VEE-BERT: Accelerating BERT Inference for Named Entity Recognition via Vote Early Exiting

Anonymous ACL submission

Abstract

Named entity recognition (NER) is of great importance for a wide range of tasks, such as medical health record understanding, document analysis, dialogue understanding. BERT and its variants are the most performing models for NER. However, these models are notorious for being large and slow during inference. Thus their usage in the industry is limited. Pilot experiments exhibit that in the NER task, BERT suffers from the severe over-thinking problem, thus motivating BERT to exit early at intermediate layers. Thus, in this work, we propose a novel method, Vote Early Exiting BERT (VEE-BERT), for improving the early exiting of BERT on NER tasks. To be able to deal with complex NER tasks with nested entities, we adopt the Biaffine NER model (Yu et al., 2020), which converts a sequence labeling task to the table filling task. VEE-BERT makeS early exiting decisions by comparing the predictions of the current layer with those of the previous layers. Experiments on six benchmark NER tasks demonstrate that our method is effective in accelerating the BERT Biaffine model's inference speed with less performance loss compared to the baseline early exiting method.

1 Introduction

Since BERT (Devlin et al., 2018), the pre-trained language models (PLMs) become the default state-of-the-art (SOTA) models for natural language processing (NLP). The recent years have witnessed the rise of many PLMs, such as GPT (Radford et al., 2019), XLNet (Yang et al., 2019), and ALBERT (Lan et al., 2020), and so forth. These BERT-style models achieved considerable improvements in many Natural Language Processing (NLP) tasks by pre-training on the unlabeled corpus and fine-tuning on labeled tasks, such as text classification, natural language inference (NLI), sequence labeling, etc. Despite their great performances, there are two issues for PLMs.

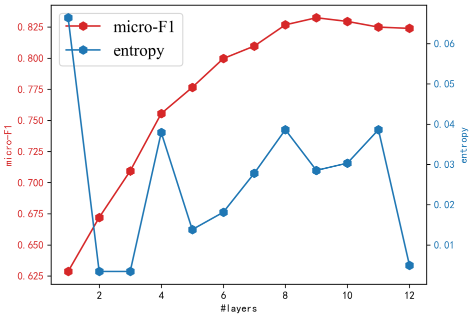
First, previous studies show that PLMs such as BERT suffer from the over-thinking problem. (Zhou et al., 2020; Zhu et al., 2021; Zhu, 2021) shows that in the sentence classification task, BERT's last few layers may be too deep for some samples. For a sentence classification task, if we insert a classifier on a certain intermediate layer and drop the deeper layers, these intermediate layers may outperform the last layer. Note that the previous literature focuses on the classification task, which is at the sentence level. Thus, one may wonder, is over-thinking present in the more fine-grained tasks like named entity recognition?

We conducted a pilot experiment on the ACE2004[1] task, which is a nested NER task, and the CONLL2003 (Tjong Kim Sang and De Meulder, 2003) task, which is a flat NER task. Figure 1(a) and 1(b) shows that the over-thinking problem is present in the NER tasks. On both tasks, layer 9 performs the best, and there are four layers that are better than the last layer. The overthinking problem motivates us to only use a portion of the BERT base to make predictions on the test samples.

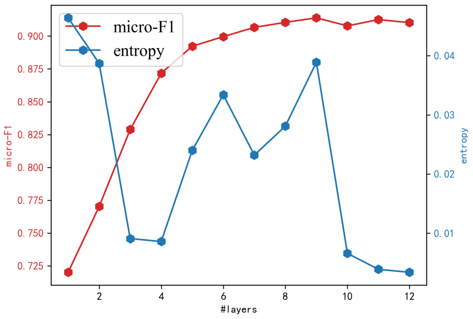
The second drawback of PLMs is their high latency. NER and other sequence labeling tasks play a central role in many application scenarios, such as question answering, document search, document-level information extraction, etc. However, these applications require low latency. For example, an online search engine needs to respond to the user's query in less than 100 milo-seconds. Thus, a NER module should be efficient and accurate. In addition, a special feature of consumer queries is that there are time intervals that the number of queries is extremely high. For example, during dinner hours, food search engines will be used much often than usual. Thus, it is important for deployed models to adjust their latency dynamically.

There exists a branch of literature focusing on making PLMs' inference more efficient via adap-

[1] https://catalog.ldc.upenn.edu/LDC2005T09



(a) Overthinking on the ACE2004 task



(b) Overthinking on the CONLL2003 task

Figure 1: This figure demonstrates that the overthinking problem prevails in the NER tasks.

tive inference (Zhou et al., 2020; Xin et al., 2020a; Liu et al., 2020). The idea of adaptive inference is to process simple examples with only shallow layers of BERT and process more difficult queries with deeper layers, thus significantly speeding up the inference time on average while maintaining high accuracy. The speed-up ratio can be easily controlled with certain hyper-parameters without re-deploying the model services or maintaining a group of models. Early exiting is one of the most important adaptive inference methods (Bolukbasi et al., 2017). As depicted in Figure 2, it implements adaptive inference by installing an early exit, i.e., an intermediate prediction layer, at each layer of BERT and early exiting "easy" samples to speed up inference. At the training stage, all the exits are jointly optimized with BERT's parameters. At the inference stage, some strategies for early exiting are designed to decide whether to exit at each layer given the currently obtained predictions (from previous and current layers) (Teerapittayanon et al., 2016; Kaya et al., 2019; Xin et al., 2020a; Zhou et al., 2020). In this mode, different samples can exit at different depths.

