks that take $\boldsymbol{x}$ as input. The VAE also encourages its posterior distribution $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ to be close to the prior $p(\boldsymbol{z})$, which is typically taken as a standard normal distribution $\mathcal{N}(0, \mathbf{I})$.

The VAE also consists of a *decoder* model, which is another distribution $p_\theta(\boldsymbol{x}|\boldsymbol{z})$ that takes as input a *random* latent code $\boldsymbol{z}$ and produces an observation $\boldsymbol{x}$. The parameters of the decoder distribution θ are defined by the outputs of another feedforward neural network, akin to the VAE decoder model. The parameters defining the VAE are learned by maximizing the following objective:

$$\mathcal{L}(\theta, \phi; \boldsymbol{x}) = \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})] - KL(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z}))$$

which provides a lower bound on the model evidence $p(\boldsymbol{x}|\theta, \phi)$, Here KL stands for the KL divergence.

Endowing the latent code $\boldsymbol{z}$ with a distribution "prepares" the VAE decoder for producing realistic looking outputs even when $\boldsymbol{z}$ is a *random* latent code not representing the encoding of any of the previously seen inputs. This makes VAE very attractive for generative models for complex data, such as images and text data such as sentences.

In particular, (Bowman et al. 2015) presented a text-generation model in which the encoder and decoder were modeled by long short-term memory (LSTM) networks. Moreover, training tricks such as KL-term annealing and dropout of inputs of the decoder were employed to circumvent the problems encountered when using the standard VAE for the task of modeling text data. Our work is in a similar vein but the key difference lies in the design of a novel VAE-LSTM architecture, specifically customized for the paraphrase generation task, where the training examples are given in form of pairs of sentences (original sentence and its paraphrased version), and both encoder and decoder of the VAE-LSTM are conditioned on the original sentence. We describe our VAE-LSTM architecture in more detail in the next section.

Model Architecture

Our training data is provided in form of N pairs $\{\boldsymbol{s}_n^{(o)}, \boldsymbol{s}_n^{(p)}\}_{n=1}^N$, with each pair consisting of the *original* sentence (denoted by superscript o) and its *paraphrase* (denoted by superscript p). For the n^{th} pair, $\boldsymbol{s}_n^{(o)} =$

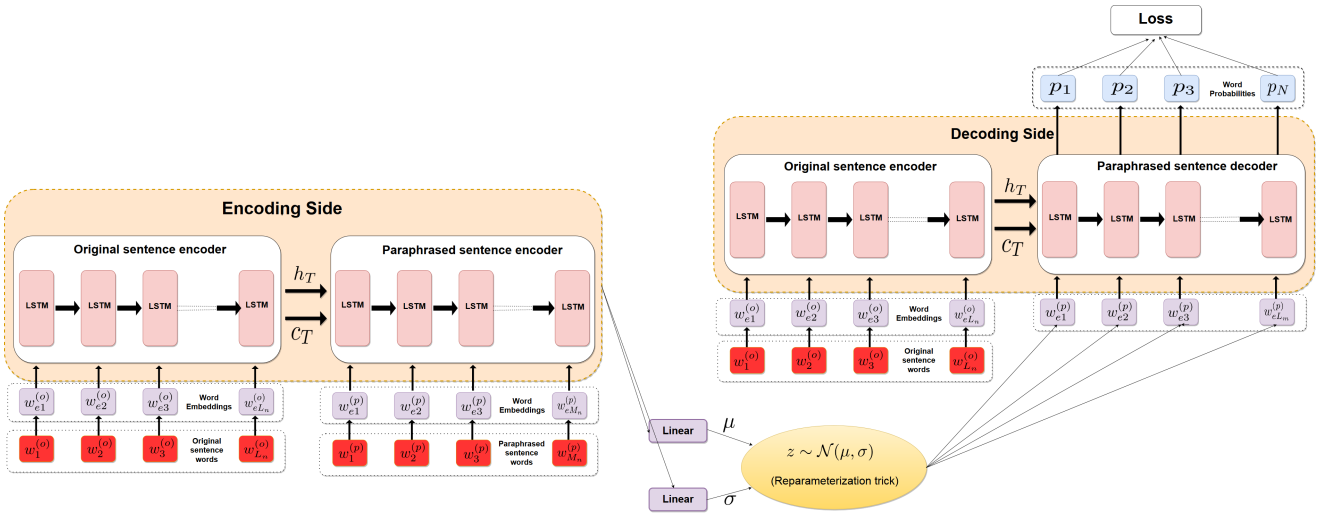

Figure 2: The block diagram of our VAE-LSTM architecture for paraphrase generation

