EATT: Knowledge Graph Integration in Transformer Architecture

Phong Vo[1,2], **Long Nguyen**[1,2*]
[1]Faculty of Information Technology, University of Science, Ho Chi Minh City, Vietnam
[2]Vietnam National University, Ho Chi Minh City, Vietnam
20120547@student.hcmus.edu.vn, nhblong@fit.hcmus.edu.vn

Abstract

The Transformer model and Transformer-based models have demonstrated their strength in machine translation tasks. However, their ability to accurately translate entities that appear in sentences has not been fully effective, which is one of the reasons for inefficiencies in semantic transfer between languages. We propose a novel method that integrates a knowledge graph (KG) into the Transformer model, called EATT, to produce more accurate translations of entities. Specifically, this method implements a cross-attention mechanism between the internal vectors in the Transformer model and the embedding vectors obtained from knowledge graph embeddings. This new method outperforms the baseline Transformer model as well as two methods named KB-Trans and KB-Trans-R, which were proposed in our previous research. The evaluation is based on the metrics: BLEU, TER, GLEU, and SBERT. Our source code is available on Github at https://github.com/VTaPo/EATT.

1 INTRODUCTION

The Transformer model (Vaswani et al., 2017) and its variants have achieved great success in machine translation because they can process all parts of a sentence at once and focus on the relationships between different parts of the sentence. However, some challenges remain, particularly in accurately understanding and generating meaning in language. One significant challenge is handling specific entities, such as names or places, that are difficult to identify correctly in the data. For example, when translating the sentence "Lionel Messi was born in Rosario" from English to Vietnamese, there are two main concerns: ensuring the overall quality of the translation and correctly translating the entity names "Lionel Messi" and "Rosario".

Even before the Transformer model was developed, researchers had studied the problem of entity translation, but the focus was mostly on other areas of Natural Language Processing (NLP), like Machine Reading (Yang and Mitchell, 2017) and Question Answering (Sun et al., 2018). Only a few studies have directly tackled the issue of out-of-vocabulary (OOV) words and tried to improve translation quality for entities. These include algorithms like BPE (Sennrich et al., 2016) and methods for querying entity information using a multilingual Knowledge Base (KB) (Moussallem et al., 2019). In this approach, the KB contains multiple representations of an entity in different languages, and these are added to both the source and target sentences to help with accurate translation. Another approach involves breaking down entities into smaller units using knowledge graphs (KGs) (Zhao et al., 2020). Here, entities and sentence pairs are split into sub-word units using BPE, and the authors combined machine translation with knowledge reasoning to help the model use knowledge from the KG more effectively during translation.

These studies still face several challenges. One major issue is the lack of focus on the importance of integrating entities into an Entity Linking system or constructing a Knowledge Graph (KG) and finding a multilingual Knowledge Base (KB) that is robust enough for effective querying. Discovering a KB that is both novel and comprehensive in terms of data coverage requires significant effort. Additionally, the effective application of information from a KB depends heavily on accurately converting the information in the KB into vector space (note that a KG is the graph-based representation of a KB).

This paper proposes a new method for integrating knowledge graphs into the Transformer model, which can leverage knowledge more thoroughly and effectively than previous approaches for English as the source language. This method, named EATT (Entity-Aware Transformer Translation), implements a cross-attention mechanism between the input sentences (where each token has been en-

*Corresponding author.

coded into numerical vectors) and the vector representations of entities in the knowledge graph (KG). Additionally, we also improve other components in our machine translation system, including the development of an entity linking system to guide entities from the input data to the knowledge graph, and the construction of a knowledge graph from the monolingual knowledge base called Wikidata5M (Wang et al., 2021) for the English language, using a knowledge graph embedding algorithm named Fast Linear (Armand et al., 2017).

In summary, our main contributions are:

- Proposing a new method named EATT that integrates knowledge graphs (KG) into the Transformer model to enhance translation quality for entities in English as the source language.
- Evaluating the EATT method across various datasets to demonstrate the generalizability of the proposed approach.
- Conducting a comprehensive evaluation using automatic evaluation metrics, semantic similarity measures, and the translation quality of entities across specific categories.

2 RELATED WORK

In this section, we explore sequence-to-sequence models, entity linking systems, and knowledge graph embedding algorithms, with an emphasis on their contributions to enhancing text processing capabilities.

