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 has achieved successful performance improvements in various supervised learning tasks, BERT is still limited by repetitive inferences on unsupervised tasks for the computation of contextual language representations. To resolve this limitation, we propose a novel deep bidirectional language model called a **T**ransformer-based **T**ext **A**utoencoder (**T-TA**). The T-TA computes contextual language representations without repetition and displays the benefits of a deep bidirectional architecture, such as that of BERT. In computation time experiments in a CPU environment, the proposed T-TA performs over six times faster than the BERT-like model on a reranking task and twelve times faster on a semantic similarity task. Furthermore, the T-TA shows competitive or even better accuracies than those of BERT on the above tasks. Code is available at https://github.com/joongbo/tta.

1 Introduction

A language model is an essential component of many natural language processing (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, the Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2019) and its variations have led to significant improvements in learning natural language representation and have achieved state-of-the-art performances on various downstream tasks such as the General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2019) and question answering (Rajpurkar et al., 2016). BERT continues to succeed in various unsupervised tasks, such as the N-best list reranking for ASR and NMT (Shin et al., 2019; Salazar et al., 2019), confirming that deep bidirectional language models are useful in unsupervised applications as well.

However, concerning its applications to unsupervised learning tasks, BERT is significantly inefficient at computing language representations at the inference stage (Salazar et al., 2019). During training, BERT adopts the *masked language modeling* (MLM) objective, which is to predict the original word of the explicitly masked word from the input sequence. Following the MLM objective, each contextual word representation should be computed by a two-step process: masking a word in the input and feeding the result to BERT. During the inference stage, this process is repeated n times to obtain the representations of all the words within a text sequence (Wang and Cho, 2019; Shin et al., 2019; Salazar et al., 2019), resulting in a computational complexity of $O(n^3)$[1] in terms of the number of words n. Hence, it is necessary to reduce the computational complexity when applying the model to situations where the inference time is critical, *e.g.*, mobile environments and real-time systems (Sanh et al., 2019; Lan et al., 2019). Considering this limitation of BERT, we submit a new research question: "Can we construct a deep bidirectional language model with a minimal inference time while maintaining the accuracy of BERT?"

In this paper, in response to the above question, we propose a novel bidirectional language model named the **T**ransformer-based **T**ext **A**utoencoder (**T-TA**), which has a reduced computational complexity of $O(n^2)$ when applying the model to unsupervised applications. The proposed model is trained with a new learning objective named *language autoencoding* (LAE). The LAE objective, which allows the target labels to be the same as the text input, is to predict every token in the input sequence simultaneously without merely copying

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

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 823–835
July 5 - 10, 2020.

the input to the output. To learn the proposed objective, we devise both 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: N-best list reranking and unsupervised semantic textual similarity. First, by conducting runtime experiments in a CPU environment, we show that the proposed T-TA is 6.35 times faster than the BERT-like model in the reranking task and 12.7 times faster in the unsupervised semantic textual similarity task. Second, despite its faster inference time, the T-TA achieves competitive performances relative to BERT on reranking tasks. Furthermore, the T-TA outperforms BERT by up to 8 points in Pearson's r on unsupervised semantic textual similarity tasks.

2 Related Works

When referring to an 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 that 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 these approaches consider an autoencoder that encodes *word-level* representations (such as BERT) without an autoregressive decoding process.

Many studies have been performed on neural network-based language models for word-level representations. Distributed word representations were proposed and attracted considerable interest, as they were considered to be fundamental building blocks for NLP tasks (Rumelhart et al., 1986; Bengio et al., 2003; Mikolov et al., 2013b). Subsequently, researchers explored contextualized representations of text where each word has a different representation depending on the context (Peters et al., 2018; Radford et al., 2018). Most recently, a Transformer-based deep bidirectional model was proposed and applied to various supervised-learning tasks with remarkable success (Devlin et al., 2019).

