Fast and Accurate Deep Bidirectional Language Representations for Unsupervised Learning

Joongbo Shin, Yoonhyung Lee, Seunghyun Yoon, Kyomin Jung
Seoul National University
Republic of Korea
{jbshin, cpi1234, mysmilish, kjung}@snu.ac.kr

Abstract

Even though BERT achieves successful performance improvements in various supervised learning tasks, applying BERT for unsupervised tasks still holds a limitation that it requires repetitive inference for computing contextual language representations. To resolve the limitation, we propose a novel deep bidirectional language model called **T**ransformer-based **T**ext **A**utoencoder (T-TA). The T-TA computes contextual language representations without repetition and has benefits of the deep bidirectional architecture like BERT. In runtime experiments on CPU environments, the proposed T-TA performs over six times faster than the BERT-based model in the reranking task and twelve times faster in the semantic similarity task. Furthermore, the T-TA shows competitive or even better accuracies than those of BERT on the above tasks[1].

1 Introduction

A language model is an essential component in many NLP applications ranging from automatic speech recognition (ASR) (Chan et al., 2016; Panayotov et al., 2015) to neural machine translation (NMT) (Sutskever et al., 2014; Sennrich et al., 2016; Vaswani et al., 2017). Recently, BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al., 2019) and its variations have brought significant improvements in learning natural language representation, and they have achieved state-of-the-art performances on various downstream tasks such as GLUE benchmark (Wang et al., 2019) and question answering (Rajpurkar et al., 2016). This success of BERT continues in various unsupervised tasks such as the N-best list reranking for ASR and NMT (Shin et al., 2019; Salazar et al., 2019), showing that deep bidirectional language models are useful in unsupervised applications as well.

However, when applying the BERT to unsupervised learning tasks, there exists significant inefficiency in computing language representations at the inference stage (Salazar et al., 2019). During training, the BERT uses the *masked language modeling* (MLM) objective, which is to predict the original ids of explicitly masked words from the input. Due to the MLM objective, each contextual word representation should be computed by a two-step process of masking a word in the input and feeding it into the BERT. During inference, therefore, this process repeats n times to obtain obtain the representations of the whole words of a text sequence (Wang and Cho, 2019; Shin et al., 2019; Salazar et al., 2019), resulting in computational complexity of $O(n^3)$[2] in terms of the number n of words. Hence, it is necessary to reduce the computational complexity when we apply the model to the case where the inference time is considered critical, *e.g.* mobile environments and real-time systems (Sanh et al., 2019; Lan et al., 2019). Faced with this limitation of the BERT, we raise a new research question: "Can we make a deep bidirectional language model that has minimal inference time while maintaining the accuracy of BERT?"

In this paper, we answer "YES" to the above question by proposing a novel bidirectional language model named **T-TA**: **T**ransformer-based **T**ext **A**utoencoder that has the reduced computational complexity of $O(n^2)$ when applying the model to the unsupervised applications. The proposed model is trained with a new learning objective named *language autoencoding* (LAE). The LAE let the target labels to be the same as the text input, and its objective is to predict every token in the input sequence at once without merely copying

[1] Code is available at https://github.com/joongbo/tta

[2] $O(n^2)$ is from the per-layer complexity of Transformer (Vaswani et al., 2017).

the input to the output. To learn the proposed objective, we devise a **diagonal masking** operation and an **input isolation** mechanism inside the T-TA based on the Transformer encoder (Vaswani et al., 2017). These components enable the proposed T-TA to compute contextualized language representations at once while maintaining the benefits of the deep bidirectional architecture of BERT.

We conduct a series of experiments on two unsupervised tasks: the N-best list reranking and the unsupervised semantic textual similarity. First, in the runtime experiments on CPU environments, we show that the proposed T-TA is 6.35 times faster than the BERT-based model in the reranking task, and 12.7 times faster in the semantic similarity task. Second, even with this faster inference, the T-TA achieves competitive performances to BERT on reranking tasks. Furthermore, the T-TA outperforms BERT up to 8 points in Pearson's r on unsupervised semantic textual similarity tasks.

2 Related Works

When referring to the autoencoder for language modeling, sequence-to-sequence learning approaches have been commonly used. These approaches encode a given sentence into a compressed vector representation, followed by a decoder which reconstructs the original sentence from the *sentence-level* representation (Sutskever et al., 2014; Cho et al., 2014; Dai and Le, 2015). To the best of our knowledge, however, none of them considered an autoencoder that encodes *word-level* representations like BERT without the autoregressive decoding process.

There have been many studies on neural network-based language models for word-level representations. Distributed word representations were proposed and gained huge interests as they were considered to be fundamental building blocks for the natural language processing tasks (Rumelhart et al., 1986; Bengio et al., 2003; Mikolov et al., 2013b). Recently, researchers explored contextualized representations of text where each word will have different representations depending on the context (Peters et al., 2018; Radford et al., 2018). More recently, the Transformer-based deep bidirectional model was proposed and applied to the various supervised-learning tasks with a huge success (Devlin et al., 2019).

