Early Exit: A Natural Capability in Transformer-based Models Unveiling by Joint Optimization

Anonymous ACL submission

Abstract

Large language models (LLMs) exhibit exceptional performance across a variety of downstream tasks. However, they encounter limitations due to slow inference speeds stemming from their extensive parameters. The early exit (EE) approach, involving obtaining results from intermediate layers for each token before reaching the final layer, offers a promising solution for accelerating auto-regressive decoding. However, additional output layers and joint optimization used in conventional EE hinder the application of EE in LLMs. In this paper, we explore the possibility of LLMs EE without additional output layers and joint optimization. Our findings indicate that EE is a natural capability within Transformer-based models not only within LLMs. While joint optimization is not strictly necessary for EE capability, it must be employed to address challenges by 1) improving the accuracy of locating the optimal EE layer through gating functions and 2) mitigating key-value copy issues. Additionally, our study reveals patterns in EE behavior from subword and part-of-speech perspective based on the llama model, and the potential possibility for EE based on sub-layers.

1 Introduction

Recently, large language models have witnessed widespread adoption in Natural Language Processing (NLP) (Brown et al., 2020; OpenAI, 2023; Touvron et al., 2023a). Owing to their extensive parameter count, LLMs as decoder-only models exhibit remarkable performance across various downstream tasks, including dialogue, question answering (QA), text summarization (TS), and even machine translation (MT) (Brown et al., 2020). As the online utilization (OpenAI, 2023) and deployment of LLMs (Touvron et al., 2023a; Workshop et al., 2022; Zhang et al., 2022) become increasingly prevalent, the substantial scale gives rise to challenges related to unaffordable computation costs and latency during the generation process.

Facing the massive computation cost, inference relies on part of layers in a model known as early exit can make the inference faster and less expensive. The conventional early exit model adds multiple additional output layers, which are often called Internal Classifiers in ResNet, to the backbone model, as shown in Fig1(a) and (b). Meanwhile, the additional output layer requires a careful finetuning stage called joint optimization for good performance (Teerapittayanon et al., 2016). This early exit framework based on joint optimization is widely used in the subsequent studies (Elbayad et al., 2019; Schuster et al., 2021, 2022).

However, the aforementioned approach based on the conventional early exit framework is well-suited for relatively smaller models, while notably inadequate for LLMs with billions of parameters attributed to two key challenges: 1) Additional output layers significantly increase the model parameters. 2) Joint optimization for additional classifiers and the backbone model is computationally expensive and may degrade the ability of LLMs learned from the pre-trained stage based on the previous work (Xin et al., 2020, 2021).

Facing this challenge, a natural idea is to exit from all layers consistently using the original final output layer in LLMs and without joint optimization. We conducted experiments on the LLaMA2 (Touvron et al., 2023b) model with several test datasets, including WMT22 in machine translation, CNN Daily Mail in summarization, NarrativeQA in question answering. We find that at last 42.38% and maximum 65.56% token could generate the same result as the final output when exiting 10 layers before computing the last layer, as illustrated in Table 1.

We also extend this experiment on the Transformer-base and Bert models, yielding the same conclusion. This result proves that the Transformer-based model has the early exit capability inherently with the original final output layer,

Layers	WMT22	CNN DM	NQA
total	88.96	94.83	96.88
5	59.42	75.72	81.86
10	42.38	59.19	65.56
15	18.96	25.01	24.66
20	6.92	10.70	9.04

Table 1: Percent of tokens that can exit early to the total tokens during the Llama-2-Chat-7B(with 32 layers) reference. We note the CNN Daily Mail to CNN DM, WMT22 to WMT, NarrativeQA to NQA for brevity.

and this capability does not rely on joint optimization. However joint optimization is a critical factor for early exit application, we observed that early exit performs better with joint optimization, as it reduces the difficulty for the gating function to find the optimal early exit layer by improving the similarity of distribution from each layer.

Additionally, we observed that early exit proves to be challenging in LLMs due to the lack of localized attention information under auto-regressive decoding. The conventional copy key-value operation helps alleviate error propagation during inference to some extent in relatively short sentences but faces challenges for the long sentence inference in LLMs.

our main contributions are listed as follows:

- We first propose that the early exit capability is inherent in the Transformer-based model and does not rely on joint optimization based on extensive experiments.
- We investigated the impact of joint optimization in early exit and noted that it contributes to the gating function by enhancing the similarity between the hidden states. Without joint optimization, the early exit capability has not disappeared, but an approximate early exit layer is hard to find. Nevertheless, identifying the early exit layer remains a challenging task even with joint optimization.
- Early exit can be easily applicable in the GLUE benchmark, but difficult in generation tasks that rely on auto-regressive decoding. Conventional copy key-value operation performs well in short sentences but exhibits limitations in LLMs in long sentence scenarios.

2 Related Work

Early Exit : Early exit (EE) is motivated by a hypothesis that certain samples are easier to predict and require less computation costs (Panda et al., 2016; Schwartz et al., 2020; Elbayad et al., 2019). Implemented within the framework of joint optimization (Teerapittayanon et al., 2016), early exit is applied to both Bert (Xin et al., 2020; Schwartz et al., 2020) and Transformer (Elbayad et al., 2019) models with a learnable depth estimator or estimate the required depths in advance (Liu et al., 2021). Although joint optimization leads a multi-exit model(or called Once-for-All model (Cai et al., 2019)), two notable weaknesses of this framework are frequently addressed: 1) Joint optimization is challenging, as the loss from early classifiers may interfere with later classifiers. Some researchers adapt dense connectivity (Huang et al., 2017) or two-stage fine-tuning (Xin et al., 2020, 2021) to fix this problem. 2) Early additional classifiers may not output a good enough result. Facing this problem, Liao et al. (2021) propose to combine both the past and future states from early and future layers to enhance current layer performance. Zhou et al. (2020) determining exit layer by additional classifiers continuous output the same result n times. Sun et al. (2021) boost performance through ensemble methods involving multiple additional classifiers. Except for the impact of joint optimization, estimating the exit layer by comparing confidence scores from the classifier and a hyperparameter threshold may not always yield satisfactory results with additional classifiers. To overcome this, Schuster et al. (2021, 2022) propose to employ conformal prediction for tuning the threshold to achieve well-calibrated predictions. (Gao et al., 2023) (Sun et al., 2022)

In the context of early exit research based on LLMs, Kavehzadeh et al. (2023) proposes to replace the standard supervised fine-tuning in joint optimization with sorted fine-tuning lead to a more flexible LLaMa2 model based on the SortedNet (Valipour et al., 2023). In the token-level early exit scenario, where the previous token exits earlier when generating the current token, a copied Key-Value caching must be used for self-attention for the current token(detail in 3.2). To avoid copying Key-Value caching and effectively use batched inference, Del Corro et al. (2023) proposes to skip the early layer and exit only after the final layer. Additionally, Bae et al. (2023) real-

izes a fast and robust early exit model by parallel decoding (Leviathan et al., 2023) to avoid the KV caching problem.

Saturation Events : Recently, a phenomenon called saturation events has provided strong support for early exit research, which the final output prediction is consistently in the top-ranked prediction launched by the Feed-Forward Networks (FFN) in hidden layers and with increasing rank (Geva et al., 2020). This implies that as the model uses more layers, it gains more confidence, meanwhile the correct output has already been predicted by earlier layers, this phenomenon has motivated the CALM (Schuster et al., 2022). Moreover, it raises the possibility that the similarity between hidden states or distributions from each layer could be helpful for generating predictions (Chuang et al., 2023). The saturation events have further motivated us research on early exit, but it is worth noting that we obtain the output after residual connection not the FFN update (Geva et al., 2020), and we found similar saturation events in various tasks and transformer-based model. Additionally, we pay more attention on could we find the early exit layer under saturation events and do we need the joint optimization.