$\{w_1^{(o)}, \ldots, w_{L_n}^{(o)}\}$ and $\boldsymbol{s}_n^{(p)} = \{w_1^{(p)}, \ldots, w_{M_n}^{(p)}\}$ denote the set of L_n words from the *original* sentence and M_n words from its *paraphrase*, respectively. In the following description, we will omit explicitly using the pair index n; e.g., we will denote a pair of original sentence and its paraphrase simply by $\boldsymbol{s}^{(o)}$ and $\boldsymbol{s}^{(p)}$, respectively. We will also use $\boldsymbol{x}^{(o)}$ and $\boldsymbol{x}^{(p)}$ to denote the vector space representations of the original sentence and its paraphrase, respectively. These representations will be learned using LSTM networks, whose parameters will be learned in an end-to-end fashion, with the rest of the model.

Fig. 1 shows a macro view (without the LSTM) of our proposed model architecture, which is essentially a VAE based generative model for each paraphrase's vector representation $\boldsymbol{x}^{(p)}$, which in turn is generated by a latent code $\boldsymbol{z}$ and the original sentence $\boldsymbol{x}^o$. In addition, unlike the standard VAE, note that our VAE decoder model $p_\theta(\boldsymbol{x}^{(p)}|\boldsymbol{z}, \boldsymbol{x}^{(o)})$ is also *conditioned* on the vector representation $\boldsymbol{x}^{(o)}$ of the original sentence. In particular, as Fig. 1 shows, the VAE encoder as well as decoder are conditioned on the original sentence.

A detailed zoomed-in view of our model architecture is shown in Fig. 2, where we show all the components, including the LSTM encoders and decoders. In particular, our model consists of three LSTM encoders and one LSTM decoder (thus a total of four LSTMs), which are employed by our VAE based architecture as follows:

- **VAE Input (Encoder) Side:** As shown in Fig. 2, two of the LSTM encoders are used on the VAE's input side. The first one converts the original sentence $\boldsymbol{s}^{(o)}$ into its vector representation $\boldsymbol{x}^{(o)}$, which is fed, along with the paraphrase version $\boldsymbol{s}^{(p)}$ of this sentence, to the next LSTM encoder. The output of this LSTM encoder ($\boldsymbol{x}^{(p)}$) is passed through a feedforward neural network to produce the mean and variance parameters i.e., ϕ, of the VAE encoder.
- **VAE Output (Decoder) Side:** As shown in Fig. 2, the VAE's output side uses an LSTM decoder which takes as input (1) the latent code $\boldsymbol{z}$, and (2) vector representation $\boldsymbol{x}^{(o)}$ (produced by the third LSTM encoder) of the original sentence. The vector representation $\boldsymbol{x}^o$ is used to initialize the LSTM decoder by feeding it to the first stage of the decoder, in contrast to the latent code $\boldsymbol{z}$ which is fed to each stage of the LSTM decoder (after being concatenated with the output of previous LSTM stage). Thus both $\boldsymbol{z}$ and $\boldsymbol{x}^o$ are used to reconstruct the paraphrased sentence $\boldsymbol{s}^{(p)}$.

Similar to the VAE, the variational lower-bound of the proposed model is given by:

$$\mathcal{L}(\theta, \phi; \boldsymbol{x}^{(p)}, \boldsymbol{x}^{(o)}) = \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x}^{(o)}, \boldsymbol{x}^{(p)})}[\log p_\theta(\boldsymbol{x}^{(p)}|\boldsymbol{z}, \boldsymbol{x}^{(o)})] - KL(q_\phi(\boldsymbol{z}|\boldsymbol{x}^{(o)}, \boldsymbol{x}^{(p)})||p(\boldsymbol{z})) \quad (1)$$

Maximizing the above lower bound trades off the expected reconstruction of the paraphrased sentence's representation $\boldsymbol{x}^{(p)}$ (given $\boldsymbol{x}^{(o)}$), while ensuring that the posterior of $\boldsymbol{z}$ is close to the prior. We train our model following the same training procedure as employed in (Bowman et al. 2015).

