What Role Does BERT Play in the Neural Machine Translation Encoder?

Anonymous ACL submission

Abstract

Pre-trained language models have been widely applied in various natural language processing tasks. But when it comes to neural machine translation, things are a little different. The differences between the embedding spaces created by BERT and NMT encoder may be one of the main reasons for the difficulty of integrating pre-trained LMs into NMT models. Previous studies illustrate the best way of integration is introducing the output of BERT into the encoder with some extra modules. Nevertheless, it is still unrevealed whether these additional modules will affect the embedding spaces created by the NMT encoder or not and what kind of information the NMT encoder takes advantage of from the output of BERT. In this paper, we start by comparing the changes of embedding spaces after introducing BERT into the NMT encoder trained on different machine translation tasks. Although the changing trends of these embedding spaces vary, introducing BERT into the NMT encoder will not affect the space of the last layer significantly. Subsequent evaluation on several semantic and syntactic tasks proves the NMT encoder is facilitated by the rich syntactic information contained in the output of BERT to boost the translation quality.

1 Introduction

Contextualized representations generated by pre-trained language models (LMs), e.g. ELMo (Peters et al., 2018), GPT-2 (Radford et al.), and BERT (Devlin et al., 2019), have proven their effectiveness on an array of downstream tasks, which is largely attributed to the richer information contained in the representations. However, Clinchant et al. (2019) and Zhu et al. (2020) demonstrated that simply utilizing BERT as the encoder of neural machine translation (NMT) model or initializing the NMT encoder with BERT yields relatively poor translation results. The explanation about the difficulty of utilizing pre-trained LMs in NMT is still an open question. Vázquez et al. (2021) proposed that the discrepancy between embedding spaces created by BERT and vanilla NMT encoder may explain the difficulty of applying BERT to the NMT model.

Recently, several effective methods of integrating BERT into the NMT encoder have been put forward (Clinchant et al., 2019; Rothe et al., 2020; Yang et al., 2020). Xu et al. (2021) utilized a tailored language model trained with bilingual texts to produce embeddings as the input to the Transformer (Vaswani et al., 2017). Despite its good performance, a great amount of bilingual corpus is not always available, not to mention that training another bilingual language model is a cost of time and money as well.

Therefore, we pay more attention to analyzing the approaches integrating widely-used pre-trained LMs, BERT (Devlin et al., 2019) for example, into NMT model. Typically, these methods compute a weighted sum based on the outputs of various attention modules using the representations generated by BERT and each NMT encoder layer (Zhu et al., 2020; Weng et al., 2020; Zhang et al., 2020, 2021). The success of these approaches makes us curious about the changes occurring in encoder embedding spaces after interacting with BERT. In addition, it also attracts us to considering what kind of information provided by BERT may boost the translation quality, semantic or syntactic? The answers to these questions may provide some hints on better utilizing pre-trained LMs in the NMT task.

To this end, we take a complementary comparison between the embedding spaces created by the vanilla Transformer encoders (Vaswani et al., 2017) and the BERT-fused (Zhu et al., 2020) encoders trained with IWSLT14 EN→DE dataset, WMT14 EN→DE dataset, and WMT17 EN→ZH dataset, respectively. We contrast the random cosine similarity (Vázquez et al., 2021), the *SelfSim*, and *IntraSim* proposed by Ethayarajh (2019) between the word representations generated by each of the encoders. Subsequently, we adopt the tasks proposed by Conneau and Kiela (2018) and Hewitt and Manning (2019) to examine the semantic and syntactic information contained in these contextualized representations.

Our experiments demonstrate that compared to keeping the characteristics of BERT embedding spaces (Vázquez et al., 2021), the additional BERT-encoder attention module can ensure the encoder keep its space characteristics, making it easier for the decoder to converge after integrating BERT. Besides, Introducing the output of BERT into the NMT encoder can provide richer syntactic information to boost the translation quality.

Our contribution can be summarized as follows:

- We analyze the differences of embedding spaces between the vanilla Transformer encoder and the encoder integrated BERT, i.e. BERT-fused encoder in this case. To the best of our knowledge, this is the first effort to investigate the discrepancy between the spaces of encoder before and after introducing pre-trained LMs.

- We find that the NMT encoder can benefit a lot from the syntactic information provided by the BERT, which may result in the improvements on the translation quality.

2 Related Work

2.1 Analysis of Contextual Representations

An increasing number of studies have been conducted to analyze the information contained in the contextual embeddings generated by pre-trained LMs. These methods can be roughly divided into two categories:

Probing Tasks. These approaches design simple neuron networks as probes to predict some properties we care about (Shi et al., 2016; McCann et al., 2017; Conneau and Kiela, 2018; Conneau et al., 2018). Hewitt and Manning (2019) designed a structural probe and find that BERT (Devlin et al., 2019) can encode some structural information of words, such as their depth in the dependency parse trees, into word representations. Merchant et al. (2020) not only utilized probing tasks but also adopted the similarity analysis methods to explore the effects of fine-tuning on the representations generated by BERT.

Quantitatively Analysis. Ethayarajh (2019) proposed two metrics, *SelfSim* and *IntraSim*, to compare the word level differences between representations generated by ELMo (Peters et al., 2018), GPT-2 (Radford et al.), and BERT (Devlin et al., 2019). Voita et al. (2019a) utilized Canonical Correlation Analysis (CCA) and mutual information to contrast the contextualized representations trained with various objectives in NMT and LM models. Vázquez et al. (2021) compared the representations spaces between NMT encoder and BERT by the means of *SelfSim* and *IntraSim* as well.

Our work is inspired by the research of Ethayarajh (2019) and Vázquez et al. (2021). We analyze the differences before and after integrating BERT into the NMT encoder with an additional module, attempting to find out how BERT affects the characteristics of embedding spaces created by the NMT encoder and what kind of information provided by BERT boosts the translation performance.