3 Background

3.1 Large Language Models

Large language model constructed based on Transformer decoder architecture with N blocks (Vaswani et al., 2017), each block consists of three sub-layers multi-head self-attention(attn), feedforward network(ffn) and layer normalization, which commonly based on the pre-norm architecture. The final hidden state of model can be represented by $\mathbf{h}_{\text{final}_t}^{N} = \text{LN}_{\text{final}}(\mathbf{h}_{\text{ffn}_t}^{N})$, in which:

$$\mathbf{h}_{\text{attn}_t}^{\ell} = \text{ATTN}(\mathbf{h}_{\text{ffn}_t}^{\ell-1}) + \mathbf{h}_{\text{ffn}_t}^{\ell-1} \quad (1)$$
$$\mathbf{h}_{\text{ffn}_t}^{\ell+1} = \text{FFN}(\mathbf{h}_{\text{attn}_t}^{\ell}) + \mathbf{h}_{\text{attn}_t}^{\ell},\ \ell \in N$$

Where, $\mathbf{h}_{\text{ffn}_t}^{\ell+1}$ is the ℓ-th block output at the t-th token, $\mathbf{h}_t^{\ell}$ is the output of block ℓ for token t, for brevity we omitted all layer normalization in model. Based on the $\mathbf{h}_t^N$, model will give a prediction by an output layer called language model head(lm_{head}):

$$p(y_t^N) = \text{Softmax}(\mathbf{W}_N^{\intercal}\mathbf{h}_{\text{final}_t}^{N}) \quad (2)$$

3.2 Early Exit

Training : The implementation of early exit involves the attaching multiple additional classifiers to the backbone model, commonly referred to as the output layer in the Transformer model or the language model head in LLMs. Then the new model will be optimized by joint optimization, wherein the sum of all loss functions is used to collectively optimize both the additional output layer and the backbone model, represented as $Loss = \sum_{i=1}^{N} Loss_i$. However, this pipeline often results in undesirable parameter growth and poses challenges in the optimization process.

Inference : To obtain the ideal exit layer for each token during the inference stage, most works decide the exit layer by comparing a value derived from the output layer with a threshold τ, which is sensitive to the distribution output from each layer and an additional fine-tuning stage.

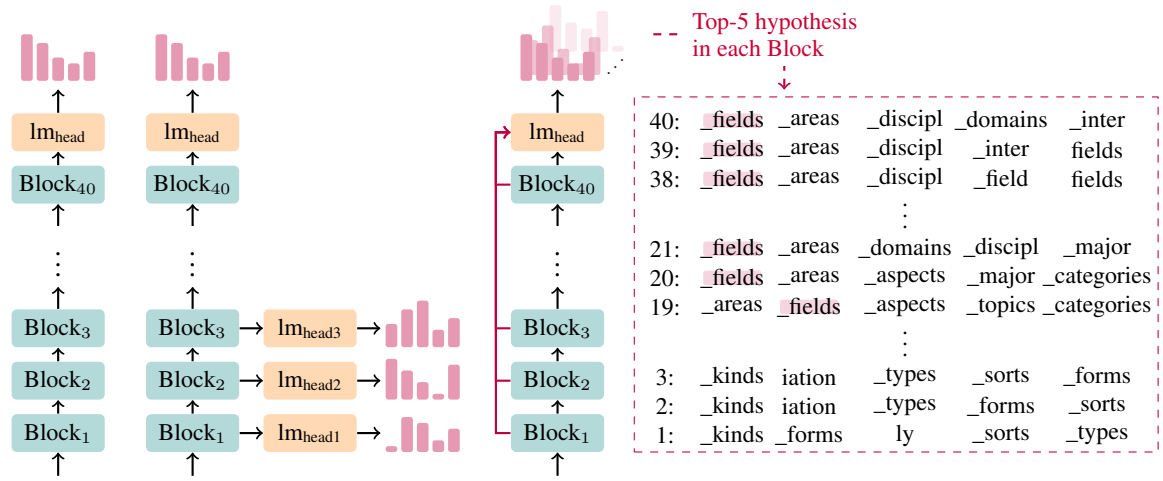
- Confidence-based (Liao et al., 2021): Exit at layer i when $\max p(y_t^i) > \tau$.
- Entropy-based (Xin et al., 2020): Exit at layer i when $entropy(p(y_t^i)) > \tau$.
- Patience-based (Zhou et al., 2020): Exit at layer i when $\forall\{\max p(y_t^i), \ldots, \max p(y_t^{i-\tau})\}$ is the same token(take Top-1 sampling for example).

During the inference stage, the model computes layer by layer, and upon reaching the target exit layer i, the model outputs the result $p(y_t^i)$ through an additional output layer $\text{lm}_{\text{head}i}$.

Copying the KV Cache : While this flexible strategy enhances potential decoding efficiency, it introduces a significant challenge for auto-regressive decoding. When we predict $p(y_t^i)$, self-attention operation requires access to all $\mathbf{K}_{<t}^{i-1}$ and $\mathbf{V}_{<t}^{i-1}$. Unfortunately, a likely scenario arises where some token $s, s < t$ may exit at layer $j(j < i-1)$, implying that obtaining $\mathbf{K}_s^{i-1}$ can only be based on $\mathbf{K}_s^{>j} = \mathbf{K}_s^j$ or $\mathbf{K}_s^{>j} = \text{ATTN}(\mathbf{h}_{\text{ffn}_s^j})$. This scenario poses a challenge as it limits the availability of information for the auto-regressive decoding process.

3.3 Experimental Setup

We experiment on three types of Transformer-based model, including the BERT and RoBERTa(encoder-only) model, the Transformer-base(encoder-decoder) model, and the Llama2(decoder-only)



(a) Llama (b) - with conventional EE (c) - with shared final output layer

Figure 1: block saturation event based on the Llama-2-Chat-13B when doing translation task on WMT22-zh2en test set, we found that the top-1 hypothesis is same with the final top-1 output far away from the final layer. Each token is selected by greedy search based on the final layer output($Block_{40}$ in Figure) while showing the Top-5 output from each block.

model. All experiments were conducted using the Transformers and Fairseq toolkit based on the GeForce RTX 3090 * 8, and A800 80GB * 4 only for Llama2 fine-tuning.

Dataset We select GLUE benchmark for bert-like model and use toolkit *glue_compute_metrics* for Acc/F1. For Transformer-base model we do experiment on WMT14 DE2EN translation tasks and compute BLEU and COMET following (Wang et al., 2019; Zheng et al., 2023). For Llama model we do experiment on three generation tasks including WMT22 translation tasks, CNN_DM, and NarrativeQA, and we select COMET (Rei et al., 2020) with `Unbabel/wmt22-comet-da` and ROUGE-L (Lin, 2004) as the metrics separately.

Model We verify the early exit capability on the pre-trained backbone model and the pre-trained model with joint optimization separately. The backbone model training stage we follow the work which is widely accepted[1] (Wang et al., 2019; Elbayad et al., 2019). For joint optimization stage we following (Teerapittayanon et al., 2016; Taori et al., 2023) without additional output layer which means we obtain output from each layer all based on the final output layer. During decoding stage, the prompt we used in LLMs is listed in A.2 and select Top-k sampling for all generation tasks.