Implementation: At the training time, the sentence and its paraphrase are given as input to the encoding side of the network, and encoded through two LSTM encoders chained together in a sequence. The output of the last LSTM cell of the second encoder is sent to a feed-forward network which outputs the parameters μ and σ. Then variable z is sampled from $\mathcal{N}(\mu, \sigma)$. At the decoding side, we first obtain the original sentence's representation through a third LSTM encoder, and then send it along with z to the decoder (RNN with LSTM cell). The decoder, at each time step, outputs the probability distribution over the vocabulary, from which, we can select a word using sampling or through beam search. The above network architecture is trained in an end-to-end fashion though back propagation, and is optimized for the loss function given in Eq. 1. At the test time, we ignore the encoding side, and only use the decoding side to generate a paraphrase given an input sentence, with z sampled from

Table 1: Different models compared in the evaluation study.

Models	Reference
Seq-to-Seq	(Sutskever, Vinyals, and Le 2014)
With Attention	(Bahdanau, Cho, and Bengio 2014)
Bi-directional LSTM	(Graves, Jaitly, and Mohamed 2013)
Residual LSTM	(Prakash et al. 2016)
Unsupervised	Ours but baseline
VAE-S	Ours but baseline
VAE-SVG	Ours
VAE-SVG-eq	Ours

standard normal distribution (prior). Having an input sentence encoder at the decoding side allows us to generate a paraphrase for the given sentence, without which, the model will generate only random sentences.

Related Work

The task of generating paraphrases of a given sentence has been dealt in great depth and in various different types of approaches. Work has been done to use Statistical Machine Translation based models for generating paraphrases. (Quirk, Brockett, and Dolan 2004) apply SMT tools, trained on large volumes of sentence pairs from news articles. (Zhao et al. 2008) proposed a model that uses multiple resources to improve SMT based paraphrasing, paraphrase table and feature function which are then combined in a log-linear SMT model. Some old methods use data-driven methods and hard coded rules such as (Madnani and Dorr 2010), (McKeown 1983b). (Hassan et al. 2007) proposes a system for lexical substitution using thesaurus methods. (Kozlowski, McCoy, and Vijay-Shanker 2003) pairs elementary semantic structures with their syntactic realization and generate paraphrases from predicate/argument structure. As mentioned in (Prakash et al. 2016), the application of deep learning models to paraphrase generation has not been explored rigorously yet. This is one of the first major works that used deep architecture for paraphrase generation and introduce the residual recurrent neural networks.

Finally, our work is also similar in spirit to other generative models for text, e.g. controllable text generation (Hu et al. 2017), which combines VAE and explicit constraints on independent attribute controls. Other prior works on VAE for text generation include (Bowman et al. 2015; Semeniuta, Severyn, and Barth 2017) which used VAEs to model holistic properties of sentences such as style, topic and various other syntactic features.

Experiments

In this section, we describe the datasets, experimental setup, evaluation metrics and the results of our experiments.

Datasets

We evaluate our framework on two datasets, one of which (MSCOCO) is for the task of standard paraphrase generation and the other (Quora) is a newer dataset for the specific problem of question paraphrase generation.

MSCOCO (Lin et al. 2014): This dataset, also used previously to evaluate paraphrase generation methods (Prakash et al. 2016), contains human annotated captions of over 120K images. Each image contains five captions from five different annotators. This dataset is a standard benchmark dataset for image caption generation task. In majority of the cases, annotators describe the most prominent object/action in an image, which makes this dataset suitable for the paraphrase generation task. The dataset has separate division for training and validation. *Train 2014* contains over 82K images and *Val 2014* contains over 40K images. From the five captions accompanying each image, we randomly omit one caption, and use the other four as training instances (by creating two source-reference pairs). We further reduce the long captions to first 15 words in order to be consistent with (Prakash et al. 2016).

