RAEE: A ROBUST RETRIEVAL-AUGMENTED EARLY EXIT FRAMEWORK FOR EFFICIENT INFERENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Deploying large language model inference remains challenging due to their high computational overhead. Early exit optimizes model inference by adaptively reducing the number of inference layers. Current methods typically train internal classifiers to determine whether to exit at intermediate layers. However, such classifier-based early exit frameworks require significant effort to train the classifiers while can only achieve comparable performance at best. To address these limitations, this paper proposes RAEE, a robust **R**etrieval-**A**ugmented **E**arly **E**xit framework for efficient inference. This paper first demonstrates that the early exit problem can be effectively modeled as a distribution prediction problem, in which the distribution is approximated through the exit information of similar data. Subsequently, it outlines the methodology for collecting exit information to construct the retrieval database. Finally, leveraging the pre-constructed retrieval database, RAEE utilizes the exit information from retrieved similar data to guide the backbone model's exit at the layer. Experimental results demonstrate that RAEE significantly accelerates inference while achieving robust zero-shot performance across eight downstream tasks.

1 INTRODUCTION

Large language models have been widely used in various application scenarios due to their excellent performance (Thoppilan et al., 2022; Touvron et al., 2023; Scao et al., 2022). However, improving the efficiency of model inference remains a critical and challenging task due to the high computational overhead involved in the process (Dao et al., 2022; Liu et al., 2023). As an advanced technique, model pruning provides a new direction for efficient inference (Valicenti et al., 2023; Ma et al., 2023). It selectively removes less important weights or connections from the neural network to reduce complexity and computational requirements without significantly degrading performance. One popular model pruning method is the early exit technique, which speeds up inference by adaptively reducing the number of inference layers.

Most early exit frameworks (Liu et al., 2020; Zhu, 2021; Xin et al., 2020; Fan et al., 2024) leverage classifiers to predict the exit layer and stop the inference at the predicted exit layer. Those early exit frameworks can be categorized into three branches according to the training strategy. The first one is **training-based early exit frameworks** (Zhu, 2021; Zhou et al., 2020; Zhu et al., 2023; Bae et al., 2023; Schuster et al., 2022), which requires training the classifiers along with the backbone models and updating all parameters of backbone models as well as classifiers. These methods introduce significant training overheads, particularly when applied to large language models. The second one is **semi-training-based early exit frameworks** (Fan et al., 2024). The backbone models in those works would not be updated, and only classifiers would be fitted to predict the exit layer. These methods may not capture the patterns between inputs and exit layers well, requiring significant human effort in feature engineering. The last one is **training-free early exit frameworks** (Sun et al., 2022), which requires no parameter updates and uses heuristics to determine the exit layer. These methods lack the generalization ability to predict the exit layer and often fail to achieve good performance. Moreover, most early exit frameworks sacrifice the model performance for acceleration (Fan et al., 2024; Sun et al., 2022; Schuster et al., 2022; Bae et al., 2023). This paper mainly focuses on improving training-free early exit methods to avoid introducing too many training overheads.

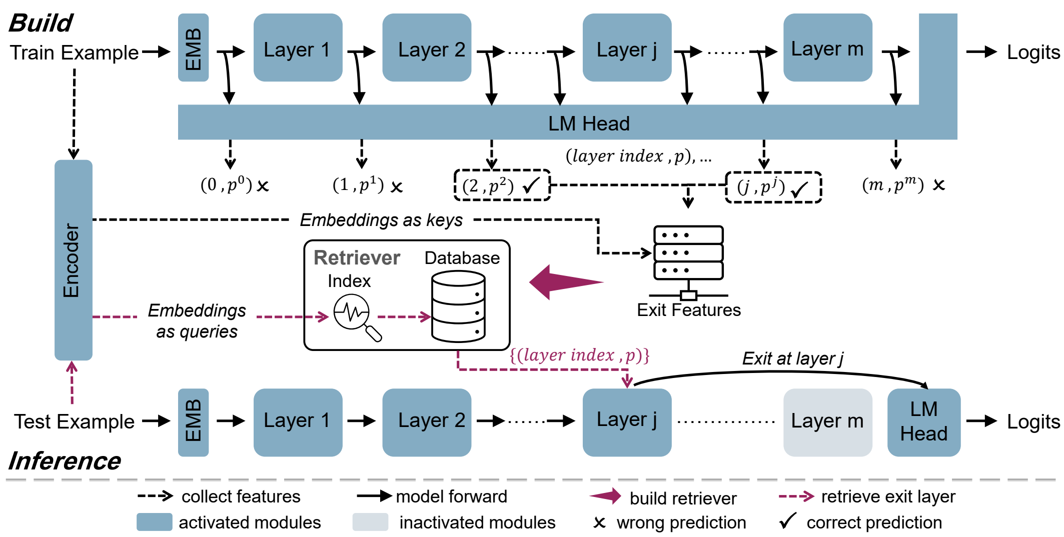


Figure 1: The overview of retrieval-augmented early exit framework. During the build phase, a retrieval database is constructed from the collected exit features, including layer indexes and their corresponding probabilities for correct predictions. During the inference phase, the framework retrieves similar data's exit information based on data embeddings to guide the model in selecting the optimal exit layer.

To address the above limitations, this paper first demonstrates that the exit layer can be predicted from an exit distribution, which can be approximated by similar data's exit information. Based on the observations, this paper proposes RAEE, a robust retrieval-augmented early exit framework for efficient inference. First, RAEE collects exit information from data. Next, RAEE builds the indexing and database to retrieve exit information from similar data. Finally, RAEE predicts the exit layer based on the top-k nearest neighbors' exit information during the inference and stops the model inference at the predicted exit layer.

We conduct comprehensive experiments to evaluate the proposed RAEE and various comparison methods on eight downstream tasks. Experimental results demonstrate that RAEE can accelerate the model inference while achieving robust model performance. Codes are available at [1].

The main contributions of this paper are:

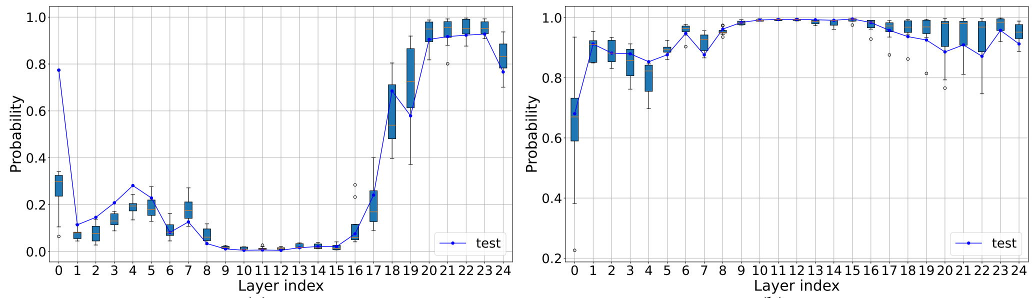
- We model the early exit problem as a distribution prediction problem and demonstrate that the exit information of similar data can approximate the exit distribution;
- We propose a robust retrieval-augmented early exit framework, RAEE, which leverages an external database to guide the early exit;
- Experimental results show that the proposed RAEE can accelerate the model inference and achieve robust performance.

2 Motivations

In this section, we demonstrate that using retrieval-based techniques is a simple yet effective way to augment the early exit framework during the inference stage.