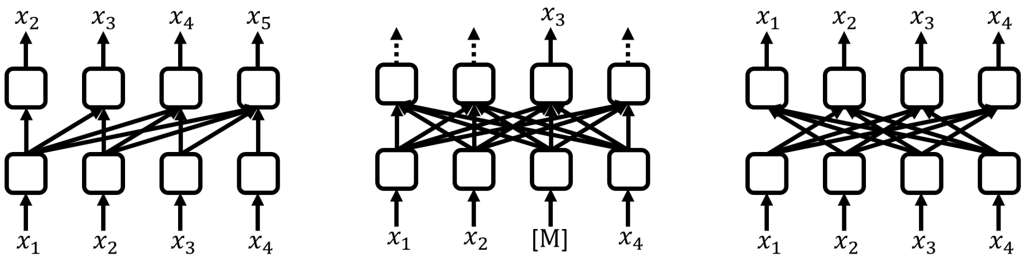
For unsupervised tasks, researchers adopted the recent language-representation models and investigated their effectiveness. One typical example is the N-best list reranking for ASR and NMT tasks. In particular, there have been researches integrating the left-to-right and the right-to-left language models (Arisoy et al., 2015; Chen et al., 2017; Peris and Casacuberta, 2015) so as to outperform conventional unidirectional language models (Mikolov et al., 2010; Sundermeyer et al., 2012) in these tasks. Furthermore, BERT-based approaches have been explored and have achieved significant performance improvements on these tasks based on the fact that; bidirectional language models yield the pseudo-log-likelihood of a given sentence; this score is useful in ranking the n-best hypotheses (Wang and Cho, 2019; Shin et al., 2019; Salazar et al., 2019).

Another line of research includes reducing the computation time and memory consumption of BERT. Lan et al. (2019) proposed parameter-reduction techniques, factorized embedding parameterization and cross-layer parameter sharing, and achieved 18 times fewer parameters and 1.7 times faster training time. With a similar research direction, Sanh et al. (2019) presented a method to pretrain a smaller model that can be finetuned for the downstream task, and achieved a 1.4 times lower parameter count with 1.6 times faster inference. However, none of these studies presented methods that directly revise BERT architecture for decreasing computational complexity during inference.

3 Language Model Baselines

The conventional language modeling is a task of predicting the i-th token x_i using its preceding context $\mathbf{x}_{<i} = [x_1, \ldots, x_{i-1}]$, and we call this objective as causal language modeling (CLM) throughout this paper following (Conneau and Lample, 2019). As shown in Figure 1a, we can obtain (left-to-right) contextualized language representations $\mathbf{H}^{\mathrm{C}} = [H_1^{\mathrm{C}}, \ldots, H_n^{\mathrm{C}}]$ at a single feeding the input sequence to the CLM-trained language model, where $H_i^{\mathrm{C}} = h^{\mathrm{C}}(\mathbf{x}_{<i})$ is the hidden representation of i-th token. This paper takes this unidirectional language model (uniLM) as our speed baseline. However, contextualized language representations obtained from the uniLM are insufficient to accurately encode a given text because future contexts cannot be used to understand the current tokens during inference.

Recently, BERT (Devlin et al., 2019) enables the full contextualization of the language repre-



(a) Causal language modeling

(b) Masked language modeling

(c) Language Autoencoding

Figure 1: Schematic diagrams of Transformer-based language models for (a) CLM, (b) MLM, and (c) LAE.

sentations by using the masked language modeling (MLM) objective. In the MLM, some tokens from the input sequence are randomly masked, and the objective is to predict the original tokens at the masked positions using only their context. As in Figure 1b, we can obtain a contextualized representation of i-th token $H_i^{\mathrm{M}} = h^{\mathrm{M}}(M_i(\mathbf{x}))$ by masking the token in the input sequence and feeding it into the MLM-trained model, where $M_i(\mathbf{x}) = [x_1, \ldots, x_{i-1}, \texttt{[MASK]}, x_{i+1}, \ldots, x_n]$ is an external masking operation. This paper takes this bidirectional language model (biLM) as our performance baseline. However, this *mask-and-predict* approach should be repeated n times to obtain the whole language representations because the learning occurs only at the masked position during the MLM training. Although the language representations are robust and accurate, this repetition causes significant inefficiency in the use of unsupervised applications such as the N-best list reranking tasks (Wang and Cho, 2019; Shin et al., 2019; Salazar et al., 2019).

4 Proposed Methods

4.1 Language Autoencoding

In this paper, we propose a new learning objective named *language autoencoding* (LAE) for obtaining fully contextualized language representations without repetition. The LAE lets the output to become the same as the input, and the objective is to predict every token in a text sequence at once without merely copying the input to output. For the proposed task, a language model should reproduce the whole input at once while avoiding the over-fitting. Otherwise, the model only outputs the representation copied from the input representation without learning any statistics of the language. To this end, information flow from the i-th input to the i-th output should be blocked inside the model shown in Figure 1c. From this LAE objective, we can obtain fully contextualized language representations $\mathbf{H}^{\mathrm{L}} = [\mathbf{H}_1^{\mathrm{L}}, \ldots, \mathbf{H}_n^{\mathrm{L}}]$ at once, where $\mathbf{H}_i^{\mathrm{L}} = \mathbf{H}^{\mathrm{L}}(\mathbf{x}_{\setminus i})$ and $\mathbf{x}_{\setminus i} = [x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n]$. The way of blocking the information flow is described in the next section.

4.2 Transformer-based Text Autoencoder

In this section, we introduce a novel architecture of a deep bidirectional language model named **T-TA**, which stands for **T**ransformer-based **T**ext **A**utoencoder, and the overall architecture of the T-TA is shown in Figure 2. As in its name, the model architecture is based on the Transformer encoder (Vaswani et al., 2017). To learn the proposed LAE, we develop a **diagonal masking** operation and an **input isolation** mechanism inside the T-TA. Both developments are designed to let the language model predict every token at once while maintaining the deep bidirectional property (see the descriptions in the following subsections). Due to the space limit, we refer to the original Transformer paper (Vaswani et al., 2017) for other details of the standard functions such as the multi-head attention, the scaled dot-product attention, layer normalization, and the position-wise fully connected feed-forward network.

4.2.1 Diagonal Masking

As shown in Figure 3, a diagonal masking operation is inside the scaled dot-product attention in order to be "self-unknown" during the inference. This operation prevents the information from flowing to the same position in the next layer by masking out the diagonal values in the input of the softmax. Specifically, the output vector at each position is the weighted sum of the value $\mathbf{V}$ at other positions, where the attention weights come from the query $\mathbf{Q}$ and the key $\mathbf{K}$.