2.2 Pre-trained LMs in NMT

After BERT (Devlin et al., 2019) was proposed, several simple methods of integrating BERT into NMT models have been presented, including utilizing the outputs of pre-trained LMs as the input embeddings (Clinchant et al., 2019) or initializing parameters of the NMT encoder by pre-trained LMs (Rothe et al., 2020).

Zhu et al. (2020) designed additional BERT-encoder and BERT-decoder attention modules and fused the representations from different attention modules in each layer of the NMT model. Similarly, APT framework utilized a layer-aware attention mechanism to fuse the output of each layer in BERT dynamically (Weng et al., 2020). Zhang et al. (2021) integrated the self attention and BERT-encoder attention into a joint attention module, proposing a three-phrase optimization strategy to train the model.

Besides, some efforts have been made to make use of different pre-trained LMs, such as mBERT (Devlin et al., 2019), XLM-R (Conneau et al., 2020), and GottBERT (Scheible et al., 2020), as the embedding layer of the NMT model. Xu et al. (2021) trained a tailored bilingual language model, BiBERT, with 146GB English texts and 145GB German texts, and achieve state-of-the-art translation performance.

3 Preliminary

3.1 Notations

Let $\mathcal{X}$ and $\mathcal{Y}$ be the source language domain and target language domain respectively, which are the sets of sentences corresponding to the languages. For any sentence $\mathbf{x} \in \mathcal{X}$ and $\mathbf{y} \in \mathcal{Y}$, $l_{\mathbf{x}}$ and $l_{\mathbf{y}}$ represent the lengths of $\mathbf{x}$ and $\mathbf{y}$. Let $\mathcal{W}$ indicate the set of words: $\mathcal{W} = \{w_1, \cdots, w_i, \cdots, w_n\}$.

We denote the encoder and decoder of Transformer and BERT as `Enc`, `Dec`, and `BERT` respectively, assuming the `Enc` consists of L layers, while `BERT` contains L_{BERT} layers. The output of the ℓ-th `Enc` layer is denoted by $\mathbf{H}_\ell^E$, which consists of a sequence of vectors $\mathbf{H}_\ell^E = [\mathbf{h}_{\ell,1}^E, \mathbf{h}_{\ell,2}^E, ..., \mathbf{h}_{\ell,n}^E]$, where $\mathbf{h}_{\ell,j}^E \in \mathbb{R}^{d_E}$. Analogously, the output of the ℓ-th `BERT` layer is written as $\mathbf{H}_\ell^B$. It is worth noting that $\mathbf{H}_0^E$ and $\mathbf{H}_0^B$ represent the output of embedding layer in `Enc` and `BERT`, respectively.

Let $\mathrm{Attn}\,(\mathbf{Q}, \mathbf{K}, \mathbf{V})$ denote the attention model, where $\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{l \times d_{\text{model}}}$ are the query, key, and value matrix, respectively. The computation of attention module can be written as:

$$\begin{aligned} &\mathrm{Attn}\,(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \\ &\mathrm{softmax}\left(\frac{(\mathbf{Q}\mathbf{W}^Q) \cdot (\mathbf{K}\mathbf{W}^K)^T}{\sqrt{d_k}}\right)\mathbf{V}\mathbf{W}^V, \end{aligned} \tag{1}$$

where $\mathbf{W}_Q \in \mathbb{R}^{d_{model} \times d_k}$, $\mathbf{W}_K \in \mathbb{R}^{d_{model} \times d_k}$, and $\mathbf{W}_V \in \mathbb{R}^{d_{model} \times d_v}$ are the parameters to be learned. $\mathrm{Attn}\,(\mathbf{Q}, \mathbf{K}, \mathbf{V})$ is implemented as a multi-head attention model, whose details can be referred to Vaswani et al. (2017).

Define $\mathrm{FFN}(\cdot)$ as Vaswani et al. (2017) did:

$$\mathrm{FFN}(\mathbf{h}) = \mathrm{ReLU}\,(\mathbf{h}\mathbf{W}_1 + \mathbf{b}_1)\,\mathbf{W}_2 + \mathbf{b}_2, \tag{2}$$

where $\mathbf{W}_1, \mathbf{W}_2 \in \mathbb{R}^{d_{model} \times d_{model}}$ and $\mathbf{b}_1, \mathbf{b}_2 \in \mathbb{R}^{d_{model}}$ are trainable parameters.

We denote the cosine similarity between two vectors $\mathbf{h}_i$ and $\mathbf{h}_j$ by $\cos\,(\mathbf{h}_i, \mathbf{h}_j)$. The Euclidean norm of vector $\mathbf{h}_i$ is denoted by $\|\mathbf{h}_i\|_2$.

3.2 Transformer

Transformer model (Vaswani et al., 2017) is one of the most effective models in a wide range of NLP tasks. The overview of its structure is shown in Figure 1.

In the ℓ-th `Enc` layer, $\mathbf{H}_\ell^E$ is computed as follows:

$$\begin{aligned} \mathbf{R}_\ell^E &= \mathrm{LN}\left(\mathbf{H}_{\ell-1}^E + \mathrm{Attn}\left(\mathbf{H}_{\ell-1}^E, \mathbf{H}_{\ell-1}^E, \mathbf{H}_{\ell-1}^E\right)\right), \\ \mathbf{H}_\ell^E &= \mathrm{LN}\left(\mathbf{R}_\ell^E + \mathrm{FFN}\left(\mathbf{R}_\ell^E\right)\right). \end{aligned}$$

