VIETNAM NATIONAL UNIVERSITY
UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING

- Graduation Thesis -

LAMPAT: Low-rank Adaptation for Multilingual Paraphrasing using Adversarial Traning

Council:	Computer Science	
Instructor:	Assoc. Prof. Quan Thanh Tho, PhD	
	Mr. Bang Ngoc Bao Tam, MEng	
Students:	Pham Khanh Trinh	1953044
	Le Minh Khoi	1952076

________________________________ Date: ____________

Assoc. Prof. Quan Thanh Tho, PhD (Instructor)
Associate Professor
Faculty of Computer Science and Engineering

________________________________ Date: ____________

Mr. Bang Ngoc Bao Tam, MEng (Instructor)
Master of Engineering
Faculty of Computer Science and Engineering

Declaration of Authenticity

We declare that this research is our own work, conducted under the supervision and guidance of Assoc. Prof. Dr. Quan Thanh Tho, PhD and Mr. Bang Ngoc Bao Tam, MEng. The result of our research is legitimate and has not been published in any forms prior to this. All materials used within this researched are collected ourselves by various sources and are appropriately listed in the references section.

In addition, within this research, we also used the results of several other authors and organizations. They have all been aptly referenced.

In any case of plagiarism, we stand by our actions and will be responsible for it. Ho Chi Minh City University of Technology - Vietnam National University HCMC therefore are not responsible for any copyright infringements conducted within our research.

Ho Chi Minh City, June 2023
Authors
Pham Khanh Trinh
Le Minh Khoi

Acknowledgements

We would like to express our deep and sincere gratitude to our supervisor, Dr. Quan Thanh Tho, for his patience, guidance, and support in our student years. We have benefited greatly from your wealth of knowledge and meticulous editing. We are extremely grateful that you took us on as students and continued to have faith in us over the years.

Our family deserves endless gratitude for their everlasting love, sacrifice, and support; hence keeping us motivated and confident. Our accomplishments and success are because they believed us. There is no word that can describe our gratitude for our family.

Abstract

Paraphrases are texts that convey the same meaning while using different words or sentence structures. Generating paraphrased sentences is a lifelong problem for natural language learning. For example, the sentence “I like to eat Pho and Fried Rice” could be expressed as "My favourite foods are Pho and Fried Rice". Paraphrases can be used as an automatic data augmentation tool for Natural Language Processing (NLP) tasks, especially for low-resource languages, where data shortages are a vital problem. The low-resource problem can arise mainly because languages are considered low-resources or domains are considered low-resources. Researchers have attempted to identify low-resource languages (LRLs) by exploring various criteria such as the number of native speakers and available datasets. LRLs are languages that lack a unique writing system, a presence on the World Wide Web, language-specific expertise, or electronic resources such as language repositories, data (monolingual and parallel), vocabulary lists, etc. The multilingual model is a type of Machine Learning model that can understand different languages. One example would be classifying whether a text is a malicious comment. Using an English Language Model, we will only be able to detect negative comments in English, but not negative comments in Spanish. However, if we try to get the model to understand both English and Spanish, it will be able to fulfil the task mentioned above. Furthermore, extending this multilingual model helps to understand more languages around the globe. Training models with data from multiple language pairs can also help a low-resource language gain more knowledge from other languages. Besides, because annotating the dataset is a labour-consuming method, self-supervised learning emerged as one of the new methods influencing many state-of-the-art models. In this work, we present a new method called LAMPAT (**L**ow-rank **A**daptation for **M**ultilingual **P**araphrasing using **A**dversarial **T**raining). Throughout the experiments, we find out that our method not only work well for English but can generalize to other languages as well.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Paraphrases try to express the new sentences with different expressions but the same meaning as the original sentences or text. Paraphrase generation focuses on synthesizing paraphrases of a given sentence automatically. This is a fundamental natural language processing task that is important for many downstream applications. For example, paraphrases can improve question answering [3], enhance the response of chatbot engines [4], and extend the coverage of semantic parsers [5]. Generating high-quality and diverse paraphrases is a fundamental yet challenging natural language processing task. Although the previous works are effectively based on generative models, there remain problems with exposure bias in recurrent neural networks, and often a failure to generate realistic sentences.

With the growth of large datasets and neural networks, recent studies have shown promising results. Research in three families of deep learning models has been conducted to generate high-quality paraphrases. The first is to generate paraphrases as a sequence-to-sequence problem, following experience from machine translation [6], while Prakash and his team [7] propose stacked residual long short-term memory networks (LSTMs). Shortly after that, several works focused on providing extra information to improve the encoder-decoder model.

However, previous works are largely based on supervised methods, which require a large amount of labelled data that is costly to collect. To address this drawback, we adopt a transfer learning approach and propose a training pipeline that enables pre-trained language models to generate high-quality paraphrases in an unsupervised method. Our work originated from Niu and his team's work [8], which consists of task adaptation, self-supervision, and a novel decoding algorithm named Dynamic Blocking (DB). Whenever the language model emits a token contained in the source sequence, DB prevents the model from outputting the subsequent source token for the next generation step. Although the previous work produces good quality for unsupervised paraphrasing, works on paraphrasing for Low-resource languages (LRLs) [9] have not attracted much attention due to the scarcity of parallel datasets as well as the lack of pre-trained language models for LRLs. However, we see this limitation as an opportunity since the unsupervised method requires little to no parallel dataset and recently, with the help of pre-trained multilingual models, we can take advantage of these models to eliminate the scarcity of LRLs.

Paraphrasing for LRLs not only contributes to the sustainable growth and diversity of the language itself, but it also addresses the shortage of data for LRLs. Paraphrase generation can be regarded as a method of augmenting data in Natural Language Processing. If we approach an LRLs' downstream task and need more data for training or validation, paraphrasing can be taken into consideration with a view to promising results.

1.2 Goals

This project is about developing a deep learning model that is capable of performing paraphrase generation with the given input. The model is expected to paraphrase in multilingual settings with many low-resource languages involved. To achieve these goals, we have to prepare general theoretical knowledge in Natural Language Processing. Moreover, we have to conduct a thorough literature review of some related works on Paraphrasing for Low-resource Languages and make a comparison between them. Then, we can come up with new solutions which hold promising results. After all, we will summarize our work, give some future directions, and prepare for the next steps.

1.3 Scope

In this Capstone Project, we aim to deal with two problems as listed below:

- **Paraphrase generation**: Given a sentence, our proposed model can generate a new sentence that shares the same semantic meaning but is different in syntax, word choice and lexical phrases. Here we focus on sentence-level paraphrasing instead of paragraph-level.
- **Multilingual**: The proposed model can paraphrase not only in one language but also in other languages as well. However, in each sentence input, all of the words/tokens will need to be in the same language.

1.4 Contribution

With the objectives and scope of this Capstone Project clearly defined, notable accomplishments have been attained. The contributions of this work can be summarized as follows:

- We have applied a parameter-efficient fine-tuning approach to a large multilingual language model, enabling it to efficiently perform the task of paraphrase generation.

- We have integrated adversarial training with parameter-efficient fine-tuning in an unsupervised learning technique to address the challenge of limited data annotation and to generate diverse and non-duplicate paraphrases.
- We have gathered and extracted data to create the first multilingual, multi-domain paraphrasing evaluation datasets, which can be utilized to evaluate the model's ability to maintain semantic coherence and produce manifold lexical choices.
- We have consolidated the most recent paraphrasing evaluation metrics, thus facilitating future advancements in this field.

1.5 Capstone Project Structure

There are five chapters in this project's proposal.

- **Chapter 1** briefly describes the motivation of the problem as well as the goal and scope of the project.
- **Chapter 2** focuses on providing preliminaries about Natural Language Processing (NLP) knowledge used in the project.
- **Chapter 3** organizes related works of the same task were brought into discussion to provide a deeper understanding of the current methods and their limitation
- **Chapter 4** illustrates our method, process to construct evaluation dataset and the result with analysis for future development
- **Chapter 5** summarizes our whole project's proposal and the plan for future development.

Chapter 2

Theoretical Background

2.1 Recurrent Neural Network

In time series data, observations are not independent of one another because the current observation depends on earlier observations. However, due to the networks' inability to store historical or previous information, traditional neural networks see each observation as an independent entity. In essence, they are completely forgetful of the past.

2.1.1 RNN

Recurrent Neural Networks (RNNs) [10], which include the reliance between data points into neural networks to add the idea of memory, became popular as a result. By learning repeating patterns, RNNs may be trained to recall concepts depending on context.

RNNs develop memory through a cell-based feedback loop. The key distinction between an RNN and a conventional neural network is this. In contrast to feed-forward neural networks, which only enable information to be transmitted between layers, the feed-back loop allows information to be passed inside a layer.

The information that is important enough to be stored in memory must then be determined using RNNs. As a result, many RNN types have developed:

Recurrent neural network (RNN) techniques Long-Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) Recurrent Neural Networks

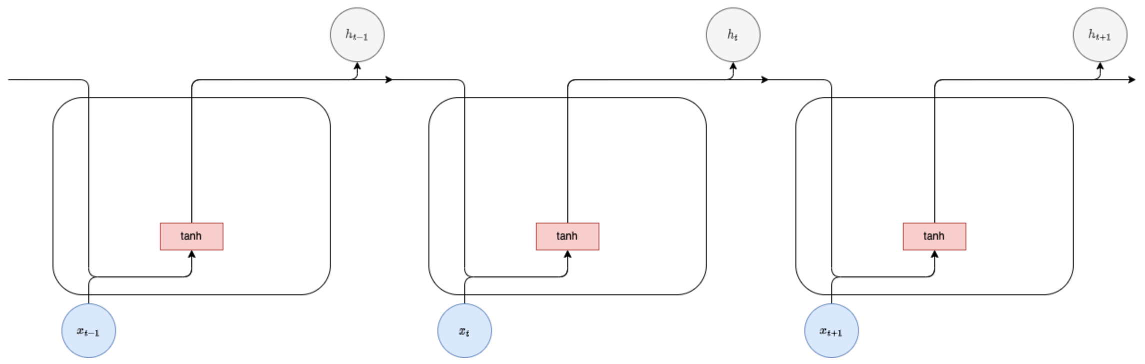


Figure 2.1: RNN unfold.

Through the feedback loop the output of one RNN cell is also used as an input by the same cell. Hence, each cell has two inputs: the past and the present. Using information of the past results in a short term memory.

For a better understanding we unroll/unfold the feedback loop of an RNN cell as illustrated by Figure 2.1. The length of the unrolled cell is equal to the number of time steps of the input sequence.

We can see how past observations are passed through the unfolded network as a hidden state. In each cell the input of the current time step **x** (present value), the hidden state **h** of the previous time step (past value) and a bias are combined and then limited by an activation function to determine the hidden state of the current time step.

$$h_t = tanh(W_x x_t + W_h h_{t-1} + b) \tag{2.1}$$

The weights **W** of the RNN are updated through a backpropagation in time (BPTT) algorithm.

RNNs can be used for one-to-one, one-to-many, many-to-one, and many-to-many predictions.

Advantages of RNN

Due to their shortterm memory RNNs can handle sequential data and identify patterns in the historical data. Moreover, RNNs are able to handle inputs of varying length.

Disadvantages of RNN

The RNN suffers from the vanishing gradient descent. In this, the gradients that are used to update the weights during backpropagation become very small. Multiplying weights with a gradient that is close to zero prevents the network from learning new weights. This stopping of learning results in the RNN forgetting what is seen in longer sequences. The problem of vanishing gradient descent increases the more layers the network has.

As the RNN only keeps recent information, the model has problems to consider observations which lie far in the past. The RNN, thus, tends to loose information over long sequences as it only stores the latest information. Hence, the RNN has only a short-term but not a long-term memory.

Moreover, as the RNN uses backpropagation in time to update weights, the network also suffers from exploding gradients and, if ReLu activation functions are used, from dead ReLu units. The first might lead to convergence issues while the latter might stop the learning.

2.1.2 LSTM

In order to solve the vanishing gradient descent problem of RNN, *GRU* and *LSTM* are proposed.

LSTMs [11] are a special type of RNNs which tackle the main problem of simple RNNs, the problem of vanishing gradients, i.e., the loss of information that lies further in the past.

The key to LSTMs is the cell state, which is passed from the input to the output of a cell. Thus, the cell state allows information to flow along the entire chain with only minor linear actions through three gates. Hence, the cell state represents the long-term memory of the LSTM. The three gates are called the forget gate, input gate, and ouput gate. These gates work as filters and control the flow of information and determine which information is kept or disregarded.

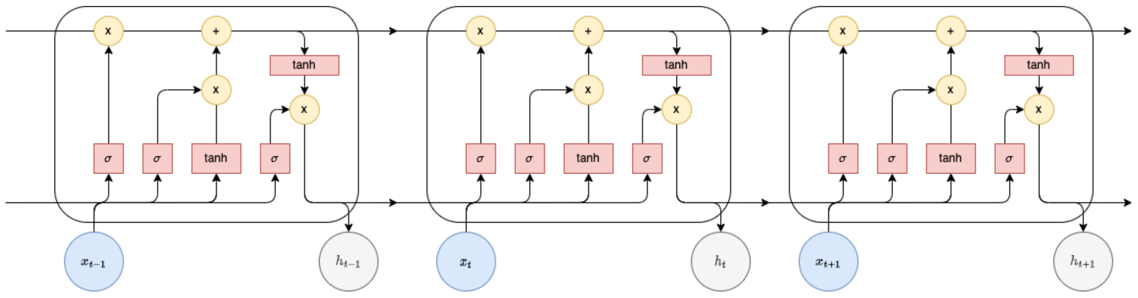


Figure 2.2: LSTM unfold.

The **forget gate** decides how much of the long-term memory shall be kept. For this, a sigmoid function is used which states the importance of the cell state. The output varies between 0 and 1 and states how much information is kept, i.e., 0, keep no information and 1, keep all information of the cell state. The output is determined by combining the current input x, the hidden state h of the previous time step, and a bias b.

$$f_t = \sigma(W_{f,x}x_t + W_{f,h}h_{t-1} + b_f) \tag{2.2}$$

The **input gate** decides which information shall be added to the cell state and thus the long-term memory. Here, a sigmoid layer decides which values are updated.

$$i_t = \sigma(W_{i,x}x_t + W_{i,h}h_{t-1} + b_i) \tag{2.3}$$

The **output gate** decides which parts of the cell state build the output. Hence, the output gate is responsible for the short-term memory.

$$o_t = \sigma(W_{o,x}x_t + W_{o,h}h_{t-1} + b_o) \tag{2.4}$$

As can be seen, all three gates are represented by the same function. Only the weights and biases differ. The cell state is updated through the forget gate and the input gate.

$$c_t = (f_t \circ c_{t-1} + i_t \circ tanh(h_{t-1})) \tag{2.5}$$

The first term in the above equation determines how much of the long-term memory is kept while the second terms adds new information to the cell state.

The hidden state of the current time step is then determined by the output gate and a tanh function which limits the cell state between -1 and 1.

$$h_t = o_t \circ tanh(c_t) \tag{2.6}$$

Advantages of LSTMs

The advantages of the LSTM are similar to RNNs with the main benefit being that they can capture patterns in the long-term and short-term of a sequence. Hence, they are the most used RNNs.

Disadvantages of LSTMs

Due to their more complex structure, LSTMs are computationally more expensive, leading to longer training times.

As the LSTM also uses the backpropagation in time algorithm to update the weights, the LSTM suffers from the disadvantages of the backpropagation (e.g., dead ReLu elements, exploding gradients).

2.1.3 GRU

Similar to LSTMs, the *GRU* [12] solves the vanishing gradient problem of simple RNNs. The difference to LSTMs, however, is that GRUs use fewer gates and do not have a separate internal memory, i.e., cell state. Hence, the GRU solely relies on the hidden state as a memory, leading to a simpler architecture.

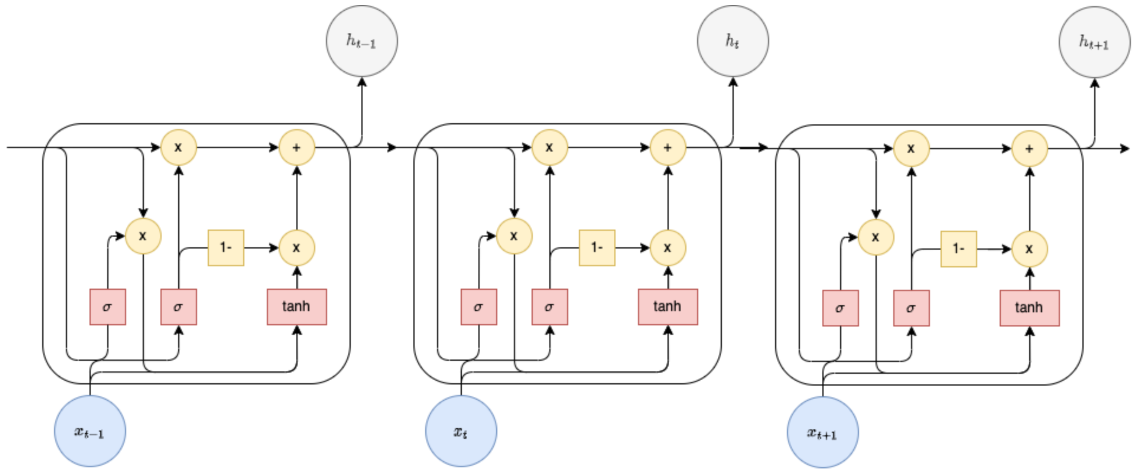


Figure 2.3: GRU unfold.

The **reset gate** is responsible for the short-term memory as it decides how much past information is kept and disregarded.

$$r_t = \sigma(W_{r,x}x_t + W_{r,h}h_{t-1} + b_r) \tag{2.7}$$

The values in the vector r are bounded between 0 and 1 by a sigmoid function and depend on the hidden state h of the previous time step and the current input x. Both are weighted using the weight matrices $\mathbf{W}$. Furthermore, a bias b is added.

The **update gate**, in contrast, is responsible for the long-term memory and is comparable to the LSTM's forget gate.

$$u_t = \sigma(W_{u,x}x_t + W_{u,h}h_{t-1} + b_u) \tag{2.8}$$

As we can see the only difference between the reset and update gate are the weights $\mathbf{W}$.

The hidden state of the current time step is determined based on a two step process. First, a candidate hidden state is determined. The candidate state is a combination of the current input and the hidden state of the previous time step and an activation function. In this example, a tanh function is used. The influence of the previous hidden state on the candidate hidden state is controlled by the reset gate

$$\hat{h}_t = tanh(W_{g,x}x_t + W_{g,h}(r_t \circ h_{t-1}) + b_g) \tag{2.9}$$

In the second step, the candidate hidden state is combined with the hidden state of the previous time step to generate the current hidden state. How the previous hidden state and the candidate hidden state are combined is determined by the update gate.

$$h_t = u_t \circ h_{t-1} + (1 - u_t) \circ \hat{h}_t \tag{2.10}$$

If the update gate gives a value of 0 then the previous hidden state is completly disregarded and the current hidden state is equal to the candidate hidden state. If the update gate gives a value of one, it is vice versa.

Advantages of GRU

Due to the simpler architecture compared to LSTMs (i.e., two instead of three gates and one state instead of two), GRUs are computationally more efficent and faster to train as they need less memory.

Moreover, GRUs haven proven to be more efficient for smaller sequences.

Disadvantages of GRU

As GRUs do not have a separate hidden and cell state they might not be able to consider observations as far into the past as the LSTM.

Similar to the RNN and LSTM, the GRU also might suffer from the disadvantages of the backpropagation in time to update the weights, i.e., dead ReLu elements, exploding gradients.

2.2 Sequence-to-sequence model

A *sequence-to-sequence* [13] model is a model that takes a sequence of items and outputs another sequence of items. In the case of *Neural Machine Translation* [14], the input is a series of words, and the output is the translated series of those words. The sequence-to-sequence utilizes *encoder-decoder architecture* [15] to encode the data and decode the information (Figure 2.4). Sequence-to-sequence models are best suited for tasks revolving around generating new sentences depending on a given input, such as *text summarization* [16], *text generation* [17] , or *question answering* [18].

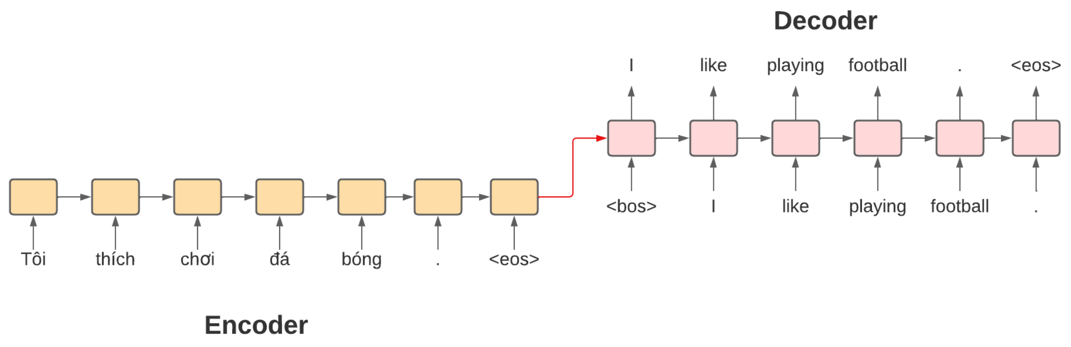


Figure 2.4: The overview of a sequence-to-sequence model of machine translation, from Vietnamese to English.