Problem Statement. Formally, early exit can be defined as follows: Given a backbone model $\mathcal{M}$ with m layers and an input x, The early exit framework aims to design an exit function or classifier $l = f(x)$ to determine whether to exit at the layer l or which layer l to exit. The final prediction y is then transformed from the intermediate output states h_l of the l-th layer. And the final prediction

[1]https://anonymous.4open.science/r/RAEE-D724



(a) (b)

Figure 2: Exit layer probabilities for two SST-2 test samples and their top-8 nearest neighbors from the SST-2 training set. Each subfigure illustrates the probability distribution across different exit layers for one test sample and its corresponding nearest neighbors. The two test samples show different probability distribution phenomena. The results are collected with the backbone model RoBERTa-Large.

probability can be formulated as,

$$P(y \mid x) = P(y \mid h_{f(x)}), \tag{1}$$

where $f(x)$ is trained or built on the downstream tasks' training data $\mathcal{D}$.

Motivations of Retrieval-Augmented Early Exit. This paper aims to leverage retrieval-based techniques to guide the early exit, which behaves like training-free-based early exit frameworks requiring no parameters to update. Existing retrieval databases use clustering and product quantization to build the retrieval indexing over millions of embeddings, which can efficiently approximate nearest-neighbor searching. The indexing building process is not resource-constrained and can run on either GPUs or CPUs. Besides, since the retrieval database stores the original data, every retrieval in the retrieval-augmented early exit can be regarded as a generalization over several semantically similar data. In contrast, those original data may be used to train classifiers or build hashing functions in other early exit frameworks. The retrieval-augmented techniques also exhibit strong adaptability to new data. The index can retrieve up-to-date information by adding new data to the retrieval database.

To further demonstrate the efficacy of retrieval-augmented early exit, this paper conducts some analysis experiments in Figure 2. It shows the distribution of probability values for predicting the correct answers across all model layers during inference, including two SST-2 test samples (represented by the blue line with dots) and their respective top-8 nearest neighbors (depicted as blue box plots at each layer) retrieved from the SST-2 training dataset. The probability values in Figure 2 are calculated as the normalized logits of the answer label tokens. Note that the reason why we can collect the probability values of SST-2 data for predicting the correct answers across all model layers is because the SST-2 task has labels that allow us to extract the relevant probability data for predicting the label token.

The experimental results in Figure 2 (a) and (b) indicate that the layer-wise probability trends for correctly predicting the answers during inference exhibit a similar pattern between the test samples and their respective nearest neighbors. At the same time, different inputs exhibit diverse behaviors in these layer-wise predictive probabilities. Notably, as observed in Figure 2 (a), the probability values for predicting the correct answers of the SST-2 test sample and its top-8 nearest neighbors at the 24th layer are lower than those at layers 20 to 23. In such cases, leveraging the probability distribution of the top-8 nearest neighbors to guide the determination of the test sample's exit layer can allow it to exit at a layer with higher probability values for correct predictions. Thus this approach not only enhances inference speed by reducing the number of inference layers but also improves the accuracy of predictions. Consequently, the experimental results demonstrate that **the probabilities of top-k nearest neighbors from the pre-collected database can approximate the layer-wise exit probabilities of new input data.** Additionally, different inputs exhibit distinct probability distributions across the model's all exit layers. These observations motivate us to propose a retrieval-augmented early exit framework.

Algorithm 3.1 Collect the exit features as keys and values for building the retrieval database.

Input: Training data $\mathcal{D} = \{(x_1^{train}, y_1^{train}), \ldots, (x_{|\mathcal{D}|}^{train}, y_{|\mathcal{D}|}^{train})\}$, backbone model $\mathcal{M}$ with m layers $\{\mathcal{L}_1, \ldots, \mathcal{L}_m\}$, encoder $\mathcal{E}$ (None value means no encoder is provided).
Output: Keys $\mathcal{K}$ and values $\mathcal{V}$.

1: $\mathcal{K} = [], \mathcal{V} = []$
2: **for** $i = 1, \ldots, |\mathcal{D}|$ **do**
3:     $v_i = []$;
4:     $h_0 = \mathcal{M}_{emb}(x_i^{train})$;
5:     **for** j=1, ..., m **do**
6:         $h_j = \mathcal{L}_j(h_{j-1})$; /* Compute the intermediate outputs of the layer j */
7:         $logits = \mathcal{M}_{lm\_head}(h_j)$; /* Predict from the layer j */
8:         $\hat{y} = \arg\max logits,\ p_i^j = \max\{\mathrm{softmax}(logits)\}$;
9:         **if** $\hat{y}$ is equal to y_i^{train} **then**
10:             Add (j, p_i^j) into v_i; /* Store the possible exit layer */
11:         **end if**
12:     **end for**
13:     Add v_i into $\mathcal{V}$;
14:     **if** $\mathcal{E}$ is None **then**
15:         Add h_0 into $\mathcal{K}$; /* Store the embeddings of backbone model when no encoder model */
16:     **end if**
17: **end for**
18: **if** $\mathcal{E}$ is not None **then**
19:     Add all $\mathcal{E}(x_i^{train})$ into $\mathcal{K}$; /* Store the embeddings of encoder */
20: **end if**
21: **return** $\mathcal{K}, \mathcal{V}$;

3 Methodology

This section presents the retrieval-augmented early exit framework in detail. First, this paper outlines the process of collecting exit features and constructing the retrieval database to facilitate early exit. Then, this paper introduces the retrieval-augmented early exit framework, denoted as **RAEE**.

3.1 Collecting the Exit Features and Building the Retrieval Database

This paper uses the collected exit features as the keys and values within the retrieval database. To avoid introducing too much retrieving overheads, this paper only retrieves once at the beginning of the backbone model. Consider the training data $\mathcal{D} = \{(x_1^{train}, y_1^{train}), \ldots, (x_{|\mathcal{D}|}^{train}, y_{|\mathcal{D}|}^{train})\}$ and a backbone model $\mathcal{M}$ with m layers $\{\mathcal{L}_1, \ldots, \mathcal{L}_m\}$. In this context, as shown in the top part of Figure 1, the keys $\mathcal{K}$ are input embeddings of training data, which can be obtained from an extra encoder model $\mathcal{E}$, such as BERT (Devlin et al., 2019), or the outputs of embedding layers in the backbone model $\mathcal{M}_{emb}$,

$$\mathcal{K} = \{e_i\}_{i=1}^{|\mathcal{D}|} = \{\mathcal{E}(x_i^{train})\}_{i=1}^{|\mathcal{D}|}. \tag{2}$$

For the values, this paper collects a set of possible exit layers l_i and corresponding probabilities p_i for each embedding e_i, i.e., $v_i = \{(l_i^j, p_i^j)\}_{j=1}^{m_i}$, where m_i indicates the number of possible exit layers for the embedding e_i. The layer l chosen as the exit layer is determined by whether the outputs of this layer h_l can be used to make the right predictions $\hat{y}$ compared to the training labels y^{train}. Then, the values $\mathcal{V}$ are all sets of possible exit layers,

$$\mathcal{V} = \{v_i\}_{i=1}^{|D|} = \left\{\{(l_i^j, p_i^j)\}_{j=1}^{m_i}\right\}_{i=1}^{|D|}. \tag{3}$$

Algorithm 3.1 shows the detailed steps of preparing keys and values. We follow the same dataset splitting used in the LM-BFF (Gao et al., 2021), *the collecting process requires no parameters to update, only model inference is performed*. When the encoder $\mathcal{E}$ is unavailable, RAEE can also leverage the hidden states generated by the backbone model $\mathcal{M}$ as embeddings for indexing purposes (Lines 14-16).

After collecting keys and values for the retrieval databases, this paper uses state-of-the-art approximate nearest neighbor search indexing, such as FAISS (Johnson et al., 2019), and efficient key-value stores to build the retrieval database.

