# Improving Prediction Certainty Estimation for Reliable Early Exiting via Null Space Projection

**Jianing He**[1], **Qi Zhang**[1], **Duoqian Miao**[1], **Yi Kun**[2], **Shufeng Hao**[3], **Hongyun Zhang**[1] and **Zhihua Wei**[1]
[1]Tongji University
[2]North China Institute of Computing Technology
[3]Taiyuan University of Technology
{jnhe, zhangqi_cs, dqmiao, zhanghongyun, zhihua_wei}@tongji.edu.cn,
kunyi.cn@gmail.com, haoshufeng@tyut.edu.cn



## Abstract

Early exiting has demonstrated great potential in accelerating the inference of pre-trained language models (PLMs) by enabling easy samples to exit at shallow layers, eliminating the need for executing deeper layers. However, existing early exiting methods primarily rely on class-relevant logits to formulate their exiting signals for estimating prediction certainty, neglecting the detrimental influence of class-irrelevant information in the features on prediction certainty. This leads to an overestimation of prediction certainty, causing premature exiting of samples with incorrect early predictions. To remedy this, we define an NSP score to estimate prediction certainty by considering the proportion of class-irrelevant information in the features. On this basis, we propose a novel early exiting method based on the Certainty-Aware Probability (CAP) score, which integrates insights from both logits and the NSP score to enhance prediction certainty estimation, thus enabling more reliable exiting decisions. The experimental results on the GLUE benchmark show that our method can achieve an average speed-up ratio of 2.19× across all tasks with negligible performance degradation, surpassing the state-of-the-art (SOTA) ConsistentEE by 28%, yielding a better trade-off between task performance and inference efficiency. The code is available at https://github.com/He-Jianing/NSP.git.

## 1 Introduction

Recently, the increasing scale of pre-trained language models (PLMs) has hindered their deployment on resource-constrained devices and latency-sensitive applications due to the high computational costs and long inference time. To address this issue, early exiting [Xin *et al.*, 2020; Zhou *et al.*, 2020; Liao *et al.*, 2021; Xin *et al.*, 2021; Balagansky and Gavrilov, 2022; Zeng *et al.*, 2024; He *et al.*, 2025a; He *et al.*, 2025b], a kind of adaptive inference strategy, has been proposed to accelerate the inference of PLMs. Specifically, each intermediate layer of PLMs is paired with an internal classifier to provide an early prediction. Once the certainty level of an early prediction reaches the threshold, the forward inference is terminated, bypassing the computation of the following deeper layers. By conducting a sample-wise inference procedure, PLMs can deal with easy samples with shallow layers and handle hard samples with deep layers. This effectively improves the inference efficiency of PLMs without sacrificing accuracy.

Typically, an early exiting method involves devising an exiting signal to estimate the certainty level of early predictions, determining whether samples exit from early layers. Researchers have developed quite a few exiting signals by measuring and seeking the characteristics of predictions with high certainty levels, including the entropy [Xin *et al.*, 2020; He *et al.*, 2024], the softmax score [Schwartz *et al.*, 2020], the patience [Zhang *et al.*, 2023; Zhu *et al.*, 2023], the energy score [Akbari *et al.*, 2022], and the patience-confidence score [Zhang *et al.*, 2022]. These methods employ heuristic exiting strategies, enabling the direct computation of exiting signals without involving any learning process and accordingly facilitating easy deployment for inference acceleration.

Notably, existing methods primarily use logits to devise their exiting signals[1]. Unfortunately, these methods exhibit a high Premature Exiting Rate, which suggests a tendency to overestimate prediction certainty based on logits. It significantly impairs task performance and hinders the model's acceleration. This is attributed to the fact that logits contain only class-relevant information in the features while ignoring the detrimental influence of class-irrelevant information on prediction certainty. Theoretically, any feature $\boldsymbol{x}$ can be orthogonally decomposed into $\boldsymbol{x} = \boldsymbol{x}^{W} + \boldsymbol{x}^{W^{\perp}}$ (per Figure 1), where $W$ is the column space of the classifier's weight matrix $\boldsymbol{W}$, and $W^{\perp}$ is the null space of $W$. We can yield that logits (i.e. $\boldsymbol{W}^{T}\boldsymbol{x}$) extract class-relevant information (i.e. the feature similarity to each class) from the component $\boldsymbol{x}^{W}$ but fail to capture class-irrelevant information from $\boldsymbol{x}^{W^{\perp}}$ due to $\boldsymbol{W}^{T}\boldsymbol{x}^{W^{\perp}} = \boldsymbol{0}$. Notably, $\boldsymbol{x}^{W^{\perp}}$ is closely related to prediction certainty since a large proportion of redundant

[1]Given that probabilities are calculated from logits, typically through operators like softmax, we collectively label methods based on logits or probabilities as logit-based methods in this paper.

(class-irrelevant) information in the feature interferes with the model's classification and reduces prediction certainty. Nevertheless, logit-based methods completely disregard the detrimental impact of class-irrelevant information on prediction certainty, inevitably resulting in an overestimation of prediction certainty. The observations raise an intriguing question: Can we enhance prediction certainty estimation by integrating both class-relevant and class-irrelevant information for more reliable exiting decisions?

In this paper, we propose a novel early exiting method based on the Certainty-Aware Probability (CAP) score, which complements the logit-based methods by considering the proportion of class-irrelevant information in the features. To this end, we first define the ratio of $\|\boldsymbol{x}^{W^\perp}\|$ and $\|\boldsymbol{x}\|$ as the NSP (null space projection) score, which provides an estimation of prediction certainty by leveraging the proportion of class-irrelevant information in the features. A higher NSP score indicates a lower level of prediction certainty. Then, we introduce the scaled NSP score as a new logit corresponding to a constructed virtual UNK (unknown) class. After appending this new logit to the original logits, the CAP score is finally defined as the softmax probability corresponding to the constructed UNK class, which can serve as a proxy for prediction uncertainty. The exiting condition is met once the CAP score falls below the threshold. From the formation of CAP, it is evident that our method integrates the class-relevant original logits (indicating the feature similarity to each class) with the class-irrelevant NSP score (indicating the proportion of class-irrelevant information in the features) to enhance prediction certainty estimation, enabling more reliable exiting decisions.