The diagonal mask becomes meaningless when

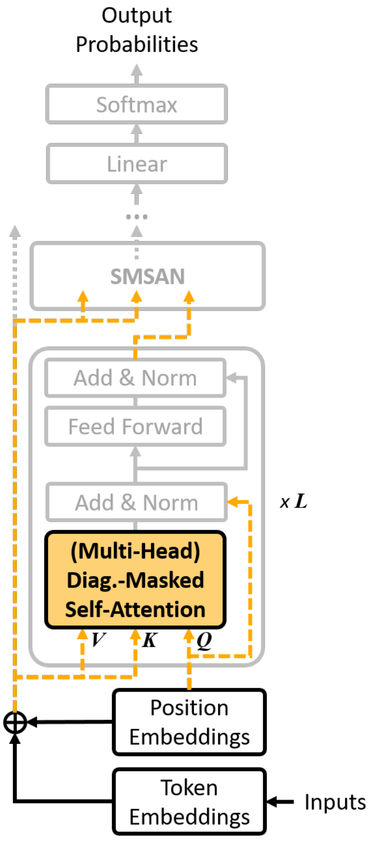


Figure 2: Architecture of our T-TA. Highlighted box and dashed arrows are newly invented in this paper.

we use it together with the residual connection or utilize it in the multi-layer architecture. To keep the self-unknown functional, we can remove the residual connection and adopt single-layer architecture. However, it is essential to utilize deep architecture to understand the intricate patterns of natural language. To this end, we further develop an architecture described in the next section.

4.2.2 Input Isolation

We now propose an input isolation mechanism in order to make the residual connection and the multi-layer architecture compatible with the diagonal masking operation. In the input isolation, the key-value inputs (**K**-**V**) of all encoding layers are isolated from the network flow, and they are fixed to the sum of the token embeddings and the position embeddings. Only query inputs (**Q**) are updated across the layers during inference by referring to the fixed output of the embedding layer.

Additionally, we input the position embeddings to the Q of the very first encoding layer in order to make the self-attention mechanism effective. Otherwise, the attention weights will be the same at

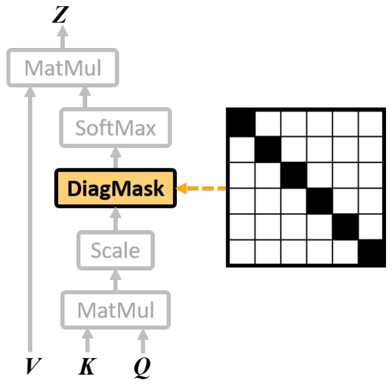


Figure 3: Diagonal masking of the scaled dot-product attention mechanism. Highlighted box and dashed arrow are newly invented in this paper.

all positions, resulting in that the first self-attention works as a simple calculator of averaging the input representations except the "self" position. Finally, we utilize the residual connection only to the query to maintain the unawareness completely. The dashed arrows in Figure 2 show this input isolation mechanism inside the T-TA.

By using the diagonal masking and input isolation together, the T-TA can have multiple encoder layers. They enable the T-TA to obtain high-quality contextual language representations with an only single feeding of a sequence.

4.3 Discussion and Analysis

Until now, we have introduced the new learning objective, language autoencoding (LAE), and the novel deep bidirectional language model, Transformer-based Text Autoencoder (T-TA). We will verify the model architecture of the proposed T-TA in Section 4.3.1, and compare our model with the recent bidirectional language model BERT in Section 4.3.2.

4.3.1 Verification of the Architecture

We here discuss how the diagonal masking with input isolation preserve "self-unknown" property in detail.

As in Figure 2, we have two embeddings, token embeddings $\mathbf{X} = [X_1, \dots, X_n]^T \in \mathbb{R}^{n \times d}$ and position embeddings $\mathbf{P} = [P_1, \dots, P_n]^T \in \mathbb{R}^{n \times d}$, where the d is an embedding dimension. From the input isolation, the key and value $\mathbf{K} = \mathbf{V} = \mathbf{X} + \mathbf{P}$ have the information of input tokens and they are *fixed* in all layers, but the query $\mathbf{Q}^l$ is *updated* across the layers during inference started from the position embeddings $\mathbf{Q}^1 = \mathbf{P}$ at the first layer.

Let us consider the l-th encoding layer's query input $\mathbf{Q}^l$ and its output $\mathbf{H}^l = \mathbf{Q}^{l+1}$. Then,

$$\begin{aligned}\mathbf{H}^l &= \text{SMSAN}(\mathbf{Q}^l, \mathbf{K}, \mathbf{V}) \\ &= g(\text{Norm}(\text{Add}(\mathbf{Q}^l, f(\mathbf{Q}^l, \mathbf{K}, \mathbf{V})))),\end{aligned} \tag{1}$$

where SMSAN$(\cdot)$ represents the Self-Masked Self-Attention Network, the encoding layer of the T-TA, $g(x) = \text{Norm}(\text{Add}(x, \text{FeedForward}(x)))$, two upper-side sub-boxes of the encoding layer, and $f(\cdot)$ is the (multi-head) diagonal-masked self-attention (DMSA) mechanism shown in Figure 2. As in Figure 3, the DMSA module computes $\mathbf{Z}^l$ as follows:

$$\begin{aligned}\mathbf{Z}^l &= f(\mathbf{Q}^l, \mathbf{K}, \mathbf{V}) = \text{DMSA}(\mathbf{Q}^l, \mathbf{K}, \mathbf{V}) \\ &= \text{SoftMax}(\text{DiagMask}(\mathbf{Q}^l\mathbf{K}^T/\sqrt{d}))\mathbf{V}.\end{aligned} \tag{2}$$

