

Master's Thesis

IMPROVING ABSTRACTIVE SUMMARIZATION  
BY UNDERSTANDING HIDDEN REPRESENTATIONS AND  
GUIDANCE ON SEMANTIC MEANING

Nguyen Thanh Nguyen

Department of Computer Science and Engineering

Ulsan National Institute of Science and Technology

2021

IMPROVING ABSTRACTIVE SUMMARIZATION  
BY UNDERSTANDING HIDDEN REPRESENTATIONS AND  
GUIDANCE ON SEMANTIC MEANING

Nguyen Thanh Nguyen

Department of Computer Science and Engineering

Ulsan National Institute of Science and Technology

# Improving Abstractive Summarization by Understanding Hidden Representations and Guidance on Semantic Meaning

A thesis/dissertation submitted to  
Ulsan National Institute of Science and Technology  
in partial fulfillment of the  
requirements for the degree of  
Master of Science

NGUYEN THANH NGUYEN

11.23.2020

Approved by

\_\_\_\_\_  
Advisor

Kwang-In Kim

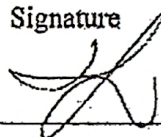
Improving Abstractive Summarization by  
Understanding Hidden Representations and  
Guidance on Semantic Meaning

Nguyen Thanh Nguyen

This certifies that the thesis/dissertation of Nguyen Thanh Nguyen is approved.

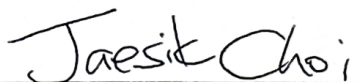
23.11.2020

Signature



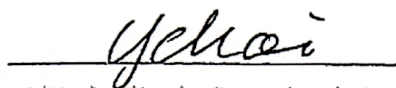
Advisor: Kwang-In Kim

Signature



Jaesik Choi: Thesis Committee Member #1

Signature



Young-ri Choi : Thesis Committee Member #2

# ABSTRACT

Abstractive Summarization is a challenging task in Text Summarization field. Most work on summarization task focus on minimizing negative log-likelihood objective function. This leads to difficulty in generating highly abstractive summaries if the dataset does not have good reference summaries. In this paper, we propose a model that follows a three-step workflow of understanding and generating abstractive summarization, which includes understanding hidden representation of the data samples by using Conditional Variational Autoencoder model; giving guidance and rewards for the model going in right direction; and using top- $p$  random sampling to generate highly abstractive summaries based on the learned features and hidden latent space. We show that having deeper understanding about the structure can obtain higher abstractedness while keeping the same semantic understanding of the document.

# TABLE OF CONTENTS

<b>1 INTRODUCTION</b> .....	<b>1</b>
<b>2 BACKGROUND</b> .....	<b>3</b>
2.1 TRANSFORMER-BASED MODEL.....	3
2.1.1 <i>Development of Seq2Seq models and attention mechanisms</i> .....	3
2.1.2 <i>Transformer Architecture</i> .....	4
2.2 CONDITIONAL VARIATIONAL AUTOENCODER.....	6
2.2.1 <i>Variational Autoencoder</i> .....	6
2.2.2 <i>Conditional Variational Autoencoder</i> .....	7
2.3 SAMPLING TECHNIQUES .....	8
2.4 METRICS IN NLP .....	8
<b>3 RELATED WORK</b> .....	<b>10</b>
<b>4 CONDITIONAL VARIATIONAL AUTOENCODER WITH REWARD AS GUIDANCE FOR TEXT SUMMARIZATION</b> .....	<b>11</b>
4.1. LEARNING INNER REPRESENTATIONS OF DOCUMENTS .....	11
4.2. GUIDANCE MODEL BY GIVING REWARDS .....	14
4.3 GENERATING SUMMARIES BY SAMPLING.....	16
<b>5 EXPERIMENTAL RESULTS</b> .....	<b>18</b>
5.1 EXPERIMENTAL SETUP .....	18
<b>6 CONCLUSION</b> .....	<b>22</b>
<b>BIBLIOGRAPHY</b> .....	<b>23</b>

## LIST OF FIGURES

2.1	Transformer architecture in Text Summarization.....	4
2.2.	Variational Autoencoder in text reconstruction.....	7
2.3.	Example of Variational Autoencoder in text reconstruction.....	7
2.4.	Conditional Variational Autoencoder graph for ELBO objective [Zhang et al, 2016].....	8
4.1.	Our Transformer-CVAE graph.....	12
4.2.	Detail structure of CVAE-Transformer for Text Summarization.....	13
4.3.	Reward system for CVAE-Transformer.....	15

## **LIST OF TABLES**

5.1.	ROUGE scores comparisons.....	18
5.2.	STS-B scores comparison.....	19
5.3.	RTE scores comparison.....	19
5.4.	Novel n-gram scores comparison.....	19
5.5.	Summarization result examples.....	21



## LIST OF ABBREVIATIONS

<b>BART</b>	Bidirectional and Auto-Regressive Transformers model
<b>CNN</b>	Convolutional Neural Network
<b>CVAE</b>	Conditional Variational Autoencoder
$\mathcal{D}_{KL}$	Kullback-Leibler Divergence
<i>GloVe</i>	Global Vectors for Word Representation.
<b>KL</b>	Kullback-Leibler
<b>LSTM</b>	Long Short-Term Memory
<b>MLP</b>	Multilayer perceptron
<b>NLP</b>	Natural Language Processing
<b>RL</b>	Reinforcement Learning
<b>RNN</b>	Recurrent Neural Network
<b>RoBERTa</b>	Robustly optimized BERT approach model
<b>ROUGE</b>	Recall-Oriented Understudy for Gisting Evaluation
<b>RTE</b>	Recognizing Textual Entailment
<i>seq2seq</i>	Sequence-to-sequence
<b>STS</b>	Semantic Text Similarity
<b>VAE</b>	Variational Autoencoder

# CHAPTER 1

## INTRODUCTION

Text summarization is a task targeting at shortening long documents into concise, coherent and fluent versions while maintaining important information concurrently. Automated text summarization refers to using machines to learn and execute the task systematically and is common among a diverse family of Natural Language Processing (NLP) tasks.

There are two fundamental approaches for Text Summarization: extractive and abstractive summarization. Extractive summarization is the task of extracting crucial sentences and major phrases to combine them into a summary by using statistic, graphical model based or neural network approaches. Abstractive summarization is the task of producing novel words, paraphrasing, and shrinking the original document texts.