Our contributions are summarized as follows:

- We reveal that current logit-based early exiting methods neglect the detrimental influence of class-irrelevant information in the features on prediction certainty, leading to an overestimation of prediction certainty and premature exiting of samples with incorrect early predictions.
- We define a class-irrelevant NSP score to estimate prediction certainty by considering the proportion of class-irrelevant information in the features.
- We propose a novel early exiting method based on the CAP score, which integrates class-relevant logits and the class-irrelevant NSP score to enhance prediction certainty estimation for more reliable exiting decisions.

Extensive experiments on the GLUE benchmark verify that our method outperforms the SOTA ConsistentEE [Zeng *et al.*, 2024] by $28\%$ in model acceleration, with negligible extra computational or storage overhead. Further analysis confirms the generality of our method across various backbones and demonstrates its effectiveness in enhancing prediction certainty estimation and delivering reliable exiting decisions.

## 2 Related Works

**Heuristic Exiting Strategy.** DeeBERT [Xin *et al.*, 2020], FastBERT [Liu *et al.*, 2020], Right-Tool [Schwartz *et al.*, 2020], and E-LANG [Akbari *et al.*, 2022] employ entropy, softmax score, and energy score to indicate prediction certainty/uncertainty, respectively. The exiting criterion is met when the certainty level exceeds a threshold. PABEE [Zhou *et al.*, 2020] relies on cross-layer consistency (i.e. patience) to determine exiting, allowing samples to exit once a sufficient number of consecutive classifiers provide the same answer. F-PABEE [Gao *et al.*, 2023], LECO [Zhang *et al.*, 2023], and BADGE [Zhu *et al.*, 2023] enhance the flexibility of PABEE by introducing softer cross-layer comparison strategies. PCEE-BERT [Zhang *et al.*, 2022] proposes a hybrid exiting strategy based on entropy and patience to guarantee reliable and flexible early exiting. These logit-based methods neglect the detrimental influence of class-irrelevant information on prediction certainty, compromising the reliability of exiting decisions. To remedy this, we propose to consider the proportion of class-irrelevant information in the features for more reliable exiting decisions.

**Learning-based Exiting Strategy.** BERxiT [Xin *et al.*, 2021], PALBERT [Balagansky and Gavrilov, 2022], and ConsistentEE [Zeng *et al.*, 2024] train neural networks to generate exiting signals. HASHEE [Sun *et al.*, 2022] and BE3R [Mangrulkar *et al.*, 2022] train networks to route each sample (or token) to an appropriate exiting layer, requiring no layer-by-layer exiting judgments. These methods incorporate the learning process to formulate their exiting strategies, introducing additional parameters and training costs. They also depend on intricate parameter tuning, and carefully designed network architectures or training objectives. In contrast, our method employs a heuristic exiting strategy where exiting signals can be computed directly, facilitating easy deployment.

**Architecture/Loss of Multi-Exit Networks.** CascadeBERT [Li *et al.*, 2021] performs early exiting in multiple cascaded complete models to provide comprehensive representations for accurate predictions. It also introduces a difficulty-aware objective to calibrate the model's predictions. GPFEE [Liao *et al.*, 2021] facilitates early predictions by integrating both past and future states. It incorporates imitation loss to train the model to approximate future states based on all past states. LeeBERT [Zhu, 2021] and GAML-BERT [Zhu *et al.*, 2021] introduce cross-layer distillation objectives to encourage mutual learning among classifiers. LECO [Zhang *et al.*, 2023] enhances the network architecture based on neural architecture search. BADGE [Zhu *et al.*, 2023] introduces block-wise bypasses to alleviate optimization conflicts among multiple classifiers. DisentangledEE [Ji *et al.*, 2023] introduces adapters to decouple generic language representations from task-specific representations. It also proposes a non-parametric classifier for improvements. Different from these methods that enhance training objectives or network architectures for early exiting models, our method focuses on improving exiting strategies. Combining our method with these orthogonal works would be an intriguing direction for further research.

## 3 Background and Motivation

### 3.1 Problem Definition

Given a BERT-style PLM with $M$ layers, we denote the hidden states at the $m$-th layer as $h^{(m)}$. To enable early exiting

on a classification task with $C$ classes during the inference stage, each intermediate layer is coupled with an internal classifier $f_m$ where $m \in \{1, 2, \cdots, M-1\}$ to provide an early prediction $p^{(m)} = f_m(h^{(m)})$, i.e., the probability distribution over the $C$ classes. Classifiers in different layers do not share parameters. The exiting criterion is met when the estimated prediction certainty (or uncertainty) of an internal classifier exceeds (or falls below) the threshold $\tau$.

### 3.2 Are Exiting Decisions Reliable?

We utilize two types of error rates to evaluate the reliability of exiting decisions from different aspects:

- *Premature Exiting Rate*, which is the frequency of making "exit" decisions when the internal classifier provides an incorrect early prediction.
- *Delayed Exiting Rate*, which is the frequency of making "continue" decisions when the internal classifier provides the correct early prediction.

The exiting decision-making can be formalized as a binary classification task, where correct early predictions are positive samples and incorrect ones are negative. The Premature Exiting Rate thus corresponds to the false positive rate, indicating the model's failure to identify negative samples (i.e., incorrect early predictions) due to overestimating prediction certainty. A higher rate suggests that the model is prone to prematurely emit samples with incorrect early predictions, ultimately impairing task performance. In contrast, the Delayed Exiting Rate corresponds to the false negative rate, indicating the model's failure to identify positive samples (i.e., correct early predictions) due to underestimating prediction certainty. As its value escalates, the model tends to delay the exiting of samples with correct early predictions, resulting in prolonged inference time. Therefore, exiting decisions are considered reliable only when both the Premature and Delayed Exiting Rates are sufficiently low.

From Figure 3, we observe that the current logit-based early exiting methods exhibit a much higher Premature Exiting Rate compared to the Delayed Exiting Rate. This indicates that these methods primarily suffer from an overestimation of prediction certainty, leading to premature exiting of samples with incorrect early predictions. We believe this is because logits represent the feature similarity to each class, namely they are class-relevant. However, there is class-irrelevant information related to prediction certainty in the feature space that is not contained in logits. This affects the estimation of prediction certainty, leading to sub-optimal exiting decisions. These observations trigger our further exploration on the missing information from features to logits.