2.1 Sequence to Sequence models

The Sequence to Sequence (Seq2Seq) model (Sutskever et al., 2014) consists of two main components: the Encoder, which takes an input sequence of characters or words (x_1, x_2, ..., x_T) and transforms them into a context vector h. The embedding layer maps the words or characters in the input text into numerical vectors in a continuous space: $e_t = \text{Embedding}(x_t)$. These embedding vectors are passed through hidden layers to produce hidden states h_t. There can be multiple stacked hidden layers, with the output of the previous layer serving as the input to the next hidden layer, and the computation function f at each layer could be a basic RNN unit, LSTM, or GRU: $h_t = f(e_t, h_{t-1})$. The context vector h is the final hidden state of the encoder: $h = h_T$. The Decoder receives the context vector h from the encoder and generates the output sequence (y_1, y_2, ..., $y_{T'}$) step by step. The initial hidden state of the decoder is usually initialized with the context vector from the encoder: $s_0 = h$. The embedding layer and hidden layers in the decoder function similarly to those in the encoder, as previously explained. The output layer is the hidden state of the decoder transformed into the probabilities of the output words through a softmax layer, where W and b are model parameters to be learned: $o_t = \text{softmax}(Ws_t + b)$. Finally, the word with the highest probability is selected as the output at each time step: $y_t = \text{argmax}(o_t)$.

2.2 Entity Linking Systems

The architecture of an Entity Linking System (EL system) varies depending on the task and system implementation, but generally, an EL system consists of two key components: the NER module and the Entity Disambiguation module. The NER module uses machine learning and deep learning models such as BiLSTM-CRF (Luo et al., 2018), BERT (Devlin, 2018), RoBERTa (Liu, 2019), etc., to recognize entities in the text by tagging them with labels according to predefined standards such as BIO (i.e. **B**egin-**I**nside-**O**utside), BILOU (i.e. **B**eginning, the **I**nside and **L**ast token of multi-token chunks while differentiate them from **U**nit-length chunks), and others. The Entity Disambiguation module's role is to accurately determine the corresponding entry in the Knowledge Graph (KG) for each entity in the source text after it has been recognized. Moreover, when there are multiple similar potential entities, guiding the system to the most accurate corresponding entity in the KG is another crucial role of the disambiguation module to reduce entity ambiguity. Various techniques can be employed to implement the entity disambiguation module, such as absolute string matching, tagging with special IDs, or linking through URL links.

2.3 Knowledge Graph Embedding Algorithms

A Knowledge Graph (KG) is a graph-based representation of a Knowledge Base (KB), where each node represents an entity and each edge represents a relationship between two entities within the knowledge base. Through a knowledge graph embedding algorithm, the entities and relationships in the KG are encoded into vector representations in a high-dimensional latent space, also known as Knowledge Graph Embeddings (KGEs). These vectors contain additional knowledge that, when integrated into the Transformer model, can help the model better un-

derstand the entities that appear in the sentences. Some notable algorithms include:

- TransE (Bordes et al., 2013) assumes that the relationship between two entities is represented by a linear transformation from the input entity to the output entity. Mathematically, let S be the set of all valid triples, S$^{'}$ be the set of all invalid triples, d be the distance metric, which can be either Euclidean or Manhattan, and γ be a hyperparameter of the model. The TransE loss function is defined as follows:

$$L = \sum_{(h,r,t)\in\mathcal{S}} \sum_{(h',r,t')\in\mathcal{S}'} \text{Loss} \quad (1)$$

$$\text{Loss} = \Big[\gamma + d(h+r,t) - d(h'+r,t')\Big]_+ \quad (2)$$

- TransR (Lin et al., 2015) maps each entity and relationship into a subspace of the vector space, allowing TransR to effectively handle multi-relational and complex relationships. When implementing the TransR algorithm, a special matrix Mr is typically constructed to perform this transformation. Mr adjusts the entity embedding vector to align with the embedding space of the relationship r. The TransR loss function is fundamentally similar to that of TransE but differs in the distance metric, which is defined using the Euclidean formula:

$$d(h,r,t) = \|M_r h + r - M_r t\|_2 \quad (3)$$

- TransD (Ji et al., 2015) extends TransR by mapping each pair of entities and relationships into different vector spaces. This allows TransD to model the relationship between entities based on the context of that relationship. However, TransD's large number of parameters increases the risk of overfitting. The TransD loss function is constructed similarly to TransR, but the transformation matrix M is computed using a much more complex formula than in TransR.

- ComplEx (Trouillon et al., 2016) is an advanced and powerful model that uses complex numbers for representation, offering greater flexibility and robustness.

3 METHODOLOGY

In this section, we review the Transformer model, explain the Fast Linear Knowledge Graph Embedding, and describe our EL system. We also present our two baseline methods, KB-Trans and KB-Trans-R, and introduce the new EATT method.