In this work, we focus on abstractive summarization, which needs more complex analysis and more advanced natural language algorithm. Most of recent Text Summarization tasks are based on deep learning or neural network and the structure is *seq2seq* model with encoder-decoder framework, in which encoder is used for building representation for words, such as word embeddings like *word2vec* or *GloVe*, and processing those representations to capture most information from the original document as possible. Decoder, meanwhile, receives encoder's representation and subsequently develop it into a probability distribution over the vocabulary. From the distribution, we can generate most proper words that can summarize the input document with the negative log-likelihood objective function.

Deep learning approaches range from Recurrent Neural Network (RNN), Long Short-term Memory (LSTM) to Transformer-based recently. Most work of abstractive summarization task only focus on cross-entropy loss with reference summary while training. Therefore, other techniques from Conditional Variation Autoencoder (CVAE) and Reinforcement Learning (RL) are also used with new losses for improving other aspects of the task. Recent achievements on Transformer are gaining more attention with better performance than RNN or LSTM due to better ability of capturing documents' underlying structures.

Despite better achievements of supervised summarization task with deep neural network models, recent models still lean towards heavily extractive if the dataset does not have well-abstractive-formed reference

summaries. The goal of this paper is experimenting a three-step workflow of understanding and generating better abstractive text summary includes:

- 1) *Understand and learn inner document latent vector spaces,*
- 2) *Understand the goal of the task with rewards and punishments system,*
- 3) *Generate highly abstractive summary while preserving semantic similarity in meaning.*

Consequently, we propose a new model that combines CVAE and reward method from RL using a pretrained baseline model that follows the above workflow to improve abstractedness while preserving semantic similarity score comparing to the baseline.

This paper includes following chapters. Chapter 2 revisits some background concepts of Transformer model, Conditional Variational Autoencoder algorithm and reward system in Reinforcement Learning. Chapter 3 mentions about related work in Text Summarization. Chapter 4 proposes our model that follows the same three-step workflow for better understanding of Text Summarization task. Chapter 5 describes the experiments' settings, dataset and results. Finally, chapter 6 wraps up with the conclusion of the thesis.

# CHAPTER 2

## BACKGROUND

### 2.1 Transformer-based Model

#### 2.1.1 Development of Seq2Seq models and attention mechanisms

Sequence-to-sequence (*seq2seq*) models in NLP are the models that can transform sequences of a specific type to another type. For instance, translation of English sentences to Spanish sentences is a sequence-to-sequence task.

Recent work on *seq2seq* mostly concentrate on neural network methods which process sentences by creating words vector space as representation. The model then can measure information from those representations of words, phrases or sentences and discover the context from the aggregated information.

After being introduced in 2014, RNN-based *seq2seq* models have gathered a lot of attention for its ability of processing sequential data, numbers, texts, video frames or even audios, for example. Considering each computational time step  $t$  as a position state for generation, RNN can generate a sequence of hidden states  $h_t$  as a function  $f$  of the previous hidden state  $h_{t-1}$  and input data for position  $t$ :

$$h_{t+1} = f(x_t, h_t) \tag{1}$$

From the hidden states, they can generate the output at each time step

$$o_t = b + W_t h_t, \tag{2}$$

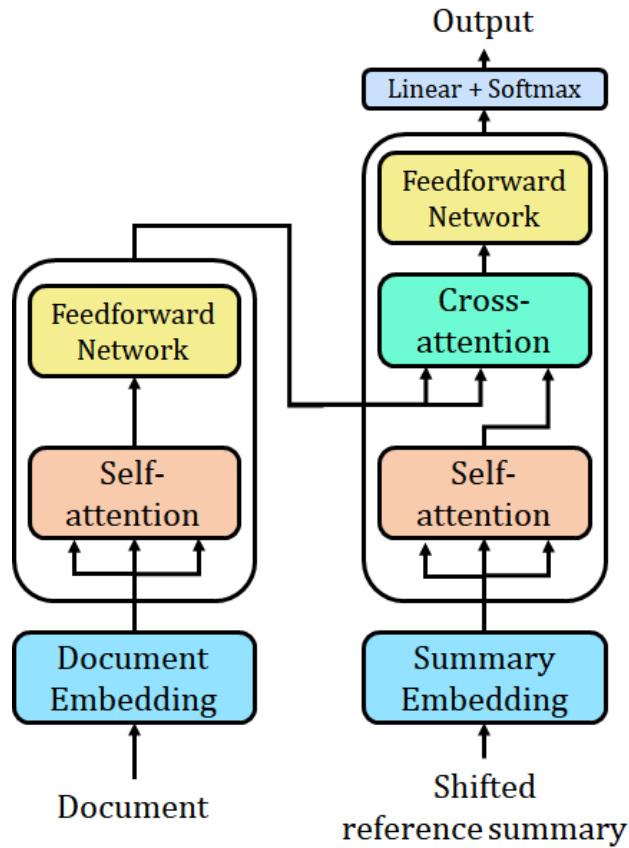
with weights  $W_t$  and bias  $b$  as learnable parameters. The performance of these *seq2seq* RNN-based models was further improved with the addition of the Attention Mechanism [Luong et al, 2015] and with the evolution to LSTM models.

RNN is a good model for sequences, but there are unavoidable limitations of *seq2seq* models when it has difficulties of handling long-range dependencies in sequence, and parallelizing when analyzing long documents becomes challenging due to its nature structure.

### 2.1.2 Transformer Architecture

In 2017, *Transformers model* [Vaswani et al, 2017] is proposed to solve the issues of parallelizing tokens in documents. Despite a combination between Convolutional Neural Networks (CNN) and attention mechanism, Transformer still has the same encoder-decoder architecture as other sequence neural language models. Principally, Transformer encoder transforms a sequence of input data  $\mathbf{x} = (x_1, \dots, x_n)$  into a multi-dimension continuous representation  $\mathbf{c} = (c_1, \dots, c_n)$ . The decoder then uses  $\mathbf{c}$  as its input to generate output sequence  $\mathbf{y} = (y_1, \dots, y_m)$  by using auto-regressive method, which is using previous generated output as the next input for next time step.

Transformer encoder architecture comprises of multi-blocks, and each block basically includes two layers: self-attention and a feed forward neural network. Decoder has the same structure, except it has another layer of cross-attention between the two previous layers, which helps it to address attention to the input, not only the previous words itself.