### 3.3 The Missing Information in Logits

Given a classification task with $C$-classes, each internal classifier consists of a fully connected layer with weight $\boldsymbol{W} \in \mathbb{R}^{N \times C}$ and bias $\boldsymbol{b} \in \mathbb{R}^C$, which transforms the feature $\boldsymbol{x} \in \mathbb{R}^N$ to the logits $\boldsymbol{l} \in \mathbb{R}^C$, i.e. $\boldsymbol{l} = \boldsymbol{W}^T\boldsymbol{x} + \boldsymbol{b}$, thus providing an early prediction $\boldsymbol{p} = \text{softmax}(\boldsymbol{l})$. To omit the bias term from the calculation of logits, we offset the original feature space by a vector $\boldsymbol{o} = (\boldsymbol{W}^T)^+\boldsymbol{b}$, where $(\cdot)^+$ denotes the Moore-Penrose inverse of the matrix:

$$\boldsymbol{l} = \boldsymbol{W}^T\boldsymbol{x'} = \boldsymbol{W}^T(\boldsymbol{x} + \boldsymbol{o}). \tag{1}$$

For convenience, in the rest of this paper, the feature $\boldsymbol{x}$ refers to the offset result. Let $W$ denote the column space of $\boldsymbol{W}$ and $W^\perp$ denote the null space of $W$. The feature $\boldsymbol{x}$ can be orthogonally decomposed into $\boldsymbol{x} = \boldsymbol{x}^W + \boldsymbol{x}^{W^\perp}$, where $\boldsymbol{x}^W$ and $\boldsymbol{x}^{W^\perp}$ are the projections of $\boldsymbol{x}$ to $W$ and $W^\perp$, respectively. Note that $\boldsymbol{x}^{W^\perp}$ satisfies $\boldsymbol{W}^T\boldsymbol{x}^{W^\perp} = \boldsymbol{0}$, which is class-irrelevant. Therefore, Eq.(1) can be rewritten as:

$$\boldsymbol{l} = \boldsymbol{W}^T\boldsymbol{x} = \boldsymbol{W}^T\boldsymbol{x}^W, \tag{2}$$

where each logit $l_i$ is the inner product of $\boldsymbol{x}^W$ and the class vector $\boldsymbol{w_i}$ (the $i$-th column of $\boldsymbol{W}$), indicating the feature similarity to the $i$-th class. Obviously, logits $\boldsymbol{l}$ extracts class-relevant information from the component $\boldsymbol{x}^W$ using $\boldsymbol{W}$ but neglect the component $\boldsymbol{x}^{W^\perp}$ that carries class-irrelevant information. Indeed, $\boldsymbol{x}^{W^\perp}$ is closely related to prediction certainty. Precisely, given a feature $\boldsymbol{x}$, a larger component $\boldsymbol{x}^{W^\perp}$ indicates more redundant (class-irrelevant) information in the feature, which can interfere with the model's classification and lead to reduced prediction certainty. Based on the above analysis, the detrimental influence of class-irrelevant information on prediction certainty is completely ignored by logit-based methods, which leads to an overestimation of prediction certainty, causing premature exiting of samples.

To remedy this, we define the ratio of the norms of $\boldsymbol{x}^{W^\perp}$ and $\boldsymbol{x}$ as the NSP (null space projection) score:

$$\text{NSP}(\boldsymbol{x}) = \frac{\|\boldsymbol{x}^{W^\perp}\|}{\|\boldsymbol{x}\|}. \tag{3}$$

The NSP score can provide an estimation of prediction certainty by considering the proportion of class-irrelevant information in the features. Geometrically, it equals to the cosine similarity between the feature and its projection to the null space. The NSP score lies between 0 and 1, and a higher value indicates a lower certainty level.

Logits reflect the feature similarity to each class, while the NSP score indicates the proportion of class-irrelevant information in the features, both of which are closely related to prediction certainty. We hypothesize that combining class-relevant logits and the class-irrelevant NSP score could enhance prediction certainty estimation, thus enabling more reliable exiting decisions. Such a solution is proposed in Section 4 by introducing the scaled NSP score as a new logit.

## 4 Methods

We propose a novel early exiting method based on the Certainty-Aware Probability (CAP) score, which integrates the class-irrelevant NSP score with class-relevant logits for reliable exiting decisions. Figure 1 provides an overview of our method. During inference, we first compute the NSP score in Eq.(3), which is the ratio of the norms of the feature $\boldsymbol{x}$ and its projection $\boldsymbol{x}^{W^\perp}$ onto the null space $W^\perp$. Then, we introduce the scaled NSP score as a new logit corresponding to a constructed virtual UNK class. Finally, the exiting signal of our method, i.e. the CAP score, is defined as the softmax probability corresponding to the new logit.

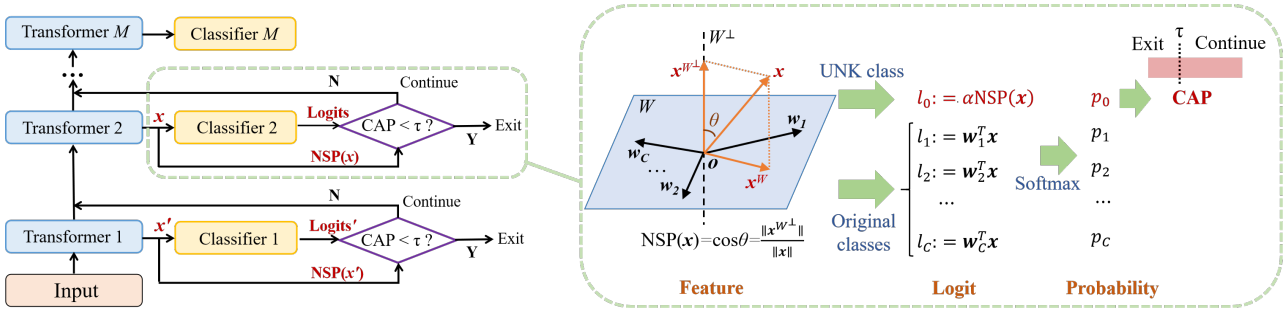



Figure 1: Method overview. Our method integrates the class-irrelevant NSP score with class-relevant logits to generate high-quality exiting signals (i.e., the CAP score). The right box details the CAP-based exiting strategy at the second layer. The subspace $W$ is spanned by the class vectors $\boldsymbol{w}_1 \sim \boldsymbol{w}_C$, i.e., the column vectors of the second-layer classifier's weight matrix. $W^\perp$ denotes the null space of $W$. Given an offset sample feature $\boldsymbol{x}$, $\boldsymbol{x}^W$ and $\boldsymbol{x}^{W^\perp}$ denote its projections onto $W$ and $W^\perp$, respectively. $\theta$ represents the angle between $\boldsymbol{x}$ and $\boldsymbol{x}^{W^\perp}$. A scaled NSP score forms a new logit $l_0$ for the virtual UNK class. $\alpha$ aligns the scale of $l_0$ with the original logits $l_1 \sim l_C$ of $C$ original classes. After softmax, the UNK probability $p_0$, i.e., the CAP score, serves as the exiting signal. Exiting occurs if CAP falls below the threshold $\tau$.