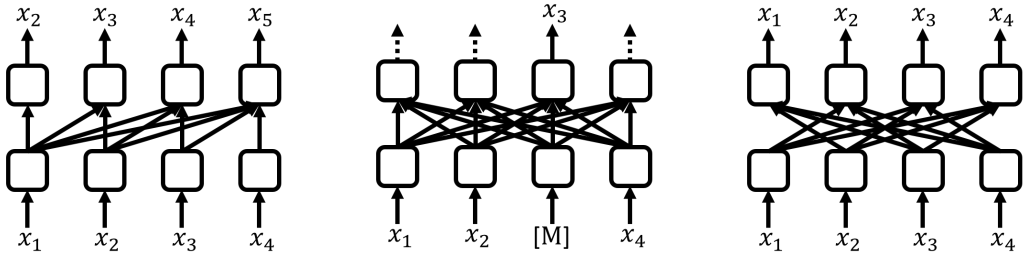
For unsupervised tasks, researchers have adopted recently developed language-representation models and investigated their effectiveness; a typical example is the N-best list reranking for ASR and NMT tasks. In particular, studies have integrated left-to-right and right-to-left language models (Arisoy et al., 2015; Chen et al., 2017; Peris and Casacuberta, 2015) 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 because bidirectional language models yield the pseudo-log-likelihood of a given sentence, and 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 involves 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 reported 18 times fewer parameters and a 1.7-fold increase in the training time. Similarly, Sanh et al. (2019) presented a method to pretrain a smaller model that can be fine-tuned for downstream tasks and achieved 1.4 times fewer parameters with a 1.6-fold increase in the inference time. However, none of these studies developed methods that directly revise the BERT architecture to reduce the computational complexity during the inference stage.

3 Language Model Baselines

In a conventional language modeling task, the i-th token x_i is predicted using its preceding context $\mathbf{x}_{<i} = [x_1, \ldots, x_{i-1}]$; throughout this paper, this objective is known as causal language modeling (CLM) following (Conneau and Lample, 2019). As shown in Figure 1a, we can obtain (left-to-right) contextualized language representations $\mathbf{H}^{\mathrm{C}} = [H_1^{\mathrm{C}}, \ldots, H_n^{\mathrm{C}}]$ after feeding the input sequence to the CLM-trained language model only once, where $H_i^{\mathrm{C}} = h^{\mathrm{C}}(\mathbf{x}_{<i})$ is the hidden representation of the 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 leveraged to understand the current tokens during the inference stage.

Recently, BERT (Devlin et al., 2019) was designed to enable the full contextualization



(a) Causal language modeling (b) Masked language modeling (c) Language autoencoding

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

of language representations by using the MLM objective, in which some tokens from the input sequence are randomly masked; 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 the 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 to the MLM-trained model, where $M_i(\mathbf{x}) = [x_1, \ldots, x_{i-1}, \texttt{[MASK]}, x_{i+1}, \ldots, x_n]$ signifies 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 all the language representations $\mathbf{H}^{\mathrm{M}} = [H_1^{\mathrm{M}}, \ldots, H_n^{\mathrm{M}}]$ because learning occurs only at the masked position during the MLM training stage. Although the resulting language representations are robust and accurate, as a consequence of this repetition, the model is significantly inefficient when applied to unsupervised tasks such as N-best list reranking (Wang and Cho, 2019; Shin et al., 2019; Salazar et al., 2019).

4 Proposed Method

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 objective, with which the output is the same as the input, is to predict every token in a text sequence simultaneously without merely copying the input to the output. For the proposed task, a language model should reproduce the whole input at once while avoiding overfitting; otherwise, the model outputs only the representation copied from the input representation without learning any statistics of the language. To this end, the flow of information from the i-th input to the i-th output should be blocked inside the model shown in Figure 1c. From the LAE objective, we can obtain fully contextualized language representations $\mathbf{H}^{\mathrm{L}} = [H_1^{\mathrm{L}}, \ldots, H_n^{\mathrm{L}}]$ all at once, where $H_i^{\mathrm{L}} = h^{\mathrm{L}}(\mathbf{x}_{\setminus i})$ and $\mathbf{x}_{\setminus i} = [x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n]$. The method for blocking the flow of information is described in the next section.

4.2 Transformer-based Text Autoencoder

In this section, we introduce the novel architecture of the proposed **T-TA** shown in Figure 2. As indicated by its name, the T-TA architecture is based on the Transformer encoder (Vaswani et al., 2017). To learn the proposed LAE objective, we develop both a **diagonal masking** operation and an **input isolation** mechanism inside the T-TA. Both developments are designed to enable the language model to predict all tokens simultaneously while maintaining the deep bidirectional property (see the descriptions in the following subsections). For brevity, we refer to the original paper on the Transformer encoder (Vaswani et al., 2017) for other details regarding the standard functions, such as the multihead attention and scaled dot-product attention mechanisms, 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 implemented inside the scaled dot-product attention mechanism to be "self-unknown" during the inference stage. This operation prevents information from flowing to the same position in the next layer by masking out the diagonal values in the input of the softmax function. 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 $\mathbf{K}$.