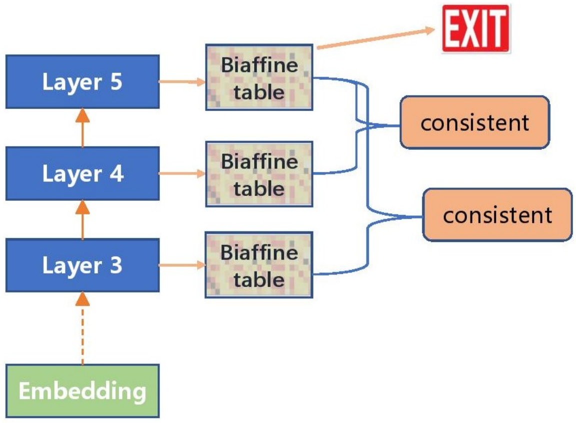


Figure 2: Our proposed VEE-BERT method for accelerating BERT biaffine NER model's inference speed. VEE-BERT compares the current layer with the previous layers. It will decide to exit at the current layer when there are enough previous layers making predictions that are consistent with the current layer's.

In order for our framework to be generally applicable, we mainly adopt the biaffine model (Yu et al., 2020) for NER. The biaffine model converts the NER task into a 2-dimensional table filling task, thus providing a solution to the nested NER problem. (Yu et al., 2020) shows that the biaffine model can achieve the state-of-the-art (SOTA) performances on not only nested NER tasks but also flat NER tasks.

In this work, we propose a novel early exiting method, VEE-BERT, designated for the NER task. Different from previous work such as DeeBERT (Xin et al., 2020b) and RightTool (Schwartz et al., 2020), we look into the consistency of different intermediate layers. At a certain intermediate layer, we first compute the biaffine logits. VEE-BERT mainly uses the KL divergence as the measurement of consistency. That is, if the KL divergence between two layers' logits is small, their predictions are consistent with each other. Our VEE-BERT compares the current layer's prediction with the lower layers'. If the number of previous intermediate layers that have consistent predictions with the current layer exceeds the patience parameter, BERT will stop further inference and exit. Intuitively, the decision of early exit is made when enough layers agree with one another and make the votes. Thus, our method can be seen as the ensemble of the current and previous layers.

Extensive experiments are conducted on the six benchmark NER tasks. Three of the tasks are nested NER tasks, ACE2004[2], ACE2005[3], GENIA (Kim et al., 2003b). We also experiment on three flat NER tasks, CONLL2003 (Tjong Kim Sang and De Meulder, 2003), OntoNotes 4.0 Chinese[4] and the Chinese MSRA task (Levow, 2006). We show that our VEE-BERT consistently achieves better performances under the same speed-up ratio, compared with a series of the previous early exiting methods. Deeper analysis and ablation studies result in the following main takeaways: (a) our method works for different pre-trained language models (PLMs) like ALBERT; (b) we compare with a wide range of consistency measures, such as edit distance, Euclidean distance, cosine similarity, and demonstrate that KL divergence performs the best.

The rest of the paper is organized as follows. First, we introduce the preliminaries for the Biaffine NER model and early exiting. Second, we elaborate on our VEE-BERT method. Third, we conduct experiments on 6 NER tasks and conduct a series of ablations studies. Finally, we conclude with possible future works.

2 Preliminaries

In this section, we introduce the necessary background for BERT and early exiting. Throughout this work, we consider the case of a NER task with samples $\{(x, y), x \in \mathcal{X}, y \in \mathcal{Y}, i = 1, 2, ..., N\}$, e.g., sentences and their NER span information, and the number of entity categories is K (including the non-entity type label). The input sequence length after BERT's subword tokenization is L.

2.1 Backbone models

In this work, we adopt BERT as the backbone model. BERT is a multi-layer Transformer (Vaswani et al., 2017) network, which is pre-trained in a self-supervised manner on a large corpus. In the ablation studies, we also use ALBERT (Lan et al., 2020) as backbones. ALBERT is more lightweight than BERT since it shares parameters across different layers, and the embedding matrix is factorized. The number of transformer layers of our backbone is denoted as M, and the hidden dimension is d.

[2]https://catalog.ldc.upenn.edu/LDC2005T09

[3]https://catalog.ldc.upenn.edu/LDC2006T06

[4]https://catalog.ldc.upenn.edu/LDC2011T03

2.2 The Biaffine model for NER

The BERT-Biaffine model (Yu et al., 2019) transform the NER task into a two-dimensional table filling task. It asks the model to identify whether the slot in the table with coordinate (s, e) corresponds to an entity with category k, that is, whether a pair of tokens (x_s, x_e) in the input sequence $x = (x_1, x_2, , , , , x_L)$ is the start and end tokens for an entity with category k. Formally, after BERT encoding, the contextualized embedding of tokens s and e are h_s and h_e (h_s, $h_e \in \mathcal{R}^d$). Then in a biaffine layer f, the score of span (s, e) is calculated by[5]

$$f(s, e) = h_s^T U h_e + W(h_s \oplus h_e) + b. \quad (1)$$

Since we need to calculate the scores for K entity categories, U is a $d \times K \times d$ tensor, and W is a $2d \times K$ tensor. $f(s, e) \in \mathcal{R}^K$ is the scores (or logits). A softmax operation will transform $f(s, e)$ into a probability distribution $p(s, e)$, which represents how likely the span (s, e) is a category k entity.