### 4.1 Null Space Projection

This subsection provides the computation of the NSP score. We offset the feature space by $\boldsymbol{o} = (\boldsymbol{W}^T)^+\boldsymbol{b}$ to omit the bias term in the computation of logits, as shown in Eq.(1). Additionally, for computational efficiency, we compute the component $\boldsymbol{x}^W$, thus obtaining $\|\boldsymbol{x}^{W^\perp}\| = \sqrt{\|\boldsymbol{x}\|^2 - \|\boldsymbol{x}^W\|^2}$.

As $\boldsymbol{x}^W$ is the projection of $\boldsymbol{x}$ onto the column space of $\boldsymbol{W}$, it can be expressed as $\boldsymbol{x}^W = \boldsymbol{W}\boldsymbol{\mu}$, where $\boldsymbol{\mu} \in \mathbb{R}^C$ denotes the combination coefficients of $\boldsymbol{w}_1 \sim \boldsymbol{w}_C$. Considering that $\boldsymbol{x}^{W^\perp}$ satisfies $\boldsymbol{W}^T\boldsymbol{x}^{W^\perp} = \boldsymbol{0}$, we have:

$$\boldsymbol{W}^T(\boldsymbol{x} - \boldsymbol{x}^W) = \boldsymbol{0}. \tag{4}$$

Thus, we construct a system of linear equations regarding the combination coefficients $\boldsymbol{\mu}$:

$$\boldsymbol{W}^T(\boldsymbol{x} - \boldsymbol{W}\boldsymbol{\mu}) = \boldsymbol{0}. \tag{5}$$

Solving Eq.(5), we get $\boldsymbol{\mu} = (\boldsymbol{W}^T\boldsymbol{W})^{-1}\boldsymbol{W}^T\boldsymbol{x}$, and consequently obtain:

$$\boldsymbol{x}^W = \boldsymbol{W}(\boldsymbol{W}^T\boldsymbol{W})^{-1}\boldsymbol{W}^T\boldsymbol{x}. \tag{6}$$

According to Eq.(3), the NSP score can be computed as:

$$\mathrm{NSP}(\boldsymbol{x}) = \frac{\sqrt{\|\boldsymbol{x}\|^2 - \|\boldsymbol{x}^W\|^2}}{\|\boldsymbol{x}\|}. \tag{7}$$

### 4.2 The CAP Score

In this subsection, we introduce the Certainty-Aware Probability (CAP) score, an exiting signal that combines the class-irrelevant NSP score and class-relevant logits. To this end, we introduce the scaled NSP score as a new logit corresponding to a constructed virtual UNK (unknown) class:

$$l_0 = \alpha \cdot \mathrm{NSP}(\boldsymbol{x}), \tag{8}$$

where $\alpha$ is the scaling parameter. Note that the NSP score cannot be directly used as a new logit since the subsequent softmax is highly sensitive to the scale of logits. Hence, we introduce the hyper-parameter $\alpha$ to align the scale of the new logit with each original logit, preventing the final CAP score from being dominated by either side due to scale mismatches.

We append $l_0$ to the original logits $l_1 \sim l_C$ and derive an extended probability distribution through softmax:

$$p_i = \frac{e^{l_i}}{\sum_{i=0}^{C} e^{l_i}}, \tag{9}$$

where $i \in \{0, 1, \cdots, C\}$. Specifically, $l_1 \sim l_C$ suggest the feature similarity to each original class, thus the corresponding $p_1 \sim p_C$ indicate the probability of the sample belonging to each original class, respectively. In contrast, the new logit $l_0$ reflects the proportion of class-irrelevant information in the feature. A larger $l_0$ value indicates a higher proportion of class-irrelevant information in the feature, making it more challenging for the model to identify which class the sample belongs to. Hence, the corresponding $p_0$ denotes the probability of the sample belonging to the constructed UNK class, which indicates the model's inability to provide an exact answer and can serve as a proxy for prediction uncertainty.

Based on the above analysis, the CAP score is defined as:

$$\mathrm{CAP}(\boldsymbol{x}) = p_0 = \frac{e^{\alpha \cdot \mathrm{NSP}(\boldsymbol{x})}}{\sum_{i=1}^{C} e^{l_i} + e^{\alpha \cdot \mathrm{NSP}(\boldsymbol{x})}}. \tag{10}$$

The CAP score lies in $(0, 1)$, which is affected by both class-relevant original logits and the class-irrelevant NSP score. We notice that samples with lower NSP scores (less class-irrelevant information) and larger original logits (higher similarity to each class) typically have lower CAP scores, showing higher prediction certainty. The exiting criterion is met when the CAP score falls below a predefined threshold. Using an information fusion strategy, our method improves prediction certainty estimation to deliver more reliable exiting decisions. The computational overhead from the CAP score is negligible compared to the encoder layer (see Section 6).

### 4.3 More Insights into CAP

**Rationale for Logit Concatenation.** Intrinsically, both the new and original logits capture feature similarity. Specifically, we treat all original classes as known classes, while the constructed virtual UNK class represents all unknown classes. Since the component $\boldsymbol{x}^{W^\perp}$ is orthogonal to all known

| Dataset | Classes | \|Train\| | \|Test\| | Task |
|---|---|---|---|---|
| SST-2 | 2 | 67k | 1.8k | Sentiment |
| MRPC | 2 | 3.7k | 1.7k | Paraphrase |
| QQP | 2 | 364k | 391k | Paraphrase |
| MNLI | 3 | 393k | 20k | NLI |
| QNLI | 2 | 105k | 5.4k | QA/NLI |
| RTE | 2 | 2.5k | 3k | NLI |

Table 1: Dataset Statistics. NLI is the Natural Language Inference task, and QA is the Question Answering task.

class vectors, it can be naturally interpreted as the class vector for the UNK class. Accordingly, the NSP score captures the cosine similarity between the feature and the UNK class vector (per Figure 1), while the original logits measure the feature's similarity to each known class vector. This semantic alignment justifies concatenating the new and original logits, ensuring that the extended logits and probability distribution carry clear and coherent physical meaning.