4 Early Exit in LLMs

Adding multiple internal output layers is a conventional approach to implement an early exit model. However, this approach is often hindered by substantial computation costs caused by joint optimization and redundancy in model parameters, which scale linearly with the vocabulary size and number of layers in LLMs. Using the final output layer at every exit position directly becomes a reasonable alternative (Del Corro et al., 2023). Additionally, early exit often involve constraining the number of internal output layers (Kavehzadeh et al., 2023) or setting a fixed exit points through methods such as parallel decoding (Bae et al., 2023) in LLMs. This challenge has motivated our interest in investigating the compatibility of early exit with shared and pre-trained final output layers without any fine-tuning stage. Accordingly, we aim to find the optimal early exit layer, which represents the upper boundary of early exit in LLMs.

Optimal Early Exit Layer To figure out the early exit behavior and not interference from other factors during decoding stage. We use the `Llama-2-7b-chat`[2] and `Llama-2-13b-chat`[3] model in ten language pairs in WMT22 machine translation tests, NarrativeQA in question answering, and CNN/DailyMail

[1] https://github.com/huggingface/transformers/tree/v4.37.2/examples/pytorch/text-classification

[2] https://huggingface.co/meta-llama/Llama-2-7b-chat

[3] https://huggingface.co/meta-llama/Llama-2-13b-chat

in summarization tasks. under the following conditions. We obtained outputs from all hidden layers using Top-1 sampling after employing the shared and pre-trained final output layer, and without joint optimization. Meanwhile, we generate next token based on the final layer output in each decoding step, denoted as $\mathcal{F}(y_{t+1}^{l}|\theta, y_t^N)$. We naturally did not copy KV Cache to future layers to prevent the error propagation during auto-regressive decoding. The optimal early exit layer was recognized when the current layer output matched the final layer output.

Results Remarkably, our experiments demonstrate that the output of intermediate layers starts to match the oracle final output before reaching the final layer at last ten layers across various tasks2. This matching is not limited to just top-1 hypotheses, as illustrated by an example from the wmt22-ZH2EN test shown in figure1 (c). This observation underscores the potential for early exit in LLMs based on the shared final output layer without the need for joint optimization.

4.1 Early Exit Capability in Various Transformer-based Model

Our experiment reveals a vast and not utilized early exit space to accelerate the decoding stage, particularly for expensive auto-regressive generation tasks based on the widely used Top-1 sampling in LLMs. Motivate by this phenomenon, a nature question is whether the phenomenon is universal or only occur in decoder-only model like LLMs?

Experiment We investigate this universalization based on multiple models and tasks which the models have an entirely different structures. Including the Transformer-base model employed in the WMT14-DE2EN machine translation task, RoBERTa in the General Language Understanding Evaluation (GLUE) benchmark.

Results We found the average early exit layer(avg) and the token percentage that can early exit(perc) based on the shared and pre-trained final output layer is very stable across different models and tasks, as shown in Table 3 and 9. This suggests that the capability for early exit is a natural feature inherent in pre-trained models and is not exclusive to LLMs but also extends to Bert-like models and Transformer models. Furthermore, the phenomenon illustrated in Figure1 (c), where the final-right token consistently ranks up not only in the Top-10 hypothetical list in each block but also repeatedly at the Top-1 rank in many internal layers, appears to represent a stronger saturation phenomenon(referred to as `block saturation`). Which suggests a stronger consistency in the results produced by each hidden layer that can be leveraged by distribution-sensitive gating functions.

Model	layer	WMT22		
		COMET-22	avg	perc
7B	**32**	79.68\79.71	**23.49**	88.23%
13B	**40**	80.65\80.68	**25.5**	92.43%
Model	layer	CNN_DM		
		Rough-1\-2\-L	avg	perc
7B	**32**	20.39\8.07\19.76	**21.11**	94.83%
13B	**40**	20.45\8.12\19.82	**24.29**	64.45%
Model	layer	NarrativeQA		
		Rough-1\-2\-L	avg	perc
7B	**32**	25.16\11.05\23.84	**20.67**	96.88%
13B	**40**	24.83\11.05\23.62	**23.47**	71.72%

Table 2: The optimal early exit layer in Llama-2-Chat-7B(7B) and Llama-2-Chat-13B(13B). We report the average result in wmt22 for simple, which avg is the average optimal early exit layer, and per is the percentage of token which can early exit in all token. We list the result in each language pair in detail at A.1.

Task	BLEU	COMET-22	avg	perc	layer
WMT	26.75	83.86	**4.92**	57.25%	6-**6**
IWSLT	32.18	70.53	**4.18**	80.01%	6-**6**

Table 3: The optimal early exit layer for decoder in Transformer-Base model on WMT14 EN2DE(WMT) and IWSLT14 DE2EN(IWSLT) based on Top-1 sampling.

5 Can Early Exit Capability Be Used Directly

Motivating Based on the early exit capability, a natural question arises: 1) can we take advantage of this capability directly to enhance decoding efficiency, and 2) can the gating function employed in previous works accurately identify the earliest layer. We experiment on GLUE benchmark based on the bert and roberta model in a simpler decoding scenario which not involve the kv cache copy operation in auto-regressive decoding.

Experiment For fair comparison, we conducted experiments based on the BERT and RoBERTa model in the GLUE benchmark, which involves tasks that do not follow auto-regressive decoding,

Model	layer	Metric	CoLA	MRPC	QNLI	QQP	RTE	MNLI	SST-2
RoBERTa	**12**	F1/Acc	56.24(Mcc)	91.28/88.23	93.09	91.32/88.50	72.56	87.76	94.26
		Avg	**3.17**	**1.32**	**5.44**	**5.5**	**8**	**7.01**	**3.83**
		Perc	81.11%	100%	92.97%	99.82%	77.26%	98.68%	100%
BERT	**12**	F1/Acc	56.49(Mcc)	87.32/82.60	91.61	91.12/88.07	70.03	84.87	92.88
		Avg	**3.11**	**2.88**	**2.87**	**2.04**	**4.34**	**6.46**	**4.67**
		Perc	93.1%	93.68%	99.65%	100%	96.39%	93.4%	99.77%

Table 4: The optimal early exit layer in roberta-base and BERT-base-uncased model.

Model	layer	Metric	CoLA	MRPC	QNLI	QQP	RTE	MNLI	SST-2
RoBERTa	12	Spd-up	2.52×	3.52×	1.8×	1.76×	2.07×	1.48×	2.43×
BERT	12	Spd-up	2.71×	2.77×	2.69×	3.18×	2.69×	1.63×	2.17×

Table 5: The saturation event in encoder-decoder model and encoder-only model.

such as classification tasks. In the case of generation tasks, we adapt token-level early exit in Transformer-base model and Llama2 model while avoiding the key-value copying operation. The backbone model was obtained following previous works[4] (Wang et al., 2019; Elbayad et al., 2019) and the model after joint optimization without additional output layer follow the BranchyNet (Teerapittayanon et al., 2016), with the loss defined as $Loss = \sum_{i=0}^{N} w_i \cdot Loss_i$. In terms of the gating function, we compare three types distribution-sensitive gating functions which is not limited by the model structure mentioned above.

Results In an ideal situation, exploiting the early exit capability in the BERT-like model is notably straightforward. Based on the optimal exit layer, approximate 2× speedup can be achieved directly on the pre-trained backbone model without any modifications and performance degradation, result presented in Table 5. However, we find identifying the optimal early exit layer precisely is formidable challenge. Based on three gating functions at the optimal thresholds, we find the distance between the early exit layer and the optimal exit layer in the pre-trained model is greater than the model after joint optimization, and using gating functions directly in the pre-trained backbone model seems unable to bring acceleration actually, as shown in Fig 2.