2.3 Teacher forcing

Teacher forcing [19] is a method for quickly and efficiently training *Recurrent Neural Network (RNN)* [20] models that use the ground truth from prior time steps as input. It is a network training method critical to the development of deep learning language models used in many applications. According to Figure 2.4, at the first time step $t = 0$, after the $<bos>$ token is fed into the RNN, a predicted token is generated. In the next time step $t + 1$, instead of feeding the previously generated token into the RNN, we feed the ground truth token at time step t from the training dataset, and it is the I token in this case. If the previous generated was, for example, token *We*, the token I will still be feed into the decoder. After that, we continue the previous steps until the end of the ground truth sentence. To conclude, teacher forcing works by using the actual or expected output from the training dataset at the current time step t as input in the next time step $t + 1$, rather than the output generated by the network. Teacher forcing is a fast and effective way to train an RNN as well as Seq2seq model that uses the output from prior time steps as input to the model. However, the approach can also result in models that are limited when the generated sequences vary from what has been previously seen by the model during the training phase, which is called exposure bias [21].

2.4 Beam search

Beam search [22] is a heuristic search algorithm that applies graph theory by expanding the chosen nodes in limited sets. Beam search constructs its search tree using breadth-first search. It only saves a limited number of the best states at each time step, called the *beam width*. After that, it will expand only those states. At the first time step, we select the k tokens, or states, with the highest predicted probabilities. Each of them will be the first token of k candidates output sequences, respectively. At each subsequent time step, based on the k candidate output sequences at the previous time step, we continue to select k candidate output sequences with the highest predicted probabilities from possible choices (Figure 2.5). Beam search can maintain tractability in large systems with insufficient amount of memory to store the entire search tree. For example, it has been used in many paraphrasing systems. To select the best words that convey the same meaning, each part is processed, and many different synonyms of the words appear. The top best synonyms according to their context and meaning are kept, and the rest are discarded. The model then evaluates the paraphrasing according to a given criterion, choosing the paraphrasing generation which best meets the criteria.

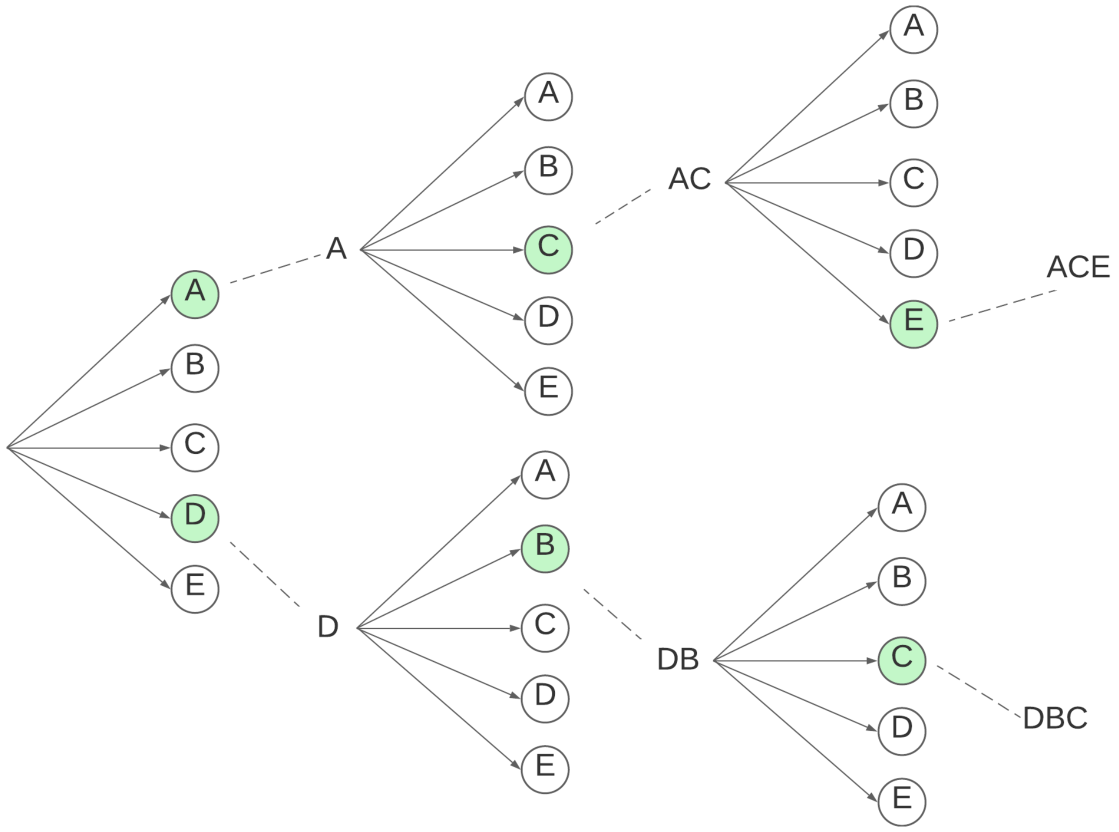


Figure 2.5: The process of beam search (beam size: 2, the maximum length of an output sequence: 3). The candidate output sequences are A, D, AC, DB, ACE and DBC.

For example, considering Figure 2.5, we initially set the beam width $k = 2$ and the max sequence length is 3. At the first time step $t = 1$, the top $k = 2$ highest probabilities are the A and D tokens. Continuing with these 2 tokens, we select the top 2 probabilities at the next time step $t = 2$, which are the C and B tokens. Last but not least, we choose the top 2 probabilities at the last time step $t = 3$ and finish with $k = 2$ sequences which are ACE and DBC.

2.5 Transformer

Transformer [1] is a deep learning model that utilizes the mechanism of *Self-attention*, differentially weighting the significance of each part of the input data. Like RNN, Transformer has Encoder and Decoder and it is used to process sequential input data. However, unlike RNNs, Transformers process the entire input in parallel, making it possible to scale the speed and capacity of sequential deep learning models. Moreover, Transformer adopts Attention [23] mechanisms that made it possible to keep track of the relations between words across long sequences in both forward and backward directions.

The Transformer Encoder is a stack of multiple layers (Figure 2.6). The first is a *Multi-head Attention* and the second is a *Position-wise Feed-forward network*. Specifically, in the Encoder Self-Attention, *Queries*, *Keys*, and *Values* are all from the outputs of the previous Encoder Layer. Inspired by the *ResNet* [24] architecture, a residual connection is used for both sublayers. This additional residual connection is immediately followed by *Layer Normalization* [25]. As a result, the Transformer Encoder outputs a vector representation for each position of the input sequence.

The Transformer Decoder is also a stack of multiple layers (Figure 2.6). Besides the two sublayers described in the Encoder, the Decoder inserts a third sublayer, known as the *Encoder-Decoder Attention*, or *Cross-Attention*, between these two. In the Encoder-Decoder Attention, Queries are from the outputs of the previous Decoder Layer, and the Keys and Values are from the Transformer Encoder outputs. In the Decoder Self-Attention, Queries, Keys, and Values are from the outputs of the previous Decoder Layer. However, each position in the Decoder is allowed to attend only to all positions in the Decoder up to that position, which is called *Masked Self-Attention*. This Masked Self-Attention preserves the autoregressive property, ensuring that the prediction only depends on those output tokens that have been generated, which can alleviate the exposure bias mentioned earlier in section 2.3.

In summary, Transformer is an instance of the Encoder-Decoder architecture, though either the Encoder or the Decoder can be used individually in practice. The transformer has had great success in NLP downstream tasks, such as machine translation. Many pre-trained models such as *GPT-2* [17], *GPT-3* [26], *BERT* [27], and *XLNet* [28] demonstrate the ability of Transformer to perform a wide variety of language processing tasks and have the potential for solving real-world problems.

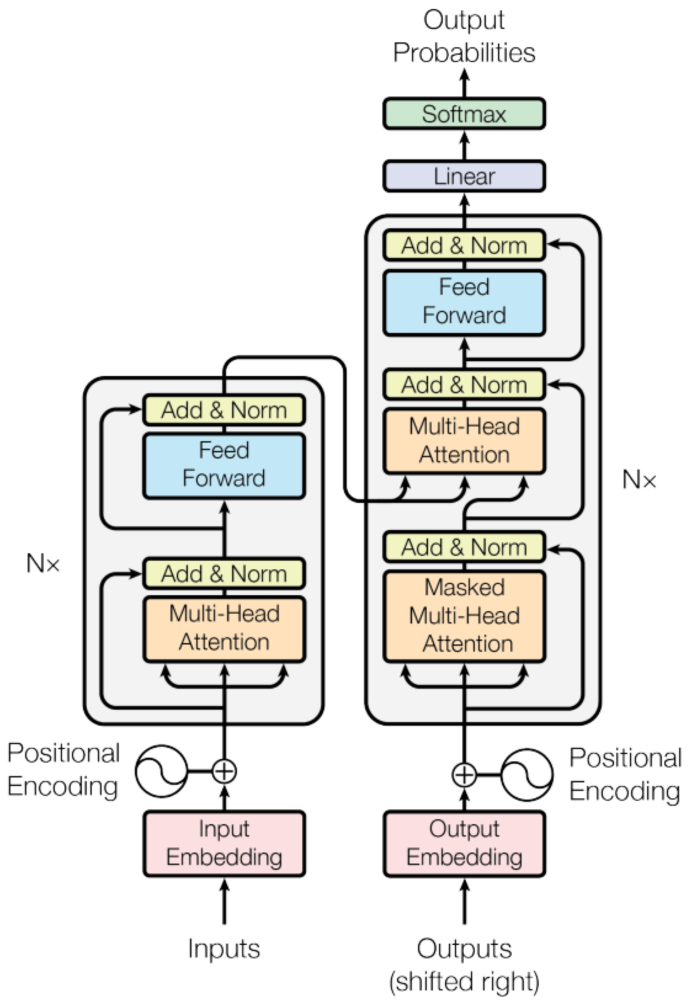


Figure 2.6: The visualization of Transformer architecture from [1].

Scaled Dot-Product Attention [1]: The input consists of queries and keys of dimension d_k, and values of dimension d_v. We compute the dot products of the query with all keys, divide each by d_k, and apply a softmax function to obtain the weights on the values. In practice, we compute the attention function on a set of queries simultaneously, packed together into a matrix Q. The keys and values are also packed together into matrices K and V. We compute the matrix of outputs as:

$$\text{Attention}\,(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2.11}$$

Multi-Head Attention [1]: Multi-head attention allows the model to jointly attend to information from different representation subspaces at different positions. With a single attention head, averaging inhibits this.

$$\begin{aligned} \text{MultiHead}(Q, K, V) &= \text{Concat}\left(\text{head}_1, \ldots, \text{head}_\text{h}\right)W^O \\ \text{where head} &= \text{Attention}\left(QW_i^Q, KW_i^K, VW_i^V\right) \end{aligned} \tag{2.12}$$

Where the projections are parameter matrices: $W_i^Q \in \mathbb{R}^{d_{model} \times d_k}$, $W_i^K \in \mathbb{R}^{d_\text{model} \times d_k}$, $W_i^V \in \mathbb{R}^{d_\text{model} \times d_v}$ and $W^O \in \mathbb{R}^{hd_v \times d_\text{model}}$
Due to the reduced dimension of each head, the total computational cost is similar to that of single-head attention with full dimensionality.

Position-wise Feed-Forward Networks [1]: In addition to attention sub-layers, each of the layers in our encoder and decoder contains a fully connected feed-forward network, which is applied to each position separately and identically. This consists of two linear transformations with a $ReLU(\cdot)$ activation in between.

$$\text{FFN}(x) = ReLU\left(xW_1 + b_1\right)W_2 + b_2 \tag{2.13}$$

While the linear transformations are the same across different positions, they use different parameters from layer to layer. Another way of describing this is as two convolutions with kernel size 1.

2.6 BERT: Bidirectional Encoder Representations from Transformer

BERT [27] leverages the attention model to get a deeper understanding of the language context. BERT is a stack of many encoder blocks. The input text is separated into tokens as in the transformer model, and each token will be transformed into a vector at the output of BERT.

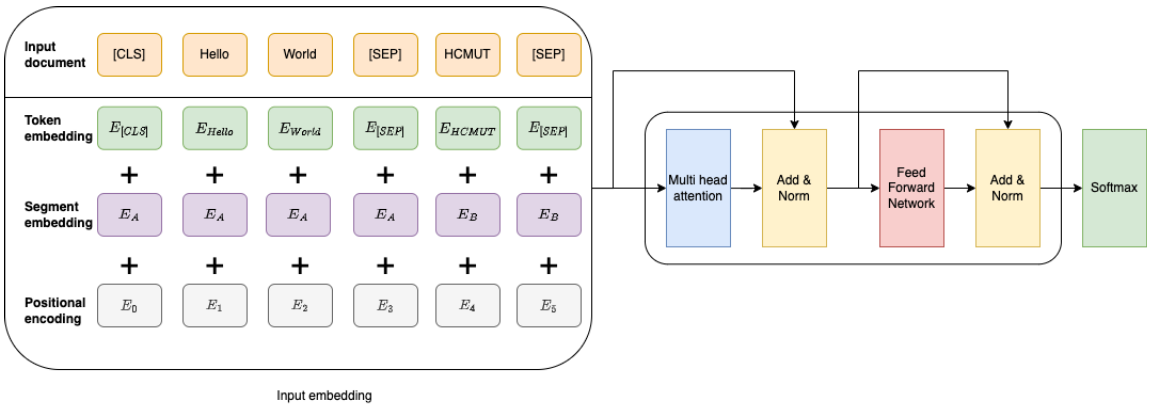


Figure 2.7: BERT architecture.

A BERT model is trained using the masked language model (MLM) and next sentence prediction (NSP) simultaneously.

Each training sample for BERT is a pair of sentences from a document. The two sentences can be consecutive in the document or not. There will be a [CLS] token prepended to the first sentence (to represent the class) and a [SEP] token appended to each sentence (as a separator). Then, the two sentences will be concatenated as a sequence of tokens to become a training sample. A small percentage of the tokens in the training sample is masked with a special token [MASK] or replaced with a random token.

Before it is fed into the BERT model, the tokens in the training sample will be transformed into embedding vectors, with the positional encodings added, and particular to BERT, with segment embeddings added as well to mark whether the token is from the first or the second sentence.

Each input token to the BERT model will produce one output vector. In a well-trained BERT model, we expect:

- output corresponding to the masked token can reveal what the original token was
- output corresponding to the [CLS] token at the beginning can reveal whether the two sentences are consecutive in the document

Then, the weights trained in the BERT model can understand the language context well.

Once having such a BERT model, it can be used for many downstream tasks. For example, by adding an appropriate classification layer on top of an encoder and feeding in only one sentence to the model instead of a pair, for example, the class token [CLS] can be taken as input for sentiment classification. It works because the output of the class token is trained to aggregate the attention for the entire input.

Another example is to take a question as the first sentence and the text (e.g., a paragraph) as the second sentence, then the output token from the second sentence can mark the position where the answer to the question rested. It works because the output of each token reveals some information about that token in the context of the entire input.

Chapter 3

Related Works

The art of paraphrasing assists many other tasks in NLP such as question answering [29], semantic parsing [30, 5], and data augmentation [31]. Recently, paraphrase generation tasks not only focus on improving the quality of the paraphrase but also the lexical diversity [32] and attribute-controlled paraphrase [33]. To the best of our knowledge, there has not been any specific work focusing on multilingual low-resource language paraphrase generation; therefore, in this section, we only point out the related works of the task paraphrase generation as well as the advantages and disadvantages of each approach.

The related works target paraphrase generation has been categorized and illustrated by the following figure.

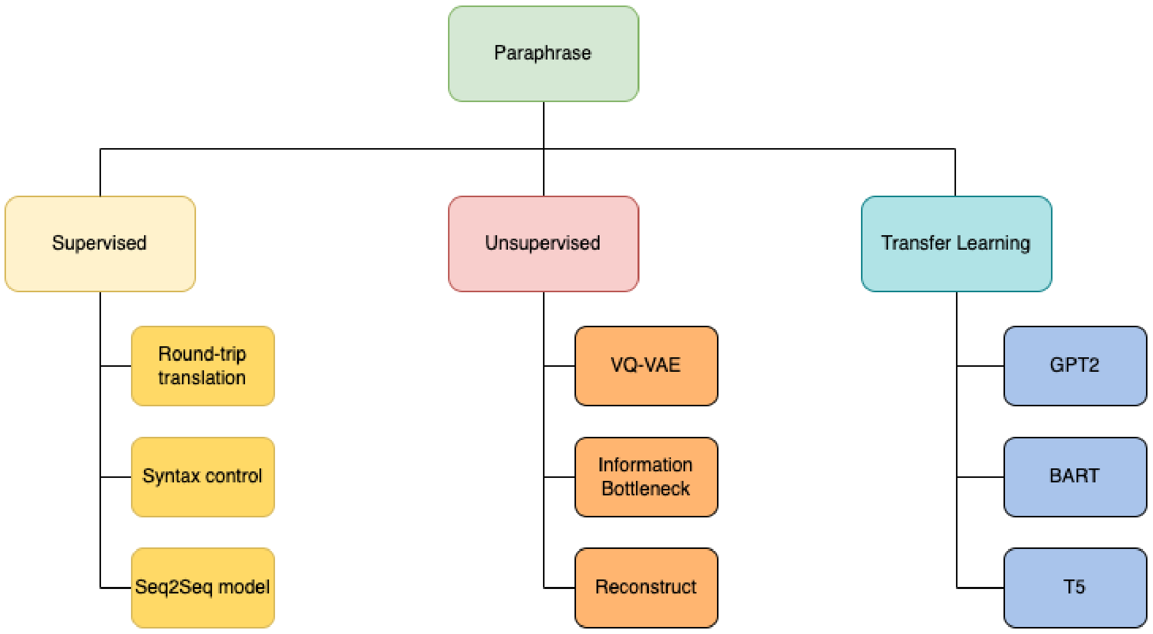


Figure 3.1: Taxonomy for the task paraphrase generation.

3.1 Supervised Approaches

Supervised approaches have been widely adopted to many tasks of NLP, even for paraphrase generation.

Neural sequence-to-sequence (Seq2Seq): To the best of our knowledge, Seq2Seq has been primarily used to tackle the task of paraphrase generation [34, 15, 35, 1]. Sometimes, such models are also used to evaluate the paraphrasing quality [36]. Besides, [37] involves using LSTM with a pair-wise discriminator to ensure that true paraphrase embeddings are close and unrelated paraphrase candidate embeddings are far.

Round-trip translation utilizes the strength of Neural Machine Translation (NMT) to translate a sentence from one language to another language and then translate it back to its original language. This method attracts much attention from the researchers since it preserves the semantic meaning through translation tasks but creates a diverse version of the original sentence using the knowledge of the NMT model [38, 39, 40, 41]. For most of the papers that used round-trip translation for paraphrase generation, the dataset ParaNMT has been the main source of training and creating the NMT models [42]. More recently, diverse paraphrasing has been addressed in order to have different sentences for each paraphrase generation [29].

Syntactic controlled paraphrase generation, in which the model will receive additive information such as the syntax of the designated sentence in order to guide it towards generating a sentence that matches the given syntax, has attracted many researchers [43, 44, 45]. However, most of these approaches require parallel data.

3.2 Unsupervised Approaches

Unsupervised paraphrasing, on the other hand, is a rather less explored and more challenging problem in NLP.

Information bottleneck theory: Ormazabal and his team [46] proposed a method that utilised the information bottleneck theory for training a paraphrasing model. They have trained a Seq2Seq model with a modification in the architecture leading to one encoder and two decoder model. Training on the dataset of translation English-French WMT14 [47] with adversarial training and information bottleneck method, together with multi-task learning objective(translation and reconstruction), the model can create paraphrase sentences due to the adversarial training and reconstruction objective while maintains the meaning of the input sentence due to the task of translation.

VQ-VAE: Roy and his team [48] proposed training a VQ-VAE [49] on monolingual dataset LM1B [50] using the nature of randomness of VAE to achieve different sentence outputs for one input sentence in addition to the discrete feature introduced by quantization of the VQ-VAE.

Reconstruction of the original sentence is also utilised as an unsupervised paraphrasing task, which is proposed by the team of Tong Niu [8]. In their work, the team trained a task-adapted model by having it reconstruct a corrupted sentence, which has been removed and shuffled some tokens, into the original sentence. By this method, the team observed that the model could generate a paraphrased version of the given input.