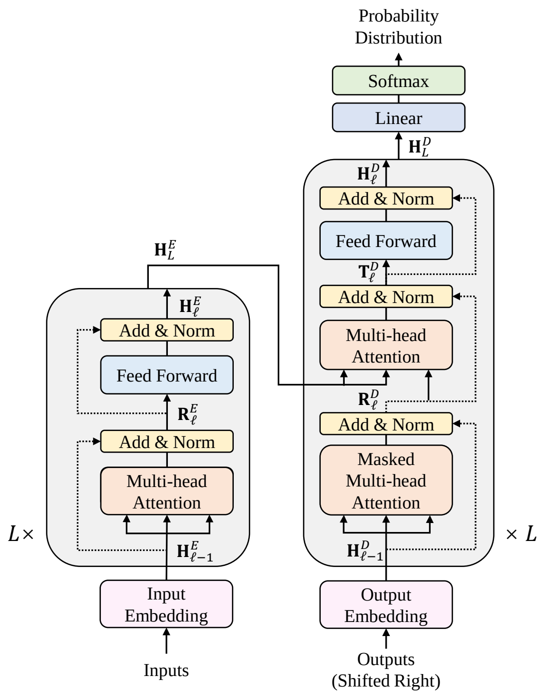


Figure 1: Overview of the structure of the vanilla Transformer.

where $\mathrm{LN}(\cdot)$ is layer normalization. The computation of $\mathrm{Attn}(\cdot)$ and $\mathrm{FFN}(\cdot)$ follows Eqn.(1) and Eqn.(2) respectively.

The computation of the output of ℓ-th `Dec` layer, $\mathbf{H}_\ell^D$, is shown below:

$$\mathbf{R}_\ell^D = \mathrm{LN}(\mathbf{H}_{\ell-1}^D + \mathrm{Attn}(\mathbf{H}_{\ell-1}^D, \mathbf{H}_{\ell-1}^D, \mathbf{H}_{\ell-1}^D)), \tag{3}$$

$$\mathbf{T}_\ell^D = \mathrm{LN}(\mathbf{R}_\ell^D + \mathrm{Attn}(\mathbf{R}_\ell^D, \mathbf{H}_L^E, \mathbf{H}_L^E)), \tag{4}$$

$$\mathbf{H}_\ell^D = \mathrm{LN}(\mathbf{T}_\ell^D + \mathrm{FFN}(\mathbf{T}_\ell^D)). \tag{5}$$

3.3 BERT-fused encoder

Up to now, there are several methods of integrating BERT into Transformer model. Among all of these approaches, `BERT-Enc Attn` and `BERT-Dec Attn` adopted by BERT-fused (Zhu et al., 2020) and BERT-JAM (Zhang et al., 2021) can significantly boost the translation quality of Transformer model. We will briefly describe the structure of the BERT-fused encoder, which is shown in Figure 2.

In the ℓ-th layer in the BERT-fused encoder, $\mathbf{H}_\ell^E$ is computed as follows:

$$\begin{aligned} \mathbf{R}_\ell^E &= \mathrm{LN}(\mathbf{H}_{\ell-1}^E + \gamma_\ell \mathrm{Attn}(\mathbf{H}_{\ell-1}^E, \mathbf{H}_{\ell-1}^E, \mathbf{H}_{\ell-1}^E) \\ &\quad + (1-\gamma_\ell)\mathrm{Attn}(\mathbf{H}_{\ell-1}^E, \mathbf{H}_{L_{\text{BERT}}}^B, \mathbf{H}_{L_{\text{BERT}}}^B)), \\ \mathbf{H}_\ell^E &= \mathrm{LN}(\mathbf{R}_\ell^E + \mathrm{FFN}(\mathbf{R}_\ell^E)), \end{aligned}$$

where $\gamma_\ell = 0.5$ in the BERT-fused model (Zhu et al., 2020).

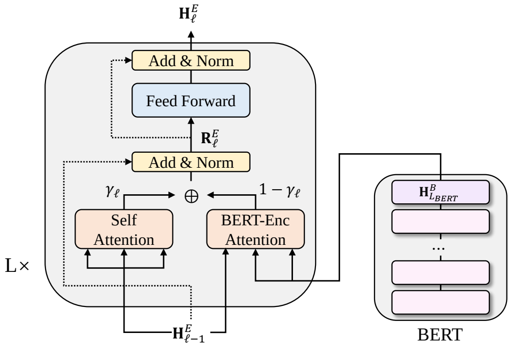


Figure 2: Overview of the structure of the BERT-fused encoder.

4 Methodology

In this section, we will introduce several metrics to evaluate the isotropy and contextuality of the embedding space. Besides, we take a brief introduction to a number of tasks which are useful for determining how much semantic information and syntactic information contained in the word representations and sentence embeddings.

4.1 Characteristics of Embedding Spaces

4.1.1 The Isotropy of Spaces

It is essential to take the characteristics of embedding spaces into consideration before comparing the similarity between word representations in different spaces. Suppose that all vectors are distributed in a narrow space, the cosine value of any two word representations will naturally approach 1, but this does not guarantee that the two words are similar to each other. Therefore, the concept of isotropy is introduced: an embedding space is described as isotropic if vectors in it are directional uniformity. Otherwise, it is called an anisotropic space.

In this paper, we estimate the level of isotropy of the embedding space by the cosine similarity and Euclean distance between the representations of uniformly randomly sampled words (Ethayarajh, 2019; Vázquez et al., 2021), denoted by *CosSim* and *EucDis*. For any two words $w_i, w_j \in \mathcal{W}$, suppose their corresponding word representations in ℓ-th layer are $\mathbf{h}^E_{\ell,pos_i}$ and $\mathbf{h'}^E_{\ell,pos_j}$. The computations of $CosSim_\ell$ and $EucDis_\ell$ can be written as:

$$CosSim_\ell(w_i, w_j) = \cos(\mathbf{h}^E_{\ell,pos_i}, \mathbf{h'}^E_{\ell,pos_j}),$$

$$EucDis_\ell(w_i, w_j) = \|\mathbf{h}^E_{\ell,pos_i} - \mathbf{h'}^E_{\ell,pos_j}\|_2.$$

If the average value of *CosSim* is concentrated around 0, vectors in the space are almost orthogonal to each other, indicating the space is more likely to be isotropic. Otherwise, the space is anisotropic. In addition, if the value of *EucDis* is relatively small on average, the embedding space is probably narrow.