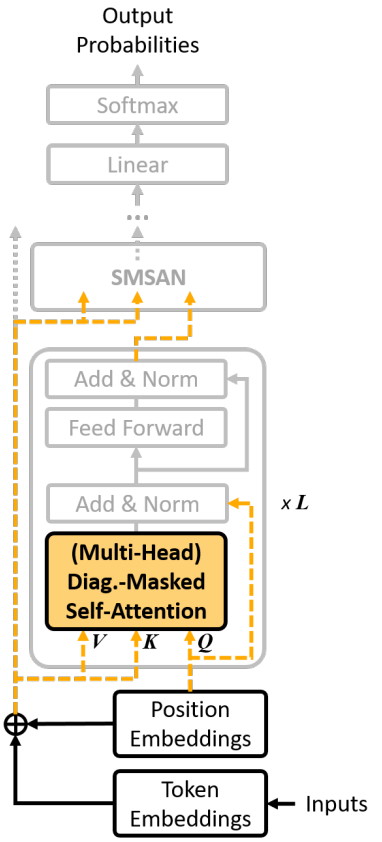


Figure 2: Architecture of our T-TA. The highlighted box and dashed arrows are the innovations presented in this paper.

The diagonal mask becomes meaningless when we use it together with a residual connection or utilize it within the multilayer architecture. To retain the self-unknown functional, we can remove the residual connection and adopt a single-layer architecture. However, it is essential to utilize a deep architecture to understand the intricate patterns of natural language. To this end, we further develop the architecture described in the next section.

4.2.2 Input Isolation

We now propose an input isolation mechanism to ensure that the residual connection and the multilayer architecture are compatible with the above-mentioned diagonal masking operation. In the input isolation mechanism, the key and value inputs (**K** and **V**, respectively) of all encoding layers are isolated from the network flow and are fixed to the sum of the token embeddings and the position embeddings. Hence, only the query inputs (**Q**) are updated across the layers during the inference stage by referring to the fixed output of the embedding layer.

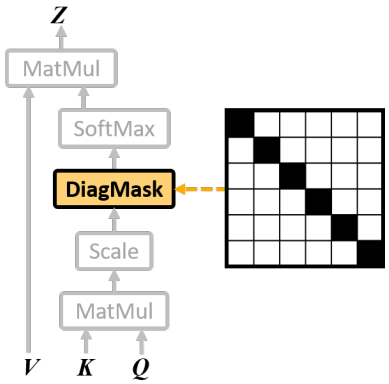


Figure 3: Diagonal masking of the scaled dot-product attention mechanism. The highlighted box and dashed arrow represent the innovations reported in this paper.

Additionally, we input the position embeddings to the **Q** of the very first encoding layer, thereby making the self-attention mechanism effective. Otherwise, the attention weights will be the same at all positions, and thus, the first self-attention mechanism will function as a simple average of all the input representations (except the "self" position). Finally, we apply the residual connection only to the query to completely maintain unawareness. The dashed arrows in Figure 2 show the proposed input isolation mechanism inside the T-TA.

By using diagonal masking and input isolation in conjunction, the T-TA can have multiple encoder layers, enabling the T-TA to obtain high-quality contextual language representations after feeding a sequence into the model only once.

4.3 Discussion and Analysis

Heretofore, we have introduced the new learning objective named LAE, and the novel deep bidirectional language model named T-TA. We will verify the architecture of the proposed T-TA in Section 4.3.1 and compare our model with the recently proposed strong baseline BERT in Section 4.3.2.

4.3.1 Verification of the Architecture

Here, we discuss how diagonal masking with input isolation preserves the "self-unknown" property in detail.

As shown in Figure 2, we have two input embeddings, namely, token embeddings $\mathbf{X} = [X_1, \ldots, X_n]^T \in \mathbb{R}^{n \times d}$ and position embeddings $\mathbf{P} = [P_1, \ldots, P_n]^T \in \mathbb{R}^{n \times d}$, where d is an embedding dimension. From the input isolation mechanism, the key and value $\mathbf{K} = \mathbf{V} = \mathbf{X} + \mathbf{P}$ have the information of the input tokens and are *fixed* in

all layers, but the query $\mathbf{Q}^l$ is *updated* across the layers during the inference stage starting from the position embeddings $\mathbf{Q}^1 = \mathbf{P}$ in the first layer.