The learning objective of the biaffine model is to assign a correct category (including the non-entity) to each valid span. Hence it is a multi-class classification problem at each slot of the two-dimensional table and can be optimized with cross-entropy loss:

$$\mathcal{L} = -\sum_{s=1}^{L} \sum_{e=s}^{L} \sum_{k=1}^{K} \mathcal{I}(y(s, e) = k) \log p_k(s, e), \quad (2)$$

where $y(s, e)$ is the ground-truth label of span (s, e), $p_k(s, e)$ is the predicted probability mass of (s, e) having label k, and $\mathcal{I}(\cdot)$ is the indicator function. After fine-tuning the BERT biaffine model, the inference procedure of the BERT biaffine model follows Yu et al. (2019).

2.3 Early-exiting Architecture

As depicted in Figure 2, early exiting architectures, or multi-exit architectures, are networks with exits[6] at each transformer layer. Since the previous literature usually considers sentence-level classification tasks, the exits are classifiers. However, since we are dealing with sequence labeling tasks formulated

[5]Note that in the BERT biaffine NER (Yu et al., 2019), two feed forward layers are designated to transform the features of h_s and h_e. However, we find that dropping the two feed forward layers and increase the learning rate for the biaffine module result in slightly better test performances.

[6]Some literature (e.g., DeeBERT (Xin et al., 2020a)) also refers to exits as off-ramps.

as two-dimensional table filling, with M exits, M separate biaffine modules $f^{(m)}$ are installed right after each layer of BERT ($m = 1, 2, ..., M$), and the scores for span (s, e) at layer m is given by:

$$f^{(m)}(s,e) = h_s^T U^{(m)} h_e + W^{(m)}(h_s \oplus h_e) + b^{(m)}. \quad (3)$$

And the loss function at each layer becomes

$$\mathcal{L}^{(m)} = -\sum_{s=1}^{L}\sum_{e=s}^{L}\sum_{k=1}^{K} \mathcal{I}(y(s,e)=k)\log p_k^{(m)}(s,e), \quad (4)$$

where $p^{(m)}(s,e) = \text{Softmax}(f^{(m)}(s,e))$ is the predicted probability distribution at exit m.

2.3.1 Training

At the training stage, all the exits are jointly optimized with a summed loss function. Following Huang et al. (2017) and Zhou et al. (2020), the loss function is the weighted average of the losses (from Equation 4):

$$\mathcal{L}^{WA} = \frac{\sum_{m=1}^{M} m * \mathcal{L}^{(m)}}{\sum_{m=1}^{M} m}. \quad (5)$$

Note that the weight m corresponds to the relative inference cost of exit m.

2.3.2 Inference

At inference, the multi-exit BERT can operate in two different modes, depending on whether the computational budget to classify an example is known or not.

Budgeted Exiting. If the computational budget is known, we can directly appoint a suitable exit m^* of BERT, $f^{(m^*)}$, to predict all queries.

Dynamic Exiting. Under this mode, after receiving a query input x, the model starts to predict on the classifiers $f^{(1)}$, $f^{(2)}$, ..., in turn in a forward pass, reusing computation where possible. It will continue to do so until it receives a signal to stop early at an exit $m^* < M$, or arrives at the last exit M. At this point, it will output the final predictions based on the current and previous predictions. Note that under this early exit setting, different samples might exit at different layers.

3 Over-thinking problems in the NER tasks

Although the over-thinking problem in the sentence classification task is investigated in Zhou et al. (2020); Zhu (2021); Zhu et al. (2021), whether BERT has the over-thinking problem in the NER task is neglected by the literature. Thus, we first conduct pilot experiments on two benchmark NER tasks, ACE2004, a nested NER task, and CoNLL2003 (Sang and De Meulder, 2003), a flat NER task. We measure the confidence level of an exit m by the average predicted distribution entropy on the two-dimensional table:

$$g^{(m)} = -\sum_{s=1}^{L}\sum_{e=s}^{L} \frac{\sum_{k=1}^{K} p_k^{(m)}(s,e) * \log p_k^{(m)}(s,e)}{\log(1/K) * (L(L+1)/2)}. \quad (6)$$

This confidence level is thoroughly studied in Teerapittayanon et al. (2016); Xin et al. (2020a). Note that low entropy indicates the current layer has high confidence.

In Figure 1(a) and 1(b), we plot each layer's average confidence level and the micro-f1 score on the development sets of ACE2004 and CoNLL2003. We can clearly see that, generally, as more layers of the pre-trained BERT are utilized during prediction, the model becomes quite confident. However, more layers may not result in better performances. On the ACE2004 task, BERT's layer 9 achieves the highest score, and the performances drop as the number of layers further increases. Similarly, results can also be found on the CoNLL2003 task.

The results in Figure 1(a) and 1(b) convey two important messages.

- First, the "overthinking" phenomenon is similar to the observations on the sentence classification task (Kaya and Dumitras, 2018; Zhou et al., 2020; Zhu, 2021). This phenomenon is a direct motivation for early exiting since some of the input queries do not need to utilize the full capacity of BERT to obtain the correct prediction.

- In the classification task, when the layer depth increases, the average confidence level will usually monotonically increase. However, we can see that in Figure 1(a) and 1(b), the depth-confidence curve drops quite drastically at the first few layers and fluctuates on deeper layers. And the first 2-3 layers can already achieve high confidence while their performances have a clear gap from the final layer. Thus, the confidence level is not an effective representation of the NER task's difficulty or performance score.