In the DMSA module, the i-th element of $\mathbf{Z}^l = [Z_1^l, \ldots, Z_n^l]^T$ is always computed by a weighted average of the fixed $\mathbf{V}$ discarding the information of i-th token X_i in V_i. To be more specific, Z_i^l is the weighted average of the $\mathbf{V}$ with the attention weight vector $\mathbf{s}_i^l$, *i.e.*, $Z_i^l = \mathbf{s}_i^l\mathbf{V}$, where $\mathbf{s}_i^l = [s_1^l, \ldots, s_{i-1}^l, 0, s_{i+1}^l, \ldots, s_n^l] \in \mathbb{R}^{1\times n}$. We here note that only the DMSA is related to the "self-unknown" since no token representation is referred to each other in the subsequent transformations from $\mathbf{Z}^l$ to $\mathbf{H}^l$. Therefore, it is guaranteed that the i-th element of the query representation in any layer, Q_i^l, never sees the corresponding token representation started from the $Q_i^1 = P_i$. Consequently, the T-TA preserves the "self-unknown" property during inference while maintaining the residual connection and multi-layer architecture.

4.3.2 Comparison with BERT

There are several differences between the strong baseline BERT (Devlin et al., 2019) and the proposed model T-TA, while both models learn deep bidirectional language representations.

- While BERT uses external masking operation in the input, T-TA has internal masking operation in the model as we intend. Also, while BERT is based on denoising autoencoder, T-TA is based on autoencoder. Due to this novel approach, the T-TA does not need *mask-and-predict* repetition during computing contextual language representations. Consequently, we reduce the computational complexity from $O(n^3)$ of BERT to $O(n^2)$ of T-TA when applying the language models to the unsupervised learning tasks.
- As in the T-TA, feeding an intact input (without masks) into BERT is also possible. However, we argue that it will significantly hurt the model performance on unsupervised applications since the MLM objective does not consider the intact token much. We include experiments that show model performance with the intact input (described in Table 1, 3, and 4). We also suggest reading previous research that reported the same opinion (Salazar et al., 2019).

5 Experiments

To evaluate the proposed method, we conduct a series of experiments. We first evaluate the contextual language representations obtained from the Transformer-based Text Autoencoder (T-TA) on the N-best list reranking tasks. We then apply our method to unsupervised semantic textual similarity (STS) tasks. The following sections will demonstrate that the proposed model is much faster than the BERT during inference (in Section 5.2) while showing competitive or even better accuracies than those of the BERT on reranking tasks (in Section 5.3) and STS tasks (in Section 5.4).

5.1 Language Model Setups

This paper mainly compares the proposed T-TA with the bidirectional language model (biLM), which is trained with the masked language modeling (MLM) objective, like BERT. For a fair comparison, each model has the same number of parameters based on the Transformer as followed: $|L| = 3$ self-attention layers with $d = 512$ input and output dimensions, $h = 8$ attention heads, and $d_f = 2048$ hidden units for the position-wise feed-forward layers. We use a *gelu* activation (Hendrycks and Gimpel, 2016) rather than the standard *relu*, following OpenAI GPT (Radford et al., 2018) and BERT (Devlin et al., 2019). We set a position embeddings to be trainable following BERT (Devlin et al., 2019) rather than a fixed sinusoid (Vaswani et al., 2017) with supported sequence lengths up to 128 tokens in our experiments. We use WordPiece embeddings (Wu et al., 2016) with a vocabulary of about $|V| \simeq 30,000$ tokens. The weights of the embedding layer and the last softmax layer of the Transformer are shared. For the speed baseline, we also implement a unidirectional language model (uniLM), which has the same number of parameters as T-TA and biLM.

For training, we make a training instance consist-

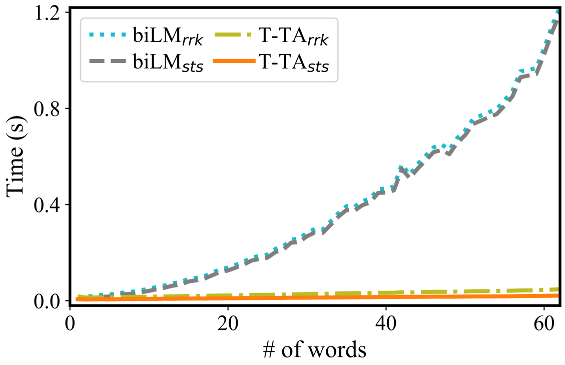


Figure 4: Average running times of each model according to the number of words on STS and reranking tasks, sub-scripted as *sts* and *rrk* respectively.

ing of a single sentence with [BOS] and [EOS] tokens at the begin and the end of each sentence. We use 64 sentences as a training batch, and train language models $1M$ steps for ASR and $2M$ steps for NMT. We train the language models with Adam (Kingma and Ba, 2014) with an initial learning rate of $1e-4$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, learning rate warm up over the first 50k steps and linear decay of the learning rate. We use a dropout probability of 0.1 on all layers. Our implementation is based on Google's official code for BERT[3].

To train language models that we implement, we use about 13GB English Wikipedia dump that has about 120M sentences. The trained models are used for reranking in neural machine translation (NMT) and unsupervised semantic textual similarity tasks. For reranking in automatic speech recognition (ASR), we use additional in-domain training data of the 4.0GB normalized text data of the official LibriSpeech corpus that has about 40M sentences.

One of the strong baseline language models, the pre-trained BERT-base-uncased (Devlin et al., 2019), is used for reranking and STS. We also include the reranking results from the traditional count-based 5-gram language models that are trained on each dataset using the KenLM library (Heafield, 2011).

5.2 Running Time Analysis

We first measure the running time of each language model for computing the contextual language representation $\mathbf{H}^L \in \mathbb{R}^{n\times d}$ of a given text sequence. In the unsupervised STS tasks, we directly use the $\mathbf{H}^L$ for the analysis. In the case of the reranking task, we need further computation; we compute Softmax$(\mathbf{H}^L\mathbf{E}^T)$ to obtain the likelihood of each token, where $\mathbf{E} \in \mathbb{R}^{|V|\times d}$ is the weight parameters of the softmax layer. Therefore, the computational complexity of reranking is bigger than that of STS.