Algorithm 3.2 Model inference with the synchronized retrieval-augmented early exit.

Input: Input x, backbone model $\mathcal{M}$ with m layers $\{\mathcal{L}_1, \ldots, \mathcal{L}_m\}$, encoder $\mathcal{E}$, indexing $\mathcal{I}$, top-k, the exit layer determination function $f(\cdot)$.
Output: Final prediction $\hat{y}$.

$h_0 = \mathcal{M}_{emb}(x)$;
if $\mathcal{E}$ is not None **then**
  $e_{query} = \mathcal{E}(x)$; /* Encode the inputs when the encoder is available */
else
  $e_{query} = h_0$; /* Use the embeddings of backbone model */
end if
$\{(v_i, dis_i)\}_{i=1}^{k} = \mathcal{I}(e_{query}, k)$; /* Retrieve the possible exit layers */
$l = f\left(\{(v_1, dis_1), \ldots, (v_k, dis_k)\}\right)$; /* Obtain the exit layer */
for $i = 1, \ldots, l$ **do**
  $h_i = \mathcal{L}_i(h_{i-1})$; /* Perform model inference with early exit */
end for
$logits = \mathcal{M}_{lm\_head}(h_l)$; /* Predict based on the layer h_l outputs */
$\hat{y} = \arg\max logits$;
return $\hat{y}$;

3.2 The Retrieval-Augmented Early Exit Framework

In this section, this paper then presents a retrieval-augmented early exit framework named RAEE to optimize the model inference. RAEE regards the exit layer as a random variable z, taking values in the set of $\{1, \ldots, m\}$, where m is the total number of layers in the backbone model $\mathcal{M}$. The probability mass function $P(z = l)$ represents the probability of the case that the backbone model exits at the layer l. With the gold label, we can observe that the random variable z follows an unknown discrete distribution F. Then, this paper shows how to leverage the retrieval database to approximate the distribution F.

Given an input x, RAEE first retrieves top-k nearest neighbors $\{v_1, \ldots, v_k\}$, where each neighbor v_i has m_i possible exit layers. Naturally, we can approximate the distribution F by estimating the probability function $P(z = l)$,

$$P(z = l \mid x) = \sum_{i=1}^{k} P(v_i \mid x) \cdot \sum_{j}^{m_i} \mathbb{1}\left(any(l_i^j = l \;\&\&\; p_i^j \geq \tau)\right) \cdot p_i^j, \tag{4}$$

where $\mathbb{1}$ is the indicator function that returns 1 if the condition is true and 0 otherwise, $any(\cdot)$ is the function that returns true if one condition is true and false otherwise, τ is the threshold for filtering the layers, the inner loop only count once since there is at most one possible exit layer of neighbor i that is equalt to l. Since different neighbors should have different contributions to the probability function $P(z = l)$, RAEE uses the reciprocal of the scaled distance between each neighbor and the query to estimate the contribution,

$$P(v_i \mid x) = \frac{\min\left(\{distance(v_j, x)\}_{j=1}^{k}\right)}{distance(v_i, x)}. \tag{5}$$

Then, RAEE designs a function $f(x)$ to determine the exit layer, which selects the layer that maximizes the probability function $P(z = l)$,

$$f(x) = \arg\max_{l} P(z = l \mid x). \tag{6}$$

Notably, when multiple exit layers have the same maximal probability, RAEE selects the earliest one.

The bottom part of Figure 1 shows the inference workflow of RAEE. Specifically, RAEE first simultaneously feeds the inputs into both the backbone model for the label predictions and the same encoder used in the building process for the query embeddings. Then, the retriever in RAEE retrieves the top-k nearest neighbors in the retrieval databases based on the query embeddings. After obtaining all possible exit layers of k nearest neighbors, RAEE computes the exit layers based on the Equations 4-6. Finally, RAEE stops the forwarding at the calculated exit layer and passes the intermediate outputs of the exit layer to the final prediction layer, e.g., LM Head in language models, to obtain the final predictions (Equation 1). This is implemented based on the Transformer library,

Table 1: Zero-shot model performance of different methods on eight downstream tasks. 'RB-L', 'EB-L', and 'T5-L' refer to RoBERTa-Large, ElasticBERT-Large, and T5-Large, respectively.

Methods	**SST-2**	**SST-5**	**MR**	**CR**	**MPQA**	**SUBJ**	**TREC**	**CoLA**	**Avg**
RB-L	83.60	34.98	80.80	79.55	67.60	51.45	32.40	2.03	54.05
EB-L	51.15	22.35	49.25	48.65	48.05	48.85	17.60	0.11	35.75
T5-L	49.31	23.12	50.40	50.90	45.40	52.75	27.60	-4.64	36.86
Llama-3-8B	62.84	26.06	59.65	72.90	51.75	52.80	8.40	0.00	41.80
Gemma-7B	49.08	28.64	50.05	50.10	50.00	48.05	14.40	-0.79	36.19
Backbone: RoBERTa-Large, ElasticBERT-Large									
HashEE (EB-L)	49.08	14.16	49.95	50.05	50.00	50.00	27.00	0.00	36.28
DeeBERT (RB-L)	52.29	18.05	50.60	50.00	75.95	80.85	16.20	0.00	42.99
AdaInfer (RB-L)	50.92	24.48	50.00	50.00	60.90	50.85	22.60	-1.62	38.52
RAEE (RB-L)	84.63	33.57	81.55	68.05	78.55	84.05	62.40	14.48	**63.41**
Backbone: T5-Large									
CALM (T5-L)	51.72	23.17	49.25	50.55	49.80	49.90	18.00	0.00	36.55
AdaInfer (T5-L)	50.11	28.14	50.35	49.80	46.30	49.95	26.00	5.22	38.23
RAEE (T5-L)	52.98	26.56	50.80	51.60	55.65	49.90	39.80	12.20	**42.44**
Backbone: Llama-3-8B									
SLEB (Llama)	54.01	21.09	51.10	49.45	55.65	49.95	14.00	0.92	37.02
AdaInfer (Llama)	53.21	18.05	53.50	50.00	49.95	47.55	16.20	0.00	36.06
RAEE (Llama)	73.05	35.25	66.45	57.95	75.05	90.05	51.80	9.55	**57.39**
Backbone: Gemma-7B									
SLEB (Gemma)	50.69	19.82	49.95	49.95	50.00	52.10	12.80	0.00	35.66
AdaInfer (Gemma)	50.92	12.62	50.00	50.00	50.00	50.60	22.60	0.00	35.84
RAEE (Gemma)	73.17	32.40	66.75	56.75	75.60	90.15	40.00	10.46	**55.66**

passing the exit layer as a parameter into the 'forward()' function and stopping the inner iteration based on the exit layer.

Algorithm 3.2 performs model inference with retrieval-augmented early exit. When the encoder $\mathcal{E}$ is unavailable (Line 5), RAEE utilizes the hidden states from the backbone model $\mathcal{M}$ as embeddings for querying. The specific layer from which the hidden states are extracted is treated as a hyperparameter. The inference process (Lines 9-11) and the retrieving process (Lines 2-8) can be executed in parallel for more efficient implementations.