**Effectiveness of CAP.** Current logit-based methods only consider the feature's similarity to known class vectors, disregarding the possibility of the sample belonging to an unknown class. This leads to an overly optimistic estimation of prediction certainty. In contrast, our CAP score considers the feature's similarity to both known and unknown classes, enabling a more objective estimation of prediction certainty.

# 5 Experiments

## 5.1 Datasets

We evaluate our method on six classification tasks of the GLUE benchmark [Wang *et al.*, 2019], including SST-2, MRPC, QNLI, RTE, QQP, and MNLI. Data statistics are shown in Table 1.

## 5.2 Backbones and Baselines

We choose the widely used BERT-base [Devlin *et al.*, 2019] as the backbone model for convincing comparison. We compare our method with two groups of representative and competing baselines. The first group encompasses all existing heuristic exiting strategies, including DeeBERT [Xin *et al.*, 2020], Right-Tool [Schwartz *et al.*, 2020], PABEE [Zhou *et al.*, 2020], PCEE-BERT [Zhang *et al.*, 2022], E-LANG [Akbari *et al.*, 2022], and F-PABEE [Gao *et al.*, 2023]. The second group encompasses competitive early exiting methods relevant to our study, including BERxiT [Xin *et al.*, 2021], PALBERT [Balagansky and Gavrilov, 2022], DisentangledEE [Ji *et al.*, 2023], and ConsistentEE [Zeng *et al.*, 2024]. For fair comparisons, HASHEE [Sun *et al.*, 2022] is not included since it employs token-level early exiting, whereas our method focuses on sentence-level early exiting.

For the first group of baselines, DeeBERT, E-LANG, and Right-Tool employ entropy, energy scores, and softmax scores as exiting signals, respectively. PABEE uses cross-layer consistency (i.e. the patience score) to determine exiting. F-PABEE combines PABEE with softer cross-layer consistency measures to ensure flexible early exiting. PCEE-BERT adopts a hybrid exiting strategy based on entropy and patience. For the second group of baselines, BERxiT introduces learning-to-exit networks to generate exiting signals that score the certainty level of early predictions. PALBERT adopts a deterministic Q-exit criterion that evaluates the cumulative distribution function of the exiting layer's probability distribution from neural networks for exit decision-making. DisentangledEE incorporates adapters to decouple generic language representations from task-specific language representations and further proposes a non-parametric classifier for improvements. It employs the patience-based exiting strategy in PABEE during inference. ConsistentEE introduces policy networks to predict the probability distribution of exiting decisions.

## 5.3 Experimental Settings

**Speed Measurement.** Following previous studies [Zhang *et al.*, 2022; Zeng *et al.*, 2024], we evaluate the model acceleration with saved layers:

$$\text{Speed-up Ratio} = \frac{\sum_{m=1}^{M} M \times N^m}{\sum_{m=1}^{M} m \times N^m}, \tag{11}$$

where $M$ is the total number of layers and $N^m$ is the number of samples exiting from the $m$th layer.

**Training.** Our implementation is based on Hugging Face's Transformers [Wolf *et al.*, 2020]. All internal classifiers are jointly trained with the backbone by minimizing the sum of their cross-entropy losses. Following previous studies [Zhou *et al.*, 2020; Zhang *et al.*, 2022], we perform a grid search over learning rates of {1e-5, 2e-5, 3e-5, 5e-5}, and batch sizes of $\{16, 32, 128\}$. The maximum sequence length is fixed at 128. We employ a linear decay learning rate scheduler and the AdamW optimizer [Loshchilov and Hutter, 2019]. We conduct experiments on two RTX4090 GPUs with 24GB.

**Inference.** Following previous studies [Zhang *et al.*, 2022; Gao *et al.*, 2023], the batch size for inference is set to 1, which mimics a common industry scenario where requests from different users arrive sequentially. We select $\alpha$ in Eq.(8) from $\{0.01, 0.1, 1.0, 10.0\}$ for each task.

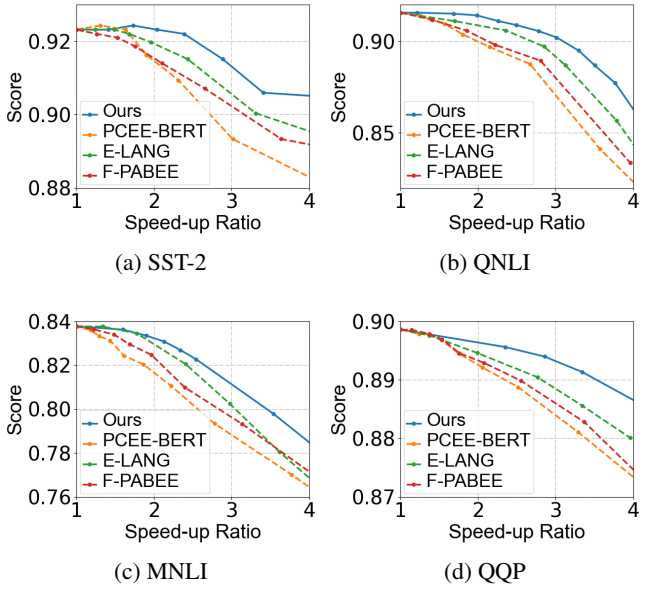
## 5.4 Overall Performance Comparison

In Table 2, we report the test results of each early exiting method on the GLUE benchmark with BERT-base as the backbone model. The speed-up ratio is approximately $2.00\times$. Overall, our method outperforms all baseline methods in nearly all tasks, which demonstrates the effectiveness of our design. Notably, compared to the backbone, our method achieves an average speed-up ratio of $2.19\times$ across all tasks with negligible performance degradation, surpassing the SOTA baseline ConsistentEE by $28\%$.