**Figure 2.1.** Transformer architecture in Text Summarization

### Self-attention mechanism

For NLP tasks, we first transform all input words into a vector space through some embedding algorithms. As in other *seq2seq* models like RNN, it is important for the model to have a mechanism for attention to all of the words and sentences in a long document after receiving the embedding space. The self-attention architecture in each block of Transformer helps the model get better attention than earlier work. After passing sentences into Transformer and receiving tokens embeddings, we will calculate the weights concurrently. Each token weight is computed by a combination of other relevant ones.

Particularly, given token  $i$  and word embedding  $x_i$ , we can get the key, query and value vectors  $k_i, q_i, v_i$  by multiplying them to learnable weights  $W_K, W_Q, W_V$  respectively

$$k_i = x_i W_K, \quad q_i = x_i W_Q, \quad v_i = x_i W_V \quad (3)$$

The attention weights are calculated by distribution of dot product between keys and queries  $q_i \cdot k_i$ , then the distribution weights will pass to the values so we can get new values for the documents. Writing in matrix form, the formula will be

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (4)$$

with  $d_k$  is key dimension, using for stabilizing gradients while training. The dot product between queries and keys can also generate attention matrix, with a meaning of where each word looking at in the current context.

### Cross-attention mechanism

Like encoder, Transformer decoder uses the same self-attention structure, but with an addition of another layer called cross-attention or co-attention between encoded input and current decoded output. The formula is the same as self-attention, but we use key from encoder and query, value from decoder. The point of this layer is to make decoder having a proper attention to the input data, or the connection between encoder and decoder is represented by this layer.

## 2.2 Conditional Variational Autoencoder

### 2.2.1 Variational Autoencoder

*Autoencoder* is a neural network that intends to learn representation for input data space. The goal of autoencoder is to learn the hidden compressed latent space of the data by an encoder. From that compressed space, we will try to reconstruct the original data after feeding the latent representation to the decoder. Due to the bottleneck in the process of learning latent space with minimum loss, we can extract functional features of original input data, and allow the model to explore broader space of data with similar features.

However, with autoencoder, there is a gap of understanding the connection between learning representations and generating new contents. The difficulty stands in the unregularized latent space, which relies on input data distribution. Problem in guarantee of the compatibility between latent space priori with content generation process is intractable for the model to solve by itself.

To resolve the issues, instead of trying to reduce dimension and encode the input values into separate points in latent space, we can learn about the distribution over that latent space. Assume that we have a priori distribution condition for  $z \sim P_\theta(z)$ , in which  $z$  is an unobserved continuous random variable, and data  $X$  will be generated conditioned on  $z$ , or  $X \sim P_\theta(X|Z)$ . In this case,  $z$  becomes the hidden probabilistic representation of  $X$ , or we can interpret it as  $q_\phi(z|x)$ . The encoder will try to learn  $q_\phi$  and decoder trying to learn  $P_\theta(X|z)$ , which means decoding the hidden  $z$  into the input space again. This algorithm, called *Variational Autoencoder* (VAE), helps the model avoid overfitting and get valuable features of latent space for better generation. The model with VAE algorithm is trained by maximizing this objective function:

$$\mathcal{L}_{\theta,\phi}(x, z) = \mathbb{E}_{z \sim q_\phi(z|x)}[\log p_\theta(x|z)] - \mathcal{D}_{KL}(q_\phi(z|x) \| p(z)). \quad (5)$$

The first term is the reconstruction loss or expected value of log likelihood of the data, taken on the distribution of generating  $z$  from the hidden distribution. This term handles the reconstruction for the original data from the latent space. The second term is the Kullback-Leibler divergence between the hidden distribution  $q_\phi$  and the priori condition on  $z$ ,  $p(z)$ . This term measures how divergent between two distributions, or how much information that can get lost if we use  $q$  as prior for  $z$ . Figure 2.2 shows the general form of Variational Autoencoder in text reconstruction task.

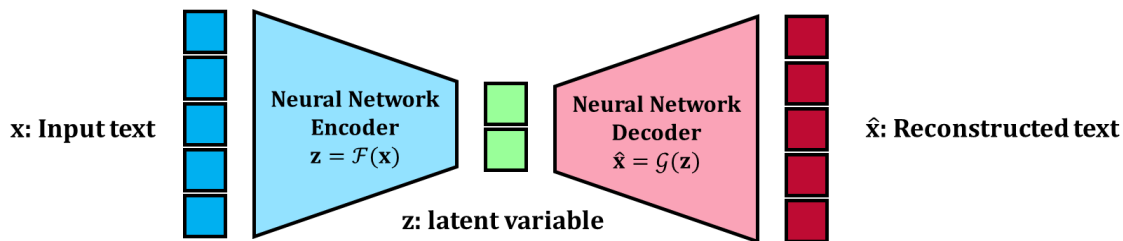


Figure 2.2. Variational Autoencoder in text reconstruction

### 2.2.2 Conditional Variational Autoencoder

VAE is good for giving prior distribution for the data and its latent space to guide the reconstruction better. However, with VAE, it is difficult for human to constrain the data model will generate after training. For example, if we already trained VAE well for summarization task, it could make concise and grammatically correct summaries, but we could not control what content of sentences it will generate. The problem is because we could not tell the model which topic or which specific collections of words it should generate later. Another problem is NLP tasks like text summarization or machine translation is they require the model to output each token based on earlier tokens. Therefore, unlike VAE when reconstructing original data, we have two conditions on the generation: input documents and the output we need to generate.

For this, VAE architecture should be changed to adapt specific tasks. Given input original documents  $X$ , we want the model to produce summary  $Y$ . In this case, VAE process will be modified as following: given observations including input variable  $\mathbf{x}$ , and latent variable  $\mathbf{z}$  is created from prior  $p_{\theta}(\mathbf{z}|\mathbf{x})$ . We have output variable  $\mathbf{y}$  generated from distribution  $p_{\theta}(\mathbf{y}|\mathbf{x}, \mathbf{z})$ . The encoder then will try learning the hidden representation distribution  $q_{\phi}(\mathbf{z}|\mathbf{x}, \mathbf{y})$  giving condition on  $\mathbf{y}$ . The decoder then will try to decode the hidden latent variable and produce output  $\mathbf{y}$  by learning distribution  $p_{\theta}(\mathbf{y}|\mathbf{z}, \mathbf{x})$ . This model is called **Conditional Variational Autoencoder (CVAE)**, in which it takes one more variable as condition to constrain the generation of the model. Figure 2.4 shows the graph of relationship between  $\mathbf{x}, \mathbf{y}$  and  $\mathbf{z}$  in CVAE.