4.1.2 The Contextuality of Spaces

Apart from the *CosSim* and *EucDis*, we also adopt two contextuality metrics presented in Ethayarajh (2019) and Vázquez et al. (2021).

SelfSim: The averaged cosine similarity between the same word in different sentences, namely different contexts. If the average value of *SelfSim* is relatively small, different contexts will make the embeddings of the same word vary. In such a space, the word representations are much more contextual.

IntraSim: The average cosine similarity between representations of words in a sentence and the mean pooled sentence embedding. The *IntraSim* reflects how context-specificity manifests in the embedding space. If *IntraSim*$(\mathbf{x})$ is high while the *SelfSim*(w), $\forall w \in \mathbf{x}$ is low, it indicates that the encoder tends to make the word representation to be contextual by gathering the representations of words in the same sentence together and keeping the word representations in different contexts away from each other.

It is worth noting that both of these two metrics need to subtract the average value of *CosSim* of the corresponding layer, ensuring the characteristics are corrected for deviation (Vázquez et al., 2021).

4.2 Semantic Information

We adopt the Sentence Textual Similarity (STS) tasks provided by SentEval[1] (Conneau and Kiela, 2018) to evaluate the information contained in the sentence embeddings generated by different encoders. Note that we use the average of embeddings of words contained in the sentence as the sentence embedding.

The STS task is first presented by Agirre et al. (2012). Given a sentence pair $\{\mathbf{x}, \mathbf{y}\}$, its object is to predict how similar the meanings of these sentences are by giving a continuous-valued score between 0 and 5.

4.3 Syntactic Information

We employ the structural probes proposed by Hewitt and Manning (2019) to evaluate the syntax information encoded by the word representations.

[1]https://github.com/facebookresearch/SentEval

More specifically, we generate the dependency parse tree of data in the SentEval (Agirre et al., 2012, 2013, 2014, 2015, 2016) using stanza (Qi et al., 2020). The probing tasks are as follows:

Distance. Predict the distance between any two words in the dependency parse tree.

Depth. Predict the depth of each word in the dependency parse tree.

We train a positive semi-determined matrix $\mathbf{B} \in \mathbb{R}^{d_{model} \times rank}$ for each task. We set $rank = 64$ in the experiments. The Spearman correlation coefficient ρ is reported as the experiment result.

Besides, we adopt three tasks provided by SentEval (Conneau and Kiela, 2018) to evaluate the syntactic information contained in the sentence embedding:

BShift. Predict whether two consecutive tokens within the sentence have been inverted.

TreeDepth. Predict the maximum depth of the syntactic tree of the sentence. It can be viewed as a simplified version of probing tasks proposed by Hewitt and Manning (2019).

TopConst. Predict the top-level constituents of constituency parse tree from 20 classes.

We train a Multi-Layer Perceptron classifier with a single hidden layer containing 50 neurons based on the sentence embeddings for each of the task and report the accuracy as the final result.

4.4 Models

The models we adopted in the comparison experiments are shown as follows[2]. In order to ensure the universality of discrepancy between these encoders, we utilize three different datasets to train the NMT models: IWSLT14 EN→DE dataset[3], WMT14 EN→DE dataset[4] and WMT17 EN→ZH dataset[5] respectively. The details of datasets, the proprocessing methods, and training settings can be referred to the Appendix A.

vanilla Transformer encoder: the encoder of a traditional Transformer model. We train a Transformer model based on Fairseq[6], a popular sequence modeling toolkit.

BERT-fused encoder: the encoder of the BERT-fused model. Note that we use the standard decoder following Eqn.(3-5) to avoid introducing other new variables. This model is implemented with Fairseq toolkit and trained along with the `bert-base-uncase`[7] provided by the HuggingFace library (Wolf et al., 2019).

BERT. We also utilize the pretrained `bert-base-uncased` model from the HuggingFace library (Wolf et al., 2019) as an auxiliary for subsequent analysis.

The BLEU scores of these two models on different test sets[8] are shown in Table 1, which are consistent with previous studies.

Models	IWSLT14 EN→DE	WMT14 EN→DE	WMT17 EN→ZH
Transformer	28.19	28.88	33.11
BERT-fused*	**30.10**	**30.07**	**34.39**

Table 1: BLEU scores trained with different datasets. Note that we change the decoder of BERT-fused model to the standard Transformer decoder.

According to the results shown in Table 1, the NMT model integrated with BERT obtains a significant boost on translation quality on the smaller size dataset. Therefore, we mainly present the comparison between models trained with IWSLT14 EN→DE dataset in the following sections. The corresponding experiment results of models trained with WMT14 EN→DE dataset and WMT17 EN→ZH dataset are displayed in Appendix B.

5 Characteristics of Embedding Spaces

Following the research of Ethayarajh (2019) and Vázquez et al. (2021), we analyze the embeddings spaces based on the data gathered from the SemEval Semantic Textual Similarity tasks from 2012 to 2016 (Agirre et al., 2012, 2013, 2014, 2015, 2016).

5.1 The Isotropy of Spaces

We first take a comparison of *CosSim* distribution, which is the indicator of the isotropy of spaces. The results are shown in Figure 3.