Figure 2 shows the performance-efficiency curves of our method and three competitive logit-based baselines on a representative subset of GLUE using the same fine-tuned multi-exit network. Our method exhibits a superior trade-off between task performance and inference efficiency across different tasks compared to the baseline methods. This demonstrates that our method effectively completes the current logit-based methods by considering the proportion of class-irrelevant information in the features suggested by NSP scores. Further analysis confirms our method's advantage in

| Method | RTE | MRPC | QQP | SST-2 | QNLI | MNLI | AVG |
|---|---|---|---|---|---|---|---|
| BERT-base | 66.4 (1.00×) | 88.9 (1.00×) | 71.2 (1.00×) | 93.5 (1.00×) | 90.5 (1.00×) | 84.6 (1.00×) | 82.5 (1.00×) |
| DeeBERT† | 64.3 (1.95×) | 84.4 (2.07×) | 70.4 (2.13×) | 90.2 (2.00×) | 85.6 (2.09×) | 74.4 (1.87×) | 78.2 (2.02×) |
| Right-Tool‡ | 64.6 (1.92×) | 84.2 (2.04×) | 70.5 (2.04×) | 89.3 (1.92×) | 86.2 (1.96×) | 77.6 (2.04×) | 78.7 (1.99×) |
| PABEE† | 64.0 (1.81×) | 84.4 (2.01×) | 70.4 (2.09×) | 89.3 (1.95×) | 88.0 (1.87×) | 79.8 (2.07×) | 79.3 (1.97×) |
| BERxiT | 65.7 (2.17×) | 86.2 (2.27×) | 70.5 (2.27×) | 91.6 (2.86×) | 89.6 (1.72×) | 82.1 (2.33×) | 81.0 (2.27×) |
| PCEE-BERT‡ | 67.1 (1.89×) | 86.4 (2.13×) | 70.9 (1.96×) | 92.3 (1.92×) | 88.8 (2.17×) | 82.2 (1.80×) | 81.3 (1.98×) |
| E-LANG‡ | 67.2 (1.96×) | 87.0 (1.98×) | 71.0 (1.89×) | 92.2 (2.05×) | 89.6 (1.89×) | 83.0 (1.96×) | 81.7 (1.96×) |
| F-PABEE‡ | 67.3 (1.85×) | 87.5 (2.16×) | 70.7 (1.92×) | 92.3 (1.96×) | 89.2 (2.14×) | 82.2 (2.08×) | 81.5 (2.02×) |
| PALBERT* | 64.3 (1.48×) | - | - | 91.8 (1.48×) | 89.1 (1.48×) | 83.0 (1.48×) | - |
| DisentangledEE | 66.8 (1.25×) | - | - | 92.9 (1.25×) | 88.5 (1.25×) | 83.0 (1.25×) | - |
| ConsistentEE | **69.0 (1.85×)** | **89.0 (1.59×)** | 70.4 (1.82×)‡ | 92.9 (1.85×) | 89.9 (1.72×) | 83.4 (1.45×) | 82.4 (1.71×) |
| **Ours** | 68.6 (1.93×) | 88.0 (2.68×) | **71.2 (2.07×)** | **93.0 (2.15×)** | **90.2 (2.40×)** | **83.4 (1.92×)** | **82.4 (2.19×)** |

Table 2: Performance comparison on the GLUE test set with BERT-base as the backbone. † and ∗ denote the results taken from GPFEE [Liao *et al.*, 2021] and DisentangledEE [Ji *et al.*, 2023], respectively. ‡ denotes the results based on our implementation. Other baseline results are from their original papers. We report F1-score for MRPC and QQP, and accuracy for other tasks. The - denotes unavailable results. Best performance values are marked in bold.



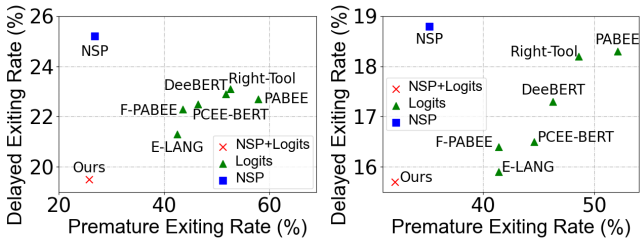

(a) SST-2 (b) QNLI

(c) MNLI (d) QQP

Figure 2: Performance-efficiency trade-off curves of different early exiting methods on four GLUE development sets.

| Method | SST-2 | | | QNLI | | |
|---|---|---|---|---|---|---|
| | $L=2$ | $L=6$ | $L=10$ | $L=2$ | $L=6$ | $L=10$ |
| PCEE-BERT | 66.3 | 73.2 | 75.2 | 54.5 | 65.9 | 70.7 |
| F-PABEE | 71.2 | 78.3 | 81.0 | 55.6 | 68.3 | 71.6 |
| E-LANG | 72.5 | 78.8 | **82.8** | 57.3 | 75.1 | 76.0 |
| **CAP (ours)** | **76.9** | **83.0** | 81.1 | **61.9** | **80.9** | **79.3** |

Table 3: DIS analysis for each exiting signal at different layers.




(a) SST-2 (b) QNLI

Figure 3: Two types of error rates for exiting decisions using different early exiting methods under a $4.00\times$ speed-up ratio.

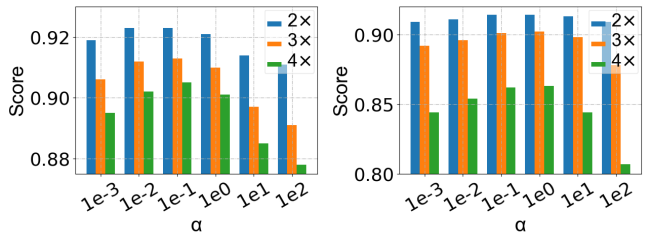
estimating prediction certainty and making reliable exiting decisions (see Section 6), offering an intrinsic explanation for its performance gains.

## 6 In-depth Analysis

**DIS Analysis for Exiting Signals.** Following previous work [Li *et al.*, 2021], we use the Difficulty Inversion Score (DIS) to evaluate the quality of prediction certainty estimation. DIS measures the consistency between the exiting signal and the sample difficulty. A higher value indicates a stronger capability of exiting signals in estimating prediction certainty. Table 3 shows the DIS of various exiting signals on the SST-2 and QNLI development sets. We observe that by incorporating the class-irrelevant NSP score, the CAP score consistently outperforms the baselines across different layers, demonstrating a more accurate estimation of prediction certainty. This is crucial for making reliable exiting decisions.

