Report on Viet-Lao Machine Translation Task from Natural Language Processing Class of Group 7

Nguyen Bao Dung 22028017 Tran Van Hiep 22028206 Tran Tuan Phong 22028081 Tran Dai Duong 22028273

***Abstract*—This report explores key methodologies and applications within Natural Language Processing (NLP), with a particular focus on transformer-based models and their performance across a variety of language tasks. Beginning with foundational concepts in NLP, the report examines tokenization, language modeling, and the evolution from traditional statistical models to deep learning architectures. Emphasis is placed on the Transformer model and its derivatives, such as BERT and GPT, highlighting their contributions to state-of-the-art performance in text classification, question answering, and sentiment analysis. The report also addresses training procedures, evaluation metrics, and challenges such as data sparsity, bias, and interpretability. Experimental results are presented to compare different model configurations, demonstrating the effectiveness of pre-trained models and transfer learning in achieving high accuracy with limited task-specific data. Overall, the report provides a comprehensive overview of modern NLP techniques and suggests future research directions to further enhance language understanding systems.**

I. Introduction

Natural Language Processing (NLP) is a pivotal field in artificial intelligence that focuses on enabling machines to understand, interpret, and generate human language. By combining computational linguistics with machine learning, NLP facilitates applications such as machine translation, sentiment analysis, and speech recognition. Recent advancements in deep learning have significantly enhanced NLP capabilities, allowing for more accurate and context-aware language models that can handle complex linguistic tasks across diverse languages [1], [2].

The Transformer architecture, introduced by [3], has revolutionized NLP by providing a highly effective framework for sequence-to-sequence tasks. Unlike traditional recurrent neural networks, Transformers rely on self-attention mechanisms to process input data in parallel, capturing long-range dependencies and contextual relationships efficiently. This architecture underpins many state-of-the-art models, such as BERT [4] and GPT [5], and is particularly well-suited for tasks like machine translation due to its scalability and performance on large datasets.

The Viet-Lao translation task addresses the challenge of translating text between Vietnamese and Lao, two languages with distinct linguistic characteristics and limited bilingual resources. Vietnamese, a tonal language with a Latin-based script, and Lao, a tonal language with its own script derived from Khmer, present unique challenges due to their syntactic differences and sparse parallel corpora. Developing an effective translation model for this low-resource language pair requires innovative approaches to data augmentation, transfer learning, and model optimization to achieve high-quality translations [6], [7].

Our attempt at the Viet-Lao translation task in the NLP class is to improve the current Transformer model result. We test out several methods including changing greedy translation to beam search generation, changing the optimizer from Adam to AdamW, varying beam search size for Transformer model. We also include the evaluation method of BLEU score and report loss value on test data to compare different methods.

Our Contribution

Our contributions can be summarized as follows:

- Varying training batch size from 32 to 128 with linear increment of 32 and observe decrement of result.
- Change greedy translation to beam search generation
- Switch optimizer from Adam to AdamW
- Combine beam search with AdamW and *observe improvements in BLEU Score*

Outlines:

- Section II: Reviews prior work in NLP and machine translation.
- Section III: Describes our data understanding and preprocessing before proceeding with translation task.
- Section IV: Discusses the baseline and backbone model architecture and progression.
- Section V: Highlights our proposed configuration to test for improvement over BLEU score.
- Section VI: Presents our findings, comparing model performance and discussing implications for Viet-Lao translation.

II. Background

Natural Language Processing As a subfield in computer science, especially artificial intelligence, Natural Language Processing (NLP) is the analysis of linguistic data, most commonly in the form of textual data such as documents or publications, using computational methods. The aims of NLP in computer science particularly, is to enable computers to process, recognize and further understand data embedded in natural language. With the rapid growth of technology, especially deep learning, NLP has been able to capture complex structures in unstructured data, allowing computers to participate in various communicative tasks such as counselling, machine translation, sentiment analysis, and automated summarization. In other fields of study, NLP can be used to structure extracted

data from the literature and integrate them with other sources of data (cross-modality). Recent advancements in NLP have been driven by the development of transformer-based models, such as BERT [8] and GPT [9], which have significantly improved performance in tasks like text classification, question answering, and language generation. Devlin et al. [8] introduced BERT, which leverages bidirectional context to achieve state-of-the-art results on multiple benchmark datasets, including GLUE and SQuAD. Radford et al. [9] proposed GPT, focusing on generative pre-training to enable robust language modeling. Additionally, studies like those by Vaswani et al. [3] have highlighted the effectiveness of attention mechanisms in sequence-to-sequence tasks, forming the backbone of modern NLP architectures. These works collectively underscore the shift towards deep learning approaches in NLP, addressing challenges in understanding and generating human-like language while paving the way for further research in cross-modal applications.