Quora: Quora released a new dataset in January 2017. The dataset consists of over 400K lines of potential question duplicate pairs. Each line contains IDs for each question in the pair, the full text for each question, and a binary value that indicates whether the questions in the pair are truly a duplicate of each-other.[2]. Wherever the binary value is 1, the question in the pair are not identical; they are rather paraphrases of each-other. So, for our study, we choose all such question pairs with binary value 1. There are a total of $155K$ such questions. In our experiments, we evaluate our model on 50K, 100K and 150K training dataset sizes. For testing, we use 4K pairs of paraphrases. Some examples of question and their generated paraphrases can be found in Table 3.

Baselines

We consider several state-of-the-art baselines for our experiments. These are described in Table 1. For MSCOCO, we report results from four baselines, with the most important of them being by (Prakash et al. 2016) using residual LSTM. Residual LSTM is also the current state-of-the-art on the MSCOCO dataset, however, it might be worth noting that we did not have their train/test split available hence results might not be comparable because of sampling bias. For the Quora dataset, there were no known baseline results, so we compare our model with (1) standard VAE model i.e., the unsupervised version, and (2) a "supervised" variant VAE-S of the unsupervised model. In the unsupervised version, the VAE generator reconstructs multiple variants of the input sentence using the VAE generative model trained only using the original sentence (without their paraphrases); in VAE-S, the VAE generator *generates* the paraphrase conditioned on the original sentence, just like in the proposed model. This VAE-S model can be thought of as a variation of the proposed model where we remove the encoder LSTM related to the paraphrase sentence from the encoder side. Alternatively, it is akin to a variation of VAE where decoder is made supervised by making it to generate "paraphrases" (instead of the reconstructing original sentence as in VAE) by conditioning the decoder on the input sentence.

[2]https://data.quora.com/First-Quora-Dataset-Release-Question-Pairs

Table 2: Examples paraphrases generated on MSCOCO.

Source	A man with luggage on wheels standing next to a white van .
Reference	A white van is *driving* through a *busy street* .
Generated	A young man standing in front of an airport . A yellow van is parked at the busy street . A white van is in the middle of a park .
Source	A table full of vegetables and fruits piled on top of each other .
Reference	*Group of mixed* vegetables sitting on a *counter top* in a kitchen .
Generated	Several plates of fruits sitting on a table top in a kitchen . Assortment of fruits and vegetables on a wooden table in a kitchen . Several types of fruit sitting on a counter top in a kitchen .

Table 3: Examples paraphrases generated on Quora.

Source	What is my old Gmail account ?
Reference	How can you *find all of your* Gmail accounts ?
Generated	Is there any way to recover my Gmail account ? How can I find my old Gmail account number? How can I get the old Gmail account password ?
Source	Which is the best training institute in Pune for digital marketing and why ?
Reference	Which is the best *digital marketing* training institute in Pune ?
Generated	Which is the best institute for digital training in Pune ? Which is the best digital Tech training institute in Hyderabad ? Which is the best digital marketing training center in Pune ?

Evaluation

Quantitative Evaluation Metrics For quantitative evaluation, we use the well-known automatic evaluation metrics[3] in machine translation domain : BLEU (Papineni et al. 2002), METEOR (Lavie and Agarwal 2007), and Translation Error Rate (TER) (Snover et al. 2006). Previous work has shown that these metrics can perform well for the paraphrase recognition task (Madnani, Tetreault, and Chodorow 2012) and correlate well with human judgments in evaluating generated paraphrases (Wubben, Van Den Bosch, and Krahmer 2010). BLEU considers exact match between reference paraphrase(s) and system generated paraphrase(s) using the concept of modified n-gram precision and brevity penalty. METEOR also uses stemming and synonyms (using WordNet) while calculating the score and is based on a combination of unigram-precision and unigram-recall with the reference paraphrase(s). TER is based on the number of edits (insertions, deletions, substitutions, shifts) required for a human to convert the system output into one of the reference paraphrases.

Qualitative Evaluation Metrics To quantify the aspects that are not addressed by automatic evaluation metrics, human evaluation becomes necessary for our problem. We collect human judgments on 100 random input sentences from both MSCOCO and Quora dataset. Two aspects are verified in human evaluation : *Relevance* of generated paraphrase with the input sentence and *Readability* of generated paraphrase. Six Human evaluators (3 for each dataset) assign a score on a continuous scale of 1-5 for each aspect per generated paraphrase, where 1 is worse and 5 is best.