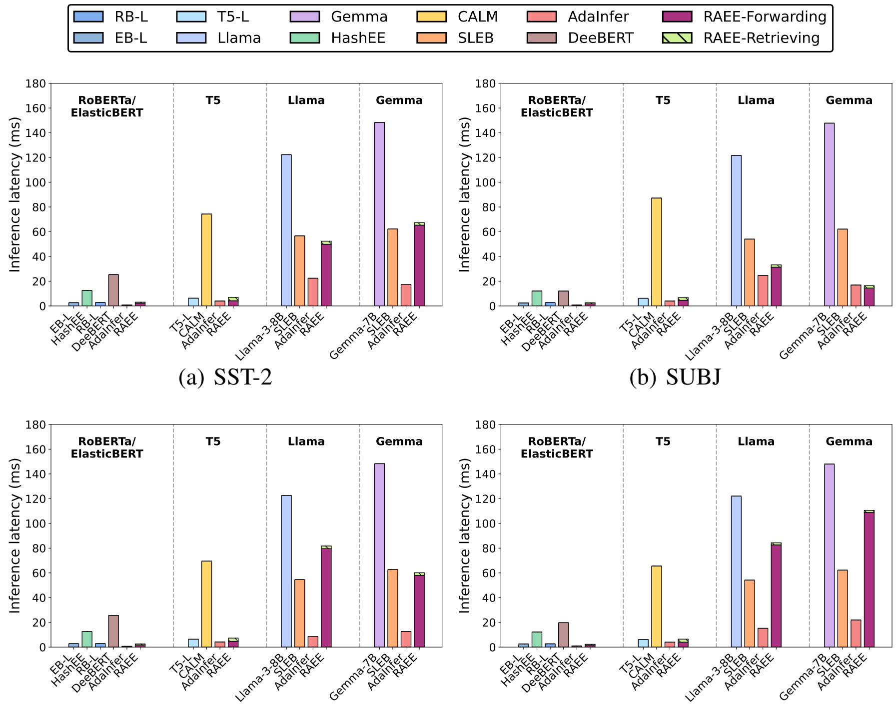
4 Experiments

In this section, we first introduce the dataset and the experimental setup. Then, we presented the main results of different methods on eight downstream tasks. We also conducted analysis experiments and ablation studies to show the impact of these factors on RAEE performance.

4.1 Dataset and Experimental Setup

Datasets We conducted comprehensive experiments across eight downstream tasks from GLUE benchmark (Wang et al., 2019), covering sentiment analysis, opinion polarity analysis, grammatical judgment, natural language inference, paraphrasing, etc.

Experimental Setup The proposed RAEE was implemented using the PyTorch framework and Transformer. We evaluated methods based on the backbone models RoBERTa-Large (Liu et al., 2019) and T5-Large (Raffel et al., 2020) on one NVIDIA GeForce RTX 4090 with 24GB GPU memory, while Llama-3-8B (Dubey et al., 2024) and Gemma-7B (Mesnard et al., 2024) on one NVIDIA A100 GPU with 40GB GPU memory. The experiments were conducted in two settings, i.e., zero-shot settings for training-free-based methods and fine-tuning settings only for semi-training-based methods. The evaluation metric is accuracy, except for the Matthew correlation coefficient for the CoLA task. The number of retrieved nearest neighbors of RAEE is set to 12 in the experiments. The threshold τ of RAEE is set to 0.9.



(a) SST-2 (b) SUBJ

(c) TREC (d) CoLA

Figure 3: Inference latency of RAEE compared with various methods on selected downstream tasks. The backbone models used in the comparisons include RoBERTa-Large/ElasticBERT-Large, T5-Large, Llama-3-8B, and Gemma-7B.

To validate the effectiveness, we compared RAEE with three types of methods. **Pretrained Models:** 1) RoBERTa-Large (Liu et al., 2019), a state-of-the-art encoder model, where the prompt-based version (Gao et al., 2021) is used; 2) ElasticBERT (Liu et al., 2022), a pre-trained multi-exit transformer model, where the large version is used in this paper; 3) T5-Large (Raffel et al., 2020), a versatile transformer-based model for various NLP tasks; 4) Llama-3-8B (Dubey et al., 2024), a pre-trained model with strength in specific language scenarios; 5) Gemma-7B (Mesnard et al., 2024), a model with the potential for outstanding performance in specific settings. **Training-Free Methods:** 1) HashEE (Sun et al., 2022), a hash-based early exit approach with ElasticBERT-Large as its backbone model; 2) CALM (Schuster et al., 2022), a classical entropy-thresholding-based early exit method with T5-Large as its backbone model, where the zero-shot setting is applied; 3) SLEB (Song et al., 2024), a method that eliminates redundant transformer blocks. **Semi-Training Methods:** 1) AdaInfer (Fan et al., 2024), an SVM-based early exiting method with our reproduced version; 2) DeeBERT (Xin et al., 2020), a classical entropy-thresholding-based early exiting method with RoBERTa-Large as its backbone model; The templates are listed in the Appendix A. More details about the experimental setup can be found in Appendix C.

4.2 Main Results

Table 1 presents the main results, comparing the performance of RAEE against different types of methods across eight downstream tasks. Experimental results show that the proposed RAEE can achieve the best zero-shot performance on average across all tasks, which is 63.41 with the backbone model RoBERTa-Large. RAEE with Gemma-7B achieves the maximal improvements over the baseline models, while RAEE with RoBERTa-Large achieves the maximal improvements over comparison methods from 36.28 to 63.41. Across eight downstream tasks, RAEE consistently improves the model performance compared to current state-of-the-art early exit frameworks.

Figure 3 shows the inference latency of RAEE and comparisons on eight downstream tasks. We show the inference latency on the selected four tasks, which covers different task types. For million-

Table 2: Model performance and inference latency of RAEE and RAEE without true predictions in the retrieval database. The backbone model is Llama-3-8B.

Models	SST-2	SST-5	MR	CR	MPQA	SubJ	TREC	CoLA	Avg
Performance ↑									
Llama	62.84	26.06	59.65	72.90	51.75	52.80	8.40	0.00	41.80
RAEE w/o	60.55	24.52	57.30	53.55	56.65	81.70	20.80	0.00	44.38
RAEE	73.05	35.25	66.45	57.95	75.05	90.05	51.80	9.55	**57.39**
Latency (ms) ↓									
Llama	122.27	122.13	122.03	121.78	121.82	121.70	122.52	122.06	122.04
RAEE w/o	37.65	115.26	37.98	31.69	54.08	34.59	112.03	91.34	64.33
RAEE	52.33	65.47	53.83	34.65	55.02	33.25	81.74	84.34	**57.58**

Table 3: The impact of retrieval number k on the distribution approximation.

k	SST-2	SST-5	MR	CR	MPQA	SUBJ	TREC	CoLA	Avg
2	79.82	32.99	76.40	68.40	77.55	81.75	61.00	7.89	60.73
4	82.22	33.17	79.20	68.05	78.65	83.60	63.60	14.74	62.90
8	84.52	34.12	80.70	68.05	78.90	84.20	63.20	12.19	63.24
12	84.63	33.57	81.55	68.05	78.55	84.05	62.40	14.48	**63.41**
16	84.98	32.17	82.05	69.00	77.90	84.00	62.60	13.00	63.21
20	85.44	32.26	81.80	68.95	78.05	83.50	62.40	12.40	63.10

level backbone models, such as RoBERTa-Large, ElasticBERT-Large, T5-Large, RAEE can achieve comparable inference efficiency. This is due to that the inference speeds of those backbone models are already fast enough, introducing too many components for early exit would degrade the inference efficiency like HashEE and DeeBERT. However, for billion-level backbone models, the acceleration of RAEE is significant. This benefits from the effectively predicted early exit layers. Although Adainfer with the backbone Llama-3-8B and Gemma-7B is much faster than RAEE, it only achieves a comparable performance of backbone models. RAEE significantly improves those backbone models' performance and also reduces the inference latency by nearly half.