Machine Translation Task Also named as Computer-Aided Translation (CAT), Machine Translation (MT) is an NLP task that translates a text paragraph from a source language to a target language (often human languages) with the help or automation of computer-based computational methods. Significant advancements in MT have been driven by pioneering research. Vaswani et al. [3] introduced the Transformer architecture, revolutionizing MT with its attention mechanism, which outperforms traditional RNN-based models in speed and accuracy, forming the foundation of modern NMT systems. Bahdanau et al. [10] proposed the attention mechanism in the context of NMT, addressing the limitations of fixed-length context in encoder-decoder models and improving translation quality for long sentences. Additionally, Sutskever et al. [11] demonstrated the effectiveness of sequence-to-sequence learning with RNNs, laying the groundwork for subsequent NMT developments. These works collectively highlight the transition from rule-based and statistical methods to data-driven neural approaches, significantly enhancing MT performance across diverse language pairs.

Deep learning optimization Deep Learning optimization has lead to many works that attempt to optimize the accuracy of the model which requires hyper-parameters tuning. The work that we base our contribution is [12]. This work suggests that using a smaller batch size, eventhough requires higher resources to train, it helps the optimizer search for a global minima, which lead to higher test accuracy.

III. Data Understanding and Preprocessing

A. The Lao Language

The Lao language is a syllabic, tonal language. In syllabic languages, words are con- structed from syllables. In tonal languages, the pitch or tone in which a word is pro- nounced can change its meaning.

The Lao script evolved from the Tai-Kadai script, is an abugida - a linguistic system in which each character represents a consonant with an inherent vowel sound in which close to that of Thai. Added diacritical marks are changes in the script that are used to modify the pronunciation, which in turn modifies the meaning.

Furthermore, the Laos language traditionally does not use spaces as word delim- iters like Vietnamese or English. This calls for tokenization strategies vastly different

from that of languages in which the words are separated by spaces.

B. Dataset Analysis, Cleaning and Preprocessing

The provided data set contains 102,000 pairs of sentences in both the source language Lao and the target language Vietnamese. Of these, 100,000 pairs are reserved for training and 2,000 for validation. By default, the data in both the Laotian and Vietnamese datasets are not clean. Most significantly, in both datasets, there are leftover components from the crawler in

3

the form of " or " marks, as well as URLs in the sentences. Addition- ally, both corpora contain unknown Unicode characters and characters from irrelevant

languages such as Chinese, Russian, etc. These residues may introduce noises and insta- bility during the training process and must be removed before tokenization.

Figure 1 provides the list of unknown words that maybe removed from the sentences.

Furthermore, a large portion of the Laotian corpus consists of sentences that con- tain letters from both the Lao language alphabet and the Roman alphabet. While machine

translation systems typically can handle these, for the sake of consistency in the initial phase of training, we divide each dataset into two parts, based on whether the sentence contains characters from the Roman alphabet or not. This ensures that the generated vocabulary would only include Laotian words. The resulting Laotian-only training and

validation set consists of 45,633 and 737 sentences, respectively. The purpose of differ- entiating good samples to bad samples is to formerly guide the model to perform well

on Laotian before learning side-language.

IV. Model Development

This section reports our baseline model development and architecture.

A. Model's Environment

1) PyTorch Lightning: PyTorch Lightning is an open-source Python library that offers a high-level interface for

PyTorch, one of the most popular machine learning and deep learning frameworks devel- oped by Meta AI. This lightweight and high-performance framework aims to simplify

deep learning experiments by decoupling research from engineering, enhancing read- ability and reproducibility. Its design facilitates the creation of scalable deep-learning

models that can easily run on distributed hardware while remaining hardware-agnostic.

Hình 1. List of faulty characters that are removed from the sentences.

2) Hydra Lightning Template: The Hydra Lightning Template for PyTorch is an adaptation of Meta's open-source

framework for simplifying the development of complex applications, especially in re- search and machine learning. Its key feature is the ability to dynamically create a hi- erarchical configuration by composition and override it through config files and the

command line and instantiate objects code-free, provides an ease to manage workflow and experiments.