According to Figure 3a, the mean value of the *CosSim* generated by vanilla Transformer encoder increases slightly towards higher layers, with the

[2]We only compare two models because of there is no published source code for other models

[3]https://workshop2014.iwslt.org/

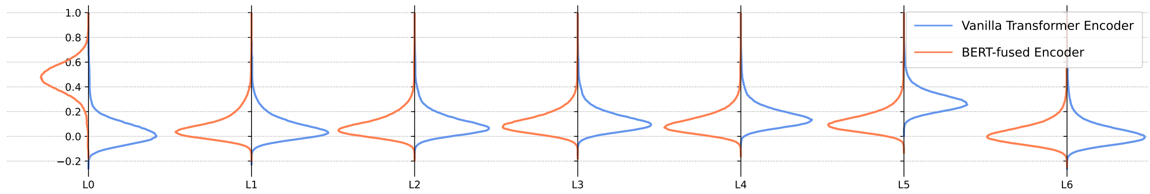
[4]https://www.statmt.org/wmt14/translation-task.html

[5]https://www.statmt.org/WMT17/translation-task.html

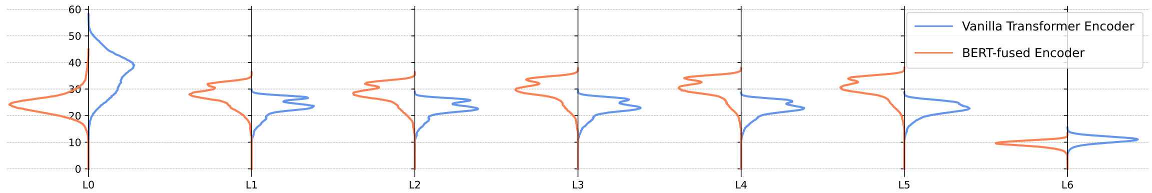
[6]https://github.com/pytorch/fairseq

[7]https://huggingface.co/bert-base-uncased/tree/main

[8]We use the concatenation of IWSLT14.TED.dev2010, IWSLT14.TEDX.dev2012, IWSLT14.TED.tst2010, IWSLT14.TED.tst2011, and IWSLT14.TED.tst2012 as the test set for IWSLT14 EN→DE; newstest2014 and newstest2017 are used as the test set for WMT14 EN→DE and WMT17 EN→ZH respectively.



(a) *CosSim* Distribution



(b) *EucDis* Distribution

Figure 3: *CosSim* (top) and *EucDis* (bottom) distributions of uniform-sampled word. Both of the vanilla Transformer encoder and BERT-fused encoder are trained with IWSLT14 EN→DE dataset. From left to right is layer 0 to layer 6.

exception of a drop at the last layer (L6). On the other hand, the *CosSim* of embeddings produced by BERT-fused encoder concentrates around 0.5 at the embedding layer (L0) and suddenly declines to 0.0 at the first encoder layer (L1), indicating the level of isotropy surges. From the first encoder layer to the last encoder layer (L6), this embedding space maintains an isotropic state except a minor fluctuation. It is worth mentioning that these two embedding spaces achieve isotropic stage in the L6 in spite of the different variation trends in previous layers.

As for the *EucDis* distribution displayed in Figure 3b, the overall changing tendency of these two embedding spaces are the same. The falling of *EucDis* value indicates the beginning wide space gradually shrinks to a relatively narrow one. Nevertheless, the space of vanilla Transformer encoder undergoes contractions twice, at the first encoder layer (L1) and the last encoder layer (L6) respectively; while the space created by BERT-fused encoder only contracts once at the last layer.

Combined the tendency of *CosSim* and *EucDis*, the NMT encoder tends to make the embeddings distributed randomly and gradually shrink the size of the effective space. The changing tendency of the size of space provides an explanation from another perspective for the feasibility of pruning the Transformer model (Voita et al., 2019b).

5.2 The Contextuality of Spaces

Afterwards, we compare the values of *SelfSim* and *IntraSim* between the vanilla Transformer encoder and BERT-fused encoder layer by layer. According to the Figure 4, these two encoder display remarkably different trends.

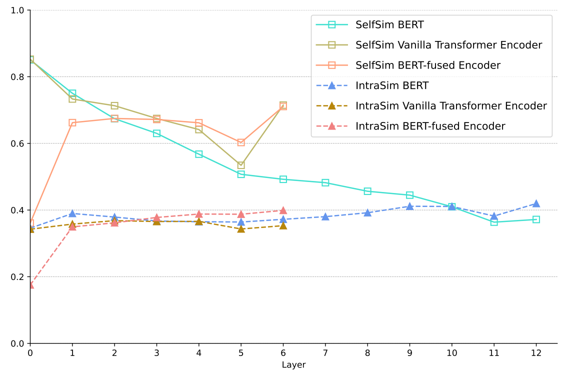


Figure 4: The *SelfSim* and *IntraSim* results of BERT, vanilla Transformer encoder, and the BERT-fused encoder. Note that layer 0 correspond to the embedding layer.

The vanilla Transformer encoder gets a relatively high *SelfSim* score in the layer 0, illustrating the embedding layer produces a less contextual representations for each word. The value of *SelfSim* declines constantly until the penultimate layer and increases suddenly in the last layer. This tendency indicates that the vanilla Transformer encoder learns to add more contextual information into word representations as the layer increases. However, vectors corresponding to the same word become much more similar in the last layer. It seems that the decoder may not need the word representation to contain too much context information.

On the contrary, the value of *SelfSim* of BERT-fused encoder rises rapidly at the first encoder layer and fluctuate slightly in the following layers. This tendency demonstrates the representations of the same words are becoming more and more similar to each other.

IntraSim value of the vanilla Transformer encoder raises slightly and then declines. Taking the declining of the value of *SelfSim* into consideration, the model generates contextual representations by gathering the words in the same sentence together. In the last layer, the value of *SelfSim* is high while the *IntraSim* value is relatively low, demonstrating that representations of the same word gather together while the representations of different words are pushed away.

In addition, the *IntraSim* value of the BERT-fused encoder increases gradually, revealing the words belong to the same sentence become similar as well. Considering its upward trend is not as steep as *SelfSim*, the information related to the word itself still dominates the change of word representation.

Based on the tendency shown in the Figure 4, we summarize two interesting findings:

- *SelfSim* and *IntraSim* scores of these two encoders are significantly close to each other in the last layer.

- Rather then imitating the characteristics of embedding space created by BERT, the representations generated by the BERT-fused encoder are still less contextual.