3.3 Transfer Learning

There have been few works exploring the art of using pre-trained models and/or transfer learning for paraphrasing such as:

BART & T5: Kajiwara and his team [51] proposed to pre-train a language such as BART [35] and T5 [15] to generate grammatically correct sentences with the same meaning as the input sentence after that, utilize the translation model to leverage the art of paraphrasing on a monolingual dataset in an unsupervised manner.

Pre-trained NMT: On the other hand, Brad and his colleagues [52] pre-trains an NMT model and then fine-tune it on the entailment generation in order to generate a sentence that conveys the same meaning as the original input.

GPT-2: Hedge and his team [53] take advantage of the tremendous knowledge encoded in a GPT-2 [17] model to fine-tune it on the task of reconstructing the original sentence, which will be later used as an unsupervised paraphrasing model.

3.4 Metrics for evaluating paraphrase

Although paraphrasing tasks have caught attention and have been researched by many researchers, paraphrasing evaluation tasks do not receive the same attention. Many of the paraphrase generation papers still use BLEU score [54] as their evaluation metrics despite its unsuitable characteristics for evaluating paraphrases. BLEU score is primarily used for Machine Translation tasks, it calculates the n-gram overlapping between two sentences. Therefore, the higher the lexical overlaps, the higher the BLEU score. However, paraphrasing is about creating sentences with different words or in other words, low lexical similarity. BERTScore [55] is a metric which uses BERT embedding to calculate the similarity in meaning between two sentences, however, BERTScore does not take into account the linguistic diversity aspect of paraphrasing, which makes it inappropriate for scoring. Niu et al. [8] proposed BERT-iBLEU which leverages both BERTScore for meaning convey and SelfBLEU (BLEU between prediction of model and the original input) for lexical diversity. ParaScore [56] also uses BERTScore for evaluating the meaning and Normalized Edit Distance (NED) to calculate the linguistic dissimilarity between original input and prediction. TER (Translation Edit Rate) [57] is also a metric that captures the semantic dissimilarity between two sequences.

3.5 Paremeter Efficient Fine-Tuning

Large Language Models (LLMs) based on the transformer architecture [1], like GPT [17], T5 [15], and BERT [27] have achieved state-of-the-art results in various Natural Language Processing (NLP) tasks. They have also started foraying into other domains, such as Computer Vision (CV) (*ViT* [58], *Stable Diffusion* [59], LayoutLM [60]) and Audio (Whisper [61], XLS-R [62]). The conventional paradigm is large-scale pre-training on generic web-scale data, followed by fine-tuning to downstream tasks. Fine-tuning these pre-trained LLMs on downstream datasets results in huge performance gains when compared to using the pre-trained LLMs out-of-the-box (zero-shot inference, for example).

However, as models get larger and larger, full fine-tuning becomes infeasible to train on consumer hardware. In addition, storing and deploying fine-tuned models independently for each downstream task becomes very expensive, because fine-tuned models are the same size as the original pre-trained model. *Parameter-Efficient Fine-tuning (PEFT)* approaches are meant to address both problems!

PEFT approaches only fine-tune a small number of (extra) model parameters while freezing most parameters of the pre-trained LLMs, thereby greatly decreasing the computational and storage costs. This also overcomes the issues of catastrophic forgetting, a behaviour observed during the full finetuning of LLMs. PEFT approaches have also shown to be better than fine-tuning in the low-data regimes and generalize better to out-of-domain scenarios.

It also helps in portability wherein users can tune models using PEFT methods to get tiny checkpoints worth a few MBs compared to the large checkpoints of full fine-tuning, e.g., bigscience/mt0-xxl takes up 40GB of storage and full fine-tuning will lead to 40GB checkpoints for each downstream dataset whereas using PEFT methods it would be just a few MBs for each downstream dataset all the while achieving comparable performance to full fine-tuning. The small trained weights from PEFT approaches are added on top of the pre-trained LLM. So the same LLM can be used for multiple tasks by adding small weights without having to replace the entire model.

In short, PEFT approaches enable you to get performance comparable to full fine-tuning while only having a small number of trainable parameters.

There are many methods that have been developed around PEFT. For example, PrefixTuning [63] fine-tuning a concatenation for each of the hidden states produced by the model. Adapter [64] adds the trainable parameter to each layer of the transformer and trains only this part while keeping the other's parameter frozen. Prompt Tuning [65] add trainable parameter directly prepend to the input embedding. LoRA

[66] replace each key and value matrix inside the self-attention mechanism with two smaller matrices and take the multiplication of them. Thus each matrix has a smaller dimension compared to the original matrix, which reduces the number of parameters training by a large margin. There are also works combining these PEFT methods or even pre-training. Tu et al. [67] pre-train the prompt of prompt tuning method by a set of tasks before fine-tuning these prompt tuning directly on the downstream task. Chowdhury et al. [68] explores the use of in-context learning together with learnable prompt, however, the team also boost the result even more by combining with the LoRA method.

3.6 Adversarial Training

Adversarial training (AT) is a technique utilized for developing robust neural networks. Adversarial training was originally proposed as a means to enhance the security of machine learning systems [69], especially for copyright detection [70], or safety critical systems like self-driving cars [71]. During the adversarial training process, the mini-batches of training samples are corrupted with adversarial perturbations, which are small alterations that cause misclassification. The corrupted samples are then employed to update network parameters until the resulting model learns to resist such attacks. People observe the opposite result for language models [72] [73], showing that adversarial training can improve both generalization and robustness. Zhu et al. [74] propose a novel adversarial training algorithm, called FreeLB (**Free L**arge-**B**atch), which adds adversarial perturbations to word embeddings and minimizes the resultant adversarial loss around input samples. The method leverages the proposed "free" training strategies [75] [76] to enrich the training data with diversified adversarial samples under different norm constraints at no extra cost than PGD-based (Projected Gradient Descent) adversarial training [77].

While computer vision researchers [77] [69] widely accepted that adversarial training negatively affects generalization, people have observed the opposite result for language models [78] [79], demonstrating that adversarial training can improve both robustness and generalization. Although pre-training neural language models on unlabeled text has proven effective, these models can still suffer catastrophic failures in adversarial scenarios. Aside from prior works in computer vision, most adversarial training solutions adopt non-targeted attacks, where the model prediction is not driven towards a specific incorrect label. In NLP, the state-of-the-art research in adversarial training tends to focus on making adversarial training less expensive (e.g., by reusing backward steps in FreeLB [74]) or regularizing rather than replacing the standard training objective (e.g., in virtual adversarial training (VAT)). Specifically, Miyato et al. [80] propose a general algorithm which regularizes the training objective by applying virtual adversarial objective function. As a result, the adversarial perturbation favors label smoothness in the embedding space, and is a hyperparameter that controls the trade-off between standard errors and robust errors. Miyato et al. [80] found that VAT is superior to conventional adversarial training, especially when labels might be noisy.

However, adversarial training is generally limited to task-specific fine-tuning, according to the recent survey of [81]. In this paper, we present the first comprehensive study on adversarial pre-training, and show that it can improve both generalization and robustness for a wide range of NLP tasks. In this downstream task, paraphrase generation, we mainly focus on the "generalization" of adversarial training. We show that adversarial training significantly improves performance of state-of-the-art models for many language generation tasks. As a consequence, we propose a unifying algorithm LAMPAT (**L**ow-rank **A**daptation for **M**ultilingual **P**araphrasing using **A**dversarial **T**raining), which modifies the standard training objective with an additional term where the inner maximization can be solved by running a number of projected gradient descent steps [77] and the outer loss is minimized as a usual cross-entropy loss. LAMPAT is generally applicable to pre-training and fine-tuning on top of any Transformer-based language models.

3.7 Multilingual Generative Language Models

Large Language Models (LLMs) have made a significant impact on AI research. These powerful, general models can take on a wide variety of new language tasks from a user's instructions. Recent studies report that autoregressive language models can successfully solve many NLP tasks via zero- and few-shot learning paradigms, which opens up new possibilities for using the pre-trained language models. A wide range of works is dedicated to expanding the possibilities and application range of language models, including such emerging areas as multilingual, multitask, and even multimodal extensions. Multilingualism allows testing the models in more challenging settings, such as zero-shot cross lingual transfer, in which task-specific supervision in a source language is used to fine-tune the model for evaluation in a target language. Several works have introduced multilingual versions of the transformer models which were initially designed for English.

For example, mBART [82], a multilingual sequence-to-sequence denoising auto-encoder. mBART is trained by applying BART architecture [35] to large-scale monolingual corpora across many languages. The input texts are noised by masking phrases and permuting sentences, and the model is learned to recover the texts. mGPT [83] introduces two autoregressive GPT-like models with 1.3 billion and 13 billion parameters trained on 60 languages from 25 language families using Wikipedia and Colossal Clean Crawled Corpus (C4 [15]). Shliazhko et al. [83] reproduce the GPT-3 architecture using GPT-2 sources and the sparse attention mechanism. Despite the differences in the architecture design and pre-training objectives, multilingual models, such as XLM-R [84], mBART [82], mT5 [85], Bloom [86], and mGPT [83], have pushed state-of-the-art results on many NLP tasks in multiple languages [87] [88]. In spite of the fact that the pre-train and fine-tune paradigm remains the dominant approach in the field, novel learning concepts and methods such as zero-shot, one-shot and few-shot techniques have been designed to balance the cost of pre-training extensive LMs and accelerate their acquisition of new skills with less supervision towards the development of general AI. GPT-2 and GPT-3 [89] models have achieved impressive performance with the few shot method on various downstream tasks given a tiny amount of supervision.

While this approach avoids additional training and expands the application of LMs, it still can lead to inconsistent results caused by the imbalance of classes in the provided supervision, shifts in frequency distributions, sub-optimal prompts, and even word order. However, this disadvantage can be overcome via a more optimal text interaction with the model [90]. Our work mostly focuses on zero-shot applications on paraphrase generation tasks, and it outperforms some of the languages in bigger monolingual models.

From pre-trained word embeddings [91] [92] to pre-trained contextualized representations [93] [94] and transformer based language models [95] [27], unsupervised representation learning has significantly improved the state of the art in natural language understanding. Most recently, Devlin et al. [27] and Conneau et al. [84] introduced mBERT and XLM, masked language models trained on multiple languages, without any cross-lingual supervision. Conneau et al. [84] further show strong improvements on unsupervised pre-training for sequence generation, they focus on the unsupervised learning of cross-lingual representations and their transfer to discriminative tasks. Conneau et al. [96] shows that monolingual BERT representations are similar across languages, explaining in part the natural emergence of multilinguality in bottleneck architectures.

The benefits of scaling language model pre-training by increasing the size of the model as well as the training data has been extensively studied in the literature. GPT [95] highlights the importance of scaling the amount of data and RoBERTa [97] shows that training BERT longer on more data leads to significant boost in performance. Inspired by this, we chose to use mGPT as our multilingual paraphrase generation model and we show that mGPT are undertuned with our downstream task, and that some improvements in the learning procedure of unsupervised generative models lead to much better performance.

3.8 Discussion

Each approach of the task paraphrase generation has its own pros and cons. The supervised method (i) creates high-quality paraphrased sentences (ii) reduces the amount of work by using the existing NMT model and (iii) implied special controls on the output but this method (i) requires many data samples and (ii) only apply to languages that have the paraphrase corpus. On the other hand, the unsupervised method can be applied to any language with the monolingual dataset, which is very suitable for LRLs but this method suffers from (i) low-quality paraphrased sentences and (ii) incomprehensible sentence generation. For transfer learning, although it has (i) huge knowledge and (ii) the ability to learn downstream tasks by fine-tuning, it still required a standard or high-quality dataset in order to generalize well on the given task and give a comprehensive qualified paraphrased sentence.

Recall that our task is to perform paraphrase generation on multilingual settings. This means that the parallel paraphrase corpus does not exist in general. For this particular reason, we choose to implement the combination of both an unsupervised approach and transfer learning. The unsupervised approach will tackle the problem of scarcity of the dataset by dealing with only monolingual datasets (which could be found for any languages). Meanwhile, the pre-trained language model will use its knowledge to create a comprehensive sentence, which is the primary source of the issues that create low-quality sentences in unsupervised settings. By combining these two approaches, we can create a high-quality paraphrased sentence of any languages.

Chapter 4

Proposed solution

As we have mentioned in Chapter 3, each of the previous works has shown their great contributions and also their limitation in the proposed method. Take supervised-approached works for an example, it is clear that although the supervised method produces high-quality paraphrasing in terms of semantic consistency and lexical diversity, the method still requires a parallel dataset of paraphrasing sentences. This is a critical issue since the parallel dataset of non-English languages, such as Vietnamese, Thailand, etc does not exist in a large amount. Due to this problem, the unsupervised methods, which do not require parallel datasets but only the monolingual dataset and the similarity of different NLP tasks such as translation and summarization, have been chosen to study in order to overcome the scarcity of the non-English paraphrase datasets. However, unsupervised methods fail to have the same paraphrasing quality as the supervised ones. To overcome this quality issue and scarcity of datasets, [2] have proposed a new decoding algorithm which is trained in an unsupervised manner but takes the advantage of the rich knowledge of pre-trained language models through transfer learning. Based on the amazing results reported by the team, we start to investigate the method in order to figure out the key recipe of their success and use those special ingredients to find a new approach for the current task.

4.1 Baseline method

4.1.1 Monolingual Paraphrasing

The team of Niu et al. [8] proposed an unsupervised paraphrasing task called reconstruction of the original sentence. They first trained a task-adapted model to reconstruct a corrupted sentence, where some tokens were removed and shuffled, back into the original sentence. After that, use the task-adapted model to generate output from the uncorrupted input with a special decoding algorithm namely Dynamic Blocking. This algorithms build up a dictionary of dynamic blocking tokens in order to block sequential token from happening. However, dynamic blocking algorithm could make the model to output non-sense tokens; therefore, a beam search is applied to tackle the problem of comprehensive output. From the dataset of uncorrupted input, the model will generate a synthetic dataset, in which each sample is the uncorrupted input and the output. From then, the synthetic dataset is used to fine-tune the pre-trained LM in order to create the unsupervised paraphrasing model.

Hedge and his team [53] leverage the extensive knowledge encoded in a GPT-2 model, fine-tuning it on the task of reconstructing the original sentence, which serves as an unsupervised paraphrasing model. From the input, the authors first remove the stop words and then shuffle the tokens. From then, the authors fine-tune GPT-2 model to generate original sentence from the corrupted input. With shuffling and removing words, the model is expected to produce new word or synonym and new sentence structure.

The majority of the above methods undergo training using datasets such as Quora Question Pair (QQP) [1] and ParaNMT [42]. QQP consists of 140k question pairs labeled as duplicates and 640k non-parallel questions. The dev and test sets have sizes of 3k and 20k, respectively. On the other hand, the ParaNMT dataset is created by back-translating sentences from Czech in the CzEng dataset.

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

To make a fair comparison, we thus train the above mentioned models on the same dataset, which is the WMT19 [98], which is just released in 2023. The number of samples are approximately 930k sentences.

4.1.2 Multilingual Paraphrasing

To our best knowledge, Thompson and Post's model [2, 36] is one of the strongest SOTA models for the multilingual paraphrasing task. To do this, the authors have trained a multilingual machine translation model namely Prism on a set of seven bilingual datasets. WikiMatrix, Global Voices [2], EuroParl, SETimes [3], United Nations, WMT Kazakh-English (1M sentence pairs) and Kazakh-Russian (200k sentence pairs) to evaluate Kazakh.

The MNMT model is based on the Transformer architecture [1]. To obtain the best result, the authors use the large version of Transformer with 8 encoder layers and 8 decoder layers since large models train more efficiently [99, 100]. To further help the model obtain a unified representation of multilingualism, for each source sentence, the language token will be prepended to it before feeding into the encoder and the same mechanism is applied to the decoder as well. This forces the Transformer encoder to produce the vector representation that can be translated into any other language, thus, making the encoder a universal language encoder.

Back to the problem of paraphrasing, the authors notice that by inputting the same language token into both the encoder and decoder, they could obtain self-translation, or in other words, a paraphrasing version of the input sequence. However, the output produced by the decoder is mostly identical to the input of the encoder, thus, resulting in a low lexical diversity paraphrasing. To prevent this, the authors proposed a new blocking algorithm in order to lower the probability that the model will produce

[2]`http://casmacat.eu/corpus/global-voices.html`
[3]`http://nlp.ffzg.hr/resources/corpora/setimes/`

identical results. The pseudocode of the algorithm is illustrated below, Figure 4.1:

```
def buildPenalties(source):
  penalties = defaultdict(list)
  for n in [1, 2, 3, 4]:
    for ngram of size n in subwords2words(source):
      prefix, word = ngram[0:-1], ngram[-1]
      for subword in targetVocab:
        if word.lower().startsWith(subword.lower()):
          penalties[prefix].append(subword)
  return penalties

def penalize(history, penalties, targetLogProbs):
  for n in [1, 2, 3, 4]:
    prefix = subwords2words(history)[-(n-1):]
    for subword in penalties[prefix]:
      targetLogProbs[id(subword)] -= alpha * (n ** beta)
```

Figure 4.1: Pseudocode of the algorithm to generate diverse paraphrasing from [2].

The algorithm will check if the current target subword vocabulary begins the last word of any sequence inside the n-grams list created from the input, then the probability will be reduced by a factor of αn^{β} where α, β is the hyperparameter and n is the n in n-grams that block the target subword.

By this reduction in probability, the authors could prevent the model from outputting the same word, thus increasing the chance that the synonym of the word will be outputted, making a highly lexical diversity paraphrasing.

4.2 Our proposed method: LAMPAT

We use transformer based models [1] to leverage a multi-head attention mechanism, which have demonstrated superiority in parallel computation and modeling long-range dependencies, compared to recurrent neural networks such as LSTM [11].

The method chosen for parameter-efficient fine-tuning is Low-rank Adaptation (LoRA)

[66]. Typically, normal task adaption is performed through fully fine-tuning, which involves updating all the parameters of the pre-trained model. However, a major drawback of fine-tuning is that the resulting model contains the same number of parameters as the original model. LoRA overcomes this by indirectly training some dense layers in a neural network through optimizing rank decomposition matrices of the dense layers' changes during adaptation, while keeping the pre-trained weights frozen. This also allows for efficient task-switching by simply replacing the matrices, resulting in reduced storage requirements and task-switching overhead. LoRA enables more efficient training and lowers the hardware barrier when using adaptive optimizers because gradients need only be calculated and optimizer states maintained for a smaller set of parameters, namely the low-rank matrices. In this work, the attention weights [1] or downstream tasks are adapted while the MLP modules are frozen to achieve simplicity and parameter-efficiency.

Consider a standard classification task with an underlying data distribution D over examples $x \in \mathbb{R}^d$ and corresponding labels y. Assume that we are given a suitable loss function $L(f(x,\theta),y)$, for example the cross-entropy loss for a neural network. As usual, $\theta \in \mathbb{R}^p$ is the set of model parameters. Our goal then is to find model parameters θ that satisfies:

$$\min_{\theta} E_{(x,y)\sim D}[L(f(x,\theta),y)] \tag{4.1}$$

Empirical risk minimization (ERM) has been tremendously successful as a way to find classifiers with small population risk. Unfortunately, ERM often does not yield models that are robust to adversarial examples. Formally, there are efficient algorithms that take an example x belonging to class c_1 as input and find examples x_{adv} such that x_{adv} is very close to x but the model incorrectly classifies x_{adv} as belonging to class $c_2 \neq c_1$. This is our motivation of creating a generalized and robust paraphrase generation model.

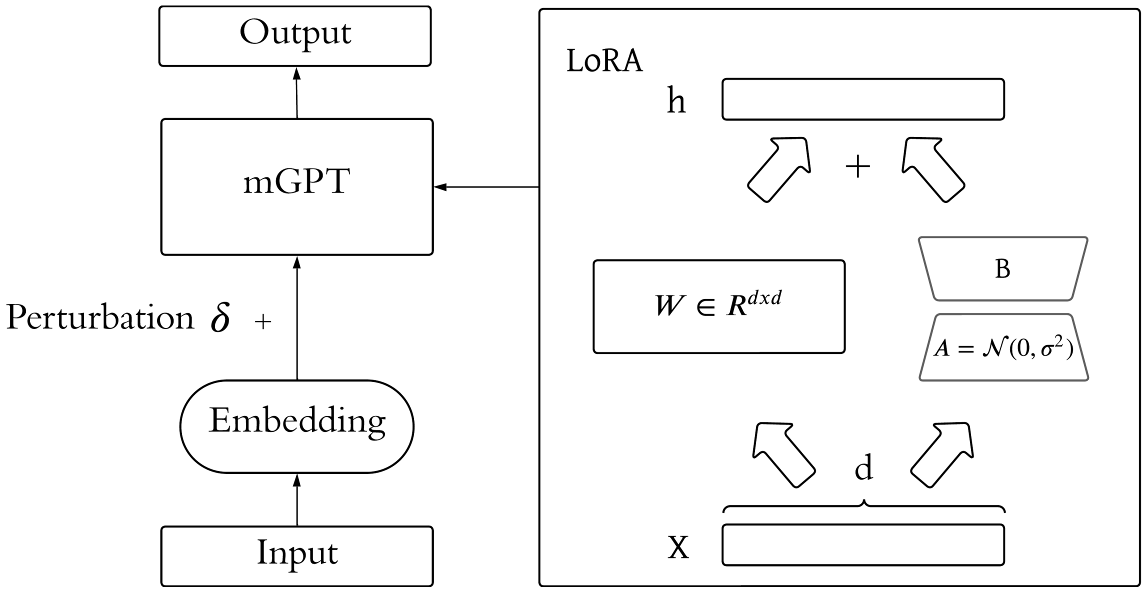


Figure 4.2: The general architecture of our proposed method: LAMPAT.