**Statistics of Exiting Decisions.** Figure 3 illustrates the two error rates for exiting decisions using different early exiting methods on the SST-2 and QNLI development sets. Compared to the current logit-based methods, utilizing the NSP score as the exiting signal can effectively reduce the Premature Exiting Rate while exhibiting a relatively high Delayed Exiting Rate. We attribute this to the NSP score's neglect of feature similarity to each class, resulting in an underestimation of prediction certainty and delayed exiting for samples with correct early predictions. In contrast, by fusing information from both class-relevant logits and the class-irrelevant NSP score, our method significantly reduces both types of error rates, demonstrating more reliable exiting decisions. This explains the superiority of our method in model acceleration.

**Impact of $\alpha$.** Figure 4 shows the impact of $\alpha$ in Eq.(8) on model acceleration. $\alpha$ is used to align the scale of the new and original logits, and a higher value indicates a stronger




(a) SST-2 (b) QNLI

Figure 4: Impact of $\alpha$ on the task performance under different speed-up ratios for SST-2 and QNLI tasks.

emphasis on the class-irrelevant NSP score compared to the class-relevant original logits. Overall, we observe that both excessively large and small values of $\alpha$ can impair the model acceleration under various speed-up ratios. This suggests an optimal trade-off between class-irrelevant and class-relevant information, which enables reliable exiting decisions. Additionally, the incorporation of the NSP score yields more significant performance improvements under high acceleration scenarios. We attribute this to the limited classification capacity of shallow classifiers. This aggravates the overestimation of prediction certainty when relying solely on the class-relevant original logits, exacerbating premature exiting of samples with erroneous early predictions. Hence, incorporating the class-irrelevant NSP score becomes increasingly crucial for reliable exiting decisions. Finally, for parameter selection, values of $\alpha$ between $0.1$ and $1.0$ typically deliver satisfactory results across different tasks and speed-up ratios.

**Computational and Storage Costs.** Table 4 presents the computational complexity for each module in our model. It is noteworthy that, at each layer, the offset vector $\boldsymbol{o}$ in Eq.(1) and the projection matrix in Eq.(6) are shared across samples during inference, requiring no repetitive calculations. Consequently, for each sample, the computational overhead introduced by making exiting decisions primarily arises from incorporating internal classifiers and calculating CAP, totaling less than 1.21M FLOPs per layer. This overhead is negligible compared to the 1813.5M of an encoder block, leading to a minimal impact on inference time. The inference time analysis presented in Table 5 further reinforces this conclusion. Additionally, the results in Table 6 indicate that our model only requires less than $0.03\%$ additional parameters compared to the backbone due to incorporating internal classifiers. The analysis above confirms our method's efficiency regarding computation and storage, suggesting strong scalability to larger datasets and backbones. Furthermore, as the model's computational complexity is primarily dominated by encoder blocks, its inference costs are approximately proportional to the number of executed layers, validating the rationale for the speed measurement in Eq.(11).

**Generality on Different PLMs.** To verify the generality of our information fusion strategy, we apply our method to ALBERT [Lan *et al.*, 2020], which is an optimized version of BERT with reduced parameters and improved efficiency. The results at around $2.00\times$ speed-up ratio are listed in Table 7. Our method still outperforms all competitive baseline methods in general, validating its generality on various PLMs.

| Module | FLOPs | |
|---|---|---|
| | $C=2$ | $C=3$ |
| Embedding | 786.4K | 786.4K |
| Encoder | 1813.5M | 1813.5M |
| Pooler | 1.2M | 1.2M |
| Classifier | 3.1K | 4.6K |
| Offset Vector* | 4.4K | 8.4K |
| Projection Matrix* | 1.8M | 3.0M |
| CAP Calculation* | 1.2M | 1.2M |

Table 4: Analysis of computational complexity. $C$ denotes the number of classes. $*$ signifies the modules introduced by our method.

| Model | Total Inference Time | Overhead vs. BERT-base |
|---|---|---|
| BERT-base | 8.36s | +0.0% |
| DeeBERT | 8.73s | +4.4% |
| Right-Tool | 8.60s | +2.9% |
| PABEE | 8.68s | +3.8% |
| E-LANG | 8.81s | +5.4% |
| Ours | 8.90s | +6.4% |

Table 5: Comparison of total inference time in the SST-2 development set. Each model disables early exiting by setting an unreachable threshold to guarantee full-layer execution. Our method incurs acceptable overhead compared to standard early exiting baselines.

| Model | #Params | |
|---|---|---|
| | $C=2$ | $C=3$ |
| BERT-base | 109.48M | 109.48M |
| Ours | +16.92K | +25.38K |

Table 6: Comparison of parameter volumes.

| Method | Speed-up | QQP | SST-2 | QNLI | MNLI | AVG |
|---|---|---|---|---|---|---|
| ALBERT-base† | 1.00× | 79.6 | 93.3 | 92.0 | 85.2 | 87.5 |
| PABEE† | 1.95× | **79.8** | 92.4 | 90.9 | 84.2 | 86.8 |
| PALBERT | 1.21× | 79.1 | 91.4 | 90.9 | 83.2 | 86.2 |
| DisentangledEE | 1.26× | 79.3 | 92.2 | 91.0 | 83.5 | 86.5 |
| **Ours** | 2.11× | 79.4 | **92.8** | **91.5** | **84.8** | **87.1** |

Table 7: Test results on a representative subset of GLUE with ALBERT-base as the backbone. The speed-up ratio is averaged across 4 tasks. We report the mean of accuracy and F1-score for QQP, and accuracy for other tasks. † denotes results taken from GPFEE [Liao *et al.*, 2021]. Other baseline results are taken from DisentangledEE [Ji *et al.*, 2023].

## 7 Conclusion

In this paper, we propose a novel early exiting method based on the CAP score, which integrates insights from both class-relevant logits and the class-irrelevant NSP score to address the overestimation of prediction certainty in current logit-based early exiting methods, enabling more reliable exiting decisions. Our method is simple yet effective. Extensive experiments on the GLUE benchmark validate its superiority across different backbones. Further analysis confirms the interpretability of our method and its efficiency in terms of storage and computation.

## Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 62376198, No. 62406225), the National Key Research and Development Program of China (No. 2022YFB3104700), and the Shanghai Baiyulan Pujiang Project (No. 08002360429).

## Contribution Statement

Duoqian Miao serves as the corresponding author and is responsible for all communications related to this manuscript.

## References

[Akbari *et al.*, 2022] Mohammad Akbari, Amin Banitalebi-Dehkordi, and Yong Zhang. E-lang: Energy-based joint inferencing of super and swift language models. *arXiv preprint arXiv:2203.00748*, 2022.