We hypothesize that the characteristics of encoder embedding space is shaped by the decoder at the same time. This less contextual representations maybe exactly what the decoder needs when executing decoding operations. Besides, considering the parameters introduced by the `BERT-Enc Attn` modules only occupy a small part of the number of parameters compared to the traditional Transformer model (14.4% to be specific), the encoder spaces may not change a lot.

Nevertheless, it is natural to consider how BERT works in the NMT encoder under this assumption. We attempt to answer this question by evaluating the outputs of encoders with different tasks in Section 5.3.

5.3 Semantic and Syntactic Tasks

In order to better handle what kind of information are utilized by the BERT-fused encoder, we not only present the experiments results on the vanilla Transformer encoder and BERT-fused encoder, but also check the outputs of `Self Attn` and `BERT-Enc Attn` module in the BERT-fused encoder, respectively. More specifically, *Self Attn* denotes the output of the BERT-fused encoder when $\gamma_\ell^E = 1.0,\ \forall \ell \in L$; *BERT-Enc Attn* denotes the output of the BERT-fused encoder when $\gamma_\ell^E = 0.0,\ \forall \ell \in L$.

5.3.1 Semantic Information

The experiment results of tasks related to the semantic information are shown in Table 2. The most notable point is that the BERT-fused model obtains a remarkably higher Spearman correlation coefficient than the mean pooled BERT embeddings with a margin of 14.56 after introducing the `BERT-Enc Attn` module.

In addition, even the *Self Attn* outperforms the vanilla Transformer encoder. Because of the existence of additional module, the `Self Attn` can focus on the semantic information. The *BERT-Enc Attn* performs better compared to *Self Attn*. We hypothesize that `BERT-Enc Attn` module focus on parsing the semantics of sentences from the contextual representations generated by BERT.

Besides, the Spearman ρ of BERT-fused model is higher than the results of utilizing `Self Attn` module and `BERT-Enc Attn` module alone. It seems that these two modules have their own emphasis, and the BERT-fused encoder finds a way to balance them.

5.3.2 Syntactic Information

As Table 3 indicates, BERT encodes rich syntax information in the word representations. Therefore, it achieves good results in probing tasks related to both word representations and sentence embeddings, especially on predicting word order and top-level constituents. On the contrary, the vanilla Transformer encoder performs poorly on these two tasks. However, it performs well on the tasks related to word representations, indicating that this encoder pays more attention to encoding structural information into word representations but neglects the overall structure of sentences.

Among all of these models, *Self Attn* performs worst on all of the tasks. Compared to its good performance on the semantic tasks, we can conclude that the `Self Attn` module prefers to focusing on extracting semantics from representations. Furthermore, the *BERT-Enc Attn* obtains a comparable score to the BERT, illustrating that outputs of BERT

Encoder	STS12	STS13	STS14	STS15	STS16	STS-B	Avg.
BERT	30.87	59.90	47.73	60.29	63.73	47.29	51.63
vanilla Transformer encoder	50.44	63.87	58.25	70.17	68.91	64.65	62.71
Self Attn	52.28	64.27	58.82	71.34	69.84	66.71	63.68
BERT-Enc Attn	51.30	69.02	58.71	71.23	**73.46**	66.02	65.09
BERT-fused encoder	**53.55**	**69.19**	**60.29**	**72.54**	73.26	**68.28**	**66.19**

Table 2: Spearman correlation coefficient ρ between cosine similarity of sentence embeddings and gold labels on STS tasks from 2012 to 2016 and STS Benchmark test set. *Self Attn* means only using the output of `Self Attn` modules in each layer of the BERT-fused encoder; *BERT-ENC Attn* represents only using the output of `BERT-ENC Attn` modules in each layer of the BERT-fused encoder.

Encoder	Distance	Depth	BShift	TreeDepth	TopConst
BERT	74.16	**78.79**	**88.77**	36.21	**72.62**
vanilla Transformer encoder	**77.36**	78.06	64.11	37.32	67.91
Self Attn	71.77	64.31	60.14	36.34	67.03
BERT-Enc Attn	71.73	74.35	85.47	37.50	72.19
BERT-fused encoder	71.85	72.34	84.31	**38.72**	71.46

Table 3: Results of syntactic probing tasks related to the word representations and sentence embeddings. Note that we use the mean pooled word representations as the sentence embeddings for the last three tasks. The higher Spearman correlation coefficient ρ for Distance and Depth tasks indicates the word representations encode richer structural information; while the lower accuracy on Bshift, TreeDepth, and TopConst tasks indicates that the sentence embeddings contain less syntactic information.

can indeed provide more syntactic information.

Combining the `Self Attn` and `BERT-Enc Attn` modules, the BERT-fused encoder obtains a significantly higher accuracy than the vanilla Transformer encoder on the BShift and TopConst tasks, proving that the BERT assists the model by providing much more information about the syntax of sentences.

This finding provides an explanation for the success of utilizing BIBERT (Xu et al., 2021). According to the results of our experiment, BIBERT performs well on the syntactic tasks, getting 82.22, 84.05, 89.96, 43.43, and 79.22 for each task. Compared to the significantly poor performance on the semantic tasks (39.56 on average), the syntax information provided by the pre-trained LMs plays an important role in boosting the translation quality.

6 Conclusion

Although pre-trained language models have been widely applied in various natural language processing tasks, it takes great efforts to introduce these models into neural machine translation model. This paper provides an analysis of the differences between the spaces created by the vanilla Transformer encoder and the encoder integrated with BERT. We find that introducing BERT through BERT-encoder attention module will not make the characteristics of original space change a lot, which may be one of the reasons for its success. Subsequent experiments concern with the semantic and syntactic information reveal that the outputs of BERT provides rich syntactic information to boost the translation quality of the NMT model.

References

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Iñigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, German Rigau, Larraitz Uria, and Janyce Wiebe. 2015. SemEval-2015 task 2: Semantic textual similarity, English, Spanish and pilot on interpretability. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 252–263, Denver, Colorado. Association for Computational Linguistics.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2014. SemEval-2014 task 10: Multilingual semantic textual similarity. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 81–91, Dublin, Ireland. Association for Computational Linguistics.