Experimental Setup

Our framework primarily uses the following experimental setup. These settings are directly borrowed from an existing implementation[4] of the paper (Bowman et al. 2015), and were not fine tuned for any of the datasets. In our setup, we do not use any external word embeddings such as Glove; rather we train these as part of the model-training. The dimension of the embedding vector is set to 300, the dimension of both encoder and decoder is 600, and the latent space dimension is 1100. The number of layers in the encoder is 1 and in decoder 2. Models are trained with stochastic gradient descent with learning rate fixed at a value of 5×10^{-5} with dropout rate of 30%. Batch size is kept at 32. Models are trained for a predefined number of iterations, rather than a fixed number of epochs. In each iteration, we sequentially pick the next batch. A fixed number of iterations makes sure that we do not increase the training time with the amount of data. When the amount of data is increased, we run fewer passes over the data as opposed to the case when there is less data. Number of units in LSTM are set to be the maximum length of the sequence in the training data.

Model Variations In addition to the model proposed in Methodology Section, we also experiment with another variation of this model. In this variation, we make the encoder of original sentence same on both sides i.e. encoder side and the decoder side. We call this model VAE-SVG-eq (SVG stands for sentence variant generation). The motivation for this variation is that having same encoder reduces the number of model parameters, and hopefully helps in learning.

Results

We performed experiments on the above mentioned datasets, and report, both qualitative and quantitative results of our approach. The qualitative results for MSCOCO and Quora datasets are given in Tables 2 and 3 respectively. In these tables, italicized and underlined parts denote interesting phrases which are different in the ground truth and generated variations respectively w.r.t. the input sentence. From both the tables, we see that variations contain many interesting phrases such as *in front of an airport*, *busy street*, *wooden table*, *recover*, *Tech training* etc. which were not encountered in input sentences. Furthermore, the paraphrases generated by our system are well-formed, semantically sensible, and grammatically correct for the most part. For example, for the MSCOCO dataset, for the input sentence *A man with luggage on wheels standing next to a white van.*, one of the variants *A young man standing in front of an airport.* is able to figure out that the situation pertains to "waiting in front of an airport", probably from the phrases *standing* and *luggage on wheels*. Similarly, for the Quora dataset, for the question *What is my old Gmail account?*, one of the variants is *Is there any way to recover my Gmail account?* which

[3] We used the software available at https://github.com/jhclark/multeval

[4] https://github.com/kefirski/pytorch_RVAE

Table 4: Results on MSCOCO dataset. Higher BLEU and METEOR score is better, whereas lower TER score is better. "Measure" column denotes the way metrics are computed over multiple paraphrases.

		MSCOCO				
Model	Measure	Beam size	Layers	BLEU	METEOR	TER
Seq-to-Seq (Sutskever, Vinyals, and Le 2014)	-	10	2	16.5	15.4	67.1
With Attention (Bahdanau, Cho, and Bengio 2014)	-	10	2	18.6	16.8	63.0
Seq-toSeq (Sutskever, Vinyals, and Le 2014)	-	10	4	28.9	23.2	56.3
Bi-directional (Graves, Jaitly, and Mohamed 2013)	-	10	4	32.8	24.9	53.7
With Attention (Bahdanau, Cho, and Bengio 2014)	-	10	4	33.4	25.2	53.8
Residual LSTM (Prakash et al. 2016)	-	10	4	37.0	27.0	51.6
Unsupervised	-	No beam	1, 2	12.8	17.5	78.8
VAE-S	Avg	No beam	1, 2	7.0	14.0	82.3
	best BLEU	No beam	1, 2	11.3	16.8	76.5
	best METEOR	No beam	1, 2	11.0	17.7	78.8
VAE-SVG (our)	Avg	No beam	1, 2	39.2	29.2	43.6
	best BLEU	No beam	1, 2	41.1	30.3	42.3
	best METEOR	No beam	1, 2	**41.7**	30.8	41.7
	Avg	10	1, 2	41.3	30.9	**40.8**
	best BLEU	10	1, 2	40.9	30.7	42.0
	best METEOR	10	1, 2	41.3	**31.0**	41.6
VAE-SVG-eq (our)	Avg	No beam	1, 2	37.3	28.5	45.1
	best BLEU	No beam	1, 2	39.2	29.5	43.9
	best METEOR	No beam	1, 2	39.8	30.0	43.4
	Avg	10	1, 2	39.6	30.2	42.3
	best BLEU	10	1, 2	39.3	30.1	43.5
	best METEOR	10	1, 2	39.7	30.4	43.2