With the observation of overthinking problems in NER, we are motivated to design an early exiting strategy to avoid unnecessary computation and increase the inference speed. Previous work (Li et al., 2021) design early exiting strategies for NER based on the confidence level, which can be seen as the NER version of BranchyNet (Teerapittayanon et al., 2016). However, our pilot experiments show that entropy is not a proper estimation for the model to determine whether the sample is well understood and predicted.

4 Vote Early Exiting

Note that the entropy-based early exiting method makes the exiting decision based on the prediction of a single intermediate layer. However, the multi-exit BERT has a series of layers that may convey important information for us. Thus, we focus on designing an early exiting mechanism based on the outputs of multiple layers.

In this work, we propose Voting Early Exiting BERT (VEE-BERT), a novel off-the-shelf early exiting method that can speedup the BERT biaffine models' inference speed without changing the fine-tuned model. VEE-BERT mines the early exiting signal from the comparison among the biaffine scores of the intermediate layers. We first elaborate on how we compare the biaffine scores of two exits and when we will consider the predictions consistent with each other, then we will present our VEE-BERT.

4.1 Consistency measures

We denote the table of distributions predicted by the biaffine exit m as $\mathcal{T}^{(m)} = \{p^{(m)}(s,e)|s,e \in 1,...,L\}$, which is a $L \times L \times K$ tensor. With two exits, m_1 and m_2 ($m_1 < m_2$), we want to measure whether and to what degree their predictions are in consistency with each other, or similar to each other. The consistency score between $\mathcal{T}^{(m_1)}$ and $\mathcal{T}^{(m_2)}$ is denoted as $C(\mathcal{T}^{(m_1)}, \mathcal{T}^{(m_2)})$. If $C(\mathcal{T}^{(m_1)}, \mathcal{T}^{(m_2)})$ increases, the predictions of m_1 and m_2 are more in consistency with each other.

We can select from the four measures:

- **Distance of probability distributions.** Since we have a probability distribution $p^{(m)}(s,e)$ at each slot of the two-dimensional table, it is natural to use the average Kullback-Leibler divergence (Kullback and Leibler, 1951):

$$C(\mathcal{T}^{(m_1)}, \mathcal{T}^{(m_2)}) = -\sum_{s=1}^{L}\sum_{e=s}^{L} \frac{D_{KL}(p^{(m_1)}(s,e)||p^{(m_2)}(s,e))}{\log(1/K) * (L(L+1)/2)}, \tag{7}$$

where $D_{KL}(p||q)$ is the KL divergence from the distribution q to p.[7]

- **Euclidean distance.** Since $\mathcal{T}^{(m_1)}$ and $\mathcal{T}^{(m_2)}$ are $L \times L \times K$ tensors, we can use the negative of Euclidean distance to measure their distance.

- **Cosine similarity.** We can reshape $\mathcal{T}^{(m_1)}$ and $\mathcal{T}^{(m_2)}$ into vectors and use the cosine similarity as the consistency measure.

- **Edit distance.** In this method, we fill the two-dimensional table with the labels that receive the maximum probability scores and reshape the table into a single-dimensional list. After the transformation, we can calculate the edit distance between the lists coming from $\mathcal{T}^{(m_1)}$ and $\mathcal{T}^{(m_2)}$, and use the negative of edit distance as the consistency measure.

We adopt the negative of average KL divergence as the default consistency measure. We will conduct ablation studies and compare the performances of different consistency measures.

4.2 VEE

Recall that early stopping (Girosi et al., 1995) occurs when the training loss becomes stable and parameter updates no longer begin to yield improves on a validation set. By analog, we make the early exiting decision based on the cross-layer comparison. The inference process of VEE-BERT is depicted in Figure 2. In VEE-BERT, we need two hyper-parameters, consistency score τ and the patience parameter cnt^*.

Assume the forward pass has arrived at the intermediate layer m, and we use exit m to obtain the table of predicted distribution $\mathcal{T}^{(m)}$. We initialize a patience counter cnt_m with value 0. We now compare $\mathcal{T}^{(m)}$ with each of the previous layer's prediction $\mathcal{T}^{(i)}$ ($i \in \{1, 2, ..., n-1\}$). If the consistency

[7] We also experiment on using alternative distance measures for probability distributions, such as Helinger coefficient (Hellinger), Jensen–Shannon divergence (Manning and Schütze, 2002), and so on, but does not result in any statistically significant improvements.

score $C(\mathcal{T}^{(i)}, \mathcal{T}^{(m)})$ is larger than the threshold τ, cnt_m will add 1. If cnt_m reaches or exceeds the pre-defined patience parameter cnt*, we can early exit at the current layer m and consider cnt_m as the final prediction. If this condition can not be satisfied at the intermediate layers, the model makes the full forward pass and use the whole BERT backbone for prediction.

Intuitively, our VEE-BERT method aggregates the predictions of the current layer m and the previous layers by asking them to vote whether they are in favor of (or being consistent with) the exit m's prediction. If there are enough previous layers voting in favor of exit m, BERT will make the call to exit early.

5 Experiments

5.1 Datasets

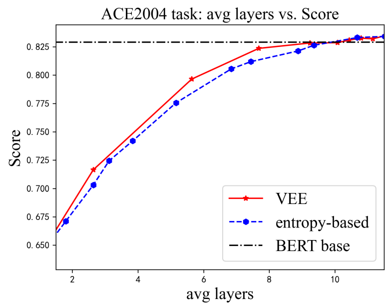
We evaluate our VEE-BERT on both nested and flat NER tasks. For the nested NER task, we use the ACE2004 task[8], ACE2005 task[9], and GENIA task (Kim et al., 2003a). For the flat NER task, we evaluate our method on the CONLL2003 (Sang and De Meulder, 2003), the OntoNotes 4.0 corpus[10], and the MSRA NER task (Levow, 2006).