B. Model's Architecture

In undertaking this project, our team opted to develop the transformer model from the

ground up. This approach was chosen to foster a deeper comprehension of the transformer architecture and its intricate components. To guide our implementation, we lever- aged insights and knowledge from various sources, most notably from OpenSpeech's [13], Hyunwoonko's [14], U. Jamil's [15]

1) Tokenizers: As mentioned in III, Lao is a tonal language with an abugida writing system. In Lao the pitch or tone in which a word is pronounced can alter its meaning. Additionally, each character in the Lao script represents a consonant with an inherent vowel sound. Because of this, we decided to encode the source language via Byte-Pair Encoding, While Vietnamese exhibits instances of what linguists refer to as scriptio continua, where words are not visually separated by spaces (examples include học, sinh, học sinh, sinh học, all being valid words), our goal is to develop a model capable of discerning the appropriate context for utilizing these combinations of syllabic units. We encode the target language with a Word Level tokenizer. We add words to our vocabulary if they appear at least twice. We ended up with a vocabulary size of 30.000 for Lao and 8.207 for Vietnamese (including the 4 special tokens <sos>, <eos>, <pad>, <unk>).

2) Number of Encoder and Decoder stacks: As outlined by Gao et al. [2], the redundancy of the encoder-decoder architecture in neu- ral machine translation becomes apparent. Given the substantial difference in vocabulary

sizes between the source and target languages, we have opted for a model configuration featuring a greater number of encoders than decoders.

3) Attention: The implementation of the Scaled Dot-Product Attention closely follows the method- ology outlined in the original paper [3]. However, the implementation of Multi-Head

Attention differs in that it omits a final linear layer, opting instead to concatenate the attention scores from the 8 heads directly. This decision is based on the recognition that attention scores inherently result from linear combinations of queries, keys, and values. Additionally, the attention scores will ultimately be passed through

4) Position-wise Feed-Forward Networks: These fully connected layers are implemented just like the original paper [3].

5) Embeddings: We originally intended to use a pre-trained Vietnamese and Lao embeddings, like the one from LaoNLP. But these embeddings kept causing RAM overflow during model training as such we decided to also train our own, smaller embeddings along with the model. The input and output embeddings map each token to a vector of size dmodel, just like in the original paper [3].

6) Positional Encoding: For Positional Encoding, we use a slightly modified version of the fixed positional en- codings written in the original paper [3]. We do division in log space for numerical stability. Our positional encoding functions are as follows:

$$PE_{pos,2i} = sin(pos \times log(10000)2i/d_{model}$$

$$PE_{pos,2i+1} = cos(pos \times log(10000)2i/d_{model}$$

V. Methodology

Our method includes four main parts where we vary the baseline model configuration to improve the performance of the Viet-Lao translation task. The baseline model is a transformer-based sequence-to-sequence model trained on a parallel Viet-Lao corpus. We explore modifications to the decoding strategy, optimization algorithm, and training hyperparameters to enhance translation quality, as measured by BLEU. Each subsection below details a specific modification and its implementation.

A. Beam Search Generation

To improve the quality of generated translations, we replace the baseline's greedy decoding strategy with beam search. Greedy decoding selects the most probable token at each time step, which can lead to suboptimal translations due to its myopic nature. Beam search, in contrast, maintains a fixed number of candidate sequences (beam width) and explores multiple hypotheses simultaneously, selecting the sequence with the highest overall probability at the end. We experiment

with beam widths of 7 and 10 to balance translation quality and computational efficiency. The beam search implementation uses a length normalization factor to prevent bias toward shorter sequences, with the normalization parameter α set to 0.6 based on preliminary experiments. This approach aims to improve the coherence and fluency of translations by considering a broader set of possible outputs.

B. Switch Optimizer from Adam to AdamW

The baseline model uses the Adam optimizer, which is effective for training deep learning models but can suffer from poor generalization due to insufficient regularization. To address this, we switch to the AdamW optimizer, which decouples weight decay from the optimization steps, providing better control over regularization. AdamW applies weight decay directly to the model parameters, which helps prevent overfitting, especially given the relatively small size of the Viet-Lao parallel corpus. We set the weight decay parameter to 0.01 and maintain the baseline learning rate of 0.0005. This modification is expected to improve the model's generalization to unseen data, leading to more robust translations, particularly for low-frequency words and complex sentence structures.

C. Beam Search and AdamW

To leverage the benefits of both decoding and optimization improvements, we combine beam search generation with the AdamW optimizer. This configuration uses the same beam search settings as described in Section V (beam width of 7 and length normalization with $\alpha = 0.6$) and the AdamW optimizer with a weight decay of 0.01. By integrating these two enhancements, we aim to achieve a synergistic effect: beam search improves the quality of the output sequences, while AdamW enhances the model's ability to generalize during training. This combined approach is expected to produce translations that are both fluent and accurate, addressing limitations in the baseline model's performance on complex Viet-Lao sentence pairs.