is very similar –but not the same– to the paraphrase available in the ground truth. It is further able to figure out that the input sentence is talking about *recovering* the account. Another variant *How can I get the old Gmail account password?* tells us that accounts are related to the password, and recovering the account might mean recovering the password as well.

In Tables 4 and 5, we report the quantitative results from various models for the MSCOCO and Quora datasets respectively. Since our models generate multiple variants of the input sentence, one can compute multiple metrics with respect to each of the variants. In our tables, we report average and best of these metrics. For average, we compute the metric between each of the generated variants and the *ground truth*, and then take the average. For computing the best variant, while one can use the same strategy, that is, compute the metric between each of the generated variants and the *ground truth*, and instead of taking average find the best value but that would be unfair. Note that in this case, we would be using the ground truth to compute the best which is not available at test time. Since we cannot use the ground truth to find the best value, we instead use the metric between the *input sentence* and the variant to get the best variant, and then report the metric between the best variant and the *ground truth*. Those numbers are reported in the *Measure* column with row best-BLEU/best-METEOR.

In Table 4, we report the results for MSCOCO dataset. For this dataset, we compare the results of our approach with existing approaches. As we can see, we have a significant improvement w.r.t. the baselines. Both variations of our supervised model i.e., VAE-SVG and VAE-SVG-eq perform better than the state-of-the-art with VAE-SVG performing

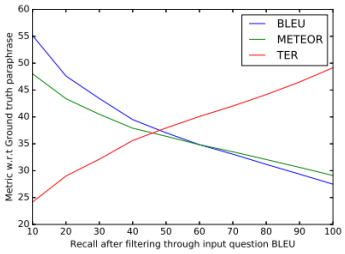

Figure 3: Recall vs BLEU/METEOR/TER after filtering the results through $x^{(o)}$-BLEU for Quora dataset

slightly better than VAE-SVG-eq. We also evaluate with respect to best variant. The best variant is computed using different metrics, that is, BLEU and METEOR, however the best variant is not always guaranteed to perform better than average since best variant is computed with respect to the input question not based on the ground truth. When using the best variant, we get improvement in all three metrics in the case of non-beam search, however when experimented with generating paraphrases through beam-search, we get further improvement for METEOR and TER however these improvement are not as significant as for the Quora dataset, as you will see below. This could be because MSCOCO is an image captioning dataset which means that dataset does not contain fully formed grammatical sentences, as one can see from the examples in Table 2. In such cases, beam search is not able to capture the structure of the sentence construction. When comparing our results with the state-of-the-art base-

Table 5: Results on Quora dataset. Higher BLEU and METEOR score is better, whereas lower TER score is better.

		Quora								
		50K			100K			150K		
Model	Measure	BLEU	METEOR	TER	BLEU	METEOR	TER	BLEU	METEOR	TER
Unsupervised (baseline)	-	8.3	12.2	83.7	10.6	14.3	79.9	11.4	14.5	78.0
VAE-S (baseline)	Avg	11.9	17.4	77.7	13.0	18.4	76.8	14.2	19.0	74.8
	best BLEU	15.8	20.1	69.4	17.5	21.6	67.1	19.8	22.6	63.9
	best METEOR	15.6	21.1	71.5	17.5	22.7	69.5	19.7	23.8	66.9
VAE-SVG (ours)	Avg	13.8	18.7	68.2	18.6	21.9	60.6	25.0	25.1	52.5
	best BLEU	17.1	21.3	63.1	22.5	24.6	55.7	30.3	28.5	47.3
	best METEOR	17.1	**22.2**	63.8	22.4	**25.5**	55.6	30.3	29.2	47.1
VAE-SVG-eq (ours)	Avg	13.9	18.8	67.1	19.0	21.7	60.0	26.2	25.7	52.1
	best BLEU	**17.4**	21.4	**61.9**	**22.9**	24.7	55.0	31.4	29.0	46.8
	best METEOR	17.3	**22.2**	62.6	**22.9**	**25.5**	**54.9**	32.0	30.0	46.1
	Avg (beam=10)	-	-	-	-	-	-	37.1	32.0	40.8
	best BLEU (beam=10)	-	-	-	-	-	-	38.0	32.9	40.0
	best METEOR (beam=10)	-	-	-	-	-	-	**38.3**	**33.6**	**39.5**