[4]https://github.com/huggingface/transformers/tree/v4.37.2/examples/pytorch/text-classification

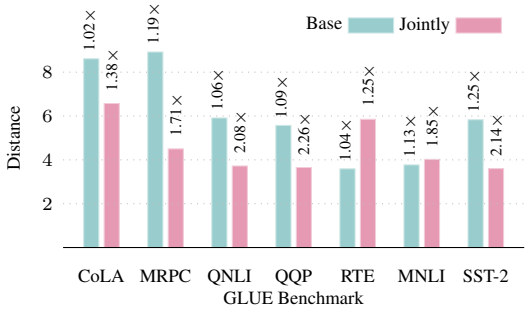


Figure 2: The average distance and speed-up between the optimal early exit layer and the exit layer from the three gating functions in RoBERTa model(more details in B.1). We constrain the performance of early exit not less than 98% original model to obtain the optimal threshold for gating functions.

5.1 Why is Joint Optimization Helpful for Early Exit

Motivating While the distance and speed-up between the optimal early exit layer and the exit layer from the gating function vary with the threshold, we found the model with joint optimization always leads a layer closer to the optimal early exit layer not only in BERT-like modelB.1 but also in the Transformer model and the Llama2 modelB.2 with more experiments. This makes us curious about why joint optimization is helpful for early exit.

Experiment We employ four distinct similarity measures to assess the likeness of the hidden state and the output distribution between each layer, including Kullback-Leibler (KL) divergence and Jensen-Shannon (JS) divergence for evaluating distribution similarity, while cosine similarity and Pearson correlation coefficient are employed to compute hidden state similarity.

Results Our observations reveal that the similarity in hidden state exhibits no significant trend after joint optimization. In contrast, there is a notable and intuitive increase in the similarity of output distributions across layers, which shown as Fig3. This reinforcement in distribution similarity enhances the consistency of output results, particularly in classification tasks with fewer category labels. Meanwhile, this distribution with more similarity drives a closer confidence level which more benefit to finding sufficiently confident outputs by fixing thresholds, shown as Fig3. This underscores the dependency of the distribution-sensitive gating functions on joint optimization. Additionally, we note a reduction in the average early exit layer when model with joint optimization.

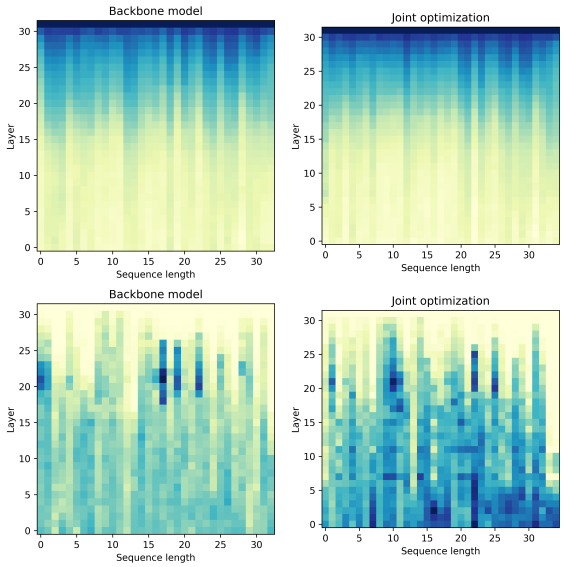


Figure 3: Example for cosine similarity of hidden state(up) and Js divergence of distribution (down) between each layer on same sentence.

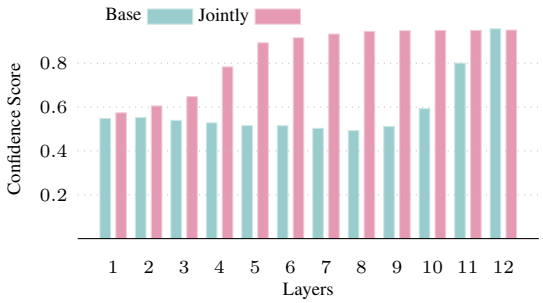


Figure 4: The average confidence score between each layer over all sample in the GLUE benchmark based on the RoBERTa model.

6 Saturation Events for Early Exit in LLMs

Motivating Although finding the optimal early exit layer is challenging directly on the pre-trained transformer-based model, it becomes easier after joint optimization. Consequently, we explore could the early exit capacity lead a actual acceleration effect in the token-level early exit scenario based on the joint optimization further.

Experiment We fine-tuning the Llama2 model based on the joint optimization following (Taori et al., 2023). Following the describe in the sec 3.2, token-level early exit for generation tasks need copy kv cache. We execute two type of kv cache including copy kv cache directly (Elbayad et al., 2019), copy $\mathbf{h}_{\text{ffn}_t}^{\ell-1}$ and recompute K and V (Schuster et al., 2022).

Results In the token-level early exit scenario, our observations indicate that early exit based on the optimal early exit layer is effective for shorter sentences. It successfully yields correct target sentences at notably low exit layers when recomputing the key-value (kv) cache. However, the direct copying of the kv cache tends to lead the model into local optima, even with joint optimization, as illustrated in Table 10. Notably, as the model starts early exit, the occurrence of local optima becomes more frequent with the target sentence length increases. Once fall into the local optima, generating the desired hypothesis becomes challenging, and the local optima is hard to avoids by existing copy kv methods11.

6.1 Trend In the Optimal Early Exit Layer

We conducted a statistical analysis of the optimal early exit layer under varying output lengths, as illustrated in Fig. 5. Our observations revealed a gradual decline in the optimal early exit layer with an increase in the length of the output sequence, particularly notable in the shorter sentences. This pattern suggests a potential reduction in the difficulty of generation, consistent with the decreasing loss presented in (Del Corro et al., 2023). However, beyond a certain length, the optimal early exit layer exhibited a slow and more unsteady descent.

To understand the reasons behind the observed unsteady patterns, we conducted a detailed analysis from both sub-word and part-of-speech perspective. A notable trend emerged in the X-to-English translation direction: Approximately 12% of to-

	SRC	Die Ware hat unter 20 Euro gekostet.
	hypothesis	optimal early exit layer
7B	The item cost less than 20 euros.</s>	[31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31]
7B-d	The item cost less than 20 euphemia.</s>	[31, 31, 31, 31, 6, 28, 17, 16, 20, 31, 20, 30, 15]
7B-c	The item cost less than 20 euros.</s>	[31, 31, 31, 31, 6, 29, 17, 9, 16, 1, 22, 13]
7B-j	The item cost less than 20 euros.</s>	[31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31]
7B-j-d	The item cost less than 20,0 - notes, your chadge, a, and a more, and. . .	[31, 31, 31, 31, 4, 15, 6, 6, 30, 4, 21, 31, 29, 31, 31, 30, 31, 18, 8, 31, 15, 9, 31, 31, 15,...]
7B-j-c	The item cost less than 20 euros.</s>	[31, 31, 31, 31, 4, 7, 6, 6, 16, 1, 11, 7]

Table 6: Token-level early exit result and exit layer of Llama-2-Chat-7B(7B) with(-J) and without joint optimization on WMT22-DE2EN test set, under the constraint that early exit only after the 4-th token is generated. For copy kv operation, we represent directly copy as -d and recompute as -c.

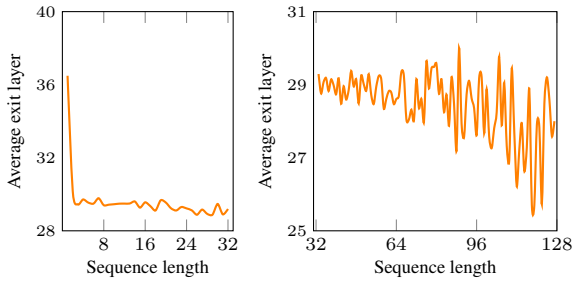


Figure 5: The relationship between the average early exit layer and sequence length based on the Llama-2-Chat-13B model in WMT22 translation tasks.