4.3 Reasons for Significant Performance Improvement

The main results demonstrate that the proposed RAEE can significantly outperform backbone models, which is not an intuitive result compared to previous early exit methods. The reasons for this lie in that **the collected exit information guides the RAEE as an error corrector**. This means that RAEE can learn from the exit information of examples that are correctly predicted by intermediate layers, but backbone models without early exit fail to predict.

To better support the above claims, we also conducted an analysis experiment using the retrieval database that only contains the exit information of data that Llama-3-8B correctly predicted without early exit. As shown in Table 2, RAEE w/o refers to the one built on only correctly predicted examples. As expected, RAEE w/o achieves comparable performance to baselines but accelerates the inference process. This is because the test data that is correctly predicted by RAEE w/o can also be correctly predicted by backbone models. However, due to a lack of exit information on examples where backbone models fail to predict, RAEE w/o also fails to predict on the test data where backbone models fail. Therefore, when providing the exit information based on examples where backbone models fail to predict but intermediate outputs succeed in predicting, RAEE can make correct predictions and exit earlier.

Table 4: The impact of retrieval database size on the distribution approximation. The percentage refers to the amount of training data that is used to build the retrieval database.

Database Size	SST-2	SST-5	MR	CR	MPQA	SUBJ	TREC	CoLA	Avg
20%	84.63	31.76	82.80	65.85	75.70	81.00	58.80	8.77	61.16
50%	83.37	32.85	80.65	66.95	77.90	83.60	61.20	11.73	62.28
100%	84.63	33.57	81.55	68.05	78.55	84.05	62.40	14.48	**63.41**

Table 5: The building cost of RAEE and different comparison methods. The results are collected with the backbone model RoBERTa-Large (ElasticBERT for HashEE).

Model	SST-2	SST-5	MR	CR	MPQA	Subj	TREC	CoLA	Avg
Time Cost (seconds)									
RAEE	91.75	111.80	112.09	27.43	110.26	103.99	71.19	108.59	92.14
HashEE	5.85	14.13	14.68	2.82	10.40	13.84	7.50	5.81	9.38
AdaInfer	91.85	124.78	129.11	41.77	122.96	120.52	66.43	114.19	101.45
Storage Overheads (MB)									
RAEE (Index)	3.4	3.7	3.8	2.0	3.8	3.6	3.1	3.7	3.4
RAEE (DB)	2.5	2.0	3.0	0.8	2.5	2.7	1.0	2.6	2.1

4.4 Ablation Study

Impact of Top-*k*: Table 3 illustrates the impact of varying the number of retrievals on the distribution approximation. As k increases, the proposed RAEE with the backbone model RoBERTa-Large improves the overall performance from 60.73 to 63.41. This suggests that more retrieved exiting information can help enhance the approximation performance. However, when k exceeds 12, the overall performance degrades from 63.41 to 63.10. The reasoning behind this may be that providing exit information from retrievals that are not quite related to the query introduces noise, misleading the final predictions. This suggests that a limited amount of relevant exit information is sufficient to approximate the exit distribution, thereby saving time in the retrieval process.

Retrieval Database Size: Table 4 shows the performance of RAEE with different sizes of retrieval databases. The size of the retrieval databases implies how similar embeddings would be retrieved, thus impacting the confidence of the provided exit information. As the database size increases, the performance of RAEE with the backbone model RoBERTa-Large increases significantly from 61.16 to 63.41 on average. This demonstrates that collecting more data can improve the generalization of RAEE, thus approximating the exit distribution more accurately.

4.5 Building Overheads

We present the building overheads in Table 5, including database size, index size, and building time costs for different methods. Specifically, the building time cost for RAEE refers to the time required to build the retriever, while the building time cost for AdaInfer pertains to the time needed for training the classifier. In the case of HashEE, the building time cost corresponds to the time taken to build the hashing buckets. For RAEE, the average time required to build the retrieval database is under 2 minutes on a single NVIDIA GeForce RTX 4090, which is considered an acceptable overhead in comparison to the time involved in fine-tuning. The index and database sizes are relatively small and can be considered negligible in comparison to the size of the backbone model.

5 Related Work

5.1 Early Exit Framework

Model inference with early exit has been a popular pruning method to reduce both computation and memory overhead on text classification or generation tasks. Most current works (Bae et al., 2023; Kong et al., 2022; Ji et al., 2023; Wolczyk et al., 2021; Hooper et al., 2023) introduce classifiers in each layer to determine whether the inference should continue. (Xin et al., 2020; Liu et al., 2020; Zhou et al., 2020; He et al., 2021) train the classifier by minimizing the differences between each layer's outputs and final outputs, then perform early-exit-based inference according to a threshold. (Liao et al., 2021) incorporates the past and future information to predict the early exit layer. Instead of training a neural network as classifiers, (Fan et al., 2024) only fit the machine learning classifiers on the extracted features for the early exit. (Zhu, 2021; Zhu et al., 2021; 2023; Zhang et al., 2023a) focus on designing novel loss functions for training a more robust classifier. (Li et al., 2021) incorporates sentence-level features as well as token-level features to predict the early exit. Different from those works, our method does not require training the classifier. (Sun et al., 2022) proposes a hash-based early exit method that uses the hashing functions to map tokens to exit layers. (Bajpai & Hanawal, 2024) introduces an online learning algorithm for early exits in BERT models, dynamically determining exit points based on confidence thresholds. (Regol et al., 2023) proposes a jointly-learned framework for early exiting and inference in dynamic neural networks, integrating gating mechanisms and intermediate inference modules. (Balagansky & Gavrilov, 2022) introduces a deterministic Q-exit criterion and revising the model architecture. Our method predicts exit layers using pre-built databases, resulting in better generalization. Other works (Jazbec et al., 2023; Li et al., 2023; Huang et al., 2018) focus on designing early exit frameworks for image classification tasks. They are optimizations in the different domains compared to our works.

5.2 Retrieval-based Augmentations

Retrieval-based augmentations (Li et al., 2022; Wang et al., 2023; Xiong et al., 2023; Cui et al., 2023; Wu et al., 2024a;b) have been widely used in various natural language processing (NLP) tasks and achieved remarkable performance. Current works mostly leverage external knowledge databases to augment generator models on various text-generation tasks, such as language modeling (Khandelwal et al., 2020; Lewis et al., 2020; Borgeaud et al., 2022), question-answering (Guu et al., 2020; Izacard & Grave, 2021), machine translation (Khandelwal et al., 2021; Wang et al., 2022), dialogue system (Cheng et al., 2023). Those works focus on improving the model's generation quality, while our work aims to use the retrieval knowledge to accelerate the model's inference. Additionally, other work like (Zhang et al., 2023b) aims to accelerate the inference process by retrieving precomputed trajectories from a knowledge base, which is specifically designed for diffusion models that differ from our target models. These works are out of the scope of the research problems in this paper.

6 Limitations

Although the proposed RAEE can improve both model performance and model efficiency, several limitations warrant discussion. The effectiveness of RAEE depends on the pre-built in-domain retrieval databases, which can well approximate the exit distribution for predictions. The framework is primarily designed for in-domain training and testing scenarios, which represent the mainstream tasks. Consequently, the out-of-domain performance of RAEE may be constrained; however, this aspect is not the primary focus of this paper and will be the subject of future research.

7 Conclusion

This paper models the early exit problem as a distribution prediction problem and observes that similar data's exit information can be used to approximate the distribution. Based on the observations, this paper proposes a retrieval-augmented early exit framework named RAEE. Experimental results show that RAEE can accelerate the model inference while significantly improving the model performance.