Eneko Agirre, Carmen Banea, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2016. SemEval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 497–511, San Diego, California. Association for Computational Linguistics.

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. SemEval-2012 task 6: A pilot on semantic textual similarity. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 385–393, Montréal, Canada. Association for Computational Linguistics.

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. *SEM 2013 shared task: Semantic textual similarity. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, pages 32–43, Atlanta, Georgia, USA. Association for Computational Linguistics.

Stephane Clinchant, Kweon Woo Jung, and Vassilina Nikoulina. 2019. On the use of BERT for neural machine translation. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 108–117, Hong Kong. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Alexis Conneau and Douwe Kiela. 2018. Senteval: An evaluation toolkit for universal sentence representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation, LREC 2018, Miyazaki, Japan, May 7-12, 2018*. European Language Resources Association (ELRA).

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2126–2136, Melbourne, Australia. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. 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)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? Comparing the geometry of BERT, ELMo, and GPT-2 embeddings. 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)*, pages 55–65, Hong Kong, China. Association for Computational Linguistics.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. 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)*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: Contextualized word vectors. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 6294–6305.

Amil Merchant, Elahe Rahimtoroghi, Ellie Pavlick, and Ian Tenney. 2020. What happens to BERT embeddings during fine-tuning? In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 33–44, Online. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. 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)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 101–108, Online. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners.

Sascha Rothe, Shashi Narayan, and Aliaksei Severyn. 2020. Leveraging pre-trained checkpoints for sequence generation tasks. *Transactions of the Association for Computational Linguistics*, 8:264–280.

Raphael Scheible, Fabian Thomczyk, Patric Tippmann, Victor Jaravine, and Martin Boeker. 2020. Gottbert: a pure german language model. *CoRR*, abs/2012.02110.

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.

Xing Shi, Inkit Padhi, and Kevin Knight. 2016. Does string-based neural MT learn source syntax? In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1526–1534, Austin, Texas. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Raúl Vázquez, Hande Celikkanat, Mathias Creutz, and Jörg Tiedemann. 2021. On the differences between BERT and MT encoder spaces and how to address them in translation tasks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: Student Research Workshop*, pages 337–347, Online. Association for Computational Linguistics.

Elena Voita, Rico Sennrich, and Ivan Titov. 2019a. The bottom-up evolution of representations in the transformer: A study with machine translation and language modeling objectives. 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)*, pages 4396–4406, Hong Kong, China. Association for Computational Linguistics.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019b. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5797–5808, Florence, Italy. Association for Computational Linguistics.

Rongxiang Weng, Heng Yu, Shujian Huang, Shanbo Cheng, and Weihua Luo. 2020. Acquiring knowledge from pre-trained model to neural machine translation. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 9266–9273. AAAI Press.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.

Haoran Xu, Benjamin Van Durme, and Kenton Murray. 2021. Bert, mbert, or bibert? A study on contextualized embeddings for neural machine translation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 6663–6675. Association for Computational Linguistics.

Jiacheng Yang, Mingxuan Wang, Hao Zhou, Chengqi Zhao, Weinan Zhang, Yong Yu, and Lei Li. 2020. Towards making the most of BERT in neural machine translation. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 9378–9385. AAAI Press.

Jia-Rui Zhang, Hongzheng Li, Shumin Shi, Heyan Huang, Yue Hu, and Xiangpeng Wei. 2020. Dynamic attention aggregation with BERT for neural machine translation. In *2020 International Joint Conference on Neural Networks, IJCNN 2020, Glasgow, United Kingdom, July 19-24, 2020*, pages 1–8. IEEE.

Zhebin Zhang, Sai Wu, Dawei Jiang, and Gang Chen. 2021. BERT-JAM: maximizing the utilization of BERT for neural machine translation. *Neurocomputing*, 460:84–94.

Jinhua Zhu, Yingce Xia, Lijun Wu, Di He, Tao Qin, Wengang Zhou, Houqiang Li, and Tie-Yan Liu. 2020. Incorporating BERT into neural machine translation. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

A Experimental Setup

A.1 Data Preprocessing

Sentences in all datasets were encoded using byte-pair encoding (BPE) (Sennrich et al., 2016) with subword-nmt[9]. FOr the IWSLT14 EN→DE and WMT14 EN→DE datasets, we adopted $10k$ and $40k$ merge operations respectively to build a shared

[9]https://github.com/rsennrich/subword-nmt

dictionary. As for the WMT18 EN→ZH dataset, the merge operation is set as $32k$.

Translation pairs were batched together by approximate sequence length. Each training batch contained a set of translation pairs containing approximately $4k$ source tokens.

A.2 Model Paramters

We follow the setup of Transformer base model (Vaswani et al., 2017). More precisely, the number of layers in the encoder and in the decoder is $L = 6$. We employ $h = 4$ attention heads for the IWSLT14 EN→DE dataset and $h = 8$ for the WMT14 EN→DE and WMT18 EN→ZH datasets. The dimensionality of input and output is $d_{model} = 512$, and the inner-layer of a feedforward networks has dimensionality $d_{ff} = 2048$.

We set dropout rate as 0.1, 0.1 and 0.25 for IWSLT14 EN→DE, WMT14 EN→DE, and WMT18 EN→ZH, respectively.

A.3 Optimizer

Adam optimizer (Kingma and Ba, 2015) is adopted with $\beta_1 = 0.9, \beta_2 = 0.98$. We vary the learning rate over the course of training according to the formula:

$$lr_{step} = \begin{cases} lr_{init} + \dfrac{step}{warmup} * (lr - lr_{init}), \\ \qquad\qquad \text{if} \quad step < warmup, \\ lr * \dfrac{\sqrt{warmup}}{\sqrt{step}}, \\ \qquad\qquad \text{if} \quad step \geq warmup. \end{cases}$$

We set $warmup = 4000$ and $lr_{init} = 1 \times 10^{-7}$ in the all training procedure. we employ $lr = 5 \times 10^{-4}$, $lr = 7 \times 10^{-4}$, and $lr = 5 \times 10^{-4}$ for each of the dataset.