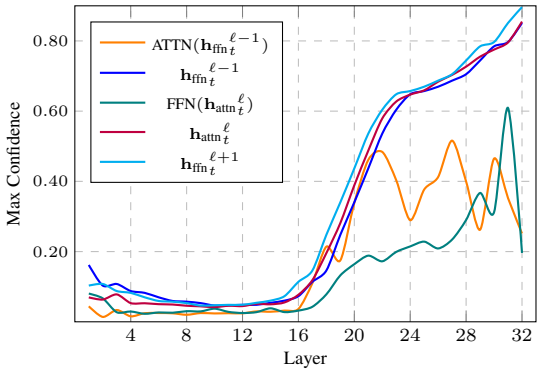


Figure 6: the Top-10 hypothesis launched by sub-layer in WMT22DE2EN.

kens contribute to forming a complete word in all decoding tokens, and the first part of a word tends to exit in deeper layers, while the remaining part exits earlier, as illustrated in . Concurrently, our examination identified distinctions among various part-of-speech categories, as illustrated in . These result underscores the potential for discerning early exit layers from a linguistic standpoint.

6.2 Early Exit and Sub-layer

We also attempt to extract the output from sub-layers, ffn module and the attn module, inspired by relevant literature (Geva et al., 2022). Our findings indicate that both the confidence score and the output token within top-10 hypotheses from skip connection is stable. Conversely, the top-10 hypotheses from module consistently demonstrate substantial variations and notably smaller confidence scores compared to the skip connection, as depicted in Fig 6(An example in C.3). This can be approximated as the primary hypotheses being preserved in the skip connect, while the residual branch incrementally incorporates the most confident hypotheses into the primary branch layer by layer according to (Geva et al., 2022), while we must omit the Softmax operation.

$$\mathbf{W}_N^\intercal \mathbf{h}_t^\ell = \mathbf{W}_N^\intercal \mathcal{F}(\mathbf{h}_t^{\ell-1}) + \mathbf{W}_N^\intercal \mathbf{h}_t^{\ell-1} \quad (3)$$

Further, we decode the same token and keep the decoding process exactly same with Figure 1 (c), and enumerated all top-10 hypotheses from each layer, as shown in Table 10. We find the consistent top-1 hypothesis not only within the block output but also across the skip connection.

7 Conclusion

Based on previous experiments, we found 1) the early exit is a natural capacity within Transformer-based models. However, leveraging this capability directly proves challenging. 2) The joint optimization approach reduces the optimal early exit layer searching difficulty by improving the hidden similarity. 3) While copy kv operations demonstrate efficacy in short sentences, their performance significantly diminishes when confronted with longer sentences. 4) Early exit based on the sub-word and sub-layer has the potential work will in the LLMs.

8 Limitations

While we do extensive experimentation with the Llama model, our research is currently constrained by the limitations of our available equipment, which has restricted us to a finite set of models. We look forward to expanding our experiment across a broader range of models in more resource scenarios, ensuring that our findings can be generalized to a wider array of environments.

References

Sangmin Bae, Jongwoo Ko, Hwanjun Song, and Se-Young Yun. 2023. Fast and robust early-exiting framework for autoregressive language models with synchronized parallel decoding. *arXiv preprint arXiv:2310.05424*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Han Cai, Chuang Gan, Tianzhe Wang, Zhekai Zhang, and Song Han. 2019. Once-for-all: Train one network and specialize it for efficient deployment. *arXiv preprint arXiv:1908.09791*.

Yung-Sung Chuang, Yujia Xie, Hongyin Luo, Yoon Kim, James Glass, and Pengcheng He. 2023. Dola: Decoding by contrasting layers improves factuality in large language models. *arXiv preprint arXiv:2309.03883*.

Luciano Del Corro, Allie Del Giorno, Sahaj Agarwal, Bin Yu, Ahmed Awadallah, and Subhabrata Mukherjee. 2023. Skipdecode: Autoregressive skip decoding with batching and caching for efficient llm inference. *arXiv preprint arXiv:2307.02628*.

Maha Elbayad, Jiatao Gu, Edouard Grave, and Michael Auli. 2019. Depth-adaptive transformer. *arXiv preprint arXiv:1910.10073*.

Xiangxiang Gao, Wei Zhu, Jiasheng Gao, and Congrui Yin. 2023. 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.

Mor Geva, Avi Caciularu, Kevin Ro Wang, and Yoav Goldberg. 2022. Transformer feed-forward layers build predictions by promoting concepts in the vocabulary space. *arXiv preprint arXiv:2203.14680*.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2020. Transformer feed-forward layers are key-value memories. *arXiv preprint arXiv:2012.14913*.

Gao Huang, Danlu Chen, Tianhong Li, Felix Wu, Laurens Van Der Maaten, and Kilian Q Weinberger. 2017. Multi-scale dense networks for resource efficient image classification. *arXiv preprint arXiv:1703.09844*.

Parsa Kavehzadeh, Mojtaba Valipour, Marzieh Tahaei, Ali Ghodsi, Boxing Chen, and Mehdi Rezagholizadeh. 2023. Sorted llama: Unlocking the potential of intermediate layers of large language models for dynamic inference using sorted fine-tuning (soft). *arXiv preprint arXiv:2309.08968*.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2023. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning*, pages 19274–19286. PMLR.

Kaiyuan Liao, Yi Zhang, Xuancheng Ren, Qi Su, Xu Sun, and Bin He. 2021. 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: Human Language Technologies*, pages 2013–2023.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Yijin Liu, Fandong Meng, Jie Zhou, Yufeng Chen, and Jinan Xu. 2021. Faster depth-adaptive transformers. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 15, pages 13424–13432.

OpenAI. 2023. Gpt-4 technical report.

Priyadarshini Panda, Abhronil Sengupta, and Kaushik Roy. 2016. Conditional deep learning for energy-efficient and enhanced pattern recognition. In *2016 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 475–480. IEEE.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. Comet: A neural framework for mt evaluation. *arXiv preprint arXiv:2009.09025*.

Tal Schuster, Adam Fisch, Jai Gupta, Mostafa Dehghani, Dara Bahri, Vinh Tran, Yi Tay, and Donald Metzler. 2022. Confident adaptive language modeling. *Advances in Neural Information Processing Systems*, 35:17456–17472.

Tal Schuster, Adam Fisch, Tommi Jaakkola, and Regina Barzilay. 2021. Consistent accelerated inference via confident adaptive transformers. *arXiv preprint arXiv:2104.08803*.

Roy Schwartz, Gabriel Stanovsky, Swabha Swayamdipta, Jesse Dodge, and Noah A Smith. 2020. The right tool for the job: Matching model and instance complexities. *arXiv preprint arXiv:2004.07453*.

Tianxiang Sun, Xiangyang Liu, Wei Zhu, Zhichao Geng, Lingling Wu, Yilong He, Yuan Ni, Guotong Xie, Xuanjing Huang, and Xipeng Qiu. 2022. A

simple hash-based early exiting approach for language understanding and generation. *arXiv preprint arXiv:2203.01670*.

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

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca.