D. Batch Size Variation

To further optimize training, we experiment with different batch sizes, as this hyperparameter can significantly affect model convergence and performance. The baseline model uses a batch size of 128. We test batch sizes of 32 and 64 to explore their impact on training stability and translation quality. Smaller batch sizes (e.g., 16) provide more frequent updates to the model parameters, potentially leading to better convergence on smaller datasets like the Viet-Lao corpus, but at the cost of increased training time. Larger batch sizes (e.g., 128) enable faster training by leveraging more parallelism but may lead to poorer generalization due to less frequent updates. We monitor the training loss and validation BLEU scores to determine the optimal batch size, ensuring a balance between computational efficiency and translation performance.

VI. RESULTS

The table in Figure I compares BLEU scores, training times, and inference times across various NLP model configurations, revealing key performance trade-offs. The baseline configuration with a batch size of 128 achieves a BLEU score of 0.25, taking 16 hours to train and 350 seconds for inference. Notably, reducing the batch size significantly improves the BLEU score: a batch size of 64 yields a BLEU score of 0.32 with 18 hours of training, while a batch size of 32 achieves the highest BLEU score of 0.4, though it requires 22 hours to train—highlighting that smaller batch sizes enhance translation quality at the cost of longer training times, with inference time remaining constant at 350 seconds. Beam search configurations with beam sizes of 7 and 10 result in BLEU scores of 0.28 and 0.32, respectively, with inference times increasing to 420 and 480 seconds, and training times of 16.5 and 18 hours, indicating a trade-off between quality and speed. Using the AdamW optimizer alone improves the BLEU score to 0.3 with 16 hours of training and 350 seconds of inference, offering a balanced option. The best overall performance comes from combining beam search (beam size 7) with AdamW, achieving a BLEU score of 0.35, with 16.5 hours of training and 421 seconds of inference, demonstrating a synergy that balances quality and efficiency. Thus, for optimal translation quality, the beam search (7) + AdamW configuration is preferable, while smaller batch sizes like 32 are ideal for maximizing BLEU scores despite extended training times.

Configuration	BLEU score	Training time	Infer time
Baseline (batch size = 128)	0.25	16 hours	350 s
Batch size = 64	0.32	18 hours	350 s
Batch size = 32	**0.4**	22 hours	350 s
Beam search (beam size = 7)	0.28	16.5 hours	420 s
Beam search (beam size = 10)	0.32	18 hours	480 s
AdamW	0.3	16 hours	350 s
Beam search(7) + AdamW	*0.35*	16.5 hours	421 s

Figure I
BLEU SCORE, TRAINING AND INFERENCE TIME FOR DIFFERENT CONFIGURATION

VII. CONCLUSION

TÀI LIỆU

[1] C. D. Manning and H. Schütze, *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.

[2] D. Jurafsky and J. H. Martin, *Speech and Language Processing*. Pearson, 2009.

[3] A. Vaswani, N. Shazeer, N. Parmar, J. Uszoreit, L. Jones, A. N. Gomez, Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017, pp. 5998–6008.

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

[5] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," OpenAI, Tech. Rep., 2018.

[6] T. Q. Nguyen and J. Salazar, "Transformers without tears: Improving the normalization of self-attention," *arXiv preprint arXiv:1910.05895*, 2020.

[7] R. Sennrich, B. Haddow, and A. Birch, "Improving neural machine translation models with monolingual data," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, 2016, pp. 86–96.

[8] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2019.

[9] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI Blog*, 2019.

[10] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *International Conference on Learning Representations*, 2015.

[11] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," *Advances in Neural Information Processing Systems*, vol. 27, 2014.

[12] S. Jastrzebski, Z. Kenton, D. Arpit, N. Ballas, A. Fischer, Y. Bengio, and A. J. Storkey, "Three factors influencing minima in SGD," *CoRR*, vol. abs/1711.04623, 2017. [Online]. Available: http://arxiv.org/abs/1711.04623

[13] S. Kim, S. Ha, and S. Cho, "Openspeech: Open-source toolkit for end-to-end speech recognition," https://github.com/openspeech-team/openspeech, 2021.

[14] hyunwoongko, "Pytorch implementation of "attention is all you need"," https://github.com/hyunwoongko/transformer, 2018.

[15] U. Jamil, "Attention is all you need implementation," https://github.com/hyunwoongko/transformer, 2018.