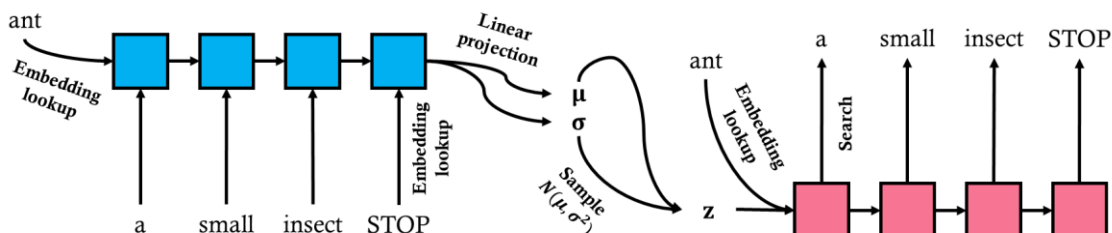
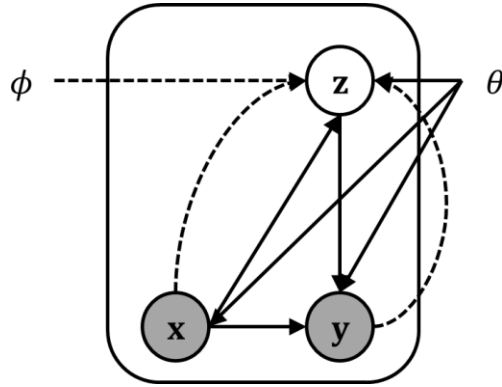


Figure 2.3. Example of Variational Autoencoder in text reconstruction



The objective function of CVAE is similar to VAE, except there is one more condition, or variable to add:

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



(6)

**Figure 2.4.** Conditional Variational Autoencoder graph for ELBO objective [Zhang et al, 2016]

## 2.3 Sampling techniques

### **Nucleus (Top-p) filtering**

Top- $k$  sampling is a technique of choosing most likely  $k$  words from the distribution. Instead of choosing a fixed subset of words, top- $p$  (nucleus) sampling [Holtzman et al, 2019] chooses the smallest possible subset of words whose cumulative probability starts to exceed  $p$ .

### **Other sampling techniques**

Besides top- $p$  and top- $k$ , beam search is also the most common strategy to sample. However, top- $p$  gives a dynamic subset sizes to choose the words comparing to the other two, which is good for generating long sequence.

## 2.4 Metrics in NLP

### **ROUGE score**

ROUGE stands for *Recall-Oriented Understudy for Gisting Evaluation*, which is used to measure the overlapping between the generated sequence and the reference sequence. It calculates how much of the

generated one can recover the ground-truth texts using both precision and recall to report F-measure. There are two types of ROUGE score that are common for NLP task like summarization:

- ROUGE-*N*: measures the n-gram overlap (which means how many continuous of n tokens are overlapping with the reference)
- ROUGE-*L*: measure the longest matching sequence of words by using Longest Common Subsequence Problem. Briefly, it computes the largest number of increasing (in terms of order) in-sequence tokens that is similar to the reference without consecutive matching.

The following two metrics are not decided by the tokens but are dedicated tasks in NLP to measure the semantic meaning and similarity.

### **Semantic Text Similarity-Benchmark**

This task is trained on a dataset of sentence pairs drawn from news headlines, video and image captions, and natural language inference data [Cer et al, 2017]. Each pair has an annotation of similarity score from 0-5 and evaluated using Pearson and Spearman correlation coefficients.

### **Recognizing Textual Entailment (RTE)**

RTE is another task to compare semantic meaning or directional relation between two texts, particularly deciding whether the meaning of hypothesis text can be inferred from another text.

### **Novel n-grams**

Novel n-grams is the metric for testing how much novel words and tokens the model can generate comparing to the original document.

## CHAPTER 3

### RELATED WORK

[See et al, 2017] used LSTM with Attention and pointer-generator mechanism to improve performance of RNN sequence-to-sequence Text Summarization. However, the Transformer based models have resulted in better performance improvement. Recent encoder-decoder approaches pre-trained on masked input objective are used to have better language understanding for summarizing documents.

**PEGASUS** [Zhang et al, 2020] proposed sentence-masked pretrained on large text corpora with the objective of generating output sequence from remaining sentences.

**ProphetNet** [Yan et al, 2020] proposed self-supervised objective approach by optimizing prediction of next  $n$  tokens simultaneously pretrained on a large-scale dataset.

**BART** [Lewis et al, 2020] proposed a denoising autoencoder approach as pretrained encoder-decoder model. Text is corrupted with a noising function and the model tries to reconstruct the original text. Text Summarization is the fine-tuned task with the document is kept as original un-denoised text.

Recently, [Wang et al, 2019] proposed a combination of Variational Autoencoder and Transformer for Text Filling task. With BART as our pretrained model, we also incorporate Variational Autoencoder to Transformer to enhance the variation for Text Summarization and use Reinforcement Learning reward system to evaluate the quality of Text Summarization results.

## CHAPTER 4

# CONDITIONAL VARIATIONAL AUTOENCODER WITH REWARD AS GUIDANCE FOR TEXT SUMMARIZATION

In chapter 2, we have introduced some models, techniques and algorithms that could serve as backgrounds for each of our steps in the workflow of understanding the structure and generating summaries. We will exploit the advantages of those fundamental models and algorithms and propose a model that can make summarization having higher variance and more controllable with these steps:

- 1) Understanding and learning the inner structure of documents in summarization task,
- 2) Make guidance for model with rewards and penalties while training,
- 3) Generate summaries by sampling and control levels of extractive and abstractive

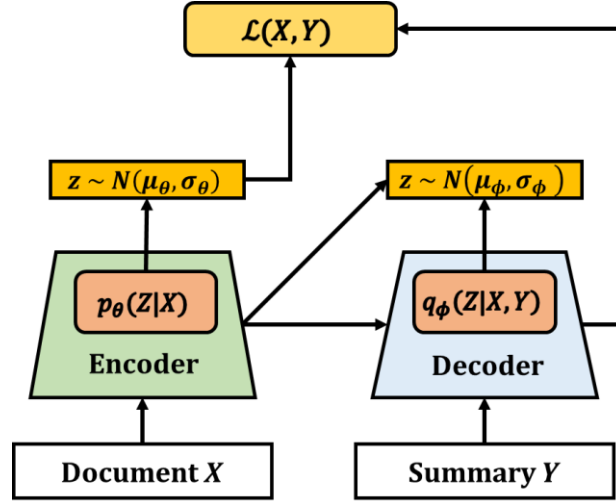
We use BART in chapter 3 as the baseline for our model, due to the combination of BERT encoder and GPT-2 decoder for autoregressive in the model.