For each data point x, we introduce a set of allowed perturbations $\mathcal{S} \subseteq \mathbb{R}^d$ that formalizes the manipulative power of the adversary. We modify the definition of population risk $E_D[L]$ by incorporating the adversary. Instead of feeding samples from the distribution D directly into the loss L, we allow the adversary δ to perturb the input first. Specifically, x is the sub-word embedding in $f(x, \theta)$. We observe that by perturbing the embedding space $x_{adv} = x + \delta$, rather than the input space, adversarial training in NLP might inadvertently bias toward on-manifold perturbation than regular perturbation, which helps generalization. Therefore, we apply perturbation to the embedding space, as in 4.2. Full label information is not available at all times. In fact, we want to output other labels than the label y. Therefore, we take the strategy to replace label y with its current approximation, $f(x, \theta)$. This approximation is not necessarily naive, because $f(x, \theta)$ shall be close to y when the number of labeled training samples is large. This is also the motivation behind inclusion of the term "virtual" in [80]. Literally, we use "virtual" labels that are probabilistically generated from $f(x, \theta)$ in place of paraphrasing labels, and compute adversarial direction based on the virtual labels. Therefore, in this study, we use the current estimate $f(x, \theta)$ in place of y. We replace the adversarial training objective Equation 4.2 in [74]:

$$\min_{\theta} E_{(x,y)\sim\mathcal{D}} \left[\max_{\delta} L\left(f_{\theta}(x+\delta), y\right)\right] \tag{4.2}$$

Instead, we follow Miyato et al. [80] to regularize the standard objective using virtual adversarial objective function:

$$\min_{\theta} E_{(x,y)\sim D} \left[\max_{\delta} [L(f(x+\delta,\theta), y) + \alpha L(f(x+\delta,\theta), f(x,\theta))]\right] \tag{4.3}$$

We use the Frobenius Norm to normalize the gradient of δ as in [74]. Besides, Madry et al. [77] demonstrated that this saddle-point problem can be solved reliably with Projected Gradient Descent (PGD), a standard method for large-scale constrained optimization, for the inner maximization. In particular, for the constraint $||\delta||_F \leq \epsilon$, with an additional assumption that the loss function is locally linear, PGD takes the next step (with step size η) in each iteration as in [74].

$$\delta \leftarrow \prod_{||\delta|| \leq \epsilon} (\delta + \eta \frac{g_{adv}}{||g_{adv}||_F}) \tag{4.4}$$

However, according to Dimitri Panteli Bertsekas [101] and Rebentrost et al. [102], although Gradient Descent algorithms locate local minima by following the path of the steepest descent, Newton's method incorporates curvature information, leading to improved convergence in many cases. Newton's method assumes that the loss is twice differentiable and uses the approximation with Hessian (second-order Taylor approximation). More specifically, the Gradient Descent's step involves descending along the linear estimate of the function, while the Newton's step involves moving the point towards the minimum of the parabola that approximates the function. Nevertheless, divergence can happen when the function is flat or almost flat in some dimension. In this case, the second derivatives are close to zero, which means their inverse becomes very large and gigantic steps will be applied in the next update. In our work, the objective function is quadratic and complex in many dimensions with respect to many languages' embedding space. Moreover, our objective function also includes the virtual adversarial labels optimization. Therefore, we choose to use the Projected-Newton Method (PNM) as a suitable approach for constraining δ with respect to ϵ in inner maximization:

$$\delta \leftarrow \prod_{H}(\delta + \eta[H]^{-1}\frac{g(\delta)}{||g(\delta)||_F}) \tag{4.5}$$

where $g(\delta) \leftarrow \nabla_\delta L(f(x+\delta,\theta), f(x,\theta))$ is the gradient of the loss with respect to δ, and $\prod_H$ performs a Newton projection step under different Hessian-defined norm in each K ascent steps, with constraint ϵ. To achieve good performance on generalization, the K-step PNM requires K forward and backward passes through the network. In the inner ascent steps of PNM, the gradients of the parameters can be obtained with almost no overhead when computing the gradients of the inputs. Shafahi et al. [75] and Zhang et al. [76] proprosed a way to accelerate adversarial training. However, we inherits the solution from FreeLB [74], which also derives good techniques from YOPO [76]. We firstly performs multiple PNM iterations, and then simultaneously accumulates the "free" parameter with gradients $\nabla_\theta L$ in each iteration. To be more specific, our training method could have a significant improvement over natural training. We acquires the K ascent steps from [74, 76], as mentioned earlier. Our method can reduce the computational cost of adversarial training using projected over constraints algorithms. It achieves similar levels while conducting fewer sweeps of forward and backward propagation, making it faster and less computationally expensive. Our method takes advantage of every propagation to update weights and allows multiple updates per iteration, potentially leading to faster convergence. By combining these factors, the K ascent steps significantly accelerates standard adversarial training. After that, the model's parameter θ is updated all at once with the accumulated gradients. By taking a descent step along the K gradients at $X + \delta_0, ..., X + \delta_{K-1}$, we can approximately calculate the following objective function as in [74]:

$$\min_\theta E_{x,y\sim D}\left[\frac{1}{K}\sum_{t=0}^{K-1}\left[\max_\delta\left[L(f(x+\delta,\theta),y) + \alpha L(f(x+\delta,\theta), f(x,\theta))\right]\right]\right] \tag{4.6}$$

The reason why $g(\delta)$ is computed with respect to $\nabla_\delta L(f(x+\delta,\theta), f(x,\theta))$, not $\nabla_\delta L(f(x+\delta,\theta), y)$, is because the model is trained in unsupervised manner and we want to steer the model towards the virtual labels instead of reconstructing the original sentence. Intuitively, our method could be a learning method with lower generalization error than Zhu et al. [74]. Equation 4.6 is equivalent to replacing the original batch x with a K-times larger virtual batch, consisting of samples whose embeddings are $X+\delta_0, ..., X+\delta_{K-1}$. While the original adversarial training Equation 4.2 minimizes the maximum risk at a single estimated point in the vicinity of each training sample, our method minimizes the maximum risk at each ascent step and steers the model towards the virtual labels at almost no overhead.

Algorithm 1 Low-rank Adaptation Multilingual Paraphrasing using Adversarial Training.

Input: X: Training samples, $f(x;\theta)$: the machine learning model parametrized by θ, σ^2: the variance of the random initialization of the perturbation δ, ϵ: perturbation bound, H: Hessian gradient matrix of the perturbation δ, τ: the global learning rate, α: the smoothing proportion of adversarial training in the augmented learning objective, η: ascent step size, N: the number of epochs, K: the number of ascent steps.

```
1 for epoch = 1...N do
2     for minibatch $B \in X$ do
3         $\delta \sim \frac{1}{\sqrt{N_\delta}}\mathcal{N}(0, \sigma^2 I)$
          for $m$ = 1...K do
4             Accumulate gradient of parameter $\theta$:
              $g_m \leftarrow g_{m-1} + \frac{1}{K}E_{(x,y)\in B}\left[\nabla_\theta L(f(x+\delta,\theta),y) + \alpha\nabla_\theta L(f(x+\delta,\theta), f(x,\theta))\right]$
              Update the perturbation $\delta$ through gradient ascent:
              $g_{adv} \leftarrow \nabla_\delta L(f(x+\delta,\theta), f(x,\theta))$
              if epoch < (N-1) then
5                 Projected Gradient Descent:
                  $\delta \leftarrow \prod_{||\delta||\leq\epsilon}(\delta + \eta\frac{g_{adv}}{||g_{adv}||_F})$
6             else
7                 Projection under Hessian matrix (Projected-Newton Method):
                  $\delta \leftarrow \prod_H(\delta + \eta[H]^{-1}\frac{g_{adv}}{||g_{adv}||_F})$
8         Update the parameter $\theta$ through gradient descent:
          $\theta \leftarrow \theta - \tau g_K$
```

Output: θ

Algorithm 1 shows the details of LAMPAT. Line 4 run K projected gradient steps to find the perturbation that maximizes the adversarial loss (local smoothness). A larger K leads to better approximation [77], however; it will be more computationally expensive. To achieve a good trade-off between speed and performance, we set $K = 2$

in our experiments. Besides, to achieve a proper trade-off smoothness between the virtual and true labels, we set the empirical smoothness rate $\alpha = 10$.

4.3 Experiment setup

4.3.1 Dataset preparation

To assess the fine-tuning, we choose to use the monolingual version of WMT19 [98] to train the model. This dataset covers a wide range of 15 languages including Arabic, Czech, German, English, Spanish, French, Hindi, Indonesian, Italian, Japanese, Kazakh, Dutch, Portuguese, Russian, Chinese. The WMT19 dataset we use is in its latest version, which is just released in 2023. However, the WMT19 dataset's version we use in the evaluation phase 4.4 is released in 2019, which is the same with [2]. In addition, our chosen model, mGPT [83], has been trained on 61 languages. Consequently, our model can work on a wide range of languages, increasing the multilingual diversity. To ease the distribution between resources of the languages, we take the uniform distribution to sample sentence, result in a large corpus of nearly 600k sentences for training set and approximately 100k sentences for validation set.

We take an unsupervised approach towards paraphrase generation and hence no parallel corpora for paraphrasing was utilized. On the other hand, we create a dataset for sentence reconstruction tasks. We corrupt the input sentence by removing all the stop words, further we randomly shuffle the tokens. We call the corrupted sentence as source sequence S and the original uncorrupted sequence as target T. Goal is to reconstruct the sentence from the keywords, or corrupted sentence.

Our stopwords originated from multilingual dictionaries [103]. After removing the stop words from sentences to form Source S, we force the model to generate diverse paraphrases by shuffling the words 33% of the time. To encourage the model to have new words in the reconstructed sentence that were not in the original sentence, we apply adversarial training as introduced in Section 4.2.

For the monolingual baseline models, for work of Niu et al. [8] and Hedge et al. [53], we train the model on sentence reconstruction objectives following the training procedures of each works.

For the multilingual baseline models, we use the bi-text of WMT19 [98] and train the Multilingual Neural Machine Translation model. For the decoding algorithm, we follows the settings of Thompson and Post [2].

4.3.2 Implementation details

We fine-tune mGPT using WMT19 [98]. The training code is implemented in PyTorch [104]. We use Adam optimizer [105] for optimizing with a standard linear decay learning rate scheduler. The training phase is conducted in 10 epochs, with a batch size of 2, and the max sequence length of a sample is 256 tokens. We set the perturbation bound $\epsilon = 10^{-5}$, the step size $\eta = 10^{-5}$, and the variance for initializing the perturbation $\sigma = 1$. We set the empirical smoothness rate $\alpha = 10$ for regularization in virtual adversarial training, and set $K = 2$ for computational efficiency. The training takes just 145 hours days on one Quadro RTX 6000. The reason is that we apply a parameter-efficient method to our approach, instead of fine-tuning the entire model. In conclusion, we have developed and set up an adversarial training approach, LAMPAT, to improve natural language generation, especially paraphrase generation. Our proposed approach adds perturbations to continuous word embeddings and utilizes the adversarial training method, thus mitigating the resultant risk in an efficient way. Empirical study shows that our method results in better generalization ability in paraphrase generation than previous works on both conventional and adversarial training.

4.4 Evaluation dataset

In order to compare our method and the baseline performance, we construct the evaluation dataset. To our best knowledge, this is the first evaluation dataset for multilingual paraphrasing tasks covering a wide range of languages including the Indo-European family to the Sino-Tibetan family.

Our evaluation dataset has two types to assess different aspects of the model: **Input-only** and **Input and Reference**. The detailed statistics of these datasets can be found in Appendix C

4.4.1 Input-only evaluation dataset

This provides only the input for the model without any ground-truth label. This type aims to test the ability of the model of producing the lexical diversity output given the input.

To construct this **Input-only** test set, we crawl the validation set from the WMT19 dataset, which is released in 2019 and available on HuggingFace [4]. Since the WMT19 dataset is used for the task Machine Translation, we thus, extract one side of the dataset in order to build the **Input-only** evaluation dataset.

The languages we extracted from the WMT19 are Czech (cs), German (de), English (en), Finish (fi), French (fr) and Chinese (zh). Most of the samples in our **Input-only** evaluation dataset are of the news domains, which is used to test the model's ability on producing a format paraphrase that conveys the same meaning.

[4]`https://huggingface.co/datasets/wmt19`

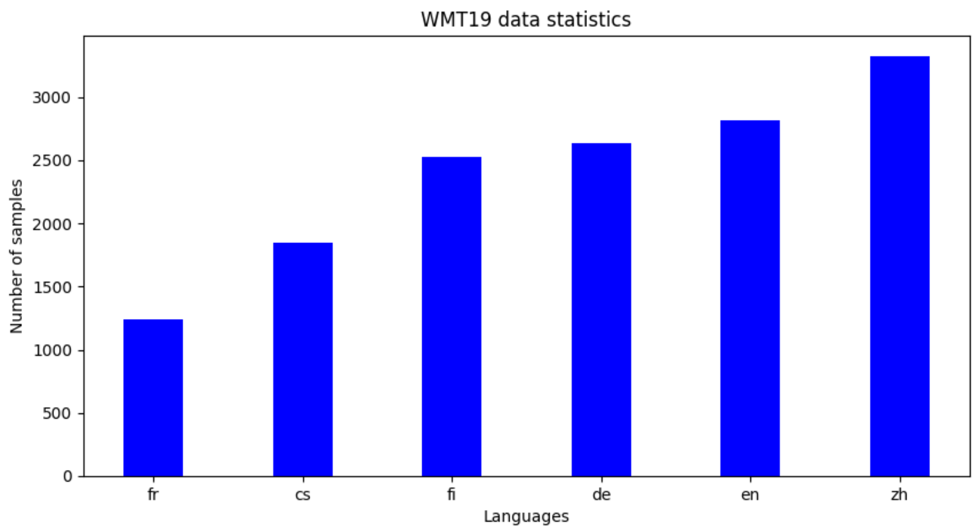


Figure 4.3: WMT19 data statistics on the number of samples per languages.

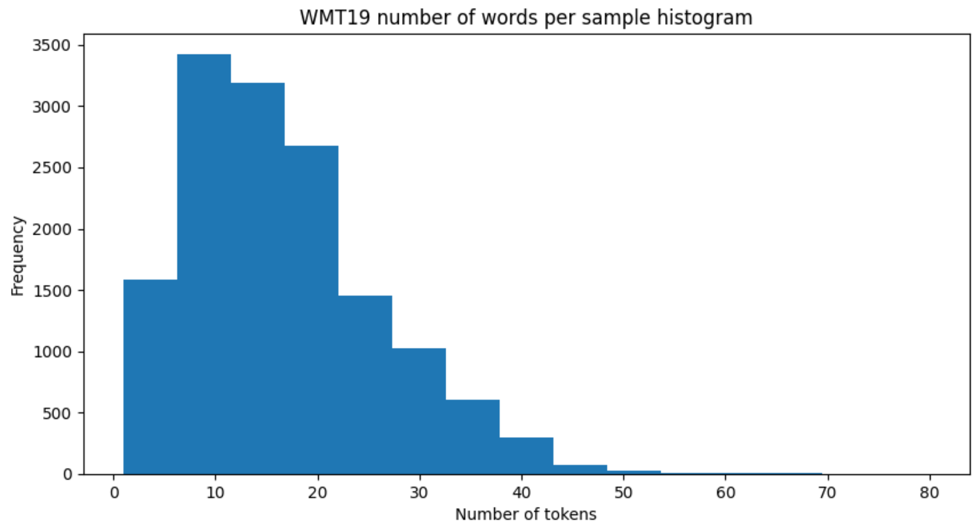


Figure 4.4: Histogram of the number of tokens of WMT19 evaluation dataset.

4.4.2 Input and reference

To increase the diversity, we also consider other dataset from different domains. In detail, we extract from three datasets with ground-truth reference: **Opusparcus**, **STAPLE**, and **PAWS-X**.

Opusparcus [106] is a paraphrase corpus for six European languages: German, English, Finnish, French, Russian, and Swedish. It consists of pairs of sentences in the same language that mean approximately the same thing. The paraphrases are extracted from the OpenSubtitles2016 corpus, which contains subtitles from movies and TV shows. The Opusparcus data have been partitioned into three types of data sets: training, development, and test sets. The training sets are large, consisting of millions of sentence pairs, and have been compiled automatically, with the help of probabilistic ranking functions. The development and test sets consist of sentence pairs that have been checked manually; each set contains approximately 1000 sentence pairs that have been verified to be acceptable paraphrases by two annotators. The score for each pair inside the Opusparcus test set ranges from 1 (not a good paraphrase) to 4 (very good paraphrase) with the step of 0.5 since there are two annotators and the score is taken as an average of them. We extract the test set of Opusparcus with a score of 4 in order to make a high-quality evaluation dataset.

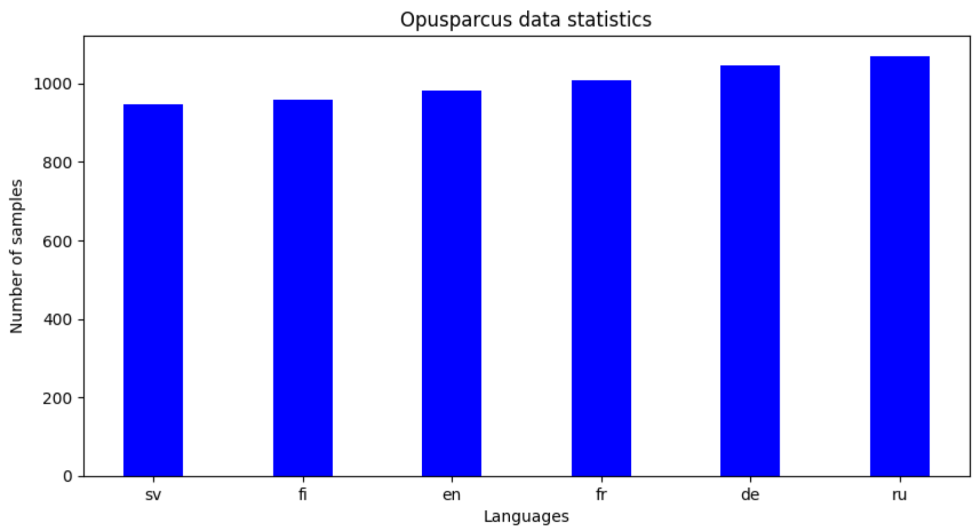


Figure 4.5: Opusparcus data statistics on the number of samples per language.

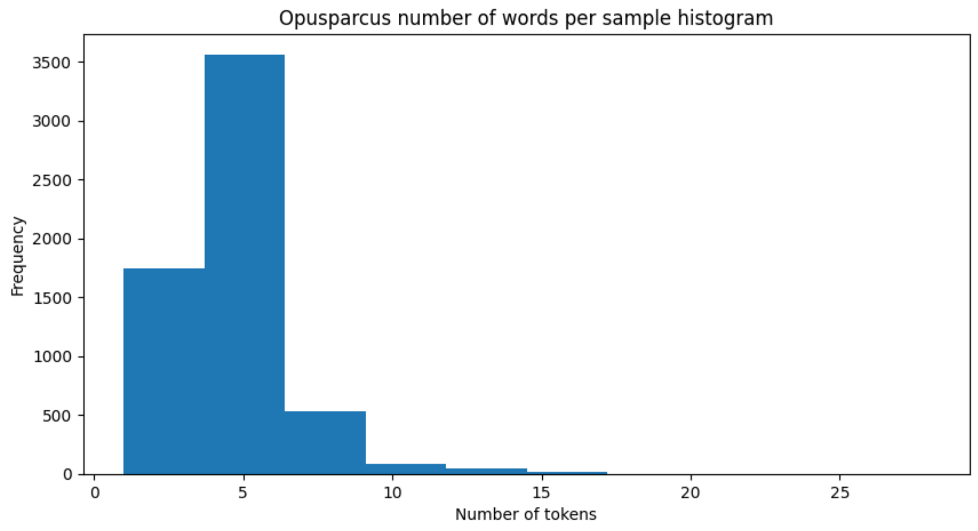


Figure 4.6: Histograms on the number of tokens of Opusparcus evaluation dataset.