In the running time measurement, we use Intel(R) Core(TM) i7-6850K CPU (3.60GHz) on the TensorFlow 1.12.0 library with Python 3.6.8 over the Ubuntu 16.04.06 LTS. In each experiment, we measured the run-time 50 times and averaged the results. Figure 4 shows that the run-time of the T-TA is faster than that of the biLM, and it becomes significant as the sentence is longer. For numerical comparison, we set the standard number of words to 20 since the average number of words in English sentences today is about 20 (DuBay, 2006). In this setup, the T-TA takes about 9.85 ms while the biLM takes about 125 ms in the STS task, showing that the T-TA is 12.7 times faster than the biLM. In the reranking task, the ratio between the T-TA and the biLM is reduced to 6.35 times (still significant), and this is because the repetition of biLM is related only to computing $\mathbf{H}^L$ not to Softmax$(\mathbf{H}^L\mathbf{E}^T)$.

For the visibility of Figure 4, we omit the run-time results of uniLM, which is also as fast as the T-TA (Appendix A.3). With this fast inference, we show that the T-TA is as accurate as BERT in the next section.

5.3 Reranking the N-best List

To evaluate language models, we conduct experiments on the unsupervised task of reranking the N-best list. In the experiments, we apply each language model to rerank the 50-best candidate sentences, which are obtained in advance using each sequence-to-sequence model on ASR and NMT. The ASR and NMT models we implement are detailed in Appendix A.1 and A.2.

We rescore the sentences by linearly interpolating two scores from a sequence-to-sequence model and each language model as follows:

$$\text{score} = (1-\lambda)\cdot \text{score}_{s2s} + \lambda \cdot \text{score}_{lm},$$

where the score$_{s2s}$ is the score from sequence-to-sequence models, score$_{lm}$ is the score from language models calculated by the sum (or mean) of the log-likelihood of each token, and the interpolation weight λ is set to a value that shows the best performance in the development set.

We note that the T-TA and biLM (also BERT) assign the pseudo-log-likelihood to the score of a

[3] https://github.com/google-research/bert

Method	dev		test	
	clean	other	clean	other
Shin et al.	7.17	19.79	7.25	20.37
w/ n-gram	5.62	16.85	5.75	17.72
w/ $*$uniSANLM$_\mathrm{w}$	6.05	17.32	6.11	18.13
w/ $*$biSANLM$_\mathrm{w}$	5.52	16.61	5.65	17.37
w/ BERT	5.24	16.56	5.38	17.46
w/ BERT$_{\setminus M}$	7.08	19.61	7.14	20.18
w/ uniLM	5.07	16.20	5.14	17.00
w/ biLM	**4.94**	**16.09**	5.14	**16.81**
w/ T-TA	4.98	**16.09**	**5.11**	16.91
Seq2Seq$_{ASR}$	4.11	12.31	4.31	13.14
w/ n-gram	3.94	11.93	4.15	12.89
w/ BERT	3.72	11.59	**3.97**	12.46
w/ BERT$_{\setminus M}$	4.09	12.26	4.28	13.15
w/ uniLM	3.82	11.73	4.05	12.63
w/ biLM	3.73	**11.53**	**3.97**	12.41
w/ T-TA	**3.67**	11.56	**3.97**	**12.38**

Table 1: WERs after reranking with each language model on LibriSpeech. 'other' sets are recorded in noisier environments than 'clean' sets. Bolds are for the best performance on each sub-task. $*$ are word-level language models from (Shin et al., 2019).

given sentence while the uniLM assigns the log-likelihood. Because the reranking task is based on relative scores of the n-best hypotheses, the fact that bidirectional language models yield the pseudo-log-likelihood of a given sentence does not matter in this task (Wang and Cho, 2019; Shin et al., 2019; Salazar et al., 2019).

5.3.1 Results on Speech Recognition

For reranking in ASR, we use prepared N-best lists obtained from dev and test sets using ***Seq2Seq$_{ASR}$*** that we train on the Librispeech ASR corpus. Additionally, we use the N-best lists obtained from (Shin et al., 2019) in order to see the robustness of the language models on testing environments. Table 1 shows the word error rates (WERs) for each method after reranking. The interpolation weights λ were 0.3 or 0.4 in all N-best lists for ASR.

We observe that the bidirectional language models trained with the LAE (T-TA) and MLM (biLM) outperform the unidirectional language model (uniLM) trained with the CLM. Performance gains from the reranking are much lower in the better base system ***Seq2Seq$_{ASR}$***, and we can see that it is challenging to rerank the N-best list using a language model if the speech recognition model performs well enough. Interestingly, the T-TA is competitive and even better than the biLM, and it may be from the gap between training and testing of the biLM: the biLM predicts multiple masks at a time when training, but predicts only one mask at a time when testing. Moreover, the 3-layer T-TA is better than the 12-layer BERT-base, showing that in-domain data is critical to the language model applications.

Finally, we note that feeding an intact input to the BERT, denoted as "w/ BERT$_{\setminus \mathrm{M}}$" in the Table 1, underperforms the others, and this shows that the *mask-and-predict* is necessary for the effective reranking.

Method	De→En	Fr→En
Seq2Seq$_{NMT}$	27.83	29.63
w/ n-gram	28.41	30.04
w/ BERT	**29.31**	**30.52**
w/ uniLM	28.80	30.21
w/ biLM	28.76	30.32
w/ T-TA	28.83	30.20

Table 2: BLEU scores after reranking with each language model on WMT13. Bolds are for the best performance on each sub-task. Underlines are for the best in our implementations.