Surat Teerapittayanon, Bradley McDanel, and Hsiang-Tsung Kung. 2016. Branchynet: Fast inference via early exiting from deep neural networks. In *2016 23rd international conference on pattern recognition (ICPR)*, pages 2464–2469. IEEE.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023a. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Mojtaba Valipour, Mehdi Rezagholizadeh, Hossein Rajabzadeh, Marzieh Tahaei, Boxing Chen, and Ali Ghodsi. 2023. Sortednet, a place for every network and every network in its place: Towards a generalized solution for training many-in-one neural networks. *arXiv preprint arXiv:2309.00255*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Qiang Wang, Bei Li, Tong Xiao, Jingbo Zhu, Changliang Li, Derek F Wong, and Lidia S Chao. 2019. Learning deep transformer models for machine translation. *arXiv preprint arXiv:1906.01787*.

BigScience Workshop, Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, et al. 2022. Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*.

Ji Xin, Raphael Tang, Jaejun Lee, Yaoliang Yu, and Jimmy Lin. 2020. Deebert: Dynamic early exiting for accelerating bert inference. *arXiv preprint arXiv:2004.12993*.

Ji Xin, Raphael Tang, Yaoliang Yu, and Jimmy Lin. 2021. Berxit: Early exiting for bert with better fine-tuning and extension to regression. In *Proceedings of the 16th conference of the European chapter of the association for computational linguistics: Main Volume*, pages 91–104.

Haoran Xu, Young Jin Kim, Amr Sharaf, and Hany Hassan Awadalla. 2023. A paradigm shift in machine translation: Boosting translation performance of large language models. *arXiv preprint arXiv:2309.11674*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. Opt: Open pre-trained transformer language models.

Tong Zheng, Bei Li, Huiwen Bao, Weiqiao Shan, Tong Xiao, and Jingbo Zhu. 2023. Partialformer: Modeling part instead of whole. *arXiv preprint arXiv:2310.14921*.

Wangchunshu Zhou, Canwen Xu, Tao Ge, Julian McAuley, Ke Xu, and Furu Wei. 2020. Bert loses patience: Fast and robust inference with early exit. *Advances in Neural Information Processing Systems*, 33:18330–18341.

A Detailed Experimental

A.1 LLaMa2 Translation Result in WMT22 Testset

We list the LLaMa2 translation result on the WMT22 test set in detail based on the template 8, shown as Table 7

Model		Llama7B	Llama13B
Total layer		**32**	**40**
DE→EN	acc	86.53 \ 87.2	87.16 \ 87.85
	avg	**22.44**	**22.62**
	perc	92.97	96.5
DE→FR	acc	76.62	79.31
	avg	**24.67**	**26.74**
	perc	88.62	93.04
EN→DE	acc	79.97 \ 79.86	81.86 \ 81.57
	avg	**24.47**	**26.96**
	perc	88.46	92.42
EN→JA	acc	74.56	73.69
	avg	**23.34**	**25.39**
	perc	85.44	89.6
EN→RU	acc	77.65	77.24
	avg	**24.35**	**26.81**
	perc	85.1	89.64
EN→ZH	acc	77.47 \ 78.48	77.11 \ 78.15
	avg	**23.12**	**26.57**
	perc	85.16	87.64
FR→DE	acc	77.31	80.57
	avg	**24.42**	**26.71**
	perc	89.36	92.75
JA→EN	acc	80.31	81.49
	avg	**22.75**	**25.05**
	perc	91.65	93.95
RU→EN	acc	85.81	86.51
	avg	**22.6**	**23.73**
	perc	92.06	95.19
ZH→EN	acc	80.54 \ 79.28	81.59 \ 80.42
	avg	**22.8**	**24.45**
	perc	90.83	94.29

Table 7: Detailed result in WMT22 translation task, `acc` is the accuracy of predicting compared with ref.A \ref.B, and we use comet as evaluation metric based on the `Unbabel/wmt22-comet-da`. `avg` and `perc` present the average minimal early exit layer per token and the percent of all decoding tokens which arise saturation event. We note Llama-2-Chat-7B as Llama7B, Llama-2-Chat-13B as Llama13B, Bigtranslation as bigtrans for simple.

A.2 Template For LLaMa2 Inference

We use the template list in Tabel 8. To verify our decoding experiment we try the ALMA style prompt and keep all other setting, and we got the same result with the LLaMA-2-7B(zero-shot) reported in paper (Xu et al., 2023), and we obtain the same result with the paper. In our experiment, we only change the demonstration in our prompt and not change other hypoparameters.

B Result Of Gating Function on the Backbone model

B.1 BERT And RoBERTa Model

we shown the distance and speed-up between the optimal early exit layer and the exit layer in detail, the threshold chosen to constrain the performance of early exit not less than 98% of the original model, is illustrated in Fig 7 and 8. Across almost all tasks, joint optimization significant enhances the accuracy of the gating function by improving the similarity of distribution output from each layer. While the gap is relatively small in the patience-based gating function, this can be attributed to the gating function exiting only at n-th continuous layer, which yields the same result, leading to a deeper exit layer and a more accurate result normally. Additionally, a noteworthy observation is the decline in the average optimal early exit layer across most tasks after joint optimization, signifying an improvement in the upper bound of early exit.

B.2 Transformer And LLaMa2 Model

We performed the same experiment on Transformer and LLaMa2 model described as B.1.

C Trend In The Optimal Early Exit Layer

C.1 Sub-word

C.2 Part-of-speech

C.3 Top-10 Hypotheses From Sub-layer And Module

We decode the same token and keep the decoding process exactly same with Figure 1 (c), and enumerated all top-10 hypotheses from each layer which consistently producing the same top-1 output as the final layer, as shown in Table 10. We find the consistent top-1 hypothesis from 20-th layer to 40-th layer, not only within the block output but also across the skip connection. However, the final output *_fields* rarely surfaced in the top-10 hypotheses from the module output, but it appear occasionally and improve the rank of final output in the top-10 hypotheses like the *_fields* in $\text{FFN}(\mathbf{h}_{\text{attn}_t^{23}})$.

D Copy KV Cache In longer sentence

Tasks	System Prompt
Translation	### Instruction: Translate `src` to `tgt`: ### Input: `real input` ### Response:
Summarization	### Instruction: Summarize the following article to a sentence: ### Input: `real input` ### Response:
Question Answering	### Instruction: I will provide a context and a question to you. You need to answer me the question based on the context. ### Context: `The context` ### Question: `Question` ### Answer:

Table 8: Prompts for generation task. For translation tasks, `src` and `tgt` is select from {English, Chinese, German, Russian, French, Japanese}, and the `real input` is the sentence to be translated

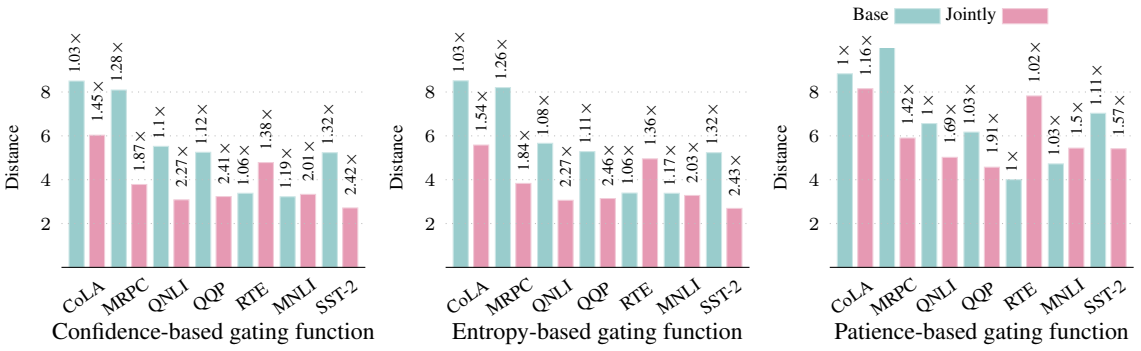


Figure 7: Roberta model.