3.1 Transformer Architecture

The Transformer model has a structure quite similar to Seq2Seq models, consisting of two main components: an encoder and a decoder. However, the Transformer can process the input sentence simultaneously. In the encoder, a list of vectors X_{source}, where each vector represents a token in the source sentence, is processed. This component uses multiple stacked encoding layers, each consisting of a self-attention layer and a feed-forward network layer. This process involves a sequence of computations carried out across these encoding layers. The output from the first encoding layer, with $self\_attn(X_{source})$ as the output of the self-attention layer, and f representing the feed-forward network with a ReLU activation function, is as follows:

$$X^{(1)}_{source} = f(self\_attn(X_{source})) \quad (4)$$

The decoder predicts the token sequence for the sentence in the source language, similar to how the encoder processes the input. The main difference is that each decoding layer includes a cross-attention layer positioned between the self-attention layer and the feed-forward layer. The cross-attention layer connects the final output from the encoder with the output from the self-attention layer in each decoding layer, allowing the model to focus on relevant parts of the input. The output of the first decoding layer is:

$$X^{(1)}_{target} = f(cross\_attn(self\_attn(X_{target}))) \quad (5)$$

Considering the computational process of the self-attention mechanism, the input is transformed into related components: the Query matrix (Q), the Key matrix (K), and the Value matrix (V). Each vector in these matrices plays a distinct role in the computational process within the self-attention mechanism. Mathematically, this can be expressed as: $\mathrm{Q} = \mathrm{XW^Q}$, $\mathrm{K} = \mathrm{XW^K}$, and $\mathrm{V} = \mathrm{XW^V}$. Here, X represents the input matrix, where each row is the embedding vector of a word. $\mathrm{W^Q}$, $\mathrm{W^K}$, and $\mathrm{W^V}$

are the weight matrices that transform the input into Q, K, and V, respectively. The computation process of the self-attention mechanism is as follows:

$$\text{attentionScores}(Q, K) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) \quad (6)$$

$$\text{output} = \text{attentionScoresV} \quad (7)$$

where $\sqrt{d_k}$ is a normalization factor to prevent excessively large values, with d_k being the dimension of the key vector.

3.2 Fast Linear Knowledge Graph Embedding

We utilized the knowledge base known as Wikidata5M (Wang et al., 2021) to construct the graph for our research. Each entity in Wikidata5M is represented by an identifier called Qid, and each relationship between two entities is represented by an identifier called Pid. The data format in Wikidata5M is similar to most other knowledge bases, where each line is a triplet in the form of <subject-s, relation-r, object-o>. For example: <Q615, P19, Q52535> corresponds to <Lionel Messi, location of birth, Rosario>, meaning Lionel Messi was born in Rosario. When considering the Wikidata5M knowledge base as a graph, we can visualize it as a graph with nodes and edges representing entities and the relationships between them.

We employed the Fast Linear algorithm to embed the entities and relationships in our knowledge graph into vector representations in a high-dimensional latent space, known as Knowledge Graph Embeddings (KGEs). This algorithm draws inspiration from the classical Bag of Words (BOW) method used in word embedding, where Fast Linear emphasizes the co-occurrence between entities and between entities and their relationships. Both BOW and Fast Linear work effectively with datasets containing discrete tokens, and we can consider a triplet <subject-s, relation-r, object-o> as discrete tokens that are correlated with each other. The generation of additional training samples from the entire set of triplets in Wikidata5M is similar to the sample generation process for the skip-gram model in word embedding. These samples, along with all triplets, are used in the training process for KGEs as follows: The entire training dataset for KGEs, generated from Wikidata5M, is passed through a classifier consisting of two loss functions. Initially, a lookup table V is randomly created, which will serve as the lookup for the initial vector representations of each discrete token. The two main loss calculations include the standard loss computation similar to word embedding in the skip-gram model and the loss calculation for predicting the object o in a triplet, where the vector x_n is a combination of the vector representations for the subject and relation in V. There are various combination methods, and we use normalization in this research. The softmax function used is hierarchical softmax to speed up operations with a large corpus. Theoretically, the optimization of the Fast Linear algorithm involves optimizing Equation 8 below:

$$\frac{1}{N}\sum_{n=1}^{N} y_n \log(f(WVx_n)) \quad (8)$$

where x_n is a normalized combination representation or a pure representation of the token of the n-th input set, y_n is the label.

3.3 Entity Linking System

We developed an Entity Linking system (EL system) as follows: We downloaded a set of all real-world aliases for all entities and relationships existing in Wikidata5M. This set was manually compiled from the information stored on the Wikidata website. In this set, an entity is not limited to a single unique real-world name but is accompanied by a list of common real-world names associated with that entity. For example, Q615 has a list of real-world aliases including M10, Messi, messi, Lionel Messi, lionel messi, Messi Lionel, messi lionel, Lionel Andrés Messi, El Pulga. We then created a dictionary data structure where the keys are the aliases, and the values are the corresponding Qid associated with that alias. This dictionary is used for exact string matching and to look up the Qid corresponding to the alias that matches the entity extracted from the sentence. The ability to cover multiple names for a single Qid reduces the ambiguity of natural language, such as name order swaps due to grammatical structure or differences in full and abbreviated names across different regions and countries.