5.3.2 Results on Machine Translation

To see the reranking performance in other domain, NMT, we prepare the N-best lists using ***Seq2Seq$_{NMT}$***[4] from the WMT-13's German-to-English and French-to-English test sets. Table 2 shows the BLEU scores for each method after reranking. Each interpolation weight becomes a value that shows the best performance on each test set with each method in NMT. The interpolation weights λ were 0.4 or 0.5 in the N-best lists for NMT.

We observe again that the bidirectional language models trained with the LAE and MLM perform better than the unidirectional language model trained with the CLM. Also, the Fr→En translation has less effect on reranking than the De→En translation because the base NMT system for Fr→En is better than that for De→En.

Seeing that the 12-layer BERT is much better than the others in reranking on NMT, it seems that

[4] Seq2Seq models for De→En and Fr→En are trained independently from t2t library (Vaswani et al., 2018)

the N-best hypotheses of the NMT model are more subtle to distinguish than those of the ASR model from the language model perspective. All reranking results in ASR and NMT demonstrate that the proposed T-TA performs efficiently like uniLM and effectively like biLM.

5.4 Unsupervised Semantic Textual Similarity

In addition to the reranking task, we apply language models to the semantic textual similarity (STS), which is the task of measuring the meaning similarity of sentence pairs. We use STS Benchmark (Cer et al., 2017) and SICK (Marelli et al., 2014), where both datasets have a set of sentence pairs with corresponding similarity scores. The evaluation metric of STS is the Pearson's r between the predicted similarity scores and the reference scores of the given sentence pairs.

In this section, we address the task of *unsupervised* STS to examine the inherent ability to obtain contextual language representations of each language model, and we mainly compare language models that are trained on the English Wikipedia dump. To compute a similarity score of a given sentence pair, we use the cosine similarity of two sentence representations, where each representation is obtained by averaging each language model's contextual language representations. Specifically, contextual representations of a given sentence are the outputs of the final encoding layer of each model, denoted as *context* in Table 3 and 4. For comparison, we use non-contextual representations, which are obtained from the outputs of the embedding layer, denoted as *embed* in Table 3 and 4. As a strong baseline for unsupervised STS tasks, we also include the 12-layer BERT model (Devlin et al., 2019), and we use the BERT in the *mask-and-predict* approach for computing contextual representations of each sentence. Note that we use the most straightforward approach for the unsupervised STS in order to focus on comparing token-level language representations.

5.4.1 Results on STS Benchmark

The STS Benchmark (STSb) has 5749/1500/1379 sentence pairs for train/dev/test splits with corresponding scores ranging from 0-5. We test language models on the STSb-dev and STSb-test using the most simple approach on the unsupervised STS. As our additional baselines, we include the results of GloVe (Pennington et al., 2014) and Word2Vec (Mikolov et al., 2013a) from the official sites of STS Benchmark[5].

Method	STSb-dev		STSb-test	
	context	*embed*	*context*	*embed*
BERT	**64.78**	-	**54.22**	-
BERT$_{\backslash M}$	59.17	60.07	47.91	48.19
BERT$_{\texttt{[CLS]}}$	29.16		17.18	
uniLM	56.25	**63.87**	39.57	**55.00**
uniLM$_{\texttt{[EOS]}}$	40.75		38.30	
biLM	59.99	-	50.76	-
biLM$_{\backslash M}$	53.20	58.80	36.51	49.08
T-TA	**71.88**	54.75	**62.27**	44.74
GloVe	-	52.4	-	40.6
Word2Vec	-	**70.0**	-	**56.5**

Table 3: Pearson's $r \times 100$ results on STS Benchmark. - denotes the infeasible value. Bolds are for the top-2 performances on each sub-task.

Table 3 shows our T-TA trained with the LAE best captures the semantic of a sentence over the Transformer-based language models. It is remarkable that our 3-layer T-TA trained on the relatively small data outperforms the 12-layer BERT trained on large data (Wikipedia + BookCorpus). Another interesting point is that embedding representations are trained better by the CLM than the other language modeling objectives, and we guess that the uniLM highly depends on the embedding layer due to its constraint of the unidirectional context.

Since the uniLM encodes all contexts in the last token `[EOS]`, we also use the last representation as to the sentence representation, but it does not outperform the averaged sentence representation. Similarly, BERT has a special token `[CLS]`, which is trained for the "next sentence prediction" objective, so we also use it to see how `[CLS]` learns sentence representation, but it significantly underperforms the others.

5.4.2 Results on SICK

We further evaluate language models on the SICK data, which consists of 4934/4906 sentence pairs for train/test splits with the scores ranging from 1-5. The results are in Table 4, and we have the same observations as STSb.

All results on unsupervised STS tasks demonstrate that the T-TA learns textual semantics best using the token-level language modeling, LAE.

[5] http://ixa2.si.ehu.es/stswiki/index.php/STSbenchmark

Method	SICK-test	
	context	*embed*
BERT	64.31	-
BERT$_{\backslash M}$	61.18	64.63
uniLM	54.20	**65.69**
biLM	58.98	-
biLM$_{\backslash M}$	53.79	62.67
T-TA	**69.49**	60.77

Table 4: Pearson's $r \times 100$ results on SICK data. - denotes the infeasible value. Bolds are for the best performance on each sub-task.

6 Conclusion

In this work, we propose a novel deep bidirectional language model named Transformer-based Text Autoencoder (T-TA) in order to eliminate the computational overload of applying BERT for unsupervised applications. Experimental results on the N-best list reranking and the unsupervised semantic textual similarity tasks demonstrate that the proposed T-TA is significantly faster than the BERT-based approach, while its encoding ability is competitive or even better than that of BERT.

Acknowledgments

K. Jung is with ASRI, Seoul National University, Korea. This work was supported by the Ministry of Trade, Industry & Energy (MOTIE, Korea) under Industrial Technology Innovation Program (No.10073144) and by the NRF grant funded by the Korea government (MSIT) (NRF2016M3C4A7952587).