PAWS-X [107] is a dataset of human-translated evaluation pairs in six different languages (French, Spanish, German, Chinese, Japanese, and Korean) that are designed to test the ability of machine learning models to identify paraphrases. It is an extension of the original PAWS dataset [108], which focused only on English. The goal of PAWS-X is to evaluate the multilingual adaptability of models and their ability to capture structure and context in a multilingual setting. The paper provides baseline numbers for three models with different capacities to capture non-local context and sentence structure, and using different multilingual training and evaluation regimes. The best-performing model is a multilingual BERT model fine-tuned on PAWS English plus machine-translated data. We extract only the test set of PAWS-X with label 1 (indicating paraphrase) for all the languages. However, Korean language has not been acknowledged by the baseline model. Therefore, we evaluate only LAMPAT model for Korean.

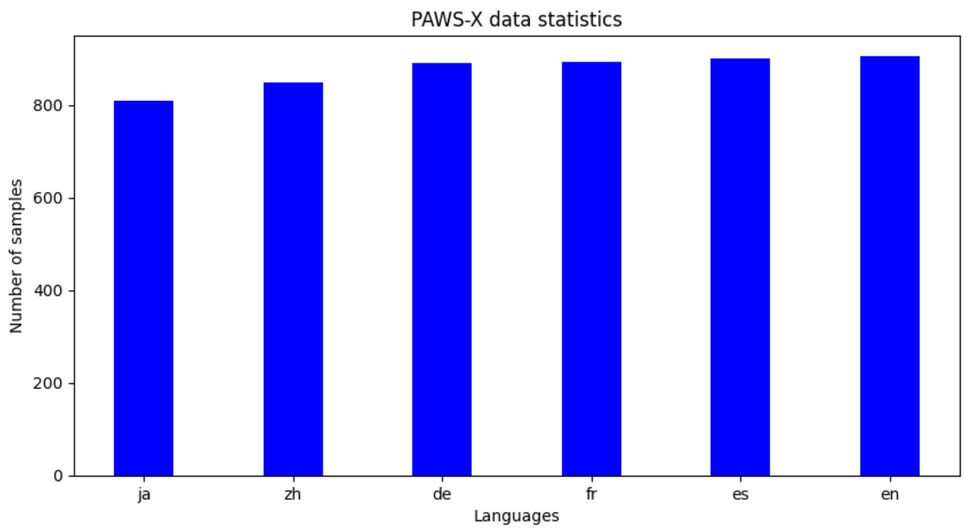


Figure 4.7: PAWS-X data statistics on the number of samples per language.

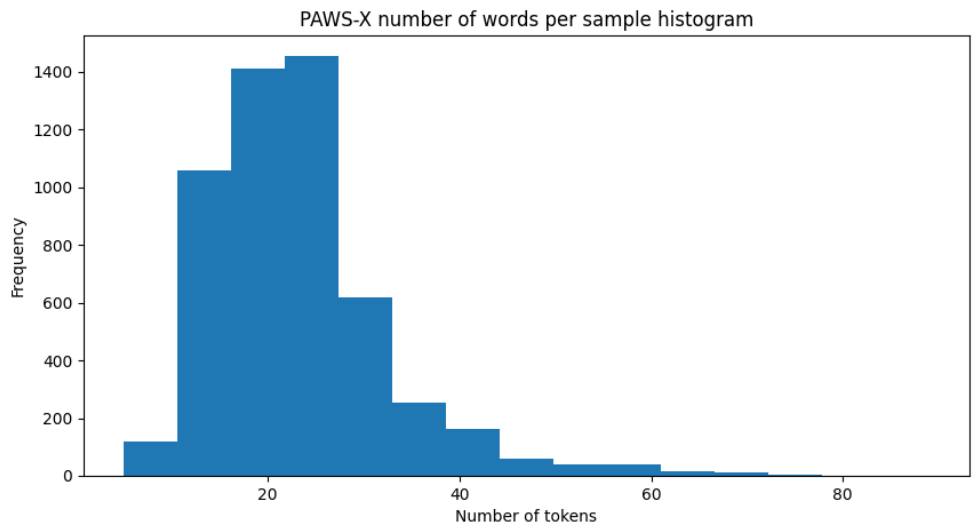


Figure 4.8: Histogram on number of tokens of PAWS-X evaluation dataset.

STAPLE [109] is a translation dataset where there is one input and multiple references. This dataset is used by Duolingo, thus, the source language is English while the target varies from Korean, Japanese, Vietnamese, Hungarian, and Portuguese. In order not to overlap with the existing languages in the other two evaluation datasets, we extract only three languages from STAPLE including Vietnamese, Hungarian, and Portuguese. Since STAPLE is the train set, we randomly extract 200 samples, each with one sentence playing the role of the input for the model and the other four sentences will be the references. We thus construct the first multi-reference multilingual paraphrase evaluation set.

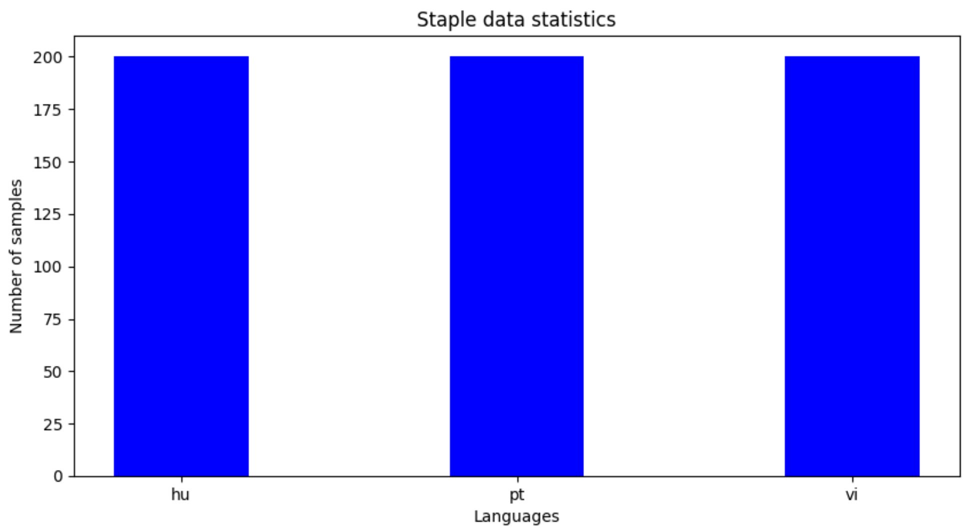


Figure 4.9: STAPLE data statistics on the number of samples per language.

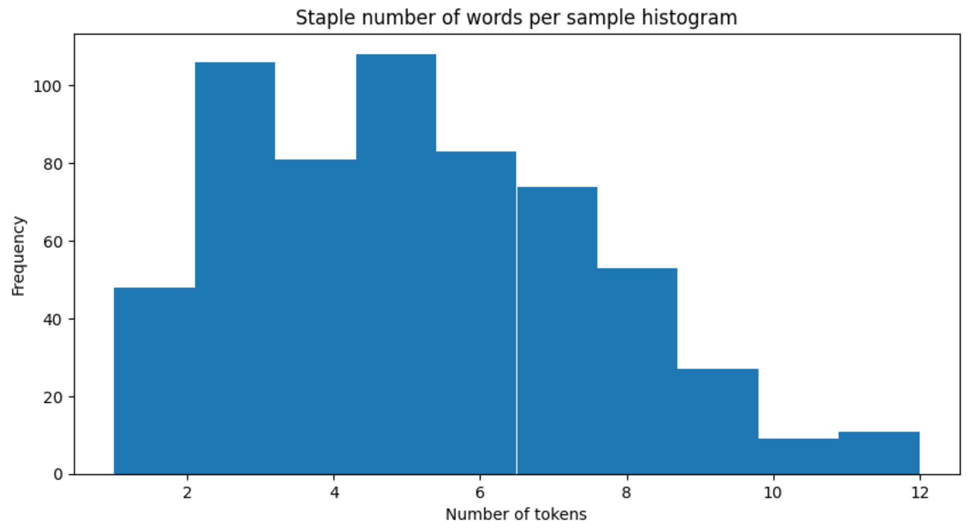


Figure 4.10: Histogram on number of tokens of STAPLE evaluation dataset.

4.5 Result and Analysis

4.5.1 Main result

For the **input-only** part, we evaluate the result of the model compared to the input by using three metrics: BERTScore-F1 [55], ParaScore [56] and BERT-iBLEU [8].

For BERTScore-F1, we use the HuggingFace implementation [5] with the mBERT [27] for scoring many languages. For ParaScore, we use the official implementation from the authors [6]. For BERT-iBLEU, we implement based on the description formula given in the paper [8].

For the **input-and-reference** part, we evaluate the result of the model using the same as **input-only** with a few modifications for ParaScore to take into account the reference. In addition, we also add SacreBLEU, SelfBLEU (SacreBLEU between input and prediction) and TER (we use the implementations from HuggingFace [7], [8]).

For all scores, the higher indicates a better model while it is the reverse for TER and SelfBLEU, in which the lower the score, the better the model.

Table 4.1: The English performance in WMT19.

	BERTScore-F1	ParaScore	BERT-iBLEU
Hegde et al. [53]	0.929	0.923	0.817
Niu et al. [8]	0.889	0.889	0.794
Thompson and Post[2]	0.921	0.903	0.818
LAMPAT (Ours)	**0.933**	**0.928**	**0.826**

[5]https://huggingface.co/spaces/evaluate-metric/bertscore
[6]https://github.com/shadowkiller33/parascore_toolkit
[7]https://huggingface.co/spaces/evaluate-metric/sacrebleu
[8]https://huggingface.co/spaces/evaluate-metric/ter

Table 4.2: The English performance in Opusparcus.

	BERTScore-F1	ParaScore	BERT-iBLEU	SacreBLEU	TER
Hegde et al. [53]	0.924	0.926	0.502	8.291	80.934
Niu et al. [8]	0.912	0.895	0.496	7.532	85.668
Thompson and Post[2]	0.949	0.925	0.569	**8.642**	74.445
LAMPAT (Ours)	**0.951**	**0.931**	**0.574**	8.395	**72.359**

Table 4.3: The English performance in PAWS-X.

	BERTScore-F1	ParaScore	BERT-iBLEU	SacreBLEU	TER
Hegde et al. [53]	0.934	0.923	**0.791**	36.804	35.372
Niu et al. [8]	0.922	0.918	0.749	35.253	35.026
Thompson and Post[2]	0.932	0.914	0.779	37.905	35.456
LAMPAT (Ours)	**0.947**	**0.939**	0.786	**38.992**	**33.887**

Overall, BERTScore-F1 is the score which measures the semantic similarity between pairs of sentences. The higher the score is, the better semantic similarity is. ParaScore is the score which measures both semantic similarity and lexical diversity between pairs of sentences. BERT-iBLEU is also the score to measure both semantic similarity and lexical diversity. TER is the metric to measure the lexical diversity between the predictions and the groundtruths. According to Table 4.1, Table 4.2, and Table 4.3, our method (LAMPAT) achieves a higher BERTScore-F1 than most of the baseline methods in English (en). LAMPAT achieves a higher ParaScore and lower TER than all of the baseline methods. However, LAMPAT's BERTScore-F1 is lower than Chowdhury et al. [68] in Opusparcus. Besides, the BERT-iBLEU of LAMPAT in PAWS-X is also less than that of Hegde et al. [53].

Table 4.4: WMT19 BERTScore-F1 performance.

	fi	de	fr	zh	en	cs
Thompson and Post[2]	0.812	0.712	0.744	0.827	0.921	0.802
LAMPAT (Ours)	**0.857**	**0.895**	**0.895**	**0.909**	**0.933**	**0.914**

Table 4.5: WMT19 ParaScore performance.

	fi	de	fr	zh	en	cs
Thompson and Post[2]	0.811	0.711	0.739	0.831	0.903	0.801
LAMPAT (Ours)	**0.868**	**0.902**	**0.902**	**0.912**	**0.928**	**0.91**

Table 4.6: WMT19 BERT-iBLEU performance.

	fi	de	fr	zh	en	cs
Thompson and Post[2]	0.618	0.505	0.478	0.646	0.818	0.628
LAMPAT (Ours)	**0.845**	**0.862**	**0.854**	**0.732**	**0.826**	**0.839**

According to Table 4.4, our method (LAMPAT) achieves a higher BERTScore-F1 than the baseline [2] in all given languages, especially in English (en). According to Table 4.5, LAMPAT achieves a higher ParaScore than the baseline in all given languages, and again we attain far better semantic similarity and lexical diversity than [2] in English. According to Table 4.6, our method achieves a higher BERT-iBLEU score than [2] in all given languages. Moreover, we attain far better semantic similarity and lexical diversity than the baseline in German, French, and English. Overall, our method has a better performance than the baseline in all given languages from WMT19 dataset , especially the English language.

Table 4.7: Opusparcus BERTScore-F1 performance.

	fi	de	fr	ru	sv	en
Thompson and Post[2]	0.804	0.801	0.794	0.863	0.842	0.949
LAMPAT (Ours)	**0.845**	**0.879**	**0.888**	**0.922**	**0.855**	**0.951**

Table 4.8: Opusparcus ParaScore performance.

	fi	de	fr	ru	sv	en
Thompson and Post[2]	0.803	0.799	0.79	0.858	0.838	0.925
LAMPAT (Ours)	**0.86**	**0.886**	**0.895**	**0.915**	**0.864**	**0.931**

Table 4.9: Opusparcus BERT-iBLEU performance.

	fi	de	fr	ru	sv	en
Thompson and Post[2]	0.646	0.581	0.642	0.665	0.606	0.569
LAMPAT (Ours)	**0.786**	**0.738**	**0.776**	**0.7**	**0.733**	**0.574**

Table 4.10: Opusparcus SacreBLEU performance.

	fi	de	fr	ru	sv	en
Thompson and Post[2]	0.322	1.134	1.563	2.085	1.156	**8.642**
LAMPAT (Ours)	**4.9**	**7.735**	**7.429**	**6.481**	**6.038**	8.395

Table 4.11: Opusparcus SelfBLEU performance.

	fi	de	fr	ru	sv	en
Thompson and Post[2]	16.758	27.122	23.807	39.667	25.724	54.533
LAMPAT (Ours)	**7.935**	**20.914**	**16.812**	**37.043**	**20.176**	**23.344**

Table 4.12: Opusparcus TER performance.

	fi	de	fr	ru	sv	en
Thompson and Post[2]	319.954	171.227	193.303	102.147	187.348	74.445
LAMPAT (Ours)	**77.405**	**72.735**	**74.554**	**72.521**	**77.309**	**72.359**

According to Table 4.7, Table 4.8, and Table 4.9, our method (LAMPAT) achieves a higher BERTScore-F1, ParaScore, and BERT-iBLEU than the baseline in all given languages, especially in English. SacreBLEU is an upgrade metric from BLEU which considers pairs of sentences. The higher SacreBLEU is, the better the quality of a sentence might be. According to Table 4.10, we achieve a better performance than [2] in most of the given languages, especially German and French. SelfBLEU and TER are the metrics to measure the lexical diversity between the predictions and the inputs, and between the predictions and the groundtruths, respectively. The lower

SelfBLEU and TER are, the better the quality of a sentence can be. According to Table 4.11 and Table 4.12, we have a way better performance than [2] in all languages. Generally speaking, our method has a better performance than the baseline in all given languages from Opusparcus dataset.

Table 4.13: PAWS-X BERTScore-F1 performance.

	ja	de	fr	zh	en	es	ko
Thompson and Post[2]	0.79	0.811	0.795	0.848	0.932	0.746	-
LAMPAT (Ours)	**0.908**	**0.913**	**0.912**	**0.934**	**0.947**	**0.929**	**0.907**

Table 4.14: PAWS-X ParaScore performance.

	ja	de	fr	zh	en	es	ko
Thompson and Post[2]	0.798	0.805	0.788	0.851	0.914	0.741	-
LAMPAT (Ours)	**0.913**	**0.923**	**0.921**	**0.93**	**0.939**	**0.93**	**0.917**

Table 4.15: PAWS-X BERT-iBLEU performance.

	ja	de	fr	zh	en	es	ko
Thompson and Post[2]	0.64	0.519	0.484	0.635	0.779	0.466	-
LAMPAT (Ours)	**0.78**	**0.879**	**0.876**	**0.677**	**0.786**	**0.865**	**0.87**

Table 4.16: PAWS-X SacreBLEU performance.

	ja	de	fr	zh	en	es	ko
Thompson and Post[2]	14.902	28.102	29.166	36.411	37.905	**27.621**	-
LAMPAT (Ours)	**29.133**	**22.326**	**24.618**	**46.581**	**38.992**	26.95	**22.754**

Table 4.17: PAWS-X SelfBLEU performance.

	ja	de	fr	zh	en	es	ko
Thompson and Post[2]	45.619	**21.504**	**23.099**	63.32	45.773	**29.004**	-
LAMPAT (Ours)	**36.0**	51.362	50.99	**56.913**	**43.722**	49.941	**49.555**

Table 4.18: PAWS-X TER performance.

	ja	de	fr	zh	en	es	ko
Thompson and Post[2]	134.581	66.147	79.658	68.637	35.456	65.632	-
LAMPAT (Ours)	**58.799**	**51.693**	**52.275**	**43.309**	**33.887**	**47.812**	**55.675**

According to Table 4.13, Table 4.14, Table 4.15, and Table 4.16, our method (LAMPAT) achieves a higher BERTScore-F1, ParaScore, BERT-iBLEU, and SacreBLEU than the Thompson and Post[2] in all given languages. According to Table 4.17 and Table 4.18, we get a lower sentence similarity, between the predictions and inputs/groundtruths, than [2] in all given languages. In general, our method has a better performance than the baseline in all given languages from PAWS-X dataset.

Table 4.19: STAPLE BERTScore-F1 performance.

	hu	pt	vi
Thompson and Post[2]	0.828	0.87	0.858
LAMPAT (Ours)	**0.862**	**0.92**	**0.877**

Table 4.20: STAPLE ParaScore performance.

	hu	pt	vi
Thompson and Post[2]	0.838	0.866	0.86
LAMPAT (Ours)	**0.869**	**0.917**	**0.898**

Table 4.21: STAPLE BERT-iBLEU performance.

	hu	pt	vi
Thompson and Post[2]	0.779	0.708	0.716
LAMPAT (Ours)	**0.789**	**0.789**	**0.814**

Table 4.22: STAPLE SacreBLEU performance.

	hu	pt	vi
Thompson and Post[2]	5.992	13.286	**21.403**
LAMPAT (Ours)	**8.163**	**16.648**	20.237

Table 4.23: STAPLE TER performance.

	hu	pt	vi
Thompson and Post[2]	80.819	74.026	62.285
LAMPAT (Ours)	**60.24**	**51.719**	**51.172**

According to Table 4.19, Table 4.20, Table 4.21, and Table 4.22, our method (LAMPAT) achieves a higher BERTScore-F1, ParaScore, BERT-iBLEU, and SacreBLEU than the Thompson and Post[2] in all given languages, even though we do not have a much better performance than [2], because the score's difference between LAMPAT and the baseline is not considerable. According to Table 4.23, we again get a lower sentence similarity, between the predictions and inputs/groundtruths, than [2] in all 3 languages: Hungarian, Portuguese, and Vietnamese. Overall, our method has a better performance than the baseline in all given languages from STAPLE dataset.

4.5.2 Human evaluation

In addition to machine evaluation, we also conduct the human evaluation of the paraphrase generated by our model. For each of the following languages: English, Vietnamese, German, French and Japanese, we randomly extract 200 sentence pairs of the input sentence and our model prediction from the evaluation dataset. After that, for each mentioned language, we ask 5 annotators to score 200 sentence pairs independently. Each annotator will be required to score the paraphrase quality on the Likert scales from 1 to 5, with 1 being the worse pair and 5 being the best.

For each language, we report the average score.

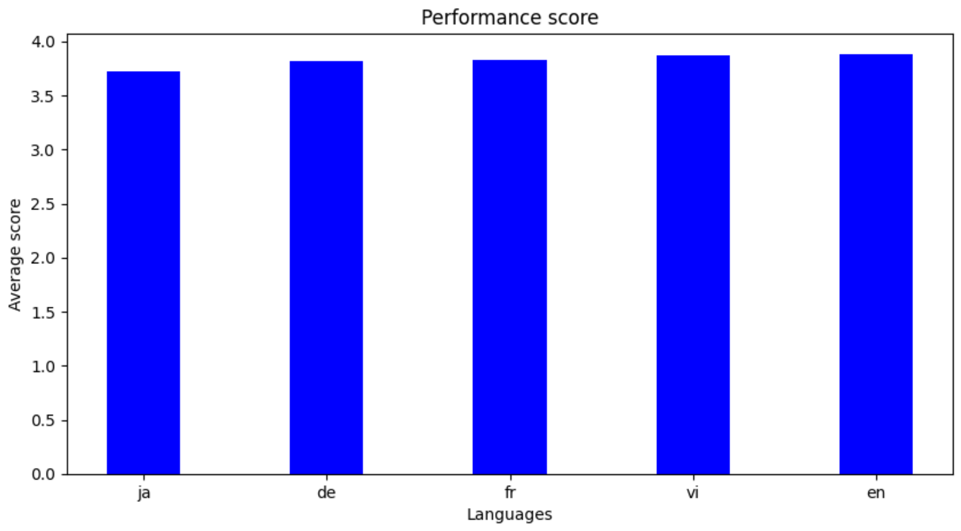


Figure 4.11: Average annotation scores for chosen languages.