3.4 KB-Trans and KB-Trans-R

We recognized the significant importance of rationally integrating the vector representations of entities in the Knowledge Graph (KG) into the internal vectors of the Transformer model. In this study, we also implemented two baseline methods for incorporating information into the Transformer

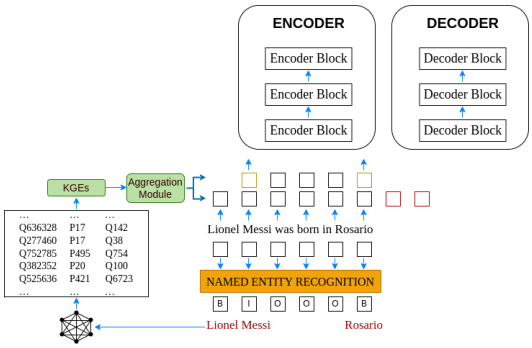

Figure 1: Knowledge-based Transformer methods.

model, named KB-Trans and KB-Trans-R, respectively.

The first method, KB-Trans, begins by extracting the entities from the source sentence. These entities are then mapped to embedding vectors constructed based on the KG, also known as Knowledge Graph Embeddings (KGEs), through the guidance of the EL system. Once the entities are mapped to the KGEs, the obtained embedding vectors are integrated into the Transformer model by concatenating the embedding vectors obtained from the KGEs with the internal vectors randomly generated within the Transformer architecture. This concatenation provides the model with additional semantic information from the entities. However, a limitation is the ambiguity caused when an entity has multiple names or variations across different languages, affecting the ability to retrieve information about the entity from the KG.

The second method, KB-Trans-R, aims to address the ambiguity left by KB-Trans. After the entities in the input sentence are identified, their corresponding Qids are retrieved, and these entities are then marked with their Qids. For example, the sentence "Lionel Messi was born in Rosario" would be marked as "Q615 was born in Q52535". The process of entity linking and extracting embedding vectors from KGEs is similar to the KB-Trans method. However, these vectors are integrated into the Transformer model by completely replacing the internal vectors of the corresponding entities generated by the Transformer's embedding layer. This method not only supplements the model with additional information but also ensures consistency, especially for entities with multiple names or variations in different languages. Additionally, it improves upon KG-Trans when data has been preprocessed with BPE. Figure 1 illustrates the general architecture of the two methods we propose. The "aggregation module" component performs the integration of information from the knowledge embedding vectors obtained from KGEs into the Transformer model. Specifically, the KB-Trans method uses concatenation, represented by red squares, while the KB-Trans-R method uses replacing to completely substitute the random internal vectors (represented by yellow squares).

3.5 Entity-Aware Transformer Translation

Although both methods provide certain improvements for the translation process, they still face certain limitations and do not offer groundbreaking interactions with the information present in the Knowledge Graph (KG). To further enhance performance and fully exploit the potential of knowledge from KGs, we propose a new method that alters the architecture of the Transformer model, named EATT.

In general, this new method focuses on implementing a cross-attention mechanism between the internal vector representations for the input sentence and the vector representations for the entities in the KG. This approach effectively leverages knowledge by avoiding the rigid reintroduction of knowledge back into the Transformer model as done by the previous two methods. EATT shares information about the entities across the entire text, thereby eliminating inconsistencies in the representation of certain entities and reducing bias related to the positional distance of entities compared to other tokens in the input sentence. Each token in the input sentence is encoded into numerical vectors and undergoes a cross-attention mechanism with the vector representations of the entities in the KG (i.e., the KGEs), enabling the model to learn complex relationships and semantic context from both data sources (Figure 2).

3.5.1 Entity Linking and Input Components

First, in terms of entity linking, after the entities in the input sentence are identified, they will be mapped to the KG to extract the corresponding knowledge embedding vectors associated with those entities. These embedding vectors will then participate in the cross-attention mechanism with the internal vectors, which are the vectors that the Transformer model encodes for the tokens in the input sentence.

Assume that the embedding layer of the Transformer model generates internal input vectors with a dimension of 512, and the KGEs also use a di-

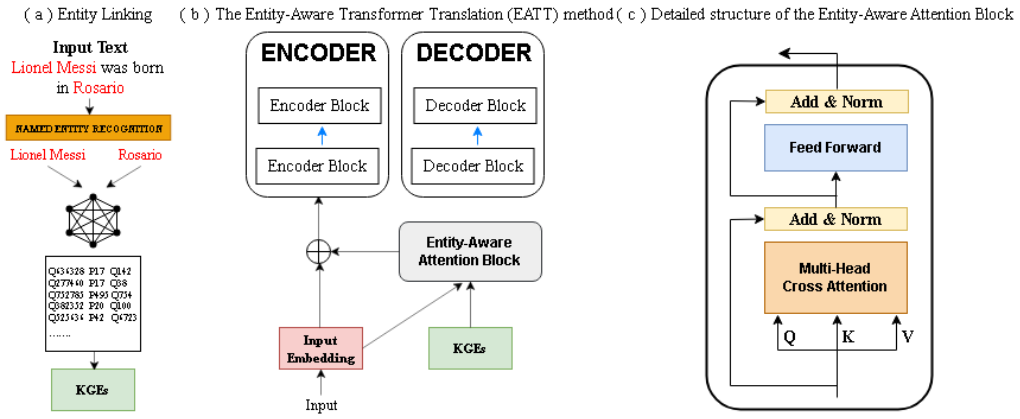

Figure 2: Overview of the EATT method.