5.2 Baselines

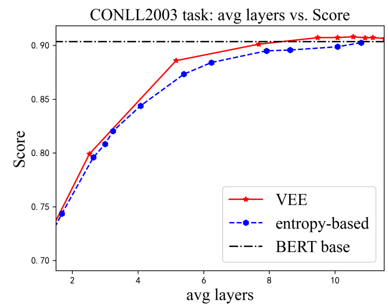
BERT biaffine (Yu et al., 2019). This baseline model uses all the BERT layers to encode sentences and make predictions by the biaffine module at the final layer.

BranchyNet (Teerapittayanon et al., 2016). This is a widely adopted method for early exiting and are utilized in Liu et al. (2020); Xin et al. (2020a); Li et al. (2021). For classification, it uses the entropy of the predicted probabilities as the confidence level and will exit if the entropy is low. BranchyNet based early exiting is the SOTA method for early exiting on the NER tasks. However, this method can not be directly adopted for the Biaffine NER model. We consider using the average entropy score on the biaffine table (in Equation 6) as the early signal. That is, if the average entropy score at an intermediate layer is lower than a pre-defined threshold τ, the model will exit.

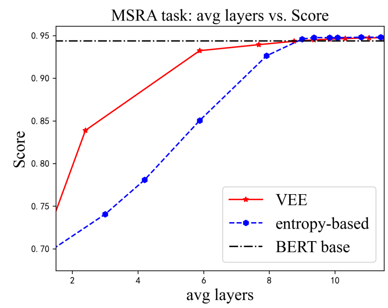
Shallow-Deep (Kaya et al., 2019) adopts the maximum probability mass of the predicted distribution as the early exiting signal for sentence classification tasks. To the best of our knowledge, this method has not been adopted for the NER task. In this work, we consider using the average of the maximum probability mass on the biaffine table as the early exiting signal.



(a) ACE2004 task



(b) CONLL2003 task



(c) MSRA task

Figure 3: Quality–efficiency trade-offs using different exiting strategies. We can see that our two versions of VEEs consistently outperform the entropy-based baseline.

5.3 Experimental settings

To perform a fair comparison, our VEEs and all baseline models adopt the same configuration as follows. English NER tasks use the open-sourced Google BERT (Devlin et al., 2019)[11] as the backbone, and the Chinese tasks adopt the BERT-www-

[8] https://catalog.ldc.upenn.edu/LDC2005T09
[9] https://catalog.ldc.upenn.edu/LDC2006T06
[10] https://catalog.ldc.upenn.edu/LDC2011T03
[11] https://huggingface.co/bert-base-uncased.

	ACE2004		ACE2005		CONLL2003		GENIA		MSRA		OntoNotes4.0	
	score	layer	score	layer	score	layer	score	layer	score	layer	score	layer
BERT biaffine	0.8290	12	0.8086	12	0.9035	12	0.7635	12	0.9438	12	0.8086	12
Shallow-Deep	0.7027	2.68	0.7024	2.78	0.8046	2.9	0.6475	2.79	0.7415	3.1	0.5678	2.92
	0.7753	5.14	0.7749	5.48	0.8736	5.42	0.7208	5.79	0.8510	5.92	0.6913	5.54
	0.8251	9.21	0.8089	9.63	0.8964	8.71	0.7612	10.1	0.9484	9.93	0.8129	9.66
BranchyNet	0.7031	2.63	0.7014	2.76	0.8082	3.0	0.6486	2.88	0.7406	3.0	0.5633	2.88
	0.7755	5.16	0.7751	5.52	0.8732	5.40	0.7212	5.76	0.8504	5.88	0.6907	5.52
	0.8263	9.36	0.8104	9.72	0.8957	8.64	0.7605	10.2	0.9475	9.84	0.8137	9.60
VEE-BERT	0.7165	2.63	0.6808	2.16	0.7990	2.52	0.6502	2.76	0.8389	2.39	0.6488	2.63
	0.7965	5.64	0.7862	5.39	0.8858	5.16	0.7288	5.40	0.9323	5.88	0.7770	5.52
	0.8283	9.24	0.8168	9.0	0.9072	9.48	0.7660	10.08	0.9458	9.84	0.8113	9.72

Table 1: Experimental results of models with BERT backbone on the GLUE's development set and test set.

ext released from (Cui et al., 2020)[12] as the backbone model. We also use ALBERT-base and ALBERT base Chinese by (Lan et al., 2020) as the backbone models for ablation studies.

Our implementation is adapted from the Huggingface Transformer Library[13]. We conduct searches on experiment settings such as the optimizer, learning rates, batch sizes, and warmup steps and discover that changing the original settings does not result in statistically significant improvements. Random seeds are also unchanged from the library for fair comparisons. We fine-tune models for at most 25 epochs; early stopping with patience eight is performed, and the best checkpoint is selected based on the dev performance score.

Experiments are done on a single NVIDIA P100 GPU with CUDA 10.1. For inference, we use a batch size of 1 since we need to perform early exiting based on each individual sample's difficulty.

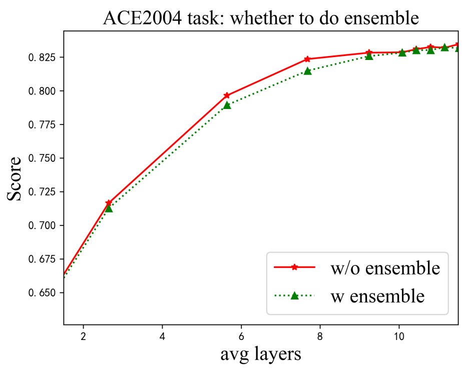


Figure 4: The results of ablation study on whether to do ensemble of all the available layers upon exiting.