Our paraphrase quality attends a high score which is nearly 4 points with the lowest score belonging to Japanese. We hypothesise that Japanese is the hieroglyphs language, thus, the vocabulary of Japanese is enormous. Our language model's vocab is huge in size but cannot cover all the words in many languages, thus, making it fail to generate high-quality paraphrases compare to other languages.

Below are some examples we randomly take from the evaluation result.

Table 4.24: German example where the first line is the input, followed by the translation and the third line is LAMPAT's generation output.

Sentence	Score
Joshua Kimmich: Lernen vom neuen Mini-Lahm des FC Bayern *(Joshua Kimmich: Learning from Bayern's new mini-lame)* Joshua Kimmich: Vom neuen FC Bayern zum Die neue Generation *(Joshua Kimmich: From the new FC Bayern to the new generation)*	4.0
Alle Maßnahmen sind umgesetzt worden. *(All measures have been implemented.)* Die Maßnahmen sind umgesetzt worden. *(The measures have been implemented.)*	4.2
Wir müssen einander lieben und unterstützen. *(We must love and support one another.)* Wir müssen uns einander lieben und unterstützen *(We must love and support one another)*	3.8
Ein 1948-er Ford Mercury fuhr durch eine Gruppe von Zuschauern im ländlichen Monroe, Georgia, und rumpelte auf die Ford-Brücke von Moore zu. *(A 1948 Ford Mercury drove through a crowd of spectators in rural Monroe, Georgia and rumbled toward Moore's Ford Bridge.)* Die Gruppe fuhr eine 1948-er Ford Mercury, die von den Zuschauern im ländlichen Georgia durch die Ford-Brücke von Moore zu *(The group drove a 1948 Ford Mercury, which was swept by the crowd in rural Georgia through the Ford Bridge from Moore.)*	4.0
Mr. Trump gewann West Virginia mit 42 Punkten, und Justice billigte nicht die demokratische Kandidatin Hillary Clinton. *(Mr. Trump won West Virginia by 42 points, and Justice did not endorse Democratic nominee Hillary Clinton.)* Hillary Clinton gewann 42 von 42 Punkten und gewann damit die demokratische Kandidatin West Virginia, die mit dem *(Hillary Clinton won 42 points out of 42, beating Democratic nominee West Virginia, who won the)*	2.0

Table 4.25: Japanese example where the first line is the input, followed by the translation and the third line is LAMPAT's generation output.

Sentence	**Score**
ロンゴは、ニュージャージー州ジャージーシティで誕生し、ニュージャージー州バヨンヌのマリスト高校およびロードアイランド大学に通っていた。 *(Born in Jersey City, New Jersey, Longo attended Marist High School in Bayonne, New Jersey and the University of Rhode Island.)* 彼は、ニュージャージー州ジャージーシティで誕生し、ニュージャージー州バヨンヌのマリスト高校およびロードアイランド大学に通っていた。 *(Born in Jersey City, New Jersey, he attended Marist High School in Bayonne, New Jersey and the University of Rhode Island.)*	4.2
演奏にはキューバ ギターではなくキューバ トレ スを使用し、このスパニッシュ ギターのテクニックを磨いていった。 *(Instead of playing a Cuban guitar, he played on a Cuban Torres and refined his technique on this Spanish guitar.)* 彼はキューバのギターではなくキューバのトレ スを使用し、このスパニッシュギターのテクニックを磨いていった *(He used a Cuban tres instead of a Cuban guitar and honed his technique on this Spanish guitar.)*	4.0
2011 年、彼は歌手マドンナの伝記「マッド・フォー・マドンナ、クイーン・オブ・ポップ」もカステルベッチ出版により発行しました。 *(In 2011, he also published a biography of the singer Madonna, Mad for Madonna, Queen of Pop, published by Castellvecchi Publishing.)* 2011年に出版された彼の伝記「フォー・マドンナ・オブ・ポップ・クイーンマッド」は、彼の歌手としてのキャリアをより多くの人に伝えたいと、彼のマドンナの	3.8

(His biography, For Madonna of Pop Queen Mad, published in 2011, said he wanted to bring his singing career to a wider audience.)	
クダワはネパール南東のバラ地区にあるナラヤニ区画の村および村開発委員会です。 *(Kudawa is a village and village development committee in the Narayani subdivision in the Bara district of southeastern Nepal.)* ネパールにおける開発の会は、ネパールにおける開発の会とナラヤニの村村の委員会の委員である。 *(The Development Society of Nepal is a member of the Narayani Village Committee.)*	3.2
英国では、一定の期間を置いて安い物件から高価な物件へと少しずつ移行していくことを指して「不動産はしご (property ladder)」という語が広く使用されています。 *(In the UK, the term "property ladder" is widely used to refer to the gradual transition from cheap to expensive properties over a period of time.)* 日本では、一定の期間を置いて安い物件から高価な物件へと少しずつ移行していくことを指して「不動産はしご(propertyladder)」という語が広く使用されている *(In Japan, the term "propertyladder" is widely used to refer to the gradual transition from low-priced properties to high-priced properties over a period of time.)*	2.0

Table 4.26: French example where the first line is the input, followed by the translation and the third line is LAMPAT's generation output.

Sentence	**Score**
Cette administration colossale emploie 32 000 personnes. *(This colossal administration employs 32,000 people.)* Cette administration emploie 32 000 personnes. *(This administration employs 32,000 people.)*	4.4
Depuis 2004, la France est divisée en huit circonscriptions pour les élections européennes. *(Since 2004, France has been divided into eight constituencies for the European elections.)* En 2004, la France est divisée en huit circonscriptions pour les élections européennes. *(In 2004, France was divided into eight constituencies for the European elections.)*	4.0
On a besoin de travailleurs transférables, résilients. *(We need transferable, resilient workers.)* Nous avons besoin de travailleurs transférables, résilients. *(We need transferable, resilient workers.)*	3.8
Bien qu'il soit bas dans les sondages, Frans Timmermans reste optimiste pour les élections européennes. *(Although he is low in the polls, Frans Timmermans remains optimistic for the European elections.)* Qu'il soit bas dans les sondages, Frans Timmermans reste optimiste pour les élections européennes. *(Although he is low in the polls, Frans Timmermans remains optimistic for the European elections.)*	4.8
L'économie européenne se porte bien, le chômage a diminué presque partout. *(The European economy is doing well, unemployment has fallen almost everywhere.)* L'économie européenne a bien diminué, mais presque partout se porte le chômage. *(The European economy has shrunk, but almost everywhere there is unemployment.)*	2.0

Table 4.27: Vietnamese example where the first line is the input, followed by the translation and the third line is LAMPAT's generation output.

Sentence	**Score**
Cô ấy sẽ lắng nghe bài diễn thuyết. *(She will listen to the lecture.)* Đó là một bài diễn thuyết mà cô ấy sẽ lắng nghe. *(It was a speech she would listen to.)*	4.4
Thịt bò thì tệ. *(Beef is bad.)* Tuy nhiên thịt bò thì tệ. *(However the beef was bad.)*	4.0
Tôi không đi bộ cũng không chạy. *(I neither walk nor run.)* Tôi không chạy bộ cũng không đi. *(I neither jog nor walk.)*	4.8
Không vấn đề! *(No problem!)* Điều này không có vấn đề! *(This is no problem!)*	4.8
Chúng tôi ăn trong khi súp nóng. *(We ate while the soup was hot.)* Chúng tôi ăn súp trong khi chúng tôi đang nóng. *(We eat soup while we are hot.)*	2.0

Table 4.28: English example where the first line is the input and the second line is LAMPAT's generation output.

Sentence	**Score**
Colin argues this is not true, but accuses Paul of incompetently managing some properties. Colin accuses Paul of incompetently managing some of his properties, and argues this is not true.	4.4
Car and tractor crash together: woman dies in terrible accident The terrible accident: a woman dies in a terrible car and tractor crash together	4.0
The performance from Frank Castorf and his directing team had been awaited with excitement. The Frank Castor and his directed team's performance had been awaited with excitement	4.8
Asked about the level of investments, Thomas Scherer doesn't want to commit himself precisely. Thomas Scherer was asked about the level of investments, and he doesn't want to commit himself precisely.	4.8
Console instead of turf – Bundesliga is getting ready for e-sport The Bundesliga is getting ready for e-sport – and it is not just the console	2.0

4.5.3 Ablation study

Parameter Efficient Fine-Tuning

In order to study the affect of PEFT method in our works, we thus experiment on some common methods in PEFT: LoRA, P-Tuning, Prefix Tuning, Prompt Tuning and Adapter. For LoRA [66], we follows the original settings $r = 16$. For Adapter [64], we follows the original settings and set the reducing factor $r = 16$. For Prefix Tuning [63], for each layers, we add the additional embeddings with the length of 2. For P-Tuning [110], we extend the input embedding by 20 trainable additional tokens. For Prompt Tuning [65], we extend the input embedding by 20 trainable additional tokens.

For each of the PEFT methods, we train on the same number of epochs with 3 random seeds and report the mean and standard deviation of ParaScore on WMT19 part of the evaluation dataset.

Table 4.29: The ParaScore of each PEFT method in WMT19 dataset.

	fi	de	fr	zh	en	cs
Prefix Tuning [63]	0.774±0.15	0.791±0.11	0.782±0.07	0.866±0.09	0.883±0.10	0.835±0.13
Prompt Tuning [65]	0.842±0.25	0.855±0.24	0.792±0.30	0.874±0.27	0.881±0.15	0.885±0.15
P-Tuning [110]	0.837±0.11	0.862±0.12	0.883±0.13	0.879±0.13	0.899±0.12	0.898±0.15
Adapter [64]	0.829±0.02	0.794±0.03	0.873±0.03	0.889±0.04	0.916±0.08	0.889±0.07
LoRA [66]	**0.846±0.03**	**0.851±0.02**	**0.889±0.09**	**0.894±0.03**	**0.926±0.04**	**0.904±0.05**

Among the PEFT methods, LoRA experiences to be the most stable method and achieves highest scores, especially for generation tasks.

Adversarial training

Since LoRA is a stable method for partially fine-tuning the LM. We adapt LoRA as the PEFT method for fine-tuning mGPT. In order to study the affect of Adversarial Training (AT) in the unsupervised learning, we employs two settings which is Projected Gradient Descent (PGD) [74] and Projected Newton Method (PNM) [101, 102] together with LoRA.

Through the experiments, we found out that PNM method has faster converged compared to PGD and LoRA only, results in a better ParaScore in some of the first epochs. Below is the ParaScore measured every 2 epochs. PNM clearly outperformed PGD and LoRA only in some first epochs and attends at highest score after 10 epochs of training.

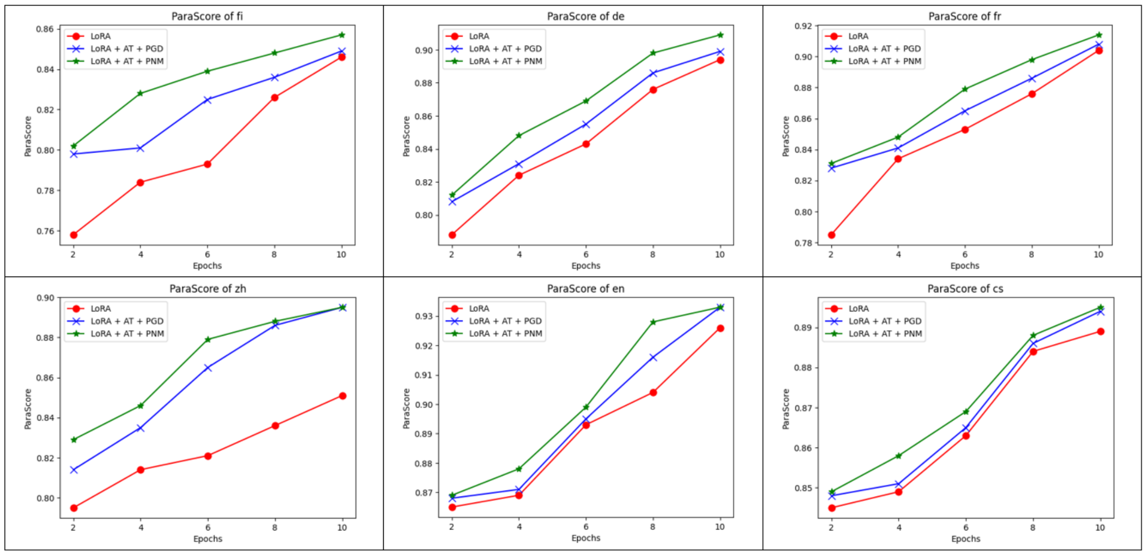


Figure 4.12: Even when all of the techniques were trained on the same dataset with the same number of epochs, the change in ParaScore of each technique is different. Although PGD seems to have a comparable performance to PNM, PNM converges faster and has a higher ParaScore than PGD in all 6 languages of WMT19.

We hypothesized that the main driving force that make PNM better compared to PGD is because it uses the Taylor expansion, which has a better approximation of the objective function. In particular, we states PNM as an optimization problem. First, we have the constraint of the perturbation $\mathcal{S} = ||\delta||_F \leq \epsilon$, with fixed ϵ, is the p-norm ball constraint. In this case, we consider this constraint as a convex set S. After each update of the gradients of the perturbation δ, they must be projected onto the constraint. On the one hand, the PGD algorithm for δ at the time step $t+1$ is:

$$\begin{aligned} \delta_{t+\frac{1}{2}} &= \delta_t - \eta g(\delta_t) \\ \delta_{t+1} &= \arg\min_{\nu \in \mathcal{S}} ||\nu - \delta_{t+\frac{1}{2}}||_2^2 \end{aligned} \tag{4.7}$$

where $g(\delta_t) = \nabla f(\delta_t)$ is the gradients with respect to δ_t, and η is the learning rate. On the other hand, δ_{t+1} can be computed with a quadratic function given Taylor approximation as followed:

$$\delta_{t+1} = \arg\min_{\nu \in \mathcal{S}} [f(\delta_t) + g(\delta_t)(\nu - \delta_t) + \frac{1}{2}(\nu - \delta_t)^T H_t (\nu - \delta_t)] \tag{4.8}$$

where H_t is the Hessian matrix of the objective function with respect to δ_t. This is also the PNM algorithm. If we suppose that $m = \nu - \delta_t$, we will have the following quadratic function:

$$\delta_{t+1} = \arg\min_{\nu \in \mathcal{S}} [f(\delta_t) + g(\delta_t)m + \frac{1}{2}m^T H_t m] \tag{4.9}$$

and to optimize δ with respect to m, we can construct the first derivative and get the optimal $m = H_t^{-1} g(\delta_t)$. Therefore, the general form of a PNM algorithm is:

$$\begin{aligned} \delta_{t+\frac{1}{2}} &= \delta_t - \eta H_t^{-1} g(\delta_t) \\ \delta_{t+1} &= \arg\min_{\nu \in \mathcal{S}} ||\nu - \delta_{t+\frac{1}{2}}||_{H_t}^2 \end{aligned} \tag{4.10}$$

where $||z||_{H_t} = \sqrt{z^T H_t z}$ is the projection of z under Hessian matrix. PNM utilizes the Taylor approximation and projected-Hessian matrix. Intuitively, it can have good approximation of the objective function and it can be faster than PGD in terms of convergence because at each iteration the perturbation δ take a step of parabola-form (from quadratic function) instead of linear-form (from linear function) in PGD algorithm. This method have an important limitation. If the approximation is accurate and the step size is small, calculating a specific value for m leads to rapid convergence. However, if the function is nearly flat in certain dimensions, divergence may occur. This happens when the second derivatives approach zero, causing the inverse to become very large and resulting in large steps. Since the Taylor approximation is accurate locally, taking large steps can move the current estimates far from regions where the Taylor approximation holds true.

4.5.4 Result analysis

With reference to Section 4.5.1 and Section4.5.2, we can state that LAMPAT has a good performance in multilingual paraphrasing. First, we achieve higher BERTScore, ParaScore, BERT-iBLEU, SacreBLEU, SelfBLEU and TER results. As a result, our method tends to preserve the semantic similarity between the input and the prediction. Probably, it is because we choose to preserve the keywords of a sentence and only remove the stopwords. Moreover, because we use a PEFT method, LoRA, to improve our method, LAMPAT can partially preserve the original distribution of its vocabulary instead of increasing its discrepancy during the training phase. Second, the higher BERTScore, ParaScore, and BERT-iBLEU are, the better the lexical diversity is. The reason why we obtain good lexical diversity compared to [2] is because we use adversarial virtual training which can steer the model prediction towards a virtual label by the perturbation embedding. Consequently, our proposed method can generate a good paraphrase of a sentence which both preserves correct semantic meaning and enhances diverse choices of lexicons.

On the other hand, LAMPAT is trained on the latest version of WMT19, which is just released in 2023. Therefore, our method can collect up-to-date information about COVID-19 and other important news. Furthermore, LAMPAT is trained with a wide range of languages including Arabic, Hindi, Indonesian, Japanese, Kazakh, etc. and it can work with Korean as well, which is not feasible to the work [2]. Based on the analysis in Section 4.5.3, we hypothesize that the usage of PNM can help us improve the performance and accuracy of LAMPAT compared to PGD. However, because PNM also has some limitations which may lead to the divergence, we address the issues by choosing the appropriate hyperparameters as presented in Section 4.3.2. For the reason that the Taylor approximation works well in local region of a function, we avoid taking large steps by using scheduler to decrease the learning rate of $g(\delta)$ and choosing a low perturbation bound.

Chapter 5

Conclusion

5.1 Summary

In this project, we successfully combine the PEFT method LoRA (Low-Rank Adaptation) with Adversarial Training to create virtual labels, which assist the unsupervised training together with the multilingual large pre-trained language model mGPT to create LAMPAT.

We find that LAMPAT outperforms the baseline by a large margin in most of the evaluation metrics that we have used. In addition to LAMPAT, we have also constructed the first multilingual multi-domain paraphrasing tasks to help contribute to the development of the multilingual NLP progress. Through experiments, we find that LAMPAT not only works well on training languages such as English, French, Japanese, etc., but can also generalize to unseen languages in the training phase such as Vietnamese with scores of nearly 4.0 on the Likert scale from 1 to 5. This holds a promising direction for future development of zero-shot cross-lingual paraphrase generation.

In conclusion, our project has made significant contributions to the field of natural language processing. We have successfully combined adversarial training with PEFT method to train a large multilingual language model for paraphrasing tasks and tackle the challenge of limited data annotation. In addition, we have fine-tuned the model in an unsupervised learning technique to address the challenge of limited data annotation and to generate diverse and non-duplicate paraphrases. Furthermore, we have constructed the first multilingual multi-domain paraphrasing evaluation dataset and consolidated up-to-date evaluation metrics for this task. These achievements are expected to facilitate future research in paraphrasing and related natural language processing tasks.

5.2 Future work

In future, we can deploy our system to a web service to evaluate our system in more practical cases and optimize LAMPAT. Moreover, we can scale up the size of the training dataset. Large Language Models are originally trained on many diverse datasets. Consequently, if we want them to work well in specific downstream tasks, we can let them train on large and well-prepared multilingual datasets. We have prepared a comprehensive Future Plan as in A.1 in Appendix A. Here is a future plan that incorporates our tasks and the desired outcomes:

1. Increase the number of languages in the training set: Collect additional language data to expand the training dataset to include nearly 40 languages.

2. Fine-tune the model on the new training dataset: Utilize the expanded training dataset to fine-tune the model, enabling it to cover more languages and improve its cross-lingual capabilities.

3. Extend the evaluation dataset: Enhance the evaluation dataset to include samples from nearly 40 languages, ensuring comprehensive coverage for assessing the model's performance.

4. Develop a new method for low-resource languages: Build upon the existing work to develop a novel method specifically targeting low-resource languages. This

method should aim to enable zero-shot cross-lingual paraphrasing, allowing the model to generalize better to languages with limited resources.

5. Evaluate the model using new metrics and human evaluation: Test the model on the updated evaluation dataset, incorporating new metrics and human evaluation. This evaluation will gauge the model's ability to generate realistic sequences not only in high-resource languages like English but also in low-resource languages such as Kazakh.

6. By following this future plan, the model will be equipped to handle a diverse range of languages, demonstrate improved performance on low-resource languages, and undergo rigorous evaluation to ensure its effectiveness across various linguistic contexts.

By following this future plan, we believe our method will be equipped to handle a diverse range of languages, demonstrate improved performance on low-resource languages, and undergo rigorous evaluation to ensure its effectiveness across various linguistic contexts.