line, the average metric of the VAE-SVG model is able to give a 10% absolute point performance improvement for the TER metric, a significant number with respect to the difference between the best and second best baseline which only stands at 2% absolute point. For the BLEU and METEOR, our best results are 4.7% and 4% absolute point improvement over the state-of-the-art.

In Table 5, we report results for the Quora dataset. As we can see, both variations of our model perform significantly better than unsupervised VAE and VAE-S, which is not surprising. We also report the results on different training sizes, and as expected, as we increase the training data size, results improve. Comparing the results across different variants of supervised model, VAE-SVG-eq performs the best. This is primarily due to the fact that in VAE-SVG-eq, the parameters of the input question encoder are shared by the encoding side and the decoding side. We also experimented with generating paraphrases through beam-search, and, unlike MSCOCO, it turns out that beam search improves the results significantly. This is primarily because beam-search is able to filter out the paraphrases which had only few common terms with the input question. When comparing the best variant of our model with unsupervised model (VAE), we are able to get more than 27% absolute point (more than 3 times) boost in BLEU score, and more than 19% absolute point (more than 2 times) boost in METEOR; and when comparing with VAE-S, we are able to get a boost of almost 19% absolute points in BLEU (2 times) and more than 10% absolute points in METEOR (1.5 times).

Table 6: Human evaluation results for paraphrase generation.

Dataset	**Input**	**Relevance**	**Readability**
MSCOCO	Ground Truth	3.38	4.68
	System Output	3.0	3.84
Quora	Ground Truth	4.82	4.94
	System Output	3.57	4.08

The results of the **qualitative human evaluation** are shown in Table 6. From the Table, we see that our method produces results which are close to the ground truth for both metrics *Readability* and *Relevance*. Note that *Relevance* of the MSCOCO dataset is 3.38 which is far from a perfect score of 5 because unlike Quora, MSCOCO dataset is an image caption dataset, and therefore allows for a larger variation in the human annotations.

Note that one can use the metric between the variant and the input question to provide filtering in the case of multiple variants, or even to decide if a variant needs to be reported or not. So in order to make the system more practical (a high precision system), we choose to report the variant only when the confidence in the variant is more than a threshold. We use the metric between *input question* and the variant to compute this confidence. Naturally this thresholding reduces the recall of the system. In Figure 3, we plot the recall for Quora dataset, after thresholding the confidence (computed using the BLEU between the variant and the input question), and the average metrics for those candidates that pass the threshold. Interestingly, we can increase the BLEU score of the system as much as up to 55% at the recall of 10%. Plots generated using other metrics such as METEOR and TER showed a similar trend.

Conclusion

In this paper we have proposed a deep generative framework, in particular, a Variational Autoencoder based architecture, augmented with sequence-to-sequence models, for generating paraphrases. Unlike traditional VAE and unconditional sentence generation model, our model conditions the encoder and decoder sides of the VAE on the input sentence, and therefore can generate *multiple* paraphrases for a given sentence in a principled way. We evaluate the proposed method on a general paraphrase generation dataset, and show that it outperforms the state-of-the-art by a significant margin, without any hyper-parameter tuning. We also evaluate our approach on a recently released question paraphrase dataset, and demonstrate its remarkable performance. The generated paraphrases are not just semantically similar to the original input sentence, but also able to capture new concepts related to the original sentence.

References

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

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

Chung, J.; Kastner, K.; Dinh, L.; Goel, K.; Courville, A. C.; and Bengio, Y. 2015. A recurrent latent variable model for sequential data. In *Advances in neural information processing systems*, 2980–2988.