[12]https://github.com/ymcui/Chinese-BERT-wwm
[13]https://github.com/huggingface/transformers

5.4 Main results

In this section, we compare quality–efficiency trade-offs of our VEEs and the baseline methods. Specifically, we use the average exit layer of all inference samples as the metric of efficiency. We choose this efficiency metric for the following reason: (1) it is linear w.r.t. the actual amount of computation; (2) according to our experiments, it is proportional to actual wall-clock runtime, and is also more stable across different runs compared with actual runtime due to randomness by other processes on the same machine.

We visualize the trade-offs on the test sets of the benchmark NER tasks in Figures 7. Detailed numbers are also shown in Table 1. Dots in the figures 7 and different rows of Table 1 are generated by varying the patience parameter cnt* and/or threshold τ. The main take-aways from the table and figures are as follows:

- On the test set, early exiting with our VEE-BERT methods saves a large amount of inference computation, with significantly less quality degradation when compared with the baseline methods.
- The entropy-based method BranchyNet slightly outperforms the max-probability-based method Shallow-Deep, especially when the average exiting layers are small.
- We can see that for most of the six tasks, utilizing the intermediate layers of BERT can outperform the whole BERT base model with fewer layers during inference. These results are consistent with the over-thinking phenomenon discussed in Section 3.
- From Figure 3(c), we can see the performance gaps between BranchyNet and our VEEs are

very significant on the Chinese NER tasks when the average exiting layers are low. A similar gap occurs on OntoNote 4.0, another benchmark Chinese NER task, which can be found in the Appendix.

5.5 Ablation studies

5.5.1 Ablation on cross-layer ensemble

Sun et al. (2021) argues that since we have a prediction module at each layer of BERT, we can conduct model ensemble by simply averaging the predicted probabilities of each layer we have go through already. Although cross-layer ensemble results in performances gains in the classification tasks like the GLUE benchmark, our experiments show that it is not always beneficial for the NER tasks. In Figure 4, we conduct the ablation studies on the ACE2004 task. We can see that in the NER task, cross-layer ensemble as in Sun et al. (2021) does not result in consistent performance improvements. At the shallow exits, ensemble distracts the current layer from making the correct decisions. However, as the layer number increases, ensemble indeed slightly improves model generalization. [14]

5.5.2 Ablation on the consistency measures

In our main experiments, we use the KL divergence as the consistency measure. To validate our choice, we now conduct ablation experiments showing KL divergence works the best among the consistency measurements we present in Subsection 4.1. The ablation results are conducted on the ACE2004 task, and the results are presented in Figure 5. We can conclude that KL divergence consistently results in better early exiting performances across different values of the patience parameter. And edit distance is the second-best consistency measure.

5.5.3 Ablation on the different backbones.

Although our main results are conducted on the BERT backbones, our VEE-BERT methods are off-the-shelf and also work well on other PLMs. We conduct experiments on the ALBERT base model. The results are shown in the Figure 6, which shows that our VEE-BERT early exiting mechanism is a plug-and-play method that can be used in different PLMs.

6 Conclusion

In this work, we investigate whether we can accelerate the inference speeds of the BERT biaffine NER

[14] Results on the MSRA task can be found in the Appendix.

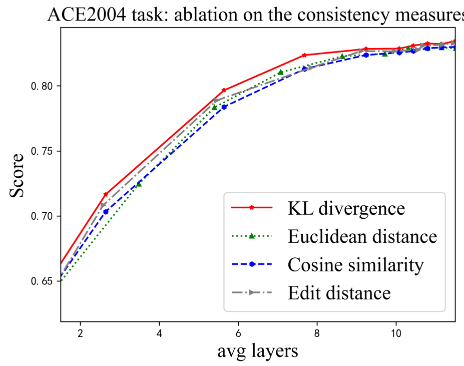


Figure 5: The results of ablation studies on the consistency measures. We can see that KL divergence consistently outperforms the other distance measures.

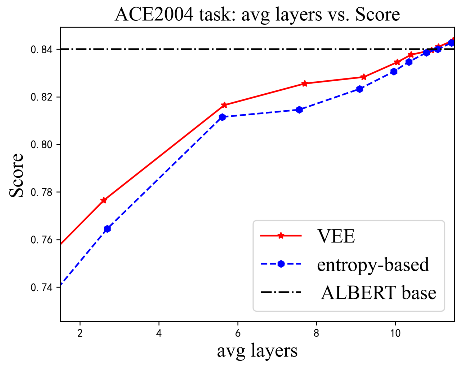


Figure 6: With ALBERT, our VEE-BERT method also performs well.

model. Pilot experiments show that the BERT biaffine model suffers from an over-thinking problem when applied in the NER tasks. To take advantage of the intermediate layers of BERT to speed up BERT inference, we propose Vote Early Exiting (VEE-BERT). Our VEE-BERT method compares the predictions of the current layer to all the previous layers. Comparison between the predictions of the two layers is made via distance measures like KL divergence, Euclidean distance, etc. If there are enough previous layers that are consistent with the current prediction, BERT will stop inference and exit at the current layer. Our VEE-BERT method mimics voting among the intermediate layers. Experiments on six benchmark NER tasks demonstrate that: (a) Our VEE-BERT method consistently outperforms the previous SOTA early exiting methods; (b) KL divergence works best with our VEE-BERT method as the consistency measure; (c) ablation studies show that our VEE-BERT is off-the-shelf and work well with other pre-trained models.