### **4.1. Learning inner representations of documents**

Documents training in deep neural network can be represented as a combination of word embeddings. However, document embeddings have remarkably high dimensions and are variant in terms of lengths. Transformer-based summarization models tend to use the embeddings directly for training, due to its strong attention mechanism, and force the machine to focus on a narrow search space for generating text. Assume that there are unobserved continuous latent random variables  $z_e$  and  $z_d$  as our hidden representations of encoder (as for original documents) and decoder (as for reference summary) respectively, and the latent variables are generated from distributions  $z_e \sim N(\mu_e, \sigma_e)$ ,  $z_d \sim N(\mu_d, \sigma_d)$ . Since summary is a concise way of expressing the original documents, we can apply CVAE method and make the model learns to minimize divergence between these two distributions.

With the addition of CVAE, we can make general latent vector spaces that characterize the distributions of all data samples. Given input variable  $\mathbf{x}$  as the original long documents vector space, and  $\mathbf{y}$  as the reference summary vector space. We will try to learn the hidden representations  $z_e$  and  $z_d$  to make the model has the ability of generating diverse summaries. Like practical conventional CVAE method, we

will assume the distributions follow multivariate Gaussian distributions. Figure 4.1 shows our combination of CVAE and Transformer model for Text Summarization.



**Figure 4.1.** Our Transformer-CVAE graph

Assume that  $f(x)$  and  $g(y)$  are encoder features and decoder features after running through baseline Transformer model. We calculate the context vector of both encoder and decoder by using mean of all features. To adapt the conditional probability of  $z_{dec}|x, y$ , we concatenate both features first and compute the mean vector for context:

$$\begin{aligned} ctx_{enc} &= \text{mean}([f(x)]) \\ ctx_{dec} &= \text{mean}([f(x); g(y)]) \end{aligned} \tag{7}$$

We put two multilayer perceptrons (MLP) on top of encoder and decoder features to make model learn  $(\mu, \sigma)$  of each encoder/decoder:

$$\begin{aligned} \begin{bmatrix} \mu_e \\ \sigma_e \end{bmatrix} &= \text{MLP}_e(ctx_{enc}), \\ \begin{bmatrix} \mu_d \\ \sigma_d \end{bmatrix} &= \text{MLP}_d(ctx_{dec}). \end{aligned} \tag{8}$$

Although model cannot backpropagation through random sampling, we can sample  $z$  from mean and variance values here using:

$$z = \sigma \cdot \epsilon + \mu, \tag{9}$$

with  $\epsilon \sim N(0,1)$ . This technique is called **reparameterization trick**, which allows the model to do backpropagation even though we are random sampling  $z$ . Given  $s(y)$  is the decoder features, or the states

of sequence  $y$  after running through Transformer decoder. After getting vector  $z_e$  from encoder, we get new decoder features by conducting a cross-attention layer, which is used in Transformer decoder, between  $z$  and the summary or decoder embedding  $g_{embed}(\hat{y})$ , with  $\hat{y}$  is the new generated sequence from the old  $s(y)$ , which is provided before going through Transformer decoder in BART.

$$s(\hat{y}) = \text{cross\_attention}(z_e, g_{embed}(\hat{y})) \quad (10)$$

Finally, we concatenate the new feature to the old decoder features to extract the output distribution over vocabulary for each token by feeding the features to the final LSTM and linear layer.

$$\begin{aligned} o_{LSTM} &= \text{LSTM}([s(\hat{y}); s(y)]), \\ o_d &= b + Wo \end{aligned} \quad (11)$$

Normally,  $z$  vector can be concatenated to decoder features to train for generating vocabulary distribution. The reason we do cross-attention together with LSTM here is for step 2 of the workflow, described in Section 4.2, in which we need to generate new tokens in training step later. The model is presented as in the following figure:

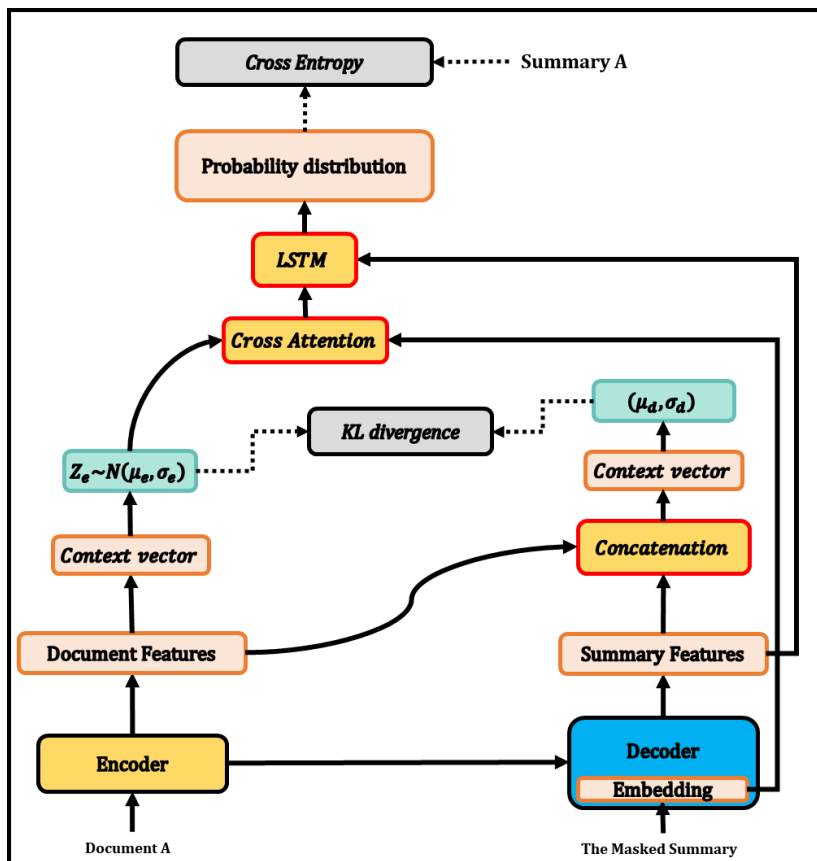


Figure 4.2. Detail structure of CVAE-Transformer for Text Summarization

Loss function of the above model is similar to CVAE objective function:

$$\text{loss}_{CVAE}(x, y) = -\mathbb{E}_{z \sim q_\phi(z|x)}[\log p_\theta(x|z)] + \mathcal{D}_{KL}(q_\phi(z|x, y) \| p(z|x)). \quad (12)$$

Empirical loss function with  $z_e$  and  $z_d$  is rewritten as

$$\text{loss}_{CVAE}(x, y) = -\frac{1}{L} \sum_{l=1}^L \log p_\theta(y|x, z^{(l)}) + \mathcal{D}_{KL}(q_\phi(z_d|x, y) \| p(z_e|x)). \quad (13)$$

with  $L$  is the number of samples and  $z^{(l)} = (z_e^l, z_d^l) = \mathcal{F}(x, y, \epsilon^{(l)})$ , with  $\epsilon^{(l)} \sim N(0,1)$  is described in reparameterization trick, and  $\mathcal{F}$  is two multilayer perceptrons (MLP) above.

#### **4.2. Guidance model by giving rewards**

Given a model with a CVAE algorithm for learning the hidden representation, our model now relies on the generation from latent variable  $z$ . To make the generative process understands deeper about how it should generate words, we propose adding a process of evaluating the ability of the model without teacher-forcing log-likelihood method. The process is similar to Reinforcement Learning if we see Text Summarization as a Reinforcement Learning (RL) task:

- 1) At time step  $t$ , we generate a state as decoder features  $s_t$ ,
- 2) Model does the action of generating predicted word  $\hat{y}_{t+1}$  by the policy, which is the probability distribution  $p(\hat{y}_{t+1}|s_t)$ ,
- 3) Model gives the reward  $r_t$  for the action of choosing  $\hat{y}_{t+1}$ ,
- 4) Run the model with predicted word  $\hat{y}_{t+1}$  as input and we get new decoder features  $s_{t+1}$ .

Our goal of training is minimizing the objective function, which is in the form of negative expected reward [Rennie et al, 2017]:

$$L(\theta) = -\frac{1}{T} \sum_{t=1}^T r_t(\hat{y}_t), \quad \nabla L(\theta) \approx -r(\hat{y}) \nabla_\theta \log p_\theta(\hat{y}) \quad (14)$$

with  $T$  is the length of the generated summary.

This new objective function is based on the generation of the sequence by our Transformer + CVAE model and we will calculate the rewards for each action of generating word.

While RL works well with RNN and LSTM, it is not resource-efficient to combine with Transformer-based model since Transformer is already a large model, and it takes huge memory when generating each

token while training. To alleviate the problem, we propose a method of *generating random sequence window* in each sample, which is motivated from text filling task. Instead of generating a whole summary, we randomly masking a window of reference tokens, which has smaller size than the reference summary length, and make the model try to predict the masking tokens without looking at reference ones.

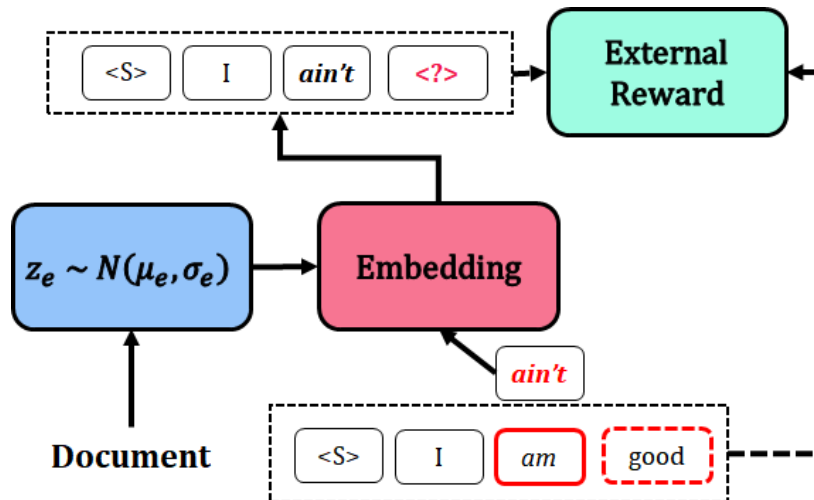


Figure 4.3. Reward system for CVAE-Transformer

Each generated token will be transformed to embedding vector through decoder embedding layer and conduct cross-attention with latent vector  $z$  at section 4.1. Through the top LSTM layer, we can construct new features for the next token. With this method of combining *random sequence window* and *decoder embedding-latent variable cross-attention*, we could avoid running a whole decoder while training.

The reward at each time step we choose is ROUGE score and Semantic Text Similarity (STS) score, which is computed via RoBERTa, a pretrained multi-task model finetuned with STS-B task. ROUGE score is a combination of ROUGE-1, ROUGE-2 and ROUGE-L. We compute the score by comparing a sequence of previous words and the current generated word with reference words. Since the first masking tokens are less sensitive to low score, we give it higher weights than the later tokens by giving it a *discount* value  $\lambda = 0.99$ . Assuming the random window starts at time step  $t_0$ , the formula for reward at time step  $t$  is represented as following:

$$r_{rouge} = \frac{1}{4}(r_{rouge1} + 2 \cdot r_{rouge2} + r_{rougeL})$$

$$r_t = \frac{1}{2} \lambda^{t-t_0} (r_{rouge} + r_{sts})$$
(15)



Applying the reward to (17), we have the final objective functions for the model:

$$\begin{aligned}
 L_{cvae}(\theta) &= -\frac{1}{T} \sum_{t=1}^T \log p_{\theta}(y_{t+1}|x_t, z_t^{(l)}) + \beta \text{KL}(q_{\phi}(z_d|x, y) || p_{\theta}(z_e|x)), \\
 L_r(\theta) &= -\frac{1}{T_0 - t_0} \sum_{t=t_0}^{T_0} r_t(\hat{y}_t), \quad \nabla L(\theta) \approx -r(\hat{y}) \nabla_{\theta} \log p_{\theta}(\hat{y}),
 \end{aligned}
 \tag{16}$$

with  $[t_0, T_0] \subseteq [1, T]$  is the window for generating word. Figure 4.3 illustrates an example of how random window sequence works with rewards system.