mension of 512. For the sentence "Lionel Messi was born in Rosario", the input internal vector matrix will have a size of 6x512 (the "Input Embedding" block in red in section (b) of Figure 2). The two entities "Lionel Messi" and "Rosario", once identified, will be mapped to the KG by the entity linking system to extract the corresponding knowledge embedding vectors for those two entities from the KG. At this point, the "KGEs" component in green in section (b) of Figure 2 will have a size of 2x512. The cross-attention mechanism between the "Input Embedding" and "KGEs" components will be carried out in a block named the Entity-Aware Attention Block before forming the complete input that goes into the encoder of the Transformer model.

3.5.2 Entity-Aware Attention Block

The detailed structure of the Entity-Aware Attention Block includes two subcomponents: the Multi-Head Cross Attention Block and a Feed Forward Neural Network. After each of these two components, there is a residual connection and normalization layer to enhance the training process's efficiency. For the Multi-Head Cross Attention Block, we follow a similar structure to the decoder component of the Transformer model, with the difference being in how the components serve the roles of query Q, key K, and value V.

The practical interpretation of these roles can be explained as follows: The original sentence contains entities that need to be learned, so this sentence acts as the information to be queried, requesting the KG to provide the necessary knowledge to answer those queries. The knowledge-encoded vectors from the KG serve both as the keys, used to match the corresponding information of the entities in the KG with the queried entities, and as the values—the knowledge that the KG returns to the Transformer model during the learning process.

Mathematically, we set Q = Input Embedding, K = KGEs, and V = KGEs. The process of single-head cross-attention between the "Input Embedding" and "KGEs" is represented mathematically as follows:

$$\text{output} = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)\text{V} \tag{9}$$

where d_k is the dimension of the key vector.

The Feed Forward network is designed with a single hidden layer using the ReLU activation function. The purpose of the Feed Forward network within the Entity-Aware Attention Block is to process the newly gathered information. Conceptually, these two processes can be simply described as follows: cross-attention—"Collect new information from the integrated knowledge", and the linear network—"Think and process this newly collected information". Additionally, in the structure of the Entity-Aware Attention Block, after each subcomponent, there is a residual connection layer and a normalization layer. These layers optimize the training process by ensuring faster convergence of the block during training and preventing information loss throughout the entire training process.

3.5.3 Entity-Aware Attention Block Output

The output of the Entity-Aware Attention Block is referred to as EAEs (Entity-Aware Embeddings). EAEs are not directly fed into the encoder; instead, they are added to the initial "Input Embedding", and the newly generated result becomes the final complete input to the Transformer model's encoder. We perform this additional addition operation,

rather than using the EAEs directly as the complete input, due to findings from our experiments, which are as follows: When using EAEs directly as input to the encoder, the generated translations were significantly shorter compared to the translations produced by the base Transformer model, as well as the translations from the KB-Trans and KB-Trans-R methods we previously introduced.

This may be because the output of the Entity-Aware Attention Block represents each token in the original input sentence "attending" to the entities in the sentence that have been supplemented with information from the KG. As a result, the translation tends to focus solely on translating these entities from the source language to the target language, leading to relatively short translations and potentially introducing unfamiliar entities into the translation. By performing the addition operation, we aim to supplement the input data with entity information before it is passed into the Transformer's encoder. This ensures that this information is learned in detail through the cross-attention mechanism.

4 EXPERIMENTS

In this section, we provide an overview of the dataset, configuration settings, experimental procedures, and a thorough analysis of the results.

4.1 Dataset

We conducted experimental evaluations on the IWSLT dataset for four language pairs as follows: English-Vietnamese (En-Vi), English-German (En-De), English-French (En-Fr), and English-Romanian (En-Ro). Additionally, we performed statistical analyses on several noteworthy parameters related to Wikidata5M, as well as the datasets and entities within these datasets. The results of these analyses are provided in Appendix A.