References

Ebru Arisoy, Abhinav Sethy, Bhuvana Ramabhadran, and Stanley Chen. 2015. Bidirectional recurrent neural network language models for automatic speech recognition. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5421–5425. IEEE.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14.

William Chan, Navdeep Jaitly, Quoc Le, and Oriol Vinyals. 2016. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4960–4964. IEEE.

Xie Chen, Anton Ragni, Xunying Liu, and Mark JF Gales. 2017. Investigating bidirectional recurrent neural network language models for speech recognition. In *INTERSPEECH*, pages 269–273.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734.

Jan K Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio. 2015. Attention-based models for speech recognition. In *Advances in neural information processing systems*, pages 577–585.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems*, pages 7057–7067.

Andrew M Dai and Quoc V Le. 2015. Semi-supervised sequence learning. In *Advances in neural information processing systems*, pages 3079–3087.

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.

William H DuBay. 2006. The classic readability studies. *Impact Information, Costa Mesa, California.*

Kenneth Heafield. 2011. Kenlm: Faster and smaller language model queries. In *Proceedings of the sixth workshop on statistical machine translation*, pages 187–197. Association for Computational Linguistics.

Dan Hendrycks and Kevin Gimpel. 2016. Bridging nonlinearities and stochastic regularizers with gaussian error linear units. *arXiv preprint arXiv:1606.08415.*

Takaaki Hori, Shinji Watanabe, Yu Zhang, and William Chan. 2017. Advances in joint ctc-attention based end-to-end speech recognition with a deep cnn encoder and rnn-lm. *Proc. Interspeech 2017*, pages 949–953.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Marco Marelli, Luisa Bentivogli, Marco Baroni, Raffaella Bernardi, Stefano Menini, and Roberto Zamparelli. 2014. Semeval-2014 task 1: Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment. In *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)*, pages 1–8.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Eleventh annual conference of the international speech communication association*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210. IEEE.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Alvaro Peris and Francisco Casacuberta. 2015. A bidirectional recurrent neural language model for machine translation. *Procesamiento del Lenguaje Natural*, 55:109–116.

Matthew 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)*, volume 1, pages 2227–2237.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1986. Learning representations by back-propagating errors. *nature*, 323(6088):533–536.

Julian Salazar, Davis Liang, Toan Q Nguyen, and Katrin Kirchhoff. 2019. Pseudolikelihood reranking with masked language models. *arXiv preprint arXiv:1910.14659*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96.

Yusuxke Shibata, Takuya Kida, Shuichi Fukamachi, Masayuki Takeda, Ayumi Shinohara, Takeshi Shinohara, and Setsuo Arikawa. 1999. Byte pair encoding: A text compression scheme that accelerates pattern matching. Technical report, Technical Report DOI-TR-161, Department of Informatics, Kyushu University.

Joonbo Shin, Yoonhyung Lee, and Kyomin Jung. 2019. Effective sentence scoring method using bert for speech recognition. In *Asian Conference on Machine Learning*, pages 1081–1093.

Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. 2012. Lstm neural networks for language modeling. In *Thirteenth annual conference of the international speech communication association*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, pages 3104–3112.

Ashish Vaswani, Samy Bengio, Eugene Brevdo, Francois Chollet, Aidan Gomez, Stephan Gouws, Llion Jones, Łukasz Kaiser, Nal Kalchbrenner, Niki Parmar, et al. 2018. Tensor2tensor for neural machine translation. In *Proceedings of the 13th Conference of the Association for Machine Translation in the Americas (Volume 1: Research Papers)*, pages 193–199.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Alex Wang and Kyunghyun Cho. 2019. Bert has a mouth, and it must speak: Bert as a markov random field language model. In *Proceedings of the Workshop on Methods for Optimizing and Evaluating Neural Language Generation*, pages 30–36.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019*.

Shinji Watanabe, Takaaki Hori, Shigeki Karita, Tomoki Hayashi, Jiro Nishitoba, Yuya Unno, Nelson-Enrique Yalta Soplin, Jahn Heymann, Matthew Wiesner, Nanxin Chen, et al. 2018. Espnet: End-to-end speech processing toolkit. *Proc. Interspeech 2018*, pages 2207–2211.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Matthew D Zeiler. 2012. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*.

A Appendices

A.1 Setups for ASR systems

This section introduces our implementation of the speech recognition system.

For the input features, we use 80-band Mel-scale spectrogram derived from the speech signal. The target sequence is processed in 5K case-insensitive sub-word units created via unigram byte-pair encoding (Shibata et al., 1999). We use an attention-based encoder-decoder model as our acoustic model. The encoder is a 5-layer bidirectional LSTM, and there are bottleneck layers that conduct linear transformation between every LSTM layers. Also, there is a VGG module before the encoder, and it reduces encoding time steps by a quarter through two max-pooling layers. The decoder is 2-layer bidirectional LSTM with location-aware attention mechanism (Chorowski et al., 2015). All the layers have 1024 hidden units. The model is trained with additional CTC objective function because the left-to-right constraint of CTC helps learn alignments between speech-text pairs (Hori et al., 2017).

Our model is trained for 20 epochs on 960h of LibriSpeech training data using Adadelta optimizer (Zeiler, 2012). Using this acoustic model, we obtain 50-best decoded sentences for each input audio through hybrid CTC-attention based scoring (Hori et al., 2017) method. For ***Seq2Seq*$_{ASR}$**, we additionally use a pre-trained RNNLM to combine the log-probability p^{lm} of RNNLM during decoding as follows:

$$\log p(y_n|y_{1:n-1}) \\ = \log p^{\mathrm{am}}(y_n|y_{1:n-1}) + \beta \log p^{\mathrm{lm}}(y_n|y_{1:n-1}), \tag{3}$$

where β is set to 0.7. We use ESPNet toolkit (Watanabe et al., 2018) for this implementation.