### **4.3 Generating summaries by sampling**

The last step of the workflow is how we can sample during training and inference stage and make the model becomes more abstractive. At RL step in 4.2, we need to predict the word by probability distribution at each time step.

Normally we can use argmax to get the words. However, to make our model generate abstractive but coherent sentence and also adapt with the reward system, we give other candidates a chance based on the probability distribution and check whether the chosen candidate can generate high reward or not. Moreover, sampling gives the model higher variance of possibilities. Therefore, we use sampling by probabilities generated from the model at each time step. We also do not want the model choosing outlier words, or unlikely words with low probabilities and break the coherent story we are generating.

From the background in 2.3, we choose top- $p$  for sampling due to its ability of having dynamic candidates in each step and not sensitive to flatten distribution. With top- $p$  and CVAE, we can have a model with high ability of generating abstractive summary.

Given a prior text and having a probability distribution for predicting next word, the task is how we can choose the next word to make our current passage and future passage becomes naturally and as abstractive as possible. The problem can be re-written as following: given context with a sequence of  $m$  tokens  $\{x_1, \dots, x_m\}$ , the task is to generate  $m + n$  continuous tokens  $\{x_{m+1}, \dots, x_{m+n}\}$ . Assume that we have probability distribution from the beginning to the last tokens as

$$p(x_{1:m+n}) = \prod_{i=1}^{m+n} p(x_i|x_1 \dots x_{i-1}).
 \tag{17}$$

[Holtzman et al, 2019] proposed a stochastic decoding method for sampling based on this decomposition, which is based on the shape of the probability distribution. At time step  $i$ , given a

distribution  $p(x|x_{i-1})$ , we define a top- $p$  subset of vocabulary  $V^{(p)} \subset V$  as the smallest set such that

$$\sum_{x \in V^{(p)}} p(x|x_{1:i-1}) \geq p \quad (18)$$

Let  $p' = \sum_{x \in V^{(p)}} p(x|x_{1:i-1})$ , the distribution is re-calculated as

$$p'(x|x_{1:i-1}) = \begin{cases} p(x|x_{1:i-1})/p', & \text{if } x \in V^{(p)} \\ 0 & \text{otherwise} \end{cases} \quad (19)$$

The task can be briefly summarized as selecting the highest probability tokens whose cumulative probability surpass the threshold  $p$ .

## CHAPTER 5

### EXPERIMENTAL RESULTS

In this section, we tested the model with CNN-Dailymail dataset, which comprises of 287K documents and articles with 3-4 highlights in each document to summarize the contents of the article. The pre-trained model we used is BART finetuned on 5 epochs.

#### 5.1 Experimental setup

First, we train the CVAE and freeze BART parameters to make the latent variable  $z$  understands the current model condition. After training for 2 epochs, we trained the whole model with CVAE and RL loss for 1 epoch.

#### ROUGE score

Supervised Learning ROUGE score			
Model	ROUGE-1	ROUGE-2	ROUGE-3
p <sub>gen<sub>cov</sub></sub> + recorder	40.44	18.15	36.9
PEGASUS	<b>44.17</b>	<b>21.47</b>	<b>41.11</b>
BART (Baseline)	44.16	21.28	40.90
BART + CVAE – Reward	40.81	17.58	37.61

**Table 5.1.** ROUGE scores comparisons

### STS Score

<b>Semantic Text Similarity Score (comparing with reference summary)</b>	
BART (Baseline)	0.3279
BART + CVAE – Reward	0.3253

**Table 5.2.** STS-B scores comparison

### RTE Score

<b>Recognizing Textual Entailment</b>	
BART (Baseline)	0.2082
BART + CVAE – Reward	0.1894

**Table 5.3.** RTE scores comparison

### Novel n-gram score

<b>Novel n-gram</b>					
<b>Model</b>	<b>1-gram</b>	<b>2-gram</b>	<b>3-gram</b>	<b>2-gram (w/o overlap)</b>	<b>3-gram (w/o overlap)</b>
<i>Reference</i>	<i>13.48</i>	<i>51.48</i>	<i>73.92</i>	<i>2.69</i>	<i>0.61</i>
BART (Baseline)	1.61	11.14	21.17	0.17	0.02
BART + CVAE–Reward	10.57	37.92	55.63	2.32	0.57

**Table 5.4.** Novel *n*-gram scores comparison

**Examples of results:**

<p><b>Article</b></p>	<p>The student was identified during an investigation by campus police and the office of student affairs and admitted to placing the noose on the tree early Wednesday, the university officials said ... (cont)... Last month a fraternity at the University of Oklahoma had its charter removed after a video surfaced showing members using the N-word and referring to lynching in a chant. Two students were expelled (cont)...</p>
<p><b>BART + CVAE - Reward</b></p>	<p>The student is "no longer on campus" and will face student conduct review, the university says. The student admitted to hanging on a tree with the noose, the university says. Two students were expelled after being caught on camera using the N-word. The incident is one of several recent racist events to affect college students.</p>
<p><b>Article</b></p>	<p>Filipinos are being warned to be on guard for flash floods and landslides as tropical storm Maysak approached the Asian island nation Saturday. Just a few days ago, Maysak gained super typhoon status thanks to its sustained 150 mph winds. It has since lost a lot of steam as it has spun west in the Pacific Ocean. It boasts steady winds of more than 70 mph (115 kph) and gusts up to 90 mph as of 5 p.m. (5 a.m. ET) Saturday ...(cont)... Authorities took preemptive steps to keep people safe such as barring outdoor activities like swimming, surfing, diving and boating in some locales, as well as a number of precautionary evacuations. ...(cont)... Dry told PNA, "We do not know what the impact will be once it will make landfall."</p>
<p><b>BART</b></p>	<p>Tropical storm Maysak is heading west in the Pacific Ocean. It boasts steady winds of more than 70 mph (115 kph) and gusts up to 90 mph. It's expected to make landfall Sunday morning on the southeastern coast of Isabela province. Authorities take preemptive steps to keep people safe such as barring outdoor activities.</p>
<p><b>BART + CVAE - Reward</b></p>	<p>Maysak is the latest tropical storm to reach the Philippines. The storm has sustained 150 mph winds and is driven west in the Pacific Ocean. Authorities have banned outdoor activities such as swimming, surfing, diving and boating. Maysak is expected to make landfall Sunday morning.</p>
<p><b>Article</b></p>	<p>The Palestinian Authority officially became the 123rd member of the International Criminal Court on Wednesday, a step that gives the court jurisdiction over alleged crimes in Palestinian territories. The formal</p>