Fader, A.; Zettlemoyer, L.; and Etzioni, O. 2014. Open question answering over curated and extracted knowledge bases. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 1156–1165. ACM.

Figueroa, A., and Neumann, G. 2013. Learning to rank effective paraphrases from query logs for community question answering. In *AAAI*, volume 13, 1099–1105.

Goodfellow, I.; Bengio, Y.; and Courville, A. 2016. *Deep learning*. The MIT Press.

Graves, A.; Jaitly, N.; and Mohamed, A.-r. 2013. Hybrid speech recognition with deep bidirectional lstm. In *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*, 273–278. IEEE.

Hassan, S.; Csomai, A.; Banea, C.; Sinha, R.; and Mihalcea, R. 2007. Unt: Subfinder: Combining knowledge sources for automatic lexical substitution. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, 410–413. Association for Computational Linguistics.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Hu, Z.; Yang, Z.; Liang, X.; Salakhutdinov, R.; and Xing, E. P. 2017. Controllable text generation. *arXiv preprint arXiv:1703.00955*.

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

Kingma, D. P., and Welling, M. 2014. Auto-encoding variational bayes. In *ICLR*.

Kingma, D. P.; Mohamed, S.; Rezende, D. J.; and Welling, M. 2014. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems*, 3581–3589.

Kozlowski, R.; McCoy, K. F.; and Vijay-Shanker, K. 2003. Generation of single-sentence paraphrases from predicate/argument structure using lexico-grammatical resources. In *Proceedings of the second international workshop on Paraphrasing-Volume 16*, 1–8. Association for Computational Linguistics.

Lavie, A., and Agarwal, A. 2007. Meteor: An automatic metric for mt evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*, 228–231. Association for Computational Linguistics.

Lin, T.-Y.; Maire, M.; Belongie, S.; Bourdev, L.; Girshick, R.; Hays, J.; Perona, P.; Ramanan, D.; Zitnick, C. L.; and Dollar, P. 2014. Microsoft coco: Common objects in context. *arXiv preprint arXiv:1405.0312*.

Madnani, N., and Dorr, B. J. 2010. Generating phrasal and sentential paraphrases: A survey of data-driven methods. *Computational Linguistics* 36(3):341–387.

Madnani, N.; Tetreault, J.; and Chodorow, M. 2012. Re-examining machine translation metrics for paraphrase identification. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 182–190. Association for Computational Linguistics.

McKeown, K. R. 1983a. Paraphrasing questions using given and new information. *Comput. Linguist.* 9(1):1–10.

McKeown, K. R. 1983b. Paraphrasing questions using given and new information. *Computational Linguistics* 9(1):1–10.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, 311–318. Association for Computational Linguistics.

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

Quirk, C.; Brockett, C.; and Dolan, W. B. 2004. Monolingual machine translation for paraphrase generation. In *EMNLP*, 142–149.

Rezende, D. J.; Mohamed, S.; and Wierstra, D. 2014. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*.

Semeniuta, S.; Severyn, A.; and Barth, E. 2017. A hybrid convolutional variational autoencoder for text generation. *arXiv preprint arXiv:1702.02390*.

Snover, M.; Dorr, B.; Schwartz, R.; Micciulla, L.; and Makhoul, J. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of association for machine translation in the Americas*, volume 200.

Sohn, K.; Lee, H.; and Yan, X. 2015. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems*, 3483–3491.

Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, 3104–3112.

Wubben, S.; Van Den Bosch, A.; and Krahmer, E. 2010. Paraphrase generation as monolingual translation: Data and evaluation. In *Proceedings of the 6th International Natural Language Generation Conference*, 203–207. Association for Computational Linguistics.

Yin, P.; Duan, N.; Kao, B.; Bao, J.; and Zhou, M. 2015. Answering questions with complex semantic constraints on open knowledge bases. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, 1301–1310. ACM.

Zhao, S.; Niu, C.; Zhou, M.; Liu, T.; and Li, S. 2008. Combining multiple resources to improve smt-based paraphrasing model. In *ACL*, 1021–1029.

Zhao, S.; Lan, X.; Liu, T.; and Li, S. 2009. Application-driven statistical paraphrase generation. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, 834–842. Association for Computational Linguistics.