References

Sangmin Bae, Jongwoo Ko, Hwanjun Song, and Se-Young Yun. Fast and robust early-exiting framework for autoregressive language models with synchronized parallel decoding. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 5910–5924, 2023.

Divya Jyoti Bajpai and Manjesh K. Hanawal. Ceebert: Cross-domain inference in early exit BERT. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pp. 1736–1748. Association for Computational Linguistics, 2024.

Nikita Balagansky and Daniil Gavrilov. PALBERT: teaching ALBERT to ponder. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack W. Rae, Erich Elsen, and Laurent Sifre. Improving language models by retrieving from trillions of tokens. In *Proceedings of the 2022 International Conference on Machine Learning (ICML)*, pp. 2206–2240, 2022.

Xin Cheng, Di Luo, Xiuying Chen, Lemao Liu, Dongyan Zhao, and Rui Yan. Lift yourself up: Retrieval-augmented text generation with self-memory. In *Proceedings of the Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2023.

Yufei Cui, Ziquan Liu, Yixin Chen, Yuchen Lu, Xinyue Yu, Xue (Steve) Liu, Tei-Wei Kuo, Miguel Rodrigues, Chun Jason Xue, and Antoni B. Chan. Retrieval-augmented multiple instance learning. In *Proceedings of the Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023 (NeurIPS)*, 2023.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Proceedings of the Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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 (NAACL-HLT)*, pp. 4171–4186. Association for Computational Linguistics, 2019.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, and et al. The llama 3 herd of models, 2024. URL `https://arxiv.org/abs/2407.21783`.

Siqi Fan, Xin Jiang, Xiang Li, Xuying Meng, Peng Han, Shuo Shang, Aixin Sun, Yequan Wang, and Zhongyuan Wang. Not all layers of llms are necessary during inference. *CoRR*, abs/2403.02181, 2024.

Tianyu Gao, Adam Fisch, and Danqi Chen. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL/IJCNLP)*, pp. 3816–3830. Association for Computational Linguistics, 2021.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. Retrieval augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, pp. 3929–3938, 2020.

Xuanli He, Iman Keivanloo, Yi Xu, Xiang He, Belinda Zeng, Santosh Rajagopalan, and Trishul Chilimbi. Magic pyramid: Accelerating inference with early exiting and token pruning. *CoRR*, abs/2111.00230, 2021.

Coleman Hooper, Sehoon Kim, Hiva Mohammadzadeh, Hasan Genc, Kurt Keutzer, Amir Gholami, and Yakun Sophia Shao. SPEED: speculative pipelined execution for efficient decoding. *CoRR*, abs/2310.12072, 2023.

Gao Huang, Danlu Chen, Tianhong Li, Felix Wu, Laurens van der Maaten, and Kilian Q. Weinberger. Multi-scale dense networks for resource efficient image classification. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

Gautier Izacard and Edouard Grave. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL): Main Volume*, pp. 874–880, 2021.

Metod Jazbec, James Urquhart Allingham, Dan Zhang, and Eric T. Nalisnick. Towards anytime classification in early-exit architectures by enforcing conditional monotonicity. In *Proceedings of the Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems (NeurIPS)*, 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)*, pp. 14128–14142, 2023.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547, 2019.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. Generalization through memorization: Nearest neighbor language models. In *Proceedings of the 8th International Conference on Learning Representations (ICLR)*, 2020.

Urvashi Khandelwal, Angela Fan, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. Nearest neighbor machine translation. In *Proceedings of the 9th International Conference on Learning Representations (ICLR)*, 2021.

Jun Kong, Jin Wang, Liang-Chih Yu, and Xuejie Zhang. Accelerating inference for pretrained language models by unified multi-perspective early exiting. In *Proceedings of the 29th International Conference on Computational Linguistics (COLING)*, pp. 4677–4686, 2022.

Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Proceedings of the Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

Xiaonan Li, Yunfan Shao, Tianxiang Sun, Hang Yan, Xipeng Qiu, and Xuanjing Huang. Accelerating BERT inference for sequence labeling via early-exit. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL/IJCNLP)*, pp. 189–199, 2021.

Yuhang Li, Tamar Geller, Youngeun Kim, and Priyadarshini Panda. SEENN: towards temporal spiking early exit neural networks. In *Proceedings of the Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2023.

Zonglin Li, Ruiqi Guo, and Sanjiv Kumar. Decoupled context processing for context augmented language modeling. In *Proceedings of the Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

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 *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT): Human Language Technologies*, pp. 2013–2023, 2021.

Weijie Liu, Peng Zhou, Zhiruo Wang, Zhe Zhao, Haotang Deng, and Qi Ju. Fastbert: a self-distilling BERT with adaptive inference time. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 6035–6044, 2020.

Xiangyang Liu, Tianxiang Sun, Junliang He, Jiawen Wu, Lingling Wu, Xinyu Zhang, Hao Jiang, Zhao Cao, Xuanjing Huang, and Xipeng Qiu. Towards efficient NLP: A standard evaluation and A strong baseline. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL): Human Language Technologies*, pp. 3288–3303, 2022.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019.

Zichang Liu, Jue Wang, Tri Dao, Tianyi Zhou, Binhang Yuan, Zhao Song, Anshumali Shrivastava, Ce Zhang, Yuandong Tian, Christopher Ré, and Beidi Chen. Deja vu: Contextual sparsity for efficient llms at inference time. In *Proceedings of the 2023 International Conference on Machine Learning (ICML)*, pp. 22137–22176, 2023.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. Llm-pruner: On the structural pruning of large language models. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Proceedings of the Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2023.

Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, Pouya Tafti, Léonard Hussenot, Aakanksha Chowdhery, Adam Roberts, Aditya Barua, Alex Botev, Alex Castro-Ros, Ambrose Slone, Amélie Héliou, Andrea Tacchetti, Anna Bulanova, Antonia Paterson, Beth Tsai, Bobak Shahriari, Charline Le Lan, Christopher A. Choquette-Choo, Clément Crepy, Daniel Cer, Daphne Ippolito, David Reid, Elena Buchatskaya, Eric Ni, Eric Noland, Geng Yan, George Tucker, George-Cristian Muraru, Grigory Rozhdestvenskiy, Henryk Michalewski, Ian Tenney, Ivan Grishchenko, Jacob Austin, James Keeling, Jane Labanowski, Jean-Baptiste Lespiau, Jeff Stanway, Jenny Brennan, Jeremy Chen, Johan Ferret, Justin Chiu, and et al. Gemma: Open models based on gemini research and technology. *CoRR*, abs/2403.08295, 2024.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67, 2020.

Florence Regol, Joud Chataoui, and Mark Coates. Jointly-learned exit and inference for a dynamic neural network : JEI-DNN. *CoRR*, abs/2310.09163, 2023.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilic, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, Jonathan Tow, Alexander M. Rush, Stella Biderman, Albert Webson, Pawan Sasanka Ammanamanchi, Thomas Wang, Benoît Sagot, Niklas Muennighoff, Albert Villanova del Moral, Olatunji Ruwase, Rachel Bawden, Stas Bekman, Angelina McMillan-Major, Iz Beltagy, Huu Nguyen, Lucile Saulnier, Samson Tan, Pedro Ortiz Suarez, Victor Sanh, Hugo Laurençon, Yacine Jernite, Julien Launay, Margaret Mitchell, Colin Raffel, Aaron Gokaslan, Adi Simhi, Aitor Soroa, Alham Fikri Aji, Amit Alfassy, Anna Rogers, Ariel Kreisberg Nitzav, Canwen Xu, Chenghao Mou, Chris Emezue, Christopher Klamm, Colin Leong, Daniel van Strien, David Ifeoluwa Adelani, and et al. BLOOM: A 176b-parameter open-access multilingual language model. *CoRR*, abs/2211.05100, 2022.