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

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

where $\text{SMSAN}(\cdot)$ is the self-masked self-attention network, namely, the encoding layer of the T-TA, $g(x) = \text{Norm}(\text{Add}(x, \text{FeedForward}(x)))$ signifies two upper subboxes of the encoding layer in Figure 2, and $f(\cdot)$ is the (multihead) diagonal-masked self-attention (DMSA) mechanism. As illustrated 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 $\mathbf{V}$ while discarding the information of the i-th token X_i in V_i. Specifically, Z_i^l is the weighted average of $\mathbf{V}$ with the attention weight vector $\mathbf{s}_i^l$, *i.e.*, $Z_i^l = \mathbf{s}_i^l\mathbf{V}$, where $\mathbf{s}_i^l = [s_1^l, \ldots, s_{i-1}^l, 0, s_{i+1}^l, \ldots, s_n^l] \in \mathbb{R}^{1\times n}$. Here, we note that the DMSA mechanism is related only to the "self-unknown" property since no token representations are referred to each other in subsequent transformations from $\mathbf{Z}^l$ to $\mathbf{H}^l$. Therefore, we can guarantee that the i-th element of the query representation in any layer, Q_i^l, never encounters the corresponding token representation starting from $Q_i^1 = P_i$. Consequently, the T-TA preserves the "self-unknown" property during the inference stage while maintaining the residual connection and multilayer architecture.

4.3.2 Comparison with BERT

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

- While BERT uses an external masking operation in the input, the T-TA has an internal masking operation in the model, as we intend. Additionally, while BERT is based on a denoising autoencoder, the T-TA is based on an autoencoder. With this novel approach, the T-TA does not need *mask-and-predict* repetition during the computing of contextual language representations. Consequently, we reduce the computational complexity from $O(n^3)$ with the BERT to $O(n^2)$ with the T-TA in applications to unsupervised learning tasks.
- As in the T-TA, feeding an intact input (without masking) into BERT is also possible. However, we argue that this process will significantly diminish the model performance in unsupervised applications since the MLM objective does not consider intact tokens much. In the next section, we include experiments that reveal the model performance with intact inputs (described in Tables 1, 3, and 4). For further reference, we also suggest a previous study 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 T-TA on 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 BERT during the inference stage (Section 5.2) while showing competitive or even better accuracies than those of BERT on reranking tasks (Section 5.3) and STS tasks (Section 5.4).

5.1 Language Model Setups

The main purpose of this paper is to compare the proposed T-TA with a biLM trained with the MLM objective. For a fair comparison, each model has the same number of parameters based on the Transformer as follows: $|L| = 3$ 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 Gaussian error linear unit (*gelu*) activation function (Hendrycks and Gimpel, 2016) rather than the standard rectified linear unit (*relu*) following OpenAI GPT (Radford et al., 2018) and BERT (Devlin et al., 2019). In our experiments, we set the 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. We use WordPiece embeddings (Wu et al., 2016) with a vocabulary of approximately $|V| \simeq 30,000$ 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 uniLM that has the same number of parameters as the T-TA and biLM.

For training, we create a training instance consisting of a single sentence with `[BOS]` and `[EOS]` tokens at the beginning and end of each sentence, respectively. We use 64 sentences as the training batch and train the language models over $1M$ 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 $1e-4$ and coefficients of $\beta_1 = 0.9$ of $\beta_2 = 0.999$; the learning rate is set to warm up over the first 50k steps, and the learning rate exhibits linear decay. We use a dropout probability of 0.1 on all layers. Our implementation is based on Google's official code for BERT[2].

To train the language models that we implement, we use an English Wikipedia dump (approximately 13 GB in size) containing approximately $120M$ sentences. The trained models are used for reranking in NMT and unsupervised STS tasks. For the ASR reranking task, we use additional in-domain training data, namely, 4.0 GB of normalized text data from the official LibriSpeech corpus containing approximately $40M$ sentences.

5.2 Runtime Analysis

We first measure the runtime of each language model to compute 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 $\mathbf{H}^L$ for the analysis. In the case of the reranking task, further computation is required: we compute $\text{Softmax}(\mathbf{H}^L\mathbf{E}^T)$ to obtain the likelihood of each token, where $\mathbf{E} \in \mathbb{R}^{|V|\times d}$ is the weight parameter of the softmax layer. Therefore, the computational complexity of the reranking task is larger than that of the STS task.