4.2 Configuration Settings

We obtained the set of aliases for all Qids in Wikidata5M thanks to the aggregation efforts and public release by the DeepGraphLearning team[1]. We applied the Fast Linear implementation from the fastText library[2], with the hyperparameters used for training KGEs with fastText provided in Appendix B. We used the spaCy library[3] for entity extraction from the data. For the Transformer model, we employed the implementation provided by fairseq[4]. The hyperparameters for all methods across all language pairs were configured as follows: 6 layers for both the encoder and decoder, 8 heads for multi-head attention, an embedding size of 512 for the model, and a feed-forward network dimension of 2048. The learning rate was set at 3e-4 for KB-Trans, and 5e-4 for the original Transformer, KB-Trans-R, and EATT. We used a dropout rate of 0.3, a label smoothing factor of 0.2, a batch size of 8000 tokens, and trained for 30 epochs. We performed a grid search to optimize certain hyperparameters: model embedding sizes of 512 or 1024, learning rates of 3e-4, 5e-4, or 7e-4, and label smoothing constants of 0, 0.1, or 0.2.

4.3 Ablation Study

We evaluated the performance of the EATT method compared to the base Transformer and the two baseline methods. To demonstrate the generalization capability of EATT, the evaluation is based on automatic evaluation metrics across four language pairs: English-Vietnamese (En-Vi), English-French (En-Fr), English-German (En-De), and English-Romanian (En-Ro). The three metrics used are BLEU (Papineni et al., 2002), TER (Snover et al., 2006), and GLEU (Mutton et al., 2007). The results in TABLE 1 show that both the KB-Trans and KB-Trans-R methods outperform the base Transformer model across all three metrics for the En-Vi language pair. A similar trend is observed in other language pairs, though there are some instances where these two methods perform slightly worse than the base Transformer on certain language pairs. Specifically, KB-Trans performs worse on the En-De pair with the GLEU metric, and KB-Trans-R performs slightly worse on the En-Fr pair with the TER metric.

Most notably, when evaluated using our EATT method, the results indicate that EATT outperforms the base Transformer model and the two baseline methods across all three metrics for all language pairs, with a significant difference in performance, demonstrating substantial improvements over the other methods. The arrows indicate whether a higher (↑) or lower (↓) score is better.

Additionally, the translations produced by the KB-Trans method for relatively long sentences exhibit fewer word reordering (swapping order or using synonyms) or word omissions compared to the

[1] https://deepgraphlearning.github.io/project/wikidata5m
[2] https://github.com/facebookresearch/fastText
[3] https://github.com/explosion/spaCy
[4] https://github.com/facebookresearch/fairseq

KB-Trans-R method. This explains why the KB-Trans-R method performs better than the KB-Trans method on the two automatic evaluation metrics, GLEU and BLEU, but slightly worse in terms of the word correction cost measured by the TER metric. In contrast, the EATT method performs well on both short and long sentences, indicating that sentence length does not affect its ability to utilize the cross-attention mechanism for learning knowledge.

4.4 SBERT semantic similarity

SBERT addresses the computational challenge by fine-tuning each sentence in a sentence pair separately and in parallel. SBERT utilizes a mean pooling layer to extract the output embedding vectors, which are then used to calculate similarity based on the cosine similarity function. TABLE 2 shows the average semantic similarity score across the entire test set of the English-Vietnamese dataset between the machine translations and the reference translations. This result demonstrates that EATT is capable of providing accurate representations of entities, resulting in more fluent and semantically rich translations. EATT can lead to significant improvements in many natural language processing tasks that rely on understanding the meaning and relationships between entities in text.

4.5 Evaluation of entity translation quality

The evaluation was conducted by examining the number of incorrectly translated entities for all three of our proposed methods as well as the baseline Transformer model. Additionally, we analyzed the number of incorrectly translated entities across specific entity types, including: Person (PER), Locations (LOC), Organizations (ORG), and Miscellaneous (MISC). The results from TABLE 3 and TABLE 4 indicate that entities of the types Person names and Organizations tend to be translated more accurately than entities of the other types when comparing the number of incorrect translations for each type across the three proposed methods. When comparing the number of improved translations between the three proposed methods and the baseline Transformer model, Person names and Location entities show the greatest number of improved translations. Notably, the EATT method continues to demonstrate superiority by achieving the highest number of correct translations.

5 LIMITATION

In general, the cases where the proposed methods could not resolve certain issues are due to the following challenges:

- Some entities in the test set, such as “Remi”, “Max Little”, and their corresponding Qids, were encoded as <unk> tokens after the data processing stage. This vocabulary size limitation is also the reason for the differences in handling by the two KG-Trans variants.

- The number of triplets containing the entity is too small in the knowledge graph: For example, the entity “Tunisian” was incorrectly translated with different meanings in the translations due to the fact that there is only one triplet containing this entity in Wikidata5M. This, combined with ambiguity in person name representation, led to the errors.

- The final limitation lies within the proposed methods themselves: Even when not encountering the aforementioned issues, the translations still did not achieve the desired quality.