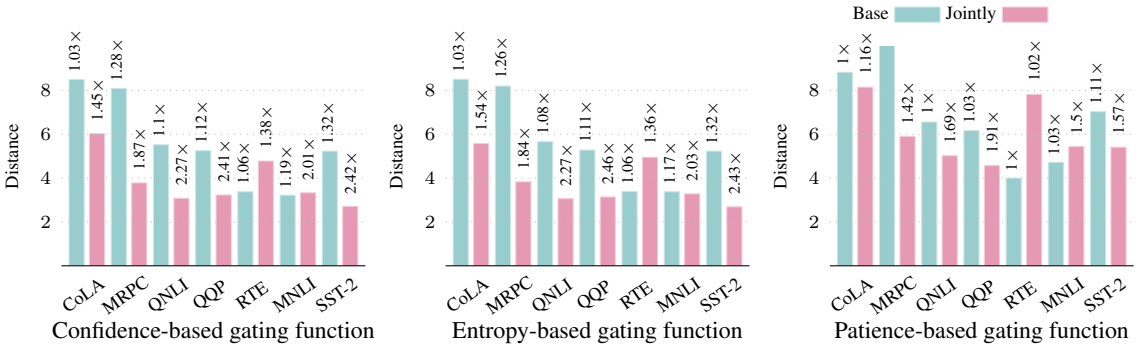


Figure 8: Bert model.

Model	RoBERTa(**12**)		BERT(**12**)	
Metric	Avg	Perc	Avg	Perc
CoLA	3.17 -> **2.26**	81.11% -> 99.90%	3.11 -> **2.40**	93.10% -> 99.90%
MRPC	1.32 -> **2.94**	100.0% -> 99.88%	2.88 -> **2.21**	93.68% -> 99.83%
QNLI	5.44 -> **2.27**	92.97% -> 99.96%	2.87 -> **1.93**	99.65% -> 99.80%
QQP	5.50 -> **1.83**	99.82% -> 99.99%	2.04 -> **1.62**	100.0% -> 99.95%
RTE	8.00 -> **3.92**	77.26% -> 99.64%	3.60 -> **1.62**	96.39% -> 99.28%
MNLI	7.01 -> **2.75**	98.68% -> 99.96%	6.46 -> **2.35**	93.40% -> 99.84%
SST-2	3.83 -> **2.34**	100.0% -> 99.77%	4.67 -> **1.58**	99.77% -> 100.0%

Table 9: The optimal early exit layer in roberta-base and BERT-base-uncased after joint optimization.

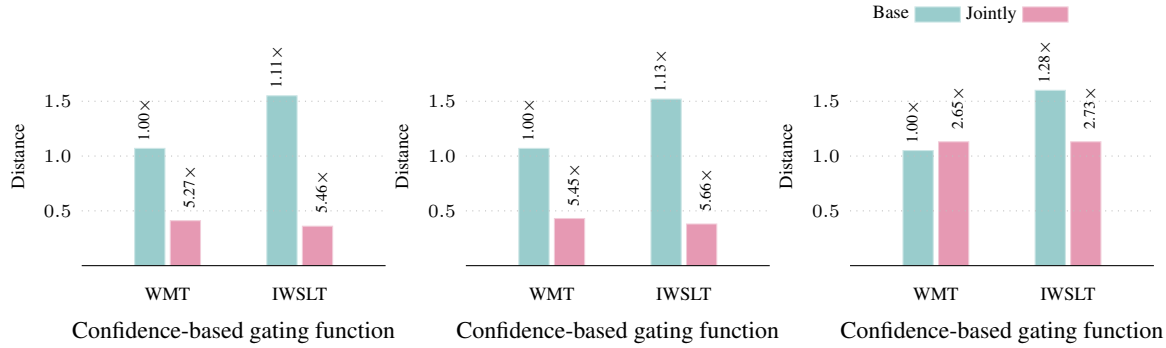


Figure 9: Transformer-base model follow DLCL (Wang et al., 2019) on the WMT14-EN2DE and DAT (Elbayad et al., 2019) on the IWSLT14-DE2EN.

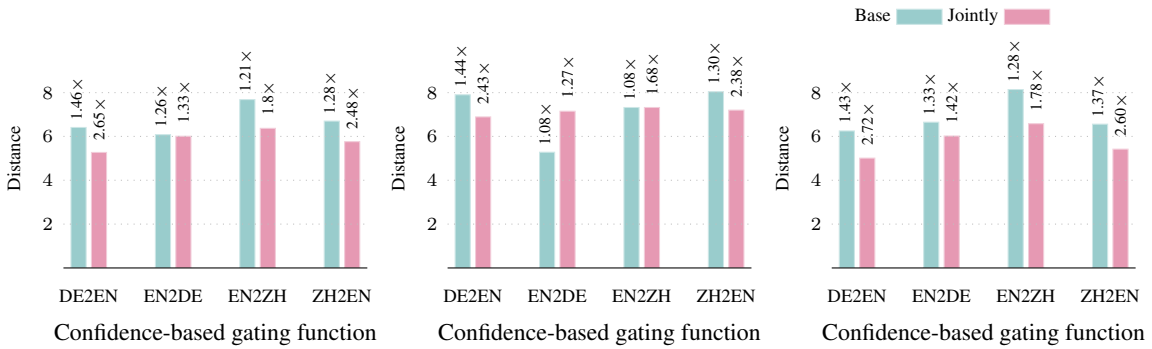


Figure 10: LLaMa2 model on the WMT22 test set.

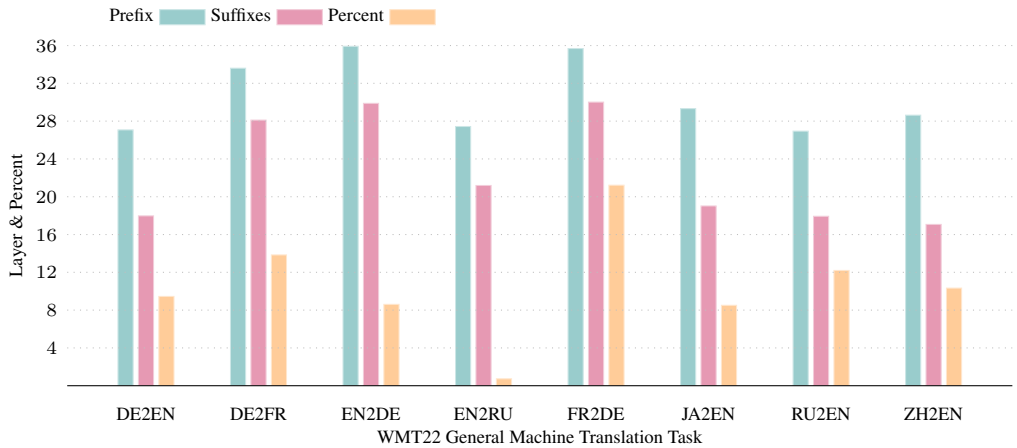


Figure 11: Translation result of LLaMa2 model on the WMT22 test set. We use prefix to represent the initial segment of a word and suffixes for the remaining part. Percent indicates the percentage of this phenomenon occurrences in relation to all tokens.