Method	dev		test	
	clean	other	clean	other
Shin et al.	7.17	19.79	7.26	20.37
oracle	3.18	12.98	3.19	13.61
***Seq2Seq*$_{ASR}$**	4.11	12.31	4.31	13.14
oracle	1.80	7.90	1.96	8.39

Table 5: Oracle WERs of the 50-best lists on LibriSpeech from each ASR system.

Table 5 shows the oracle word error rates (WERs) of the 50-best lists, which are measured assuming that the best sentence is always picked from the candidates. We also include the oracle WERs from the 50-best lists of (Shin et al., 2019).

A.2 Setups for NMT systems

We implement the standard Transformer model (Vaswani et al., 2017) using Tensor2Tensor library (Vaswani et al., 2018) for machine translation. Both the encoder and decoder of the Transformer consist of 6 layers with 512 hidden units, and the number of the self-attention heads is 8. The maximum number of input tokens is set to 256. We use the shared vocabulary of size 32k. For effective training, we let the token embedding layer and the last softmax layer share their weights. The other hyperparameters of our translation system follow the standard `transformer_base_single_gpu` setting in Google's official Tensor2Tensor repository[6].

We train the baseline model on the standard WMT18 French-English and German-English datasets for 250k steps using Adam optimizer (Kingma and Ba, 2014). We use linear-warmup-square-root-decay learning rate scheduling with the default learning rate 2.5e-4 and warmup steps 16k. Using this baseline translation model, we obtain 50-best decoded sentences for each source through the beam-search. The oracle BLEU scores for the NMT system are shown in Table 6.

Method	WMT13	
	De→En	Fr→En
***Seq2Seq*$_{NMT}$**	27.83	29.63
oracle	38.18	39.58

Table 6: Oracle BLEUs of the 50-best lists on WMT

A.3 Running time of uniLM and T-TA

As mentioned in Section 5.2, we also measure execution times of the uniLM we implement. Figure 5 shows that the averaged run-times of the uniLM and the T-TA for the number of words in a sentence. Since we use subword tokens, the number n_w of words and the number n of tokens can be different $n_w \leq n$.

[6] https://github.com/tensorflow/tensor2tensor

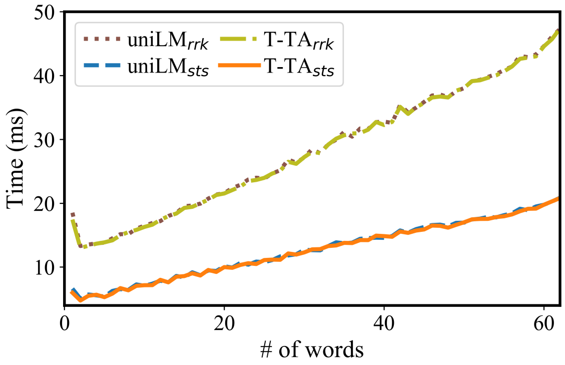


Figure 5: Running times according to the number of words for uniLM and T-TA.

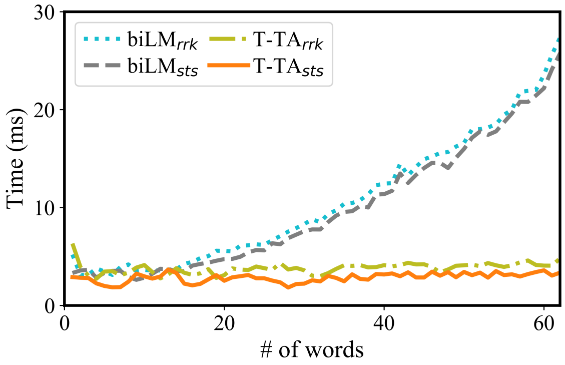


Figure 6: Running times according to the number of words for biLM and T-TA on GPU-augmented environment.

A.4 Running time on GPU

Additionally, we also measure execution times on a GPU-augmented environment (using GeForce GTX 1080 Ti). Figure 6 shows that the averaged run-times of the biLM and the T-TA for the number of words in a sentence. In our 20-words standard, the T-TA takes about 2.51 ms and biLM takes about 4.72 ms in the STS task, showing that the T-TA is 1.88 times faster than the biLM. Compared to the CPU-only environment, the speed difference was reduced due to the GPU supports. Seeing Figure 4, however, the CPU-only environment and the GPU-augmented environment have a similar tendency: the longer the sentence, the more significant the difference between the T-TA and the biLM.

A.5 Perplexity and Reranking

In general, perplexity (PPL) is a measure of how well language models trained. To see the alignment of PPL and reranking, we compute PPL of reference sentences from the Librispeech dev-clean and test-clean set using each language model. We can get pseudo-perplexity (pPPL) from biLM and T-TA since they do not follow the product rule, unlike uniLM. We note that we compute subword-level (p)PPL (not word-level); these values are valid only in our vocabulary.

	Method [WER]	(p)PPL$_a$	(p)PPL$_m$
dev clean	uniLM [3.82]	341.5	70.80
	biLM [3.73]	(76.49)	(11.93)
	T-TA [3.67]	(293.4)	(11.69)
test clean	uniLM [4.05]	495.5	73.18
	biLM [3.97]	(75.43)	(12.72)
	T-TA [3.97]	(590.0)	(12.43)

Table 7: (pseudo)Perplexities and corresponding WERs of language models on LibriSpeech.

We can find that WERs are better aligned with the median of pPPL$_m$ than the averaged pPPL$_a$. Interestingly, the pPPL$_a$ of T-TA is similar to the PPL$_a$ of uniLM, but the pPPL$_m$ of T-TA is similar to that of biLM. We additionally find that if the length of a sentence is short, T-TA shows a very high perplexity, even higher than uniLM.