References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” *Advances in Neural Information Processing Systems*, vol. 2017-December, no. Nips, pp. 5999–6009, 2017.

[2] B. Thompson and M. Post, “Paraphrase Generation as Zero-Shot Multilingual Translation: Disentangling Semantic Similarity from Lexical and Syntactic Diversity,” in *5th Conference on Machine Translation, WMT 2020 - Proceedings*, pp. 561–570, Aug. 2020.

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

[4] Z. Yan, N. Duan, J. Bao, P. Chen, M. Zhou, Z. Li, and J. Zhou, “Docchat: An information retrieval approach for chatbot engines using unstructured documents,” in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 516–525, 2016.

[5] J. Berant and P. Liang, “Semantic parsing via paraphrasing,” *52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014 - Proceedings of the Conference*, vol. 1, pp. 1415–1425, 2014.

[6] Y. Cheng, Y. Liu, Q. Yang, M. Sun, and W. Xu, “Neural machine translation with pivot languages,” *arXiv preprint arXiv:1611.04928*, 2016.

[7] 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.

[8] T. Niu, S. Yavuz, Y. Zhou, N. S. Keskar, H. Wang, and C. Xiong, “Unsupervised Paraphrasing with Pretrained Language Models,” in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, (Stroudsburg, PA, USA), pp. 5136–5150, Association for Computational Linguistics, 2021.

[9] S. Ranathunga, E.-S. A. Lee, M. P. Skenduli, R. Shekhar, M. Alam, and R. Kaur, “Neural Machine Translation for Low-Resource Languages: A Survey,” pp. 1–35, 2021.

[10] O. I. Abiodun, A. Jantan, A. E. Omolara, K. V. Dada, N. A. Mohamed, and H. Arshad, “State-of-the-art in artificial neural network applications: A survey,” *Heliyon*, vol. 4, no. 11, p. e00938, 2018.

[11] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[12] F. A. Gers, J. Schmidhuber, and F. Cummins, “Learning to forget: Continual prediction with lstm,” *Neural computation*, vol. 12, no. 10, pp. 2451–2471, 2000.

[13] I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to sequence learning with neural networks,” *CoRR*, vol. abs/1409.3215, 2014.

[14] Z. Tan, S. Wang, Z. Yang, G. Chen, X. Huang, M. Sun, and Y. Liu, “Neural machine translation: A review of methods, resources, and tools,” *AI Open*, vol. 1, pp. 5–21, 2020.

[15] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of Machine Learning Research*, vol. 21, pp. 1–67, 2020.

[16] M. Allahyari, S. A. Pouriyeh, M. Assefi, S. Safaei, E. D. Trippe, J. B. Gutierrez, and K. J. Kochut, "Text summarization techniques: A brief survey," *CoRR*, vol. abs/1707.02268, 2017.

[17] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language Models are Unsupervised Multitask Learners," 2020.

[18] Z. Wang, "Modern question answering datasets and benchmarks: A survey," 2022.

[19] R. J. Williams and D. Zipser, "A learning algorithm for continually running fully recurrent neural networks," *Neural Computation*, vol. 1, no. 2, pp. 270–280, 1989.

[20] D. E. Rumelhart and J. L. McClelland, *Learning Internal Representations by Error Propagation*, pp. 318–362. 1987.

[21] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, "Scheduled sampling for sequence prediction with recurrent neural networks," *CoRR*, vol. abs/1506.03099, 2015.

[22] C.-M. U. S. Dept., "Speech understanding systems: summary of results of the five-year research effort at Carnegie-Mellon University.," 4 2015.

[23] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014.

[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015.

[25] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," 2016.

[26] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," *CoRR*, vol. abs/2005.14165, 2020.

[27] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, vol. 1, no. Mlm, pp. 4171–4186, 2019.

[28] Z. Yang, Z. Dai, Y. Yang, J. G. Carbonell, R. Salakhutdinov, and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," *CoRR*, vol. abs/1906.08237, 2019.

[29] A. Gupta, A. Agarwal, P. Singh, and P. Rai, "A Deep Generative Framework for Paraphrase Generation," *32nd AAAI Conference on Artificial Intelligence, AAAI 2018*, pp. 5149–5156, sep 2018.

[30] S. Wu, B. Chen, C. Xin, X. Han, L. Sun, W. Zhang, J. Chen, F. Yang, and X. Cai, "From paraphrasing to semantic parsing: Unsupervised semantic parsing via synchronous semantic decoding," *ACL-IJCNLP 2021 - 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, pp. 5110–5121, 2021.

[31] M. Maimaiti, Y. Liu, H. Luan, Z. Pan, and M. Sun, "Improving Data Augmentation for Low-Resource NMT Guided by POS-Tagging and Paraphrase Embedding," *ACM Transactions on Asian and Low-Resource Language Information Processing*, vol. 20, pp. 1–21, nov 2021.

[32] Y. Cao and X. Wan, "DivGAN: Towards Diverse Paraphrase Generation via Diversified Generative Adversarial Network," in *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 2411–2421, Association for Computational Linguistics, 2020.

[33] D. Zeng, H. Zhang, L. Xiang, J. Wang, and G. Ji, "User-Oriented Paraphrase Generation with Keywords Controlled Network," *IEEE Access*, vol. 7, pp. 80542–80551, 2019.

[34] A. Fenogenova, "Russian paraphrasers: Paraphrase with transformers," *Proceedings of the 8th BSNLP Workshop on Balto-Slavic Natural Language Processing, BSNLP 2021 - Co-located with the 16th European Chapter of the Association for Computational Linguistics, EACL 2021*, pp. 11–19, 2021.

[35] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," 2019.

[36] B. Thompson and M. Post, "Automatic machine translation evaluation in many languages via zero-shot paraphrasing," *EMNLP 2020 - 2020 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, pp. 90–121, 2020.

[37] B. N. Patro, V. K. Kurmi, S. Kumar, and V. Namboodiri, "Learning semantic sentence embeddings using sequential pair-wise discriminator," in *Proceedings of the 27th International Conference on Computational Linguistics*, (Santa Fe, New Mexico, USA), pp. 2715–2729, Association for Computational Linguistics, Aug. 2018.

[38] Z. Guo, Z. Huang, K. Q. Zhu, G. Chen, K. Zhang, B. Chen, and F. Huang, "Automatically Paraphrasing via Sentence Reconstruction and Round-trip Translation," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pp. 3815–3821, International Joint Conferences on Artificial Intelligence Organization, aug 2021.

[39] Y. Cai, Y. Cao, and X. Wan, "Revisiting Pivot-Based Paraphrase Generation: Language Is Not the Only Optional Pivot," *EMNLP 2021 - 2021 Conference on Empirical Methods in Natural Language Processing, Proceedings*, pp. 4255–4268, 2021.

[40] Y. Guo, Y. Liao, X. Jiang, Q. Zhang, Y. Zhang, and Q. Liu, "Zero-Shot Paraphrase Generation with Multilingual Language Models," 2019.

[41] C. Federmann, O. Elachqar, and C. Quirk, "Multilingual Whispers: Generating Paraphrases with Translation," in *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, (Stroudsburg, PA, USA), pp. 17–26, Association for Computational Linguistics, 2019.

[42] J. Wieting and K. Gimpel, "ParaNMT-50M: Pushing the Limits of Paraphrastic Sentence Embeddings with Millions of Machine Translations," pp. 451–462, nov 2017.

[43] W. Chen, J. Tian, L. Xiao, H. He, and Y. Jin, "A Semantically Consistent and Syntactically Variational Encoder-Decoder Framework for Paraphrase Generation," in *Proceedings of the 28th International Conference on Computational Linguistics*, (Stroudsburg, PA, USA), pp. 1186–1198, International Committee on Computational Linguistics, 2020.

[44] J. Sun, X. Ma, and N. Peng, "AESOP: Paraphrase Generation with Adaptive Syntactic Control," *EMNLP 2021 - 2021 Conference on Empirical Methods in Natural Language Processing, Proceedings*, pp. 5176–5189, 2021.

[45] T. Goyal and G. Durrett, "Neural Syntactic Preordering for Controlled Paraphrase Generation," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, (Stroudsburg, PA, USA), pp. 238–252, Association for Computational Linguistics, 2020.

[46] A. Ormazabal, M. Artetxe, A. Soroa, G. Labaka, and E. Agirre, "Principled Paraphrase Generation with Parallel Corpora," in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, pp. 1621–1638, Association for Computational Linguistics, 2022.

[47] O. Bojar, C. Buck, C. Federmann, B. Haddow, P. Koehn, J. Leveling, C. Monz, P. Pecina, M. Post, H. Saint-Amand, R. Soricut, L. Specia, and A. Tamchyna, "Findings of the 2014 workshop on statistical machine translation," in *Proceedings of the Ninth Workshop on Statistical Machine Translation*, (Baltimore, Maryland, USA), pp. 12–58, Association for Computational Linguistics, June 2014.

[48] A. Roy and D. Grangier, "Unsupervised paraphrasing without translation," *ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, pp. 6033–6039, 2020.

[49] A. Van Den Oord, O. Vinyals, *et al.*, "Neural discrete representation learning," *Advances in neural information processing systems*, vol. 30, 2017.

[50] C. Chelba, T. Mikolov, M. Schuster, Q. Ge, T. Brants, P. Koehn, and T. Robinson, "One billion word benchmark for measuring progress in statistical language modeling," *arXiv preprint arXiv:1312.3005*, 2013.

[51] T. Kajiwara, B. Miura, and Y. Arase, "Monolingual transfer learning via bilingual translators for style-sensitive paraphrase generation," *AAAI 2020 - 34th AAAI Conference on Artificial Intelligence*, pp. 8042–8049, 2020.

[52] F. Brad and T. Rebedea, "Neural paraphrase generation using transfer learning," *INLG 2017 - 10th International Natural Language Generation Conference, Proceedings of the Conference*, pp. 257–261, 2017.

[53] C. Hegde and S. Patil, "Unsupervised Paraphrase Generation using Pre-trained Language Models," 2020.

[54] 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 - ACL '02*, vol. 371, (Morristown, NJ, USA), p. 311, Association for Computational Linguistics, 2002.

[55] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, "BERTScore: Evaluating Text Generation with BERT," Feb. 2020.

[56] L. Shen, L. Liu, H. Jiang, and S. Shi, "On the Evaluation Metrics for Paraphrase Generation," in *EMNLP 2022 - 2022 Conference on Empirical Methods in Natural Language Processing, Proceedings*, pp. 3178–3190, Feb. 2022.

[57] M. Snover, B. Dorr, R. Schwartz, L. Micciulla, and J. Makhoul, "A study of translation edit rate with targeted human annotation," in *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas: Technical Papers*, (Cambridge, Massachusetts, USA), pp. 223–231, Association for Machine Translation in the Americas, Aug. 8-12 2006.

[58] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[59] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695, 2022.

[60] Y. Xu, M. Li, L. Cui, S. Huang, F. Wei, and M. Zhou, "Layoutlm: Pre-training of text and layout for document image understanding," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1192–1200, 2020.

[61] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," *arXiv preprint arXiv:2212.04356*, 2022.

[62] A. Babu, C. Wang, A. Tjandra, K. Lakhotia, Q. Xu, N. Goyal, K. Singh, P. von Platen, Y. Saraf, J. Pino, *et al.*, "Xls-r: Self-supervised cross-lingual speech representation learning at scale," *arXiv preprint arXiv:2111.09296*, 2021.

[63] X. L. Li and P. Liang, "Prefix-tuning: Optimizing continuous prompts for generation," *ACL-IJCNLP 2021 - 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, pp. 4582–4597, 2021.

[64] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly, "Parameter-efficient transfer learning for nlp," in *International Conference on Machine Learning*, pp. 2790–2799, PMLR, 2019.

[65] B. Lester, R. Al-Rfou, and N. Constant, "The Power of Scale for Parameter-Efficient Prompt Tuning," *EMNLP 2021 - 2021 Conference on Empirical Methods in Natural Language Processing, Proceedings*, pp. 3045–3059, 2021.

[66] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "LoRA: Low-Rank Adaptation of Large Language Models," pp. 1–26, 2021.

[67] L. Tu, C. Xiong, and Y. Zhou, "Prompt-Tuning Can Be Much Better Than Fine-Tuning on Cross-lingual Understanding With Multilingual Language Models," 2022.

[68] J. R. Chowdhury, Y. Zhuang, and S. Wang, "Novelty Controlled Paraphrase Generation with Retrieval Augmented Conditional Prompt Tuning," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 10, pp. 10535–10544, 2022.

[69] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," 2015.

[70] P. Saadatpanah, A. Shafahi, and T. Goldstein, "Adversarial attacks on copyright detection systems," 2019.

[71] C. Xiao, R. Deng, B. Li, F. Yu, M. Liu, and D. Song, "Characterizing adversarial examples based on spatial consistency information for semantic segmentation," 2018.

[72] T. Miyato, A. M. Dai, and I. Goodfellow, "Adversarial training methods for semi-supervised text classification," 2021.

[73] Y. Cheng, L. Jiang, and W. Macherey, "Robust neural machine translation with doubly adversarial inputs," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, (Florence, Italy), pp. 4324–4333, Association for Computational Linguistics, July 2019.

[74] C. Zhu, Y. Cheng, Z. Gan, S. Sun, T. Goldstein, and J. Liu, "Freelb: Enhanced adversarial training for natural language understanding," 2020.

[75] A. Shafahi, M. Najibi, A. Ghiasi, Z. Xu, J. Dickerson, C. Studer, L. S. Davis, G. Taylor, and T. Goldstein, "Adversarial training for free!," 2019.

[76] D. Zhang, T. Zhang, Y. Lu, Z. Zhu, and B. Dong, "You only propagate once: Accelerating adversarial training via maximal principle," 2019.

[77] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," 2019.

[78] L. Pereira, X. Liu, F. Cheng, M. Asahara, and I. Kobayashi, "Adversarial training for commonsense inference," in *Proceedings of the 5th Workshop on Representation Learning for NLP*, (Online), pp. 55–60, Association for Computational Linguistics, July 2020.

[79] H. Cheng, X. Liu, L. Pereira, Y. Yu, and J. Gao, "Posterior differential regularization with f-divergence for improving model robustness," in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, (Online), pp. 1078–1089, Association for Computational Linguistics, June 2021.

[80] T. Miyato, S. ichi Maeda, M. Koyama, and S. Ishii, "Virtual adversarial training: A regularization method for supervised and semi-supervised learning," 2018.

[81] S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, and J. Gao, "Deep learning based text classification: A comprehensive review," 2021.

[82] Y. Liu, J. Gu, N. Goyal, X. Li, S. Edunov, M. Ghazvininejad, M. Lewis, and L. Zettlemoyer, "Multilingual denoising pre-training for neural machine translation," 2020.

[83] O. Shliazhko, A. Fenogenova, M. Tikhonova, V. Mikhailov, A. Kozlova, and T. Shavrina, "mgpt: Few-shot learners go multilingual," 2022.

[84] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov, "Unsupervised cross-lingual representation learning at scale," 2020.

[85] L. Xue, N. Constant, A. Roberts, M. Kale, R. Al-Rfou, A. Siddhant, A. Barua, and C. Raffel, "mT5: A Massively Multilingual Pre-trained Text-to-Text Transformer," *NAACL-HLT 2021 - 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference*, pp. 483–498, 2021.

[86] T. L. Scao, A. Fan, C. Akiki, E. Pavlick, S. Ilić, D. Hesslow, R. Castagné, A. S. Luccioni, F. Yvon, M. Gallé, J. Tow, A. M. Rush, S. Biderman, A. Webson, P. S. Ammanamanchi, T. Wang, B. Sagot, N. Muennighoff, A. V. del Moral, O. Ruwase, R. Bawden, S. Bekman, A. McMillan-Major, I. Beltagy, H. Nguyen,

L. Saulnier, S. Tan, P. O. Suarez, V. Sanh, H. Laurençon, Y. Jernite, J. Launay, M. Mitchell, C. Raffel, A. Gokaslan, A. Simhi, A. Soroa, A. F. Aji, A. Alfassy, A. Rogers, A. K. Nitzav, C. Xu, C. Mou, C. Emezue, C. Klamm, C. Leong, D. van Strien, D. I. Adelani, D. Radev, E. G. Ponferrada, E. Levkovizh, E. Kim, E. B. Natan, F. De Toni, G. Dupont, G. Kruszewski, G. Pistilli, H. Elsahar, H. Benyamina, H. Tran, I. Yu, I. Abdulmumin, I. Johnson, I. Gonzalez-Dios, J. de la Rosa, J. Chim, J. Dodge, J. Zhu, J. Chang, J. Frohberg, J. Tobing, J. Bhattacharjee, K. Almubarak, K. Chen, K. Lo, L. Von Werra, L. Weber, L. Phan, L. B. Allal, L. Tanguy, M. Dey, M. R. Muñoz, M. Masoud, M. Grandury, M. Šaško, M. Huang, M. Coavoux, M. Singh, M. T.-J. Jiang, M. C. Vu, M. A. Jauhar, M. Ghaleb, N. Subramani, N. Kassner, N. Khamis, O. Nguyen, O. Espejel, O. de Gibert, P. Villegas, P. Henderson, P. Colombo, P. Amuok, Q. Lhoest, R. Harliman, R. Bommasani, R. L. López, R. Ribeiro, S. Osei, S. Pyysalo, S. Nagel, S. Bose, S. H. Muhammad, S. Sharma, S. Longpre, S. Nikpoor, S. Silberberg, S. Pai, S. Zink, T. T. Torrent, T. Schick, T. Thrush, V. Danchev, V. Nikoulina, V. Laippala, V. Lepercq, V. Prabhu, Z. Alyafeai, Z. Talat, A. Raja, B. Heinzerling, C. Si, E. Salesky, S. J. Mielke, W. Y. Lee, A. Sharma, A. Santilli, A. Chaffin, A. Stiegler, D. Datta, E. Szczechla, G. Chhablani, H. Wang, H. Pandey, H. Strobelt, J. A. Fries, J. Rozen, L. Gao, L. Sutawika, M. S. Bari, M. S. Al-shaibani, M. Manica, N. Nayak, R. Teehan, S. Albanie, S. Shen, S. Ben-David, S. H. Bach, T. Kim, T. Bers, T. Fevry, T. Neeraj, U. Thakker, V. Raunak, X. Tang, Z.-X. Yong, Z. Sun, S. Brody, Y. Uri, H. Tojarieh, A. Roberts, H. W. Chung, J. Tae, J. Phang, O. Press, C. Li, D. Narayanan, H. Bourfoune, J. Casper, J. Rasley, M. Ryabinin, M. Mishra, M. Zhang, M. Shoeybi, M. Peyrounette, N. Patry, N. Tazi, O. Sanseviero, P. von Platen, P. Cornette, P. F. Lavallée, R. Lacroix, S. Rajbhandari, S. Gandhi, S. Smith, S. Requena, S. Patil, T. Dettmers, A. Baruwa, A. Singh, A. Cheveleva, A.-L. Ligozat, A. Subramonian, A. Névéol, C. Lovering, D. Garrette, D. Tunuguntla, E. Reiter, E. Taktasheva, E. Voloshina, E. Bogdanov, G. I. Winata, H. Schoelkopf, J.-C. Kalo, J. Novikova, J. Z. Forde, J. Clive, J. Kasai, K. Kawamura, L. Hazan, M. Carpuat, M. Clinciu, N. Kim, N. Cheng,

O. Serikov, O. Antverg, O. van der Wal, R. Zhang, R. Zhang, S. Gehrmann, S. Pais, T. Shavrina, T. Scialom, T. Yun, T. Limisiewicz, V. Rieser, V. Protasov, V. Mikhailov, Y. Pruksachatkun, Y. Belinkov, Z. Bamberger, Z. Kasner, A. Rueda, A. Pestana, A. Feizpour, A. Khan, A. Faranak, A. Santos, A. Hevia, A. Unldreaj, A. Aghagol, A. Abdollahi, A. Tammour, A. Haji-Hosseini, B. Behroozi, B. Ajibade, B. Saxena, C. M. Ferrandis, D. Contractor, D. Lansky, D. David, D. Kiela, D. A. Nguyen, E. Tan, E. Baylor, E. Ozoani, F. Mirza, F. Ononiwu, H. Rezanejad, H. Jones, I. Bhattacharya, I. Solaiman, I. Sedenko, I. Nejadgholi, J. Passmore, J. Seltzer, J. B. Sanz, K. Fort, L. Dutra, M. Samagaio, M. Elbadri, M. Mieskes, M. Gerchick, M. Akinlolu, M. McKenna, M. Qiu, M. Ghauri, M. Burynok, N. Abrar, N. Rajani, N. Elkott, N. Fahmy, O. Samuel, R. An, R. Kromann, R. Hao, S. Alizadeh, S. Shubber, S. Wang, S. Roy, S. Viguier, T. Le, T. Oyebade, T. Le, Y. Yang, Z. Nguyen, A. R. Kashyap, A. Palasciano, A. Callahan, A. Shukla, A. Miranda-Escalada, A. Singh, B. Beilharz, B. Wang, C. Brito, C. Zhou, C. Jain, C. Xu, C. Fourrier, D. L. Periñán, D. Molano, D. Yu, E. Manjavacas, F. Barth, F. Fuhrimann, G. Altay, G. Bayrak, G. Burns, H. U. Vrabec, I. Bello, I. Dash, J. Kang, J. Giorgi, J. Golde, J. D. Posada, K. R. Sivaraman, L. Bulchandani, L. Liu, L. Shinzato, M. H. de Bykhovetz, M. Takeuchi, M. Pàmies, M. A. Castillo, M. Nezhurina, M. Sänger, M. Samwald, M. Cullan, M. Weinberg, M. De Wolf, M. Mihaljcic, M. Liu, M. Freidank, M. Kang, N. Seelam, N. Dahlberg, N. M. Broad, N. Muellner, P. Fung, P. Haller, R. Chandrasekhar, R. Eisenberg, R. Martin, R. Canalli, R. Su, R. Su, S. Cahyawijaya, S. Garda, S. S. Deshmukh, S. Mishra, S. Kiblawi, S. Ott, S. Sang-aroonsiri, S. Kumar, S. Schweter, S. Bharati, T. Laud, T. Gigant, T. Kainuma, W. Kusa, Y. Labrak, Y. S. Bajaj, Y. Venkatraman, Y. Xu, Y. Xu, Y. Xu, Z. Tan, Z. Xie, Z. Ye, M. Bras, Y. Belkada, and T. Wolf, "BLOOM: A 176B-Parameter Open-Access Multilingual Language Model," nov 2022.