To measure the runtime, we use an Intel(R) Core(TM) i7-6850K CPU (3.60 GHz) and the TensorFlow 1.12.0 library with Python 3.6.8 on Ubuntu 16.04.06 LTS. In each experiment, we measure the runtime 50 times and average the results.

Figure 4 shows that the T-TA exhibits faster runtimes than the biLM, and the gap between the T-TA and biLM increases as the sentence becomes longer. To facilitate a numerical comparison, we set the standard number of words to 20, which is approximately the average number of words in a contemporary English sentence (DuBay, 2006). In this setup, in the STS tasks, the T-TA takes approximately 9.85 ms, while the biLM takes approximately 125 ms; hence, the T-TA is 12.7 times faster than the biLM. In the reranking task, the T-TA is 6.35 times faster than the biLM (which is still significant); this reduction occurs because the repetition of the biLM is related only to computing $\mathbf{H}^L$ rather than $\text{Softmax}(\mathbf{H}^L\mathbf{E}^T)$.

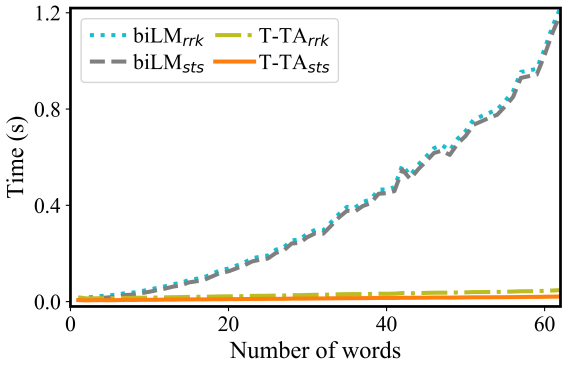


Figure 4: Average runtimes of each model according to the number of words on STS and reranking tasks, subscripted as *sts* and *rrk*, respectively.

For the visual clarity of Figure 4, we omit the runtime results of the uniLM, which is as fast as the T-TA (see Appendix B.1). With such a fast inference time, we next demonstrate that the T-TA is as accurate as BERT.

5.3 Reranking the N-best List

To evaluate the language models, we conduct experiments on the unsupervised task of reranking the N-best list. In these 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 Appendices 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 score_{s2s} is the score from the sequence-to-sequence model, score_{lm} is the score from the language model calculated by the sum (or mean) of the log-likelihood of each token, and the interpolation weight λ is set to a value that leads to the best performance in the development set.

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

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

We note that the T-TA and biLM (including BERT) assign the pseudo-log-likelihood to the score of a given sentence, whereas the uniLM assigns the log-likelihood. Because the reranking task is based on the relative scores of the n-best hypotheses, the fact that the bidirectional models yields the pseudo-log-likelihood of a given sentence does not impact this task (Wang and Cho, 2019; Shin et al., 2019; Salazar et al., 2019).

5.3.1 Results on ASR

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

First, we confirm that the bidirectional models trained with the LAE (T-TA) and MLM (biLM) objectives consistently outperform the uniLM trained with the CLM objective. The performance gains from reranking are much lower in the better base system ***Seq2Seq*$_{ASR}$**, and it is evidently 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 with (or even better than) the biLM; this may result from the gap between the 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 are critical to language model applications.

Finally, we note that feeding an intact input to BERT (the corresponding model is denoted as "w/ BERT$_{\backslash M}$" in Table 1) causes the model to underperform relative to the other models, demonstrating that the *mask-and-predict* approach is necessary for effective reranking.

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$_w$	6.05	17.32	6.11	18.13
w/ $*$biSANLM$_w$	5.52	16.61	5.65	17.37
w/ BERT	5.24	16.56	5.38	17.46
w/ BERT$_{\backslash M}$	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$_{\backslash M}$	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. The 'other' sets are recorded in noisier environments than the 'clean' sets. Bold font denotes the best performance on each subtask, and $*$ signifies a word-level language model from Shin et al. (2019).

5.3.2 Results on NMT

To compare the reranking performances in another domain, NMT, we again prepare N-best lists using ***Seq2Seq*$_{NMT}$**[3] from the WMT13 German-to-English (De→En) and French-to-English (Fr→En) test sets. Table 2 shows the bilingual evaluation understudy (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 are 0.4 or 0.5 in the N-best lists for NMT.