6 CONCLUSIONS

The new EATT method achieved promising results across multiple language pairs and various experimental evaluation groups compared to the baseline models. Additionally, EATT has the potential to be applied in more general language understanding tasks, where understanding entities and their relationships is crucial. EATT’s cross-attention mechanism allows it to learn complex relationships between entities, leading to a more comprehensive understanding of their connections within a given text. The accurate entity linking system employed by EATT ensures that the model can access comprehensive information about each entity, improving the retrieval of relevant knowledge for question answering. EATT leverages the knowledge graph to create contextualized entity representations, enabling the model to distinguish between the same entity used in different contexts. Future developments that further optimize the utilization of knowledge graphs: applying the HyperGraph Transformer architecture with extended reasoning chains of triplets and integrating knowledge bases (KB) at the document level to accurately translate not only entities but also rare terms.

Table 1: Experimental results (the highest results are marked in bold).

Models	Dataset	GLEU(↑)	TER(↓)	BLEU(↑)
Transformer	EN-VI	33.58	52.89	29.35
KB-Trans	EN-VI	33.65	52.57	29.36
KB-Trans-R	EN-VI	33.76	52.76	29.64
EATT	EN-VI	**34.04**	**52.44**	**30.00**
Transformer	EN-DE	32.49	56.52	26.30
KB-Trans	EN-DE	32.37	56.37	26.42
KB-Trans-R	EN-DE	32.36	56.43	26.53
EATT	EN-DE	**32.51**	**55.86**	**26.68**
Transformer	EN-FR	42.56	44.83	39.77
KB-Trans	EN-FR	43.09	44.46	40.27
KB-Trans-R	EN-FR	42.88	45.02	39.93
EATT	EN-FR	**43.25**	**44.25**	**40.41**
Transformer	EN-RO	20.63	68.77	16.10
KB-Trans	EN-RO	21.73	67.30	16.72
KB-Trans-R	EN-RO	21.27	66.80	16.97
EATT	EN-RO	**22.02**	**65.33**	**18.01**

Table 2: SBERT Average Semantic Similarity Score on En-Vi (the highest results are marked in bold).

Model	Average SBERT
Transformer	0.825
KB-Trans	0.835
KB-Trans-R	0.835
EATT	**0.840**

Table 3: Comparison of translation quality between Transformer model and proposed methods.

Model	#correct translation	#incorrect translation
Transformer	161	35
KG-Trans	176	20
KG-Trans-R	179	17
EATT	**181**	**15**

Acknowledgments

This research is funded by University of Science, VNU-HCM under grant number CNTT 2024-03.

References

Joulin Armand, Grave Edouard, Bojanowski Piotr, Nickel Maximilian, and Mikolov Tomas. 2017. Fast linear model for knowledge graph embeddings. *arXiv e-prints*.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26.

Jacob Devlin. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (volume 1: Long papers)*, pages 687–696.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the AAAI conference on artificial intelligence*, volume 29.

Y Liu. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ling Luo, Zhihao Yang, Pei Yang, Yin Zhang, Lei Wang, Hongfei Lin, and Jian Wang. 2018. An attention-based bilstm-crf approach to document-level chemical named entity recognition. *Bioinformatics*, 34(8):1381–1388.

Diego Moussallem, Mihael Arčan, Axel-Cyrille Ngonga Ngomo, and Paul Buitelaar. 2019. Augmenting neural machine translation with knowledge graphs. *arXiv preprint arXiv:1902.08816*.

Andrew Mutton, Mark Dras, Stephen Wan, and Robert Dale. 2007. Gleu: Automatic evaluation of sentence-level fluency. In *Proceedings of the 45th Annual*

Table 4: Translation quality on entity types: PER, LOC, ORG, and MISC of three proposed methods and baseline model.

Type	Transformer		KG-Trans		KG-Trans-R		EATT	
	Correct	Incorrect	Correct	Incorrect	Correct	Incorrect	Correct	Incorrect
PER	31	11	36	6	38	4	38	4
LOC	57	11	61	7	62	6	63	5
ORG	53	4	55	2	55	2	55	2
MISC	20	9	24	5	24	5	25	4

Meeting of the Association of Computational Linguistics, pages 344–351.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. 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*, pages 223–231.

Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William W Cohen. 2018. Open domain question answering using early fusion of knowledge bases and text. *arXiv preprint arXiv:1809.00782*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *International conference on machine learning*, pages 2071–2080. PMLR.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2021. Kepler: A unified model for knowledge embedding and pre-trained language representation. *Transactions of the Association for Computational Linguistics*, 9:176–194.

Bishan Yang and Tom Mitchell. 2017. Leveraging knowledge bases in LSTMs for improving machine reading. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1436–1446, Vancouver, Canada. Association for Computational Linguistics.