References

Tolga Bolukbasi, J. Wang, O. Dekel, and Venkatesh Saligrama. 2017. Adaptive neural networks for efficient inference. In *ICML*.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Shijin Wang, and Guoping Hu. 2020. Revisiting pre-trained models for chinese natural language processing. *ArXiv*, abs/2004.13922.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Federico Girosi, Michael J. Jones, and Tomaso A. Poggio. 1995. Regularization theory and neural networks architectures. *Neural Computation*, 7:219–269.

Ernst Hellinger. Neue begründung der theorie quadratischer formen von unendlichvielen veränderlichen. *Journal für die reine und angewandte Mathematik*, 1909:210 – 271.

Gao Huang, Danlu Chen, T. Li, Felix Wu, L. V. D. Maaten, and Kilian Q. Weinberger. 2017. Multi-scale dense convolutional networks for efficient prediction. *ArXiv*, abs/1703.09844.

Arzoo Katiyar and Claire Cardie. 2018. Nested named entity recognition revisited. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 861–871, New Orleans, Louisiana. Association for Computational Linguistics.

Y. Kaya, Sanghyun Hong, and T. Dumitras. 2019. Shallow-deep networks: Understanding and mitigating network overthinking. In *ICML*.

Yigitcan Kaya and Tudor Dumitras. 2018. How to stop off-the-shelf deep neural networks from overthinking. *ArXiv*, abs/1810.07052.

Jin-Dong Kim, Tomoko Ohta, Yuka Tateisi, and Junichi Tsujii. 2003a. Genia corpus - a semantically annotated corpus for bio-textmining. *Bioinformatics*, 19 Suppl 1:i180–2.

Jin-Dong Kim, Tomoko Ohta, Yuka Tateisi, and Jun'ichi Tsujii. 2003b. Genia corpus—a semantically annotated corpus for bio-textmining. *Bioinformatics (Oxford, England)*, 19 Suppl 1:i180–2.

Solomon Kullback and R. A. Leibler. 1951. On information and sufficiency. *Annals of Mathematical Statistics*, 22:79–86.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. *ArXiv*, abs/1909.11942.

Gina-Anne Levow. 2006. The third international Chinese language processing bakeoff: Word segmentation and named entity recognition. In *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*, pages 108–117, Sydney, Australia. Association for Computational Linguistics.

Xiaonan Li, Yunfan Shao, Tianxiang Sun, Hang Yan, Xipeng Qiu, and Xuanjing Huang. 2021. Accelerating bert inference for sequence labeling via early-exit. *ArXiv*, abs/2105.13878.

Weijie Liu, P. Zhou, Zhe Zhao, Zhiruo Wang, Haotang Deng, and Q. Ju. 2020. Fastbert: a self-distilling bert with adaptive inference time. *ArXiv*, abs/2004.02178.

Wei Lu and Dan Roth. 2015. Joint mention extraction and classification with mention hypergraphs. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 857–867, Lisbon, Portugal. Association for Computational Linguistics.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074, Berlin, Germany. Association for Computational Linguistics.

Christopher D. Manning and Hinrich Schütze. 2002. Foundations of statistical natural language processing. In *SGMD*.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Erik F Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *arXiv preprint cs/0306050*.

Roy Schwartz, Gabriel Stanovsky, Swabha Swayamdipta, Jesse Dodge, and Noah A. Smith. 2020. The right tool for the job: Matching model and instance complexities. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6640–6651, Online. Association for Computational Linguistics.

Tianxiang Sun, Yunhua Zhou, Xiangyang Liu, Xinyu Zhang, Hao Jiang, Zhao Cao, Xuanjing Huang, and Xipeng Qiu. 2021. Early exiting with ensemble internal classifiers. *ArXiv*, abs/2105.13792.

Surat Teerapittayanon, Bradley McDanel, and H. T. Kung. 2016. Branchynet: Fast inference via early exiting from deep neural networks. *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 2464–2469.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, L. Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *ArXiv*, abs/1706.03762.

Wei Wu, Yuxian Meng, Fei Wang, Qinghong Han, Muyu Li, Xiaoya Li, Jie Mei, Ping Nie, Xiaofei Sun, and Jiwei Li. 2019. Glyce: Glyph-vectors for chinese character representations. In *NeurIPS*.

J. Xin, Raphael Tang, J. Lee, Y. Yu, and Jimmy Lin. 2020a. Deebert: Dynamic early exiting for accelerating bert inference. *ArXiv*, abs/2004.12993.

Ji Xin, Raphael Tang, Jaejun Lee, Yaoliang Yu, and Jimmy Lin. 2020b. DeeBERT: Dynamic early exiting for accelerating BERT inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2246–2251, Online. Association for Computational Linguistics.

Z. Yang, Zihang Dai, Yiming Yang, J. Carbonell, R. Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *NeurIPS*.

Juntao Yu, Bernd Bohnet, and Massimo Poesio. 2020. Named entity recognition as dependency parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6470–6476, Online. Association for Computational Linguistics.

Xiang Yu, Agnieszka Falenska, and Jonas Kuhn. 2019. Dependency length minimization vs. word order constraints: An empirical study on 55 treebanks. In *Proceedings of the First Workshop on Quantitative Syntax (Quasy, SyntaxFest 2019)*, pages 89–97, Paris, France. Association for Computational Linguistics.