We confirm again that the bidirectional models trained with the LAE and MLM objectives perform better than the uniLM trained with the CLM objective. Additionally, the Fr→En translation has less effect on the reranking than the De→En translation because the base NMT system for Fr→En is better than that for De→En. The 12-layer BERT model appears much better than the other models at reranking on NMT; hence, the N-best hypotheses of the NMT model seem to be more indistinguish-

[3]The ***Seq2Seq*** models for De→En and Fr→En are trained independently using the t2t library (Vaswani et al., 2018).

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	<u>30.32</u>
w/ T-TA	<u>28.83</u>	30.20

Table 2: BLEU scores after reranking with each language model on WMT13. Bold font denotes the best performance on each subtask, and the underlined values signify the best performances in our implementations.

Method	STS-B-dev		STS-B-test	
	context	*embed*	*context*	*embed*
BERT	**64.78**	-	**54.22**	-
BERT$_{\setminus M}$	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$_{\setminus M}$	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 the STS-B dataset. "-" denotes an infeasible value, and bold font denotes the top 2-performing models on each subtask.

able than those of the ASR model from a language modeling perspective.

All the reranking results on the ASR and NMT tasks demonstrate that the proposed T-TA performs both efficiently (similar to the uniLM) and effectively (similar to the biLM).

5.4 Unsupervised STS

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

In this section, we address the *unsupervised* STS task to examine the inherent ability of each language model to obtain contextual language representations, and we mainly compare the language models that are trained on the English Wikipedia dump. To compute the 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 representations. Specifically, the contextual representations of a given sentence are the outputs of the final encoding layer of each model, denoted as *context* in Tables 3 and 4. For comparison, we use noncontextual representations, which are obtained from the outputs of the embedding layer, denoted as *embed* in Tables 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 employ BERT in the *mask-and-predict* approach for computing the contextual representations of each sentence. Note that we use the most straightforward approach for the unsupervised STS task to focus on comparing token-level language representations.

5.4.1 Results on STS-B

The STS-B dataset has 5749/1500/1379 sentence pairs with train/dev/test splits and corresponding scores ranging from 0 to 5. We test the language models on the STS-B-dev and STS-B-test sets using the simplest approach on the unsupervised STS task. As 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[4].

Table 3 shows our T-TA trained with the LAE objective best captures the semantics of a sentence over the Transformer-based language models. Remarkably, our 3-layer T-TA trained on a relatively small dataset outperforms the 12-layer BERT trained on a larger dataset (Wikipedia + BookCorpus). Furthermore, the embedding representations are trained better by the CLM objective than by the other language modeling objectives; we suppose that the uniLM depends strongly on the embedding layer due to its unidirectional context constraint.

Since the uniLM encodes all contexts in the last token, `[EOS]`, we also use the last representation as the sentence representation; however, this approach does not outperform the average sentence

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

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

Table 4: Pearson's $r \times 100$ results on the SICK dataset. "-" denotes an infeasible value, and bold font denotes the best performance on each subtask.

representation. Similarly, BERT has a special token, `[CLS]`, which is trained for the "next sentence prediction" objective; thus, we also use the `[CLS]` token to see how this model learns the sentence representation, but it significantly underperforms the other models.

5.4.2 Results on SICK

We further evaluate the language models on the SICK dataset, which consists of 4934/4906 sentence pairs with training/testing splits and scores ranging from 1 to 5. The results are in Table 4, from which we obtain the same observations as those reported for STS-B.

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

6 Conclusion

In this work, we propose a novel deep bidirectional language model, namely, the T-TA, to eliminate the computational overload of applying BERT to unsupervised applications. Experimental results on N-best list reranking and unsupervised STS tasks demonstrate that the proposed T-TA is significantly faster than the BERT-like approach, and its encoding ability is competitive with (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 the Industrial Technology Innovation Program (No.10073144) and by the NRF grant funded by the Korean 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*.

Appendix

A Implementation Details

A.1 Setup for the ASR System

This section introduces our implementation of the ASR system.