Yang Zhao, Lu Xiang, Junnan Zhu, Jiajun Zhang, Yu Zhou, and Chengqing Zong. 2020. Knowledge graph enhanced neural machine translation via multi-task learning on sub-entity granularity. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4495–4505.

A Statistics Appendix

In Appendix A, we will provide statistical data for the IWSLT dataset across four language pairs, these statistics relate to entities and the number of sentences containing the entities in the three parts train, valid, test of each dataset. We also provide statistics regarding the number of entities, relationships, and triplets in the Wikidata5M.

Table 5: Statistics about Wikidata5M.

#entities	#relations	#triplets
4,813,491	825	21,354,359

Table 6: Statistics about IWSLT EN-VI.

Features	Train	Valid	Test
#total sentences	133,317	1553	1268
#sentences w/ entities	25,721	208	262
#total entities	36,566	264	382
#unique entities	7,967	161	196
Max entities/sentence	14	8	6

B FastText Hyper-parameters Appendix

In Appendix B, we present the parameters used for training KGEs through the fastText library. The hy-

Table 7: Statistics about IWSLT EN-DE.

Features	Train	Valid	Test
#total sentences	209,522	3,889	5,078
#sentences w/ entities	42,015	601	909
#total entities	59,284	828	1225
#unique entities	11,154	440	587
Max entities/sentence	44	8	8

Table 8: Statistics about IWSLT EN-FR.

Features	Train	Valid	Test
#total sentences	236,653	3,888	5,599
#sentences w/ entities	47,498	613	1024
#total entities	67,172	842	1390
#unique entities	12,172	444	652
Max entities/sentence	44	8	8

Table 9: Statistics about IWSLT EN-RO.

Features	Train	Valid	Test
#total sentences	133,333	914	1,678
#sentences w/ entities	25,068	161	206
#total entities	34,969	229	290
#unique entities	7,607	146	159
Max entities/sentence	41	7	6

perparameter Dimension represents the number of dimensions used for the KGEs, and the Loss function is Hierarchical softmax, as mentioned ear802 lier, to speed up processing with extremely large corpora. TABLE 10 shows the values for the hyperparameters.

Table 10: Hyper-parameters in the fastText model.

Hyper-parameter	Value
Model type	Cbow
Dimension	512
Window size	2
Learning rate	0.01
Loss	Hierarchical softmax
Epochs	100

C Correct Translations Appendix

In Appendix C, we illustrate some examples of correct translations produced by the proposed methods compared to the traditional Transformer model. The results show that the proposed methods have a superior ability to translate entities compared to the baseline model; however, there are still minor differences in the translations between the methods.

Table 11: Example 1.

SRC	I am helping the North Korean people.
REF	Tôi đang giúp người Bắc Triều Tiên.
Transformer	Tôi đang giúp người **Hàn Quốc**.
KB-Trans	Tôi đang giúp đỡ những người **Bắc Triều Tiên**.
KB-Trans-R	Tôi đang giúp người **Bắc Triều Tiên**.
EATT	Tôi đang giúp đỡ những người **Bắc Triều Tiên**.

Table 12: Example 2.

SRC	I started this as a tryout in Western Australia.
REF	Tôi bắt đầu điều này bằng việc thử sức ở Tây Úc.
Transformer	Tôi bắt đầu điều này khi thử ở **miền Tây**.
KB-Trans	Tôi bắt đầu điều này khi thử ở **miền Tây nước Úc**.
KB-Trans-R	Tôi bắt đầu điều này khi thử ở **miền Tây nước Úc**.
EATT	Tôi đã bắt đầu điều này ở **miền Tây nước Úc**.

Table 13: Example 3.

SRC	We might produce the next George Washington Carver.
REF	Có thể ta sẽ sản sinh ra George Washington Carver tiếp theo.
Transformer	Chúng ta có thể sản xuất ra **George Stone lo lắng**.
KB-Trans	Chúng ta có thể tạo ra **George Washington Carver tiếp theo**.
KB-Trans-R	Có thể tạo ra **tiếp theo của George Washington**.
EATT	Chúng ta có thể sản xuất ra những **tiếp theo của George Carver**.

Table 14: Example 4.

SRC	This is South Central: liquor stores, fast food, vacant lots.
REF	Đây là vùng Trung Nam: cửa hàng rượu, đồ ăn nhanh, đất hoang.
Transformer	Đây là **Trung tâm: các cửa hàng đóng kín, đồ ăn nhanh, bỏ đi rất nhiều**.
KB-Trans	Đây là **vùng Trung tâm Phía Nam: cửa hàng nước ngọt, đồ ăn nhanh, vùng trống**.
KB-Trans-R	Đây là **vùng Trung tâm Phía Nam: cửa hàng nước ngọt, đồ ăn nhanh, vùng trống**.
EATT	Đây là **Trung tâm Phía Nam: trang cửa hàng, thức ăn nhanh, bỏ trống rất nhiều** .