[Balagansky and Gavrilov, 2022] Nikita Balagansky and Daniil Gavrilov. Palbert: Teaching albert to ponder. *Advances in Neural Information Processing Systems*, 35:14002–14012, 2022.

[Devlin *et al.*, 2019] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*, pages 4171–4186. Association for Computational Linguistics, 2019.

[Gao *et al.*, 2023] Xiangxiang Gao, Wei Zhu, Jiasheng Gao, and Congrui Yin. F-pabee: flexible-patience-based early exiting for single-label and multi-label text classification tasks. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.

[He *et al.*, 2024] Jianing He, Qi Zhang, Weiping Ding, Duoqian Miao, Jun Zhao, Liang Hu, and Longbing Cao. De$^3$-bert: Distance-enhanced early exiting for BERT based on prototypical networks. *CoRR*, abs/2402.05948, 2024.

[He *et al.*, 2025a] Jianing He, Qi Zhang, Hongyun Zhang, Xuanjing Huang, Usman Naseem, and Duoqian Miao. COSEE: consistency-oriented signal-based early exiting via calibrated sample weighting mechanism. In *AAAI-25*, pages 24023–24031. AAAI Press, 2025.

[He *et al.*, 2025b] Jianing He, Qi Zhang, Hongyun Zhang, and Duoqian Miao. Two-stage early exiting from globality towards reliability. *CAAI Transactions on Intelligence Technology*, 2025.

[Ji *et al.*, 2023] Yixin Ji, Jikai Wang, Juntao Li, Qiang Chen, Wenliang Chen, and Min Zhang. Early exit with disentangled representation and equiangular tight frame. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 14128–14142, 2023.

[Lan *et al.*, 2020] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: A lite BERT for self-supervised learning of language representations. In *ICLR*. OpenReview.net, 2020.

[Li *et al.*, 2021] Lei Li, Yankai Lin, Deli Chen, Shuhuai Ren, Peng Li, Jie Zhou, and Xu Sun. Cascadebert: Accelerating inference of pre-trained language models via calibrated complete models cascade. In *EMNLP (Findings)*, pages 475–486. Association for Computational Linguistics, 2021.

[Liao *et al.*, 2021] Kaiyuan Liao, Yi Zhang, Xuancheng Ren, Qi Su, Xu Sun, and Bin He. A global past-future early exit method for accelerating inference of pre-trained language models. In *NAACL-HLT*, pages 2013–2023. Association for Computational Linguistics, 2021.

[Liu *et al.*, 2020] Weijie Liu, Peng Zhou, Zhiruo Wang, Zhe Zhao, Haotang Deng, and Qi Ju. Fastbert: a self-distilling BERT with adaptive inference time. In *ACL*, pages 6035–6044. Association for Computational Linguistics, 2020.

[Loshchilov and Hutter, 2019] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR (Poster)*. OpenReview.net, 2019.

[Mangrulkar *et al.*, 2022] Sourab Mangrulkar, Ankith MS, and Vivek Sembium. Be3r: Bert based early-exit using expert routing. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 3504–3512, 2022.

[Schwartz *et al.*, 2020] Roy Schwartz, Gabriel Stanovsky, Swabha Swayamdipta, Jesse Dodge, and Noah A. Smith. The right tool for the job: Matching model and instance complexities. In *ACL*, pages 6640–6651. Association for Computational Linguistics, 2020.

[Sun *et al.*, 2022] Tianxiang Sun, Xiangyang Liu, Wei Zhu, Zhichao Geng, Lingling Wu, Yilong He, Yuan Ni, Guotong Xie, Xuanjing Huang, and Xipeng Qiu. A simple hash-based early exiting approach for language understanding and generation. In *ACL (Findings)*, pages 2409–2421. Association for Computational Linguistics, 2022.

[Wang *et al.*, 2019] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *ICLR (Poster)*. OpenReview.net, 2019.

[Wolf *et al.*, 2020] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *EMNLP (Demos)*, pages 38–45. Association for Computational Linguistics, 2020.

[Xin *et al.*, 2020] Ji Xin, Raphael Tang, Jaejun Lee, Yaoliang Yu, and Jimmy Lin. Deebert: Dynamic early exiting for accelerating BERT inference. In *ACL*, pages 2246–2251. Association for Computational Linguistics, 2020.

[Xin *et al.*, 2021] Ji Xin, Raphael Tang, Yaoliang Yu, and Jimmy Lin. Berxit: Early exiting for BERT with better

fine-tuning and extension to regression. In *EACL*, pages 91–104. Association for Computational Linguistics, 2021.

[Zeng *et al.*, 2024] Ziqian Zeng, Yihuai Hong, Hongliang Dai, Huiping Zhuang, and Cen Chen. Consistentee: A consistent and hardness-guided early exiting method for accelerating language models inference. In *Thirty-Eighth AAAI Conference on Artificial Intelligence*, pages 19506–19514. AAAI Press, 2024.

[Zhang *et al.*, 2022] Zhen Zhang, Wei Zhu, Jinfan Zhang, Peng Wang, Rize Jin, and Tae-Sun Chung. PCEE-BERT: accelerating BERT inference via patient and confident early exiting. In *NAACL-HLT (Findings)*, pages 327–338. Association for Computational Linguistics, 2022.

[Zhang *et al.*, 2023] Jingfan Zhang, Ming Tan, Pengyu Dai, and Wei Zhu. Leco: Improving early exiting via learned exits and comparison-based exiting mechanism. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 4: Student Research Workshop)*, pages 298–309, 2023.

[Zhou *et al.*, 2020] Wangchunshu Zhou, Canwen Xu, Tao Ge, Julian J. McAuley, Ke Xu, and Furu Wei. BERT loses patience: Fast and robust inference with early exit. In *NeurIPS*, 2020.

[Zhu *et al.*, 2021] Wei Zhu, Xiaoling Wang, Yuan Ni, and Guotong Xie. 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, 2021.

[Zhu *et al.*, 2023] Wei Zhu, Peng Wang, Yuan Ni, Guotong Xie, and Xiaoling Wang. Badge: speeding up bert inference after deployment via block-wise bypasses and divergence-based early exiting. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 5: Industry Track)*, pages 500–509, 2023.

[Zhu, 2021] Wei Zhu. 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, 2021.