Tal Schuster, Adam Fisch, Jai Gupta, Mostafa Dehghani, Dara Bahri, Vinh Tran, Yi Tay, and Donald Metzler. Confident adaptive language modeling. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

Jiwon Song, Kyungseok Oh, Taesu Kim, Hyungjun Kim, Yulhwa Kim, and Jae-Joon Kim. SLEB: streamlining llms through redundancy verification and elimination of transformer blocks. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024.

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 *Findings of the Association for Computational Linguistics (ACL)*, pp. 2409–2421, 2022.

Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, YaGuang Li, Hongrae Lee, Huaixiu Steven Zheng, Amin Ghafouri, Marcelo Menegali, Yanping Huang, Maxim Krikun, Dmitry Lepikhin, James Qin, Dehao Chen, Yuanzhong Xu, Zhifeng Chen, Adam Roberts, Maarten Bosma, Yanqi Zhou, Chung-Ching Chang, Igor Krivokon, Will Rusch, Marc Pickett, Kathleen S. Meier-Hellstern, Meredith Ringel Morris, Tulsee Doshi, Renelito Delos Santos, Toju Duke, Johnny Soraker, Ben Zevenbergen, Vinodkumar Prabhakaran, Mark Diaz, Ben Hutchinson, Kristen Olson, Alejandra Molina, Erin Hoffman-John, Josh Lee, Lora Aroyo, Ravi Rajakumar, Alena Butryna, Matthew Lamm, Viktoriya Kuzmina, Joe Fenton, Aaron Cohen, Rachel Bernstein, Ray Kurzweil, Blaise Agüera y Arcas, Claire Cui, Marian Croak, Ed H. Chi, and Quoc Le. Lamda: Language models for dialog applications. *CoRR*, abs/2201.08239, 2022.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971, 2023.

Tim Valicenti, Justice Vidal, and Ritik Patnaik. Mini-gpts: Efficient large language models through contextual pruning. *CoRR*, abs/2312.12682, 2023.

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 *Proceedings of the 7th International Conference on Learning Representations (ICLR)*, 2019.

Shuohang Wang, Yichong Xu, Yuwei Fang, Yang Liu, Siqi Sun, Ruochen Xu, Chenguang Zhu, and Michael Zeng. Training data is more valuable than you think: A simple and effective method by retrieving from training data. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 3170–3179, 2022.

Weizhi Wang, Li Dong, Hao Cheng, Xiaodong Liu, Xifeng Yan, Jianfeng Gao, and Furu Wei. Augmenting language models with long-term memory. In *Proceedings of the Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2023.

Maciej Wolczyk, Bartosz Wójcik, Klaudia Balazy, Igor T. Podolak, Jacek Tabor, Marek Smieja, and Tomasz Trzcinski. Zero time waste: Recycling predictions in early exit neural networks. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pp. 2516–2528, 2021.

Shangyu Wu, Ying Xiong, Yufei Cui, Xue Liu, Buzhou Tang, Tei-Wei Kuo, and Chun Jason Xue. Improving natural language understanding with computation-efficient retrieval representation fusion. *CoRR*, abs/2401.02993, 2024a.

Shangyu Wu, Ying Xiong, Yufei Cui, Haolun Wu, Can Chen, Ye Yuan, Lianming Huang, Xue Liu, Tei-Wei Kuo, Nan Guan, and Chun Jason Xue. Retrieval-augmented generation for natural language processing: A survey. *CoRR*, abs/2407.13193, 2024b.

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

Ying Xiong, Xin Yang, Linjing Liu, Ka-Chun Wong, Qingcai Chen, Yang Xiang, and Buzhou Tang. EARA: improving biomedical semantic textual similarity with entity-aligned attention and retrieval augmentation. In *Findings of the Association for Computational Linguistics (EMNLP)*, pp. 8760–8771, 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 (ACL): Student Research Workshop*, pp. 298–309, 2023a.

Kexun Zhang, Xianjun Yang, William Yang Wang, and Lei Li. Redi: Efficient learning-free diffusion inference via trajectory retrieval. In *Proceedings of the 2023 International Conference on Machine Learning (ICML)*, pp. 41770–41785, 2023b.

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 *Proceedings of the Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

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 (ACL/IJCNLP)*, pp. 2968–2980, 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 (EMNLP)*, pp. 3033–3044, 2021.

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 The 61st Annual Meeting of the Association for Computational Linguistics (ACL): Industry Track*, pp. 500–509, 2023.

A TEMPLATES ON ALL TASKS

Table 6 provides an overview of the manual templates and selected label words used for each dataset with the backbone model RoBERTa-Large (Liu et al., 2019) in this paper. These templates and label words were created following LM-BFF (Gao et al., 2021).

Table 6: Templates and label words with the backbone model RoBERTa-Large.

Task	Prompts	Label word
SST-2	`[CLS]` x It was `[MASK]`. `[SEP]`	"0":"terrible", "1":"great"
SST-5	`[CLS]` x It was `[MASK]`. `[SEP]`	"0":"terrible","1": "bad", "2": "okay","3": "good","4": "great"
MR	`[CLS]` x It was `[MASK]`. `[SEP]`	"0":"terrible", "1":"great"
CR	`[CLS]` x It was `[MASK]`. `[SEP]`	"0":"terrible", "1":"great"
MPQA	`[CLS]` x It was `[MASK]`. `[SEP]`	"0":"terrible", "1":"great"
SUBJ	`[CLS]` x This is `[MASK]`. `[SEP]`	"0":"subjective", "1":"objective"
TREC	`[CLS]` `[MASK]` x `[SEP]`	"0":"Description","1":"Entity","2":"Expression", "3":"Human","4":"Location","5":"Number"
CoLA	`[CLS]` x It was `[MASK]`. `[SEP]`	"0":"incorrect", "1":"correct"

Table 7 provides an overview of the manual templates and selected label words used for each dataset with the backbone model T5-Large (Raffel et al., 2020), Llama-3-8B (Dubey et al., 2024) and Gemma-7B (Mesnard et al., 2024) in this paper.

Table 7: Templates and label words with the backbone model T5-Large, Llama-3-8B and Gemma-7B.

Task	Prompts	Label word
SST-2	What is the sentiment of the sentence x ? Print negative or positive. The answer is	"0":"negative", "1":"positive"
SST-5	What is the sentiment of the sentence x '? Print terrible, bad, okay, good or great. The answer is	"0":"terrible","1": "bad", "2": "okay","3": "good", "4": "great"
MR	What is the sentiment of the sentence x ? Print negative or positive. The answer is	"0":"negative", "1":"positive"
CR	What is the sentiment of the sentence x ? Print negative or positive. The answer is	"0":"negative", "1":"positive"
MPQA	What is the sentiment of the sentence x ? Print negative or positive. The answer is	"0":"negative", "1":"positive"
SUBJ	What is the subjectivity of the sentence x ? Print subjective or objective. The answer is	"0":"subjective", "1":"objective"
TREC	Print the category for the sentence x : description, entity, expression, person, location or quantity. The answer is	"0":"description","1":"entity", "2":"expression","3":"person", "4":"location","5":"quantity"
CoLA	Is the sentence x grammatically acceptable? Print no or yes. The answer is	"0":"no", "1":"yes"

B EXIT LAYERS

Table 8 compares the average exit layers of the RAEE method against two other method types across eight downstream tasks. Experimental results show that the RAEE method can exit earlier, thus reducing computational overhead during model inference. This result also aligns with the expectations in the motivation example. This suggests that the RAEE method can accurately approximate the gold exit layer distribution by using the retrieval database. Although AdaInfer exits earlier than the RAEE

Table 8: Exit layers of RAEE and different types of methods on 8 downstream tasks. The sum of the number of layers in the encoder and the decoder counts the number of layers for T5-large (Raffel et al., 2020).