	accession was marked with a ceremony at The Hague, in the Netherlands, where the court is based.
<b>BART</b>	<b>The Palestinian Authority becomes the 123rd member of the International Criminal Court.</b> The formal accession was marked with a ceremony at The Hague, in the Netherlands, where the court is based.
<b>BART + CVAE - Reward</b>	Palestinian Authority joins International Criminal Court, in a formal accession ceremony at the Hague, in the Netherlands.
<b>Article</b>	Michele Bachmann is comparing President Obama to the co-pilot of the doomed Germanwings flight. ... <i>(cont)</i> ... Andreas Lubitz, the co-pilot of Germanwings Flight 9525, is accused by authorities of deliberately crashing the plane in the French Alps. He died in the crash along with 149 other crew and passengers. Many comments posted on her Facebook page blasted the former representative. ... <i>(cont)</i> ... The congresswoman, who sought the GOP presidential nomination in 2012, said Obama had a "condescending smile on his face and laughed at me."
<b>BART</b>	Michele Bachmann is comparing President Obama to the co-pilot of the doomed Germanwings flight. Bachmann: "With his Iran deal, Barack Obama is for the 300 million souls of the United States what Andreas Lubitz was for the 150 souls on the German Wings flight". Many comments posted on her Facebook page blasted the former representative.
<b>BART + CVAE - Reward</b>	Michele Bachmann is criticized for comparing President Obama to the co-pilot of the Germanwings flight. "He is for the 300 million souls of the United States what Andreas Lubitz was for the 150 souls on the German Wings flight," she says. Andreas Lubitz was involved in a crash in March, and Lubitz died in the crash. Bachmann also said Obama was "condescending" to her after a meeting in 2012.

**Table 5.5.** Summarization result examples

## CHAPTER 6

### CONCLUSION

In this research, we propose a novel approach for Abstractive Summarization using understanding of latent representations of documents, guidance with semantic meaning and generating summary with random sampling.

Our test proves that having good understanding in hidden representations together with the guidance of external metrics in Reinforcement Learning can help model generate highly abstractive summary.

Latent representation in Conditional Variational Autoencoder makes the model get the **global critical features** from documents. Semantic and ROUGE rewards guide the model **generating correct information**. Random sampling **enhances abstractedness** of generated summary.

Despite of not outperforming in ROUGE score, our model shows that it can generate higher abstractive summaries while still keep semantic similarity on par with baseline model. The model could be used as a useful paraphrasing tool for baseline model.

From the results of the experiments, we can also see that BART is a very extractive model when trained on CNN-Dailymail dataset. Our model is an example of an abstractive summarization model which can have many ways for generating summaries rather than relying on the reference ones. Therefore, ROUGE score might not be an important metric for abstractive summarization.

## BIBLIOGRAPHY

- [Luong et al, 2015] Luong, M. T., Pham, H., & Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. *Conference on Empirical Methods in Natural Language Processing*, 1412–1421.
- [Vaswani et al, 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30, 5998-6008.
- [Zhang et al, 2016] Zhang, B., Xiong, D., Su, J., Duan, H., & Zhang, M. (2016). Variational neural machine translation. *Conference on Empirical Methods in Natural Language Processing*, 521–530.
- [Holtzman et al, 2019] Holtzman, A., Buys, J., Du, L., Forbes, M., & Choi, Y. (2019). The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.
- [Cer et al, 2017] Cer, D., Diab, M., Agirre, E., Lopez-Gazpio, I., & Specia, L. (2017). Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, 1–14.
- [See et al, 2017] See, A., Liu, P. J., & Manning, C. D. (2017). Get to the point: Summarization with pointer-generator networks. *Annual Meeting of the Association for Computational Linguistics*, 1073–1083.
- [Zhang et al, 2020] Zhang, J., Zhao, Y., Saleh, M., & Liu, P. (2020, November). Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning* (pp. 11328-11339). PMLR.
- [Yan et al, 2020] Yan, Y., Qi, W., Gong, Y., Liu, D., Duan, N., Chen, J., ... & Zhou, M. (2020). Prophetnet: Predicting future n-gram for sequence-to-sequence pre-training. *Findings of the Association for Computational Linguistics: EMNLP*, 2401–2410.
- [Lewis et al, 2020] Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., & Zettlemoyer, L. (2020). Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.



[Rennie et al, 2017] Rennie, S. J., Marcheret, E., Mroueh, Y., Ross, J., & Goel, V. (2017). Self-critical sequence training for image captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 7008-7024).

[Wang et al, 2019] Wang, T., & Wan, X. (2019, August). T-CVAE: Transformer-Based Conditioned Variational Autoencoder for Story Completion. In *IJCAI* (pp. 5233-5239).

## **ACKNOWLEDGEMENTS**

I would like to pay my deepest regards and thanks of gratitude to my advisor, Professor Jaesik Choi. Although I could not fulfil all my tasks with high responsibility, he ceaselessly supports me during my most tough time. Without his conscientiously guiding, I would have not been able to finish this research. I have learned a lot and his advice will always be crucial factors for my future research and work.

I truly would like to send my deepest appreciation to my administrative advisor, Professor Kwang In Kim, for giving me invaluable comments for the incomplete parts in my research. I also would like to thank Professor Young-ri Choi for being in my committee as well as helping me broaden my knowledge during our collaboration work. I am also grateful to thank everyone in Statistical Artificial Intelligence Lab for all of your wholehearted support during my time at SAIL.

Finally, I owe my biggest thanks to my family and my friends, who have always been supporting and helping me unconditionally. Thank you everyone once again for all of the supports I have received during this Master journey.