Wangchunshu Zhou, Canwen Xu, Tao Ge, Julian McAuley, Ke Xu, and Furu Wei. 2020. Bert loses patience: Fast and robust inference with early exit. *ArXiv*, abs/2006.04152.

Wei Zhu. 2021. LeeBERT: Learned early exit for BERT with cross-level optimization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2968–2980, Online. Association for Computational Linguistics.

Wei Zhu, Xiaoling Wang, Yuan Ni, and Guotong Xie. 2021. GAML-BERT: Improving BERT early exiting by gradient aligned mutual learning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3033–3044, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

A Example Appendix

A.1 Introduction to the NER tasks

ACE2004 and ACE2005 The two datasets each contain 7 entity categories. For each entity type, there are annotations for both the entity mentions and mention heads. For fair comparison, we exactly follow the data pre-processing strategy in Lu and Roth (2015) to split the data into 80%, 10%, 10% for train, development and test set respectively.

GENIA Kim et al. (2003a). For GENIA, we use the GENIA v3.0.2 corpus. We preprocess the dataset following the same settings of Katiyar and Cardie (Katiyar and Cardie, 2018).

CoNLL2003 (Sang and De Meulder, 2003) is an English dataset with four types of named entities: Location, Organization, Person and Miscellaneous. We followed data processing protocols in Ma and Hovy (2016).

OntoNotes 4.0 is a Chinese dataset and consists of text from news domain. OntoNotes 4.0 annotates 18 named entity types. In this paper, we take the same data split as Wu et al. (2019).

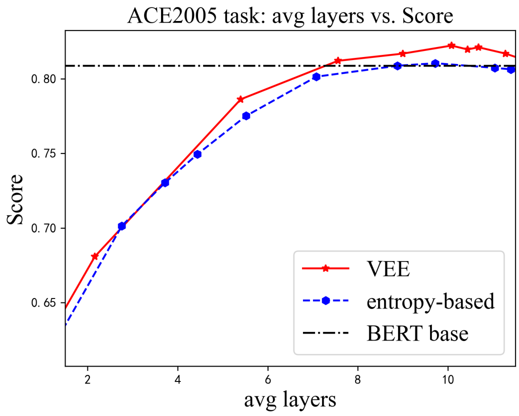
MSRA (Levow, 2006) is a Chinese dataset and performs as a benchmark dataset. Data in MSRA is collected from news domain and is used as shared task on SIGNAN backoff 2006. There are three types of named entities.

A.2 Quality–efficiency trade-offs on 3 NER tasks

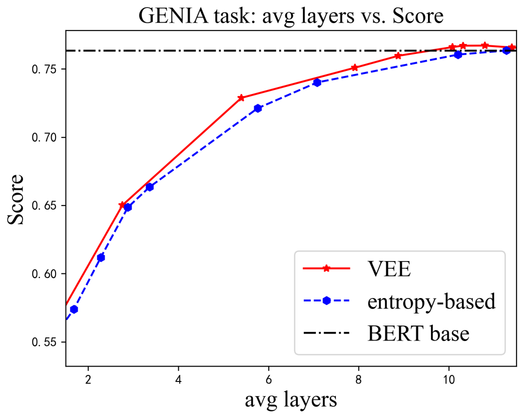
In the main content, we present the quality–efficiency trade-offs curves for 3 benchmark NER tasks. And here we put the results of ACE2005, GENIA, and OntoNotes 4.0 in the appendix, due to length limit of the main content.

A.3 Ablation study on the cross-layer ensemble

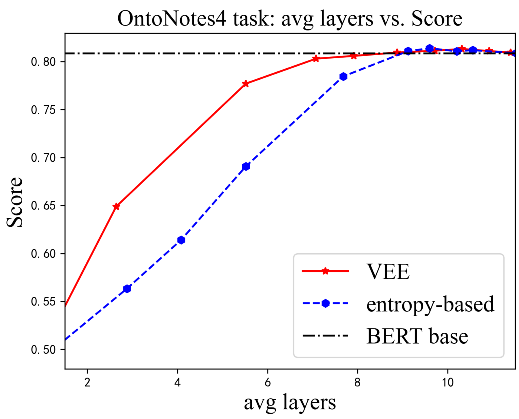
In Figure 4, we conduct the ablation studies on the ACE2004 task. Similar results (in Figure 8) can be observed on the MSRA task. The ensemble results in worse performances at the shallow layers and slight improvements at the deeper layers.



(a) ACE2005 task



(b) GENIA task



(c) OntoNotes 4.0 task

Figure 7: Quality–efficiency trade-offs using different exiting strategies. We can see that our two versions of VEEs consistently outperform the entropy-based baseline.

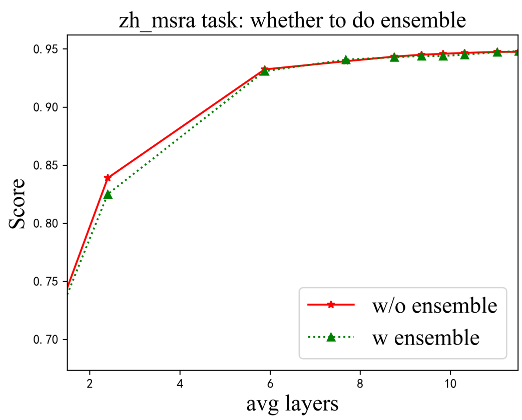


Figure 8: The performance comparison on the MSRA task between VEE-BERT without cross-layer ensemble and that with.