Model	**SST-2**	**SST-5**	**MR**	**CR**	**MPQA**	**Subj**	**TREC**	**CoLA**	**Avg**
RB-L	24	24	24	24	24	24	24	24	24
EB-L	24	24	24	24	24	24	24	24	24
T5-L	48	48	48	48	48	48	48	48	48
Llama-3-8B	32	32	32	32	32	32	32	32	32
Gemma-7B	28	28	28	28	28	28	28	28	28
Backbone: RoBERTa-Large, ElasticBERT-Large									
DeeBERT	22.95	24.00	23.33	8.98	15.90	10.36	24.00	18.31	18.48
AdaInfer (RB-L)	1.00	0.00	1.46	1.00	18.00	1.10	0.00	4.00	3.32
RAEE (RB-L)	18.55	13.93	18.71	15.35	17.20	13.59	12.82	12.48	15.33
Backbone: T5-L									
AdaInfer (T5-L)	6.34	0.00	7.72	0.00	1.00	1.00	0.00	1.00	2.13
RAEE (T5-L)	22.27	18.74	21.88	26.84	18.05	19.06	27.29	18.55	21.59
Backbone: Llama-3-8B									
SLEB (Llama)	13.00	13.00	13.00	13.00	13.00	13.00	13.00	13.00	13.00
AdaInfer (Llama)	4.00	0.00	3.18	3.00	1.00	4.71	0.00	2.00	2.24
RAEE (Llama)	11.77	15.70	12.43	7.04	12.83	6.58	20.06	21.04	13.43
Backbone: Gemma-7B									
SLEB (Gemma)	11.00	11.00	11.00	11.00	11.00	11.00	11.00	11.00	11.00
AdaInfer (Gemma)	1.00	0.00	1.04	1.00	3.00	1.00	0.00	2.00	1.13
RAEE (Gemma)	11.00	17.62	11.70	3.29	14.72	0.51	9.50	20.06	11.05

method, it exhibits quite poor performance, as shown in Table 1. The reason may be that during the zero-shot inference scenario, the collected features can only provide limited information for the SVM, thus resulting in unstable prediction performance.

C Implementation Details

This section lists the implementation details.

- For DeeBERT(Xin et al., 2020), we use RoBERTa-Large as its backbone model. Since DeeBERT(Xin et al., 2020) is a classical entropy-thresholding-based early-exit method, it requires first fine-tuning the backbone model on the downstream task and then updating all but the last off-ramp, for a fair comparison, we only update the off-ramp in DeeBERT on each downstream task. We also use RoBERTa-large as the backbone model and train all off-ramps for 50 epochs (much larger than the default setting of 10 epochs). Other experimental settings for DeeBERT(Xin et al., 2020) remain as default.

- For CALM (Schuster et al., 2022), we use T5-Large (Raffel et al., 2020) as its backbone model. CALM (Schuster et al., 2022) is also a classical entropy-thresholding-based early-exit method, and we evaluate it under the zero-shot setting.

- For SLEB(Song et al., 2024), we use Llama-3-8b (Dubey et al., 2024) and Gemma-7B (Mesnard et al., 2024) as its backbone model. SLEB(Song et al., 2024) tackles the limitation of early exit methods by eliminating redundant transformer blocks. Since the proposed RAEE exits at about 40% layers, for a fair comparison, we also set the hyper-parameter `num_remove_blocks` of SLEB(Song et al., 2024) as int($60\% \cdot$ num_layers) for comparable efficiency.

D RETRIEVED EXAMPLES OF RAEE

We show two examples from the SST-2 task and their retrieved top-k data samples. As shown in Table 9 and Table 10, the retrieved samples are semantically similar to the query sentence, demonstrating the proposed RAEE's efficacy.

Table 9: Examples of data and corresponding retrieved data.

Query/Top-K	**Sentence**	**Label**
Query	although laced with humor and a few fanciful touches, the film is a refreshingly serious look at young women.	1
Top-1	the film is hard to dismiss – moody, thoughtful, and lit by flashes of mordant humor.	1
Top-2	the movie enters a realm where few non-porn films venture, and comes across as darkly funny, energetic, and surprisingly gentle.	1
Top-3	the movie, despite its rough edges and a tendency to sag in certain places, is wry and engrossing.	1
Top-4	metaphors abound, but it is easy to take this film at face value and enjoy its slightly humorous and tender story.	1
Top-5	it may not be particularly innovative, but the film's crisp, unaffected style and air of gentle longing make it unexpectedly rewarding.	1
Top-6	it has its faults, but it is a kind, unapologetic, sweetheart of a movie, and mandy moore leaves a positive impression.	1
Top-7	although frailty fits into a classic genre, in its script and execution it is a remarkably original work.	1
Top-8	unlike lots of hollywood fluff, this has layered, well-developed characters and some surprises.	1
Top-9	as broad and cartoonish as the screenplay is, there is an accuracy of observation in the work of the director, frank novak, that keeps the film grounded in an undeniable social realism.	1
Top-10	though its rather routine script is loaded with familiar situations, the movie has a cinematic fluidity and sense of intelligence that makes it work more than it probably should.	1
Top-11	it tends to remind one of a really solid woody allen film, with its excellent use of new york locales and sharp writing.	1
Top-12	though a touch too arthouse 101 in its poetic symbolism, heaven proves to be a good match of the sensibilities of two directors.	1

Table 10: Examples of data and corresponding retrieved data (Cond).

Query/Top-K	Sentence	Label
Query	... a boring parade of talking heads and technical gibberish that will do little to advance the linux cause.	0
Top-1	a vile, incoherent mess... a scummy ripoff of david cronenberg's brilliant 'videodrome.	0
Top-2	completely creatively stillborn and executed in a manner that i'm not sure could be a single iota worse... a soulless hunk of exploitative garbage.	0
Top-3	contrived, maudlin and cliche-ridden... if this sappy script was the best the contest received, those rejected must have been astronomically bad.	0
Top-4	could as easily have been called ' under siege 3: in alcatraz '... a cinematic corpse that never springs to life.	0
Top-5	little more than a stylish exercise in revisionism whose point...is no doubt true, but serves as a rather thin moral to such a knowing fable.	0
Top-6	a thoroughly awful movie – dumb, narratively chaotic, visually sloppy...a weird amalgam of 'the thing' and a geriatric scream.	0
Top-7	on a cutting room floor somewhere lies...footage that might have made no such thing a trenchant, ironic cultural satire instead of a frustrating misfire.	0
Top-8	...while certainly clever in spots, this too-long, spoofy update of shakespeare's macbeth does n't sustain a high enough level of invention.	0
Top-9	worthless, from its pseudo-rock-video opening to the idiocy of its last frames.	0
Top-10	comes across as a relic from a bygone era, and its convolutions... feel silly rather than plausible.	0
Top-11	a tired, unnecessary retread...a stale copy of a picture that was n't all that great to begin with.	0
Top-12	(less a movie than) an appalling, odoriferous thing...so rotten in almost every single facet of production that you'll want to crawl up your own in embarrassment.	0