[87] A. Conneau, G. Lample, R. Rinott, A. Williams, S. R. Bowman, H. Schwenk, and V. Stoyanov, "Xnli: Evaluating cross-lingual sentence representations,"

2018.

[88] M. Artetxe, S. Ruder, and D. Yogatama, "On the cross-lingual transferability of monolingual representations," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Association for Computational Linguistics, 2020.

[89] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," in *Advances in Neural Information Processing Systems*, vol. 2020-Decem, may 2020.

[90] T. Z. Zhao, E. Wallace, S. Feng, D. Klein, and S. Singh, "Calibrate before use: Improving few-shot performance of language models," 2021.

[91] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," 2013.

[92] J. Pennington, R. Socher, and C. Manning, "GloVe: Global vectors for word representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (Doha, Qatar), pp. 1532–1543, Association for Computational Linguistics, Oct. 2014.

[93] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, (New Orleans, Louisiana), pp. 2227–2237, Association for Computational Linguistics, June 2018.

[94] T. Schuster, O. Ram, R. Barzilay, and A. Globerson, “Cross-lingual alignment of contextual word embeddings, with applications to zero-shot dependency parsing,” in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, (Minneapolis, Minnesota), pp. 1599–1613, Association for Computational Linguistics, June 2019.

[95] A. Radford and K. Narasimhan, “Improving language understanding by generative pre-training,” 2018.

[96] A. Conneau, S. Wu, H. Li, L. Zettlemoyer, and V. Stoyanov, “Emerging cross-lingual structure in pretrained language models,” in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, (Online), pp. 6022–6034, Association for Computational Linguistics, July 2020.

[97] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, “Roberta: A robustly optimized bert pre-training approach,” 2019.

[98] W. Foundation, “Acl 2019 fourth conference on machine translation (wmt19), shared task: Machine translation of news,”

[99] Z. Li, E. Wallace, S. Shen, K. Lin, K. Keutzer, D. Klein, and J. E. Gonzalez, “Train Large, Then Compress: Rethinking Model Size for Efficient Training and Inference of Transformers,” June 2020.

[100] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei, “Scaling Laws for Neural Language Models,” Jan. 2020.

[101] D. P. Bertsekas, “Projected newton methods for optimization problems with simple constraints,” *SIAM Journal on Control and Optimization*, vol. 20, no. 2, pp. 221–246, 1982.

[102] P. Rebentrost, M. Schuld, L. Wossnig, F. Petruccione, and S. Lloyd, “Quantum gradient descent and newton’s method for constrained polynomial optimization,” *New Journal of Physics*, vol. 21, p. 073023, jul 2019.

[103] S. ISO, “Stopwords iso.” `https://github.com/stopwords-iso/stopwords-iso`, 2020.

[104] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, “Pytorch: An imperative style, high-performance deep learning library,” in *Advances in Neural Information Processing Systems 32*, pp. 8024–8035, Curran Associates, Inc., 2019.

[105] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 2017.

[106] M. Creutz, “Open subtitles paraphrase corpus for six languages,” *LREC 2018 - 11th International Conference on Language Resources and Evaluation*, no. 2005, pp. 1364–1369, 2019.

[107] Y. Yang, Y. Zhang, C. Tar, and J. Baldridge, “PAWS-X: A Cross-lingual Adversarial Dataset for Paraphrase Identification,” in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, (Stroudsburg, PA, USA), pp. 3685–3690, Association for Computational Linguistics, 2019.

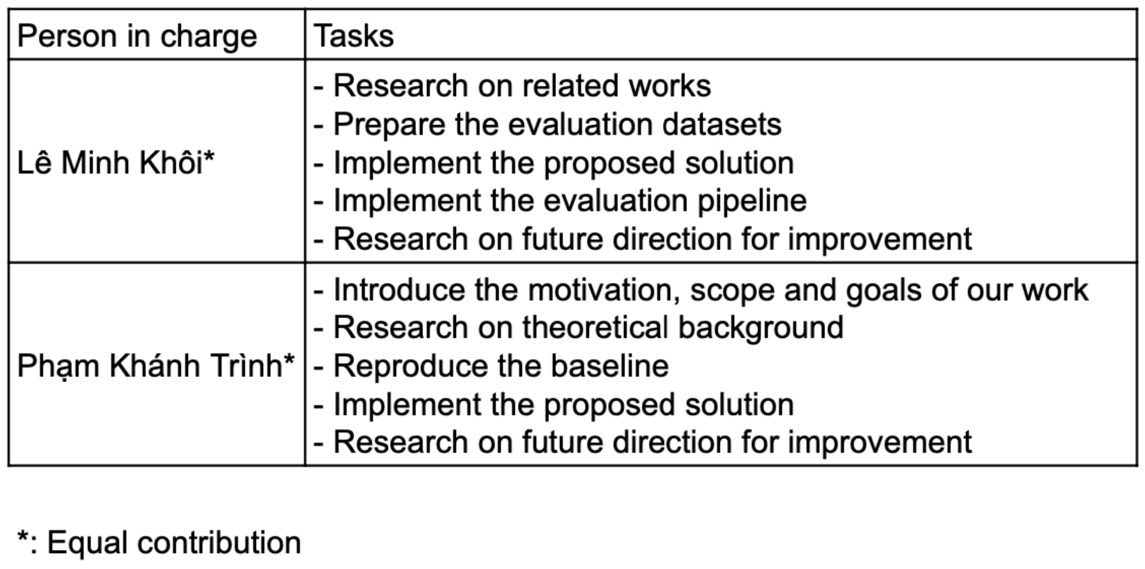
[108] Y. Zhang, J. Baldridge, and L. He, “PAWS: Paraphrase Adversaries from Word Scrambling,” in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, (Minneapolis, Minnesota), pp. 1298–1308, Association for Computational Linguistics, June 2019.

[109] Duolingo, "Data for the 2020 Duolingo Shared Task on Simultaneous Translation And Paraphrase for Language Education (STAPLE)," 2020.

[110] X. Liu, K. Ji, Y. Fu, W. Tam, Z. Du, Z. Yang, and J. Tang, "P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks," in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, (Dublin, Ireland), pp. 61–68, Association for Computational Linguistics, May 2022.

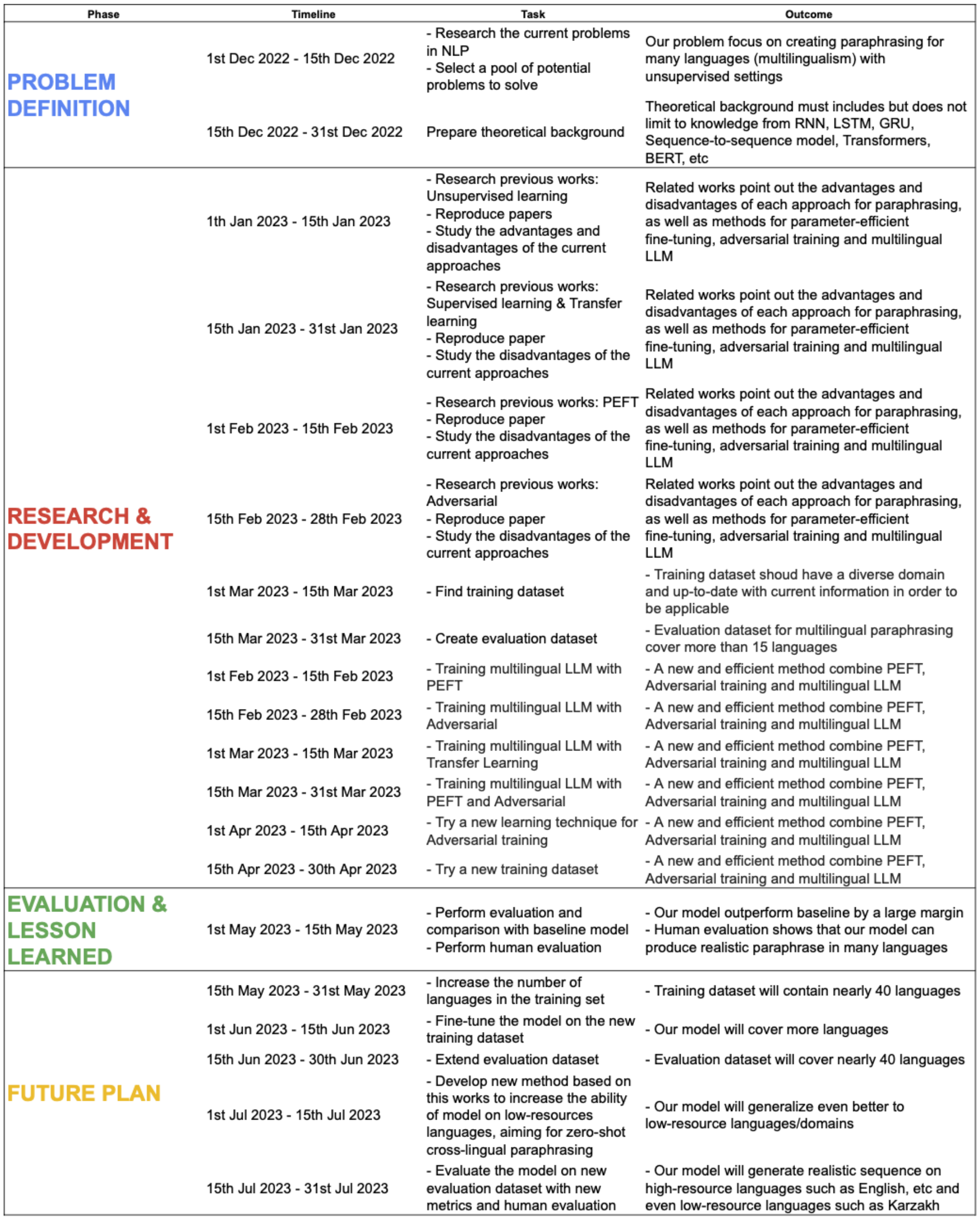
Appendix A

Work assignment



Person in charge	Tasks
Lê Minh Khôi*	- Research on related works - Prepare the evaluation datasets - Implement the proposed solution - Implement the evaluation pipeline - Research on future direction for improvement
Phạm Khánh Trình*	- Introduce the motivation, scope and goals of our work - Research on theoretical background - Reproduce the baseline - Implement the proposed solution - Research on future direction for improvement

*: Equal contribution



Phase	Timeline	Task	Outcome
PROBLEM DEFINITION	1st Dec 2022 - 15th Dec 2022	- Research the current problems in NLP - Select a pool of potential problems to solve	Our problem focus on creating paraphrasing for many languages (multilingualism) with unsupervised settings
	15th Dec 2022 - 31st Dec 2022	Prepare theoretical background	Theoretical background must includes but does not limit to knowledge from RNN, LSTM, GRU, Sequence-to-sequence model, Transformers, BERT, etc
RESEARCH & DEVELOPMENT	1th Jan 2023 - 15th Jan 2023	- Research previous works: Unsupervised learning - Reproduce papers - Study the advantages and disadvantages of the current approaches	Related works point out the advantages and disadvantages of each approach for paraphrasing, as well as methods for parameter-efficient fine-tuning, adversarial training and multilingual LLM
	15th Jan 2023 - 31st Jan 2023	- Research previous works: Supervised learning & Transfer learning - Reproduce paper - Study the disadvantages of the current approaches	Related works point out the advantages and disadvantages of each approach for paraphrasing, as well as methods for parameter-efficient fine-tuning, adversarial training and multilingual LLM
	1st Feb 2023 - 15th Feb 2023	- Research previous works: PEFT - Reproduce paper - Study the disadvantages of the current approaches	Related works point out the advantages and disadvantages of each approach for paraphrasing, as well as methods for parameter-efficient fine-tuning, adversarial training and multilingual LLM
	15th Feb 2023 - 28th Feb 2023	- Research previous works: Adversarial - Reproduce paper - Study the disadvantages of the current approaches	Related works point out the advantages and disadvantages of each approach for paraphrasing, as well as methods for parameter-efficient fine-tuning, adversarial training and multilingual LLM
	1st Mar 2023 - 15th Mar 2023	- Find training dataset	- Training dataset shoud have a diverse domain and up-to-date with current information in order to be applicable
	15th Mar 2023 - 31st Mar 2023	- Create evaluation dataset	- Evaluation dataset for multilingual paraphrasing cover more than 15 languages
	1st Feb 2023 - 15th Feb 2023	- Training multilingual LLM with PEFT	- A new and efficient method combine PEFT, Adversarial training and multilingual LLM
	15th Feb 2023 - 28th Feb 2023	- Training multilingual LLM with Adversarial	- A new and efficient method combine PEFT, Adversarial training and multilingual LLM
	1st Mar 2023 - 15th Mar 2023	- Training multilingual LLM with Transfer Learning	- A new and efficient method combine PEFT, Adversarial training and multilingual LLM
	15th Mar 2023 - 31st Mar 2023	- Training multilingual LLM with PEFT and Adversarial	- A new and efficient method combine PEFT, Adversarial training and multilingual LLM
	1st Apr 2023 - 15th Apr 2023	- Try a new learning technique for Adversarial training	- A new and efficient method combine PEFT, Adversarial training and multilingual LLM
	15th Apr 2023 - 30th Apr 2023	- Try a new training dataset	- A new and efficient method combine PEFT, Adversarial training and multilingual LLM
EVALUATION & LESSON LEARNED	1st May 2023 - 15th May 2023	- Perform evaluation and comparison with baseline model - Perform human evaluation	- Our model outperform baseline by a large margin - Human evaluation shows that our model can produce realistic paraphrase in many languages
FUTURE PLAN	15th May 2023 - 31st May 2023	- Increase the number of languages in the training set	- Training dataset will contain nearly 40 languages
	1st Jun 2023 - 15th Jun 2023	- Fine-tune the model on the new training dataset	- Our model will cover more languages
	15th Jun 2023 - 30th Jun 2023	- Extend evaluation dataset	- Evaluation dataset will cover nearly 40 languages
	1st Jul 2023 - 15th Jul 2023	- Develop new method based on this works to increase the ability of model on low-resources languages, aiming for zero-shot cross-lingual paraphrasing	- Our model will generalize even better to low-resource languages/domains
	15th Jul 2023 - 31st Jul 2023	- Evaluate the model on new evaluation dataset with new metrics and human evaluation	- Our model will generate realistic sequence on high-resource languages such as English, etc and even low-resource languages such as Karzakh

Figure A.1: Capstone Project's plan and log.

Appendix B

Metric formula

Below are the formulas for each metric that we have used to evaluate our model performance as well as the baseline performance.

B.1 BERTScore

The sequence x and y are fed into the BERT model result in two tensors. Denote $\mathbf{x}$ and $\mathbf{y}$ as the list of token embedding emitted by the BERT model for sequence x and y. The formula for BERTScore recall, precision and F1 score is given as below:

$$R_{BERT} = \frac{1}{|\mathbf{x}|} \sum_{\mathbf{x_i} \in \mathbf{x}} \underset{\mathbf{y_j} \in \mathbf{y}}{max} \frac{\mathbf{x_i}^{\mathsf{T}} \mathbf{y_j}}{|\mathbf{x_i}||\mathbf{y_j}|} \tag{B.1}$$

$$P_{BERT} = \frac{1}{|\mathbf{y}|} \sum_{\mathbf{y_j} \in \mathbf{y}} \underset{\mathbf{x_i} \in \mathbf{x}}{max} \frac{\mathbf{x_i}^{\mathsf{T}} \mathbf{y_j}}{|\mathbf{x_i}||\mathbf{y_j}|} \tag{B.2}$$

$$F1_{BERT} = \frac{2 * P_{BERT} * R_{BERT}}{P_{BERT} + R_{BERT}} \tag{B.3}$$

B.2 ParaScore

For the ParaScore with reference, the input for ParaScore will be input sequence X, model prediction C, and the reference R.

$$ParaScore(X, C, R) = max(Sim(X, C), Sim(C, R)) + \omega \cdot DS(X, C) \tag{B.4}$$

where
$Sim(\cdot, \cdot) = BERTScore(\cdot, \cdot)$
ω is the hyper-parameter

$$DS(X, C) = \begin{cases} \gamma & d > \gamma \\ d \cdot \dfrac{\gamma + 1}{\gamma} - 1 & 0 \leq d \leq \gamma \end{cases} \tag{B.5}$$

where $d = NED(X, C)$ and $NED(x, y) = \dfrac{edit\_distance(x, y)}{max(|x|, |y|)}$
If there is no reference, the ParaScore will be defined as below:

$$ParaScore(X, C) = Sim(X, C) + \omega \cdot DS(X, C) \tag{B.6}$$

B.3 BERT-iBLEU

BERT-iBLEU is the harmony between semantic similarity and lexical diversity. BERT-iBLEU receives the input sequence x and the model prediction y.

$$\text{BERT-iBLEU}(x, y) = \left(\frac{\beta * \text{BERTScore-F1}(x, y)^{-1} + 1.0 * (1.0 - \text{BLEU}(x, y))^{-1}}{\beta + 1.0} \right)^{-1} \tag{B.7}$$

B.4 BLEU

BLEU measures the lexical similarity between two sequence x and y by measuring the overlapping n-gram sequence between the two.

Denote by p_n the precision of n-grams, which is the ratio of the number of matched n-grams in the predicted and target sequences to the number of n-grams in the predicted sequence. Besides, let len_{ref} and len_{pred} be the numbers of tokens in the reference (label) sequence and the predicted sequence, respectively.

BLEU score is defined as below:

$$exp\left(min\left(0, 1 - \frac{len_{label}}{len_{pred}}\right)\right)\prod_{n=1}^{k} p_n^{1/2^n} \tag{B.8}$$

For SacreBLEU, $k = 4$ and there are some language specific tokenizer to help detect n-grams correctly.

B.5 TER

$$TER = \frac{\text{Number of edit}}{\text{Average number of words of the reference sequence}} \tag{B.9}$$

in which the `Number of edit` is calculated through Algorithm 2.

Algorithm 2 Translation Edit Rate: Algorithm to determine Number of Edit.

```
Input: Prediction $P$; Reference $R$
$E \leftarrow \infty$
for all $r \in R$ do
    $p' \leftarrow P$
    $e \leftarrow 0$
    while Number of shifts reduced edit distance $\neq 0$ do
        Find shift $s$ that most reduce min-edit-distance(p', r)
        if $s$ reduces edit distance then
            $h' \leftarrow$ apply $s$ to $h'$
            $e \leftarrow e + 1$
    $e \leftarrow e + min - edit - distance(p', r)$
    if $e < E$ then
        $E \leftarrow e$
return E
Output: $E$
```

Appendix C

Evaluation dataset statistics

The following table demonstrates the number of samples per language per dataset among with their release years.

Name	Language	Number of sample	Release year
WMT19	French	1243	2019
	Czech	1843	
	Finish	2527	
	German	2635	
	English	2819	
	Chinese	3319	
Opusparcus	Swedish	947	2018
	Finish	958	
	English	982	
	French	1007	
	German	1047	
	Russian	1068	
PAWS-X	Japanese	810	2019
	Chinese	849	
	German	891	
	French	893	
	Spanish	900	
	English	905	
STAPLE	Hungarian	200	2020
	Vietnamese	200	
	Portugese	200	

Table C.1: Number of sample per language per dataset and the release year of each dataset inside the evaluation dataset

Appendix D

The 12th Science and Technology Symposium for OISP Students Submission Paper

A paper we submitted to the 12th Science and Technology Symposium for OISP Students that demonstrate LAMPAT.