B Other Experiment Results

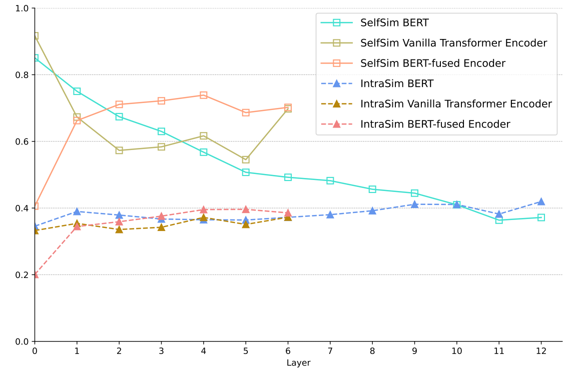


Figure 5: The *SelfSim* and *IntraSim* results of BERT, vanilla Transformer encoder, and the BERT-fused encoder. Both of the vanilla Transformer encoder and BERT-fused encoder are trained with WMT14 EN→DE dataset. Note that layer 0 correspond to the embedding layer.

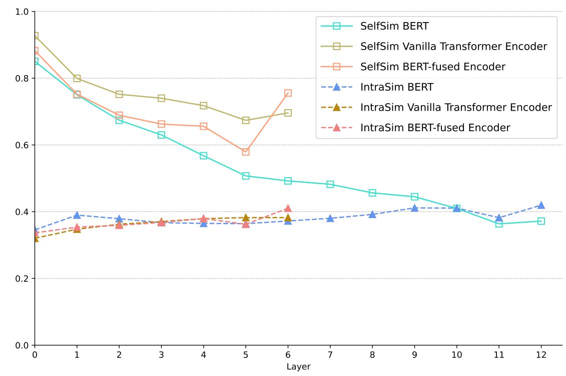
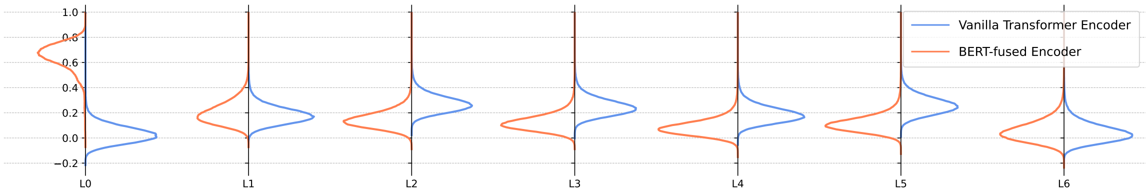
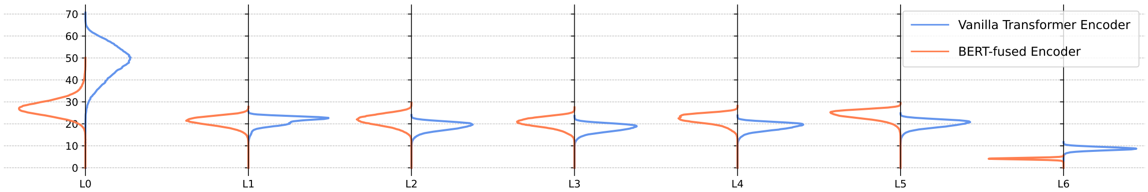


Figure 6: The *SelfSim* and *IntraSim* results of BERT, vanilla Transformer encoder, and the BERT-fused encoder. Both of the vanilla Transformer encoder and BERT-fused encoder are trained with WMT17 EN→ZH dataset. Note that layer 0 correspond to the embedding layer.

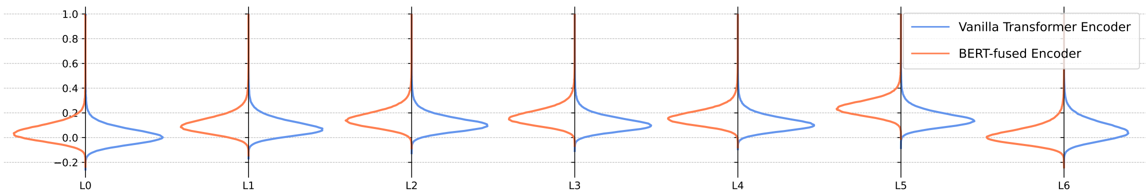


(a) Cosine Similarity Distribution

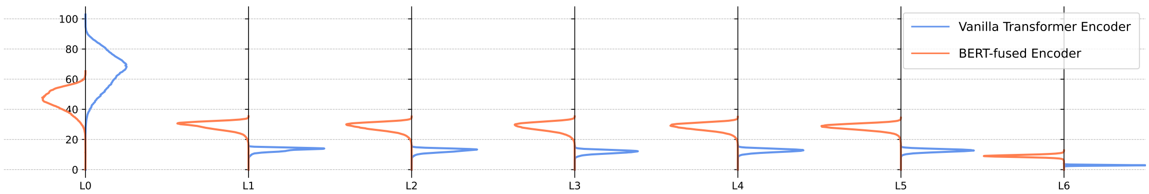


(b) Euclidean Distance Distribution

Figure 7: *CosSim* (top) and *EucDis* (bottom) distributions of uniform-sampled word. Both of the vanilla Transformer encoder and BERT-fused encoder are trained with WMT14 EN→DE dataset. From left to right is layer 0 to layer 6.



(a) Cosine Similarity Distribution



(b) Euclidean Distance Distribution

Figure 8: *CosSim* (top) and *EucDis* (bottom) distributions of uniform-sampled word. Both of the vanilla Transformer encoder and BERT-fused encoder are trained with WMT17 EN→ZH dataset. From left to right is layer 0 to layer 6.