For the input features, we use an 80-band Mel-scale spectrogram derived from the speech signal. The target sequence is processed in 5K case-insensitive subword 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 long short-term memory (LSTM) network, and there are bottleneck layers that conduct a linear transformation between every LSTM layer. Additionally, there is a VGG module before the encoder, and it reduces the number of encoding time steps by one-quarter through two max-pooling layers. The decoder is a 2-layer bidirectional LSTM network with a location-aware attention mechanism (Chorowski et al., 2015). All the layers have 1024 hidden units. The model is trained with an additional connectionist temporal classification (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 960 h of LibriSpeech training data using the Adadelta optimizer (Zeiler, 2012). Using this acoustic model, we obtain the 50 best decoded sentences for each input audio file through the hybrid CTC-attention-based scoring (Hori et al., 2017) method. For ***Seq2Seq***$_{ASR}$, we additionally use a pretrained recurrent neural network language model (RNNLM) to combine the log-probability p^{lm} of the RNNLM during decoding as follows:

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

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

Table 5 shows the oracle word error rates (WERs) of the 50 best lists 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).

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.

A.2 Setup for the NMT System

We implement the standard Transformer model (Vaswani et al., 2017) using the Tensor2Tensor library (Vaswani et al., 2018) for NMT. Both the encoder and the decoder of the Transformer consist of 6 layers with 512 hidden units, and the number of self-attention heads is 8. The maximum number of input tokens is set to 256, and we use a 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[5].

We train the baseline model on the standard WMT13 Fr→En and De→En datasets with 250k steps using the 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 number of warmup steps (16k). Using this baseline translation model, we obtain the 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 BLEU scores of the 50 best lists on WMT13

B Additional Experiments

B.1 Runtimes of the uniLM and T-TA

As mentioned in Section 5.2, we also measure the runtimes of the uniLM we implement. Figure 5

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

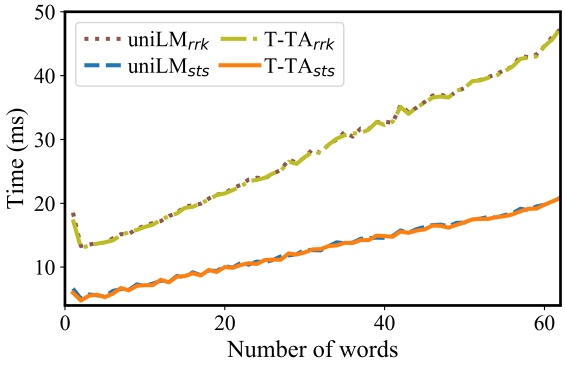


Figure 5: Runtimes according to the number of words for the uniLM and T-TA.

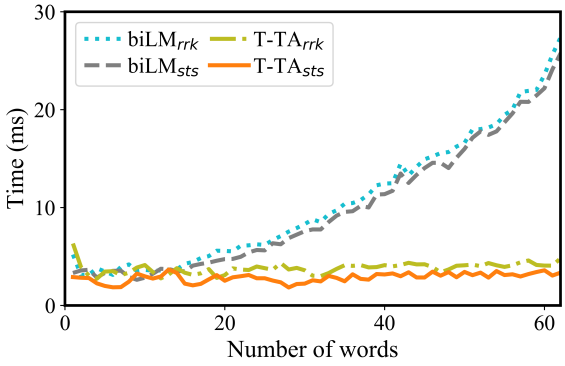


Figure 6: Runtimes according to the number of words for the biLM and T-TA in the GPU-augmented environment.

shows the average runtimes of the uniLM and the T-TA for the number of words in a sentence. Since we use subword tokens, the number of words n_w and the number of tokens n can be different ($n_w \leq n$).

B.2 Runtimes on a GPU

Additionally, we similarly measure the runtimes in a GPU-augmented environment (using GeForce GTX 1080 Ti). Figure 6 shows the average runtimes of the biLM and the T-TA for the number of words in a sentence. In our 20-word standard in the STS task, the T-TA takes approximately 2.51 ms, whereas biLM takes approximately 4.72 ms, showing that the T-TA is 1.88 times faster than the biLM. Compared to the CPU-only environment, the speed difference is significantly reduced due to the support offered by the GPU. Considering Figure 4, however, the CPU-only environment and GPU-augmented environment show a similar tendency: the longer the sentence is, the more significant the difference in the runtime between the T-TA and the biLM.

B.3 Perplexity and Reranking

In general, perplexity (PPL) is a measure of how well the language model is trained. To investigate the alignment of the PPL and reranking, we compute the PPL of reference sentences from the LibriSpeech dev-clean and test-clean sets using each language model. We can obtain the pseudoperplexity (pPPL) from the biLM and T-TA since they do not follow the product rule, unlike the uniLM. Note that we compute the 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 the language models on LibriSpeech.

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