Exit position	Top-10 hypotheses
$\mathbf{h}_{\text{ffn}_t}^{40}$	_fields, _areas, _discipl, _domains, _inter, fields, _field, _subjects, _major, _indust
$\mathbf{h}_{\text{attn}_t}^{40}$	_fields, _areas, _discipl, _inter, _field, fields, _major, _se, _domains, _subjects
FFN($\mathbf{h}_{\text{attn}_t}^{40}$)	<s>, _edific, textt, _departamento, metros, _religios, _communes, , _interfaces, Portail
$\mathbf{h}_{\text{ffn}_t}^{39}$	_fields, _areas, _discipl, _inter, fields, _field, _domains, _major, _subjects, _se
ATTN($\mathbf{h}_{\text{ffn}_t}^{39}$)	<0x0A>, „ _, -, ., _and, _(, _the, _in, _C
$\mathbf{h}_{\text{ffn}_t}^{39}$	_fields, _areas, _discipl, _inter, fields, _field, _domains, _major, _subjects, _se
$\mathbf{h}_{\text{attn}_t}^{39}$	_fields, _areas, _discipl, _field, fields, _domains, _inter, _subjects, _major, _maj
FFN($\mathbf{h}_{\text{attn}_t}^{39}$)	_, _A, _(, _C, <0x0A>, _R, _just, _in, _g, _G
$\mathbf{h}_{\text{ffn}_t}^{38}$	_fields, _areas, _discipl, _field, fields, _domains, _inter, _subjects, _major, _maj
ATTN($\mathbf{h}_{\text{ffn}_t}^{38}$)	_. . . , _..., _., _, _covering, _cover, „ _and, _coverage, _..
⋮	⋮
$\mathbf{h}_{\text{ffn}_t}^{23}$	_fields, _areas, _field, _discipl, _categories, fields, _dici, _domains, _subjects, _topics
$\mathbf{h}_{\text{attn}_t}^{23}$	_fields, _areas, _field, _categories, _discipl, _domains, fields, Fields, _topics, _dici
FFN($\mathbf{h}_{\text{attn}_t}^{23}$)	_subject, aban, _rising, _fields, engo, enten, gew, chten, _nich, _branches
$\mathbf{h}_{\text{ffn}_t}^{22}$	_areas, _fields, _discipl, _categories, _domains, _area, _topics, _aspects, _dici, _major
ATTN($\mathbf{h}_{\text{ffn}_t}^{22}$)	_field, _fields, _Field, Field, fields, field, Fields, _research, _, _campo
⋮	⋮
$\mathbf{h}_{\text{ffn}_t}^{20}$	_fields, _areas, _aspects, _major, _categories, _topics, _discipl, _types, _subjects, _area
$\mathbf{h}_{\text{attn}_t}^{20}$	_areas, _fields, _aspects, _topics, _subjects, _categories, _major, _discipl, _types, _area
FFN($\mathbf{h}_{\text{attn}_t}^{20}$)	yl, _natural, _str, _kind, _pure, _proven, _un, _extreme, _underlying, _flav
$\mathbf{h}_{\text{ffn}_t}^{19}$	_areas, _fields, _aspects, _topics, _categories, _subjects, _types, _major, _discipl, _area
ATTN($\mathbf{h}_{\text{ffn}_t}^{19}$)	_territ, _fields, _field, cipl, _sector, _discipline, _territory, sci, _domains, _indust

Table 10: Top-10 hypotheses from block output $\mathbf{h}_{\text{ffn}_t}^{\ell+1}$, ffn module FFN($\mathbf{h}_{\text{attn}_t}^{\ell}$), attn module ATTN($\mathbf{h}_{\text{ffn}_t}^{\ell-1}$) and skip connect $\mathbf{h}_{\text{attn}_t}^{\ell}$, $\mathbf{h}_{\text{ffn}_t}^{\ell-1}$ based on the final output layer.

	SRC	**Denn falls tatsächlich etwas passieren sollte wie ein Brand, Einbruch, Erdbeben, Alieninvasion etc. wäre es tatsächlich zu viel Verantwortung für K1, sich um K2 zu kümmern.**
	hypothesis	optimal early exit layer
7B-d	If something were to happen like a fire, burglary, earthquake, or alien invasion, it would be too much responsibility for K1 to take care of K2.</s>	[31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31]
7B-d	If something were to happen like a fire, a break-in, an earthquake, an aldeorrde2-20220-20086; a pre-lift-de-20086; a pre-lift-de-41 M-de-41 M-de-41 M-de-41 M-de-41, by 4 M-de-m-lam-lam-lam-lam-lam-lam-lam-m-m-m-m-m-m-m-m-	[31, 31, 31, 31, 5, 16, 26, 21, 16, 30, 31, 29, 28, 19, 29, 16, 14, 18, 26, 26, 26, 31, 30, 29, 31, 30, 30, 30, 28, 29, 31, 29, 31, 31, 30, 29, 29, 31, 29, 31, 29, 31, 31, 26, 28, 31, 31, 30, 29, 18, 14, 27, 16, 31, 22, 30, 27, 12, 17, 12, 30, 31, 29, 30, 29, 12, 29, 31, 28, 26, 29, 15, 28, 28, 28, 17, 28, 12, 28, 28, 28, 17, 28, 11, 28, 19, 31, 30, 31, 30, 29, 29, 29, 14, 29, 30, 30, 31, 31, 30, 30, 15, 31, 29, 31, 29, 31, 29, 31, 14, 31, 13, 29, 27, 28, 13, 28, 17, 28, 16, 26, 15, 28, 17, 28, 17, 28, 17]
7B-c	If something were to happen like a fire, theft, earthquake, or alien invasion, it would be too much for K1elo to take care of K2 inoculation.i</s>	[31, 31, 31, 31, 5, 16, 26, 16, 16, 31, 10, 22, 16, 15, 19, 31, 18, 24, 0, 16, 30, 17, 13, 22, 15, 1, 30, 19, 17, 30, 30, 30, 29, 0, 24, 24, 16, 30, 31, 29, 29, 31, 27]
7B-j	If something were to happen like a fire, break-in, earthquake, or alien invasion, it would be too much responsibility for K1 to take care of K2.</s>	[31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31]
7B-j-d	If something were to happen like a "bright" - a Br. in-turn-in-k-bene 2166 (m bu bu 11/tre bu bu-d bu-/M. (M. (M. (as (c. (M. (c. (c. (c. (c. (c. (c. (c a c-c a c c c c a c a c a c a c a c a c a c a c a c a c a glun the ra t. The ra a t. a t.	[31, 31, 31, 31, 5, 12, 7, 31, 29, 31, 26, 31, 7, 30, 30, 31, 20, 27, 18, 20, 7, 24, 30, 31, 31, 24, 22, 31, 15, 31, 31, 27, 31, 30, 22, 7, 31, 27, 29, 29, 29, 31, 29, 31, 9, 31, 27, 31, 23, 27, 30, 7, 19, 7, 7, 31, 31, 31, 31, 7, 30, 7, 6, 11, 7, 6, 6, 7, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 6, 6, 31, 28, 31, 29, 31, 27, 28, 28, 28, 31, 6, 31, 6, 31, 6, 31, 6, 31, 6, 31, 6, 31, 6, 31, 6, 31, 6, 31, 6, 31, 6, 10, 31, 31, 31, 31, 29, 31, 31, 31, 27, 29, 31, 31, 29, 30]
7B-j-c	If something were to happen like a fire, a (catastrophic) alien invasion, (a)n (alien) invasion, (a)nd (al (alien), (al (al), (al (al), (al (al), (al (al), (al (al), (al (al), (al (al), (al (al), (al (al), (al (al), (al (al).</s>	[31, 31, 31, 31, 5, 16, 7, 10, 7, 7, 31, 31, 8, 2, 2, 8, 31, 0, 7, 7, 15, 23, 5, 30, 30, 30, 13, 7, 15, 7, 6, 6, 4, 24, 7, 8, 31, 12, 15, 29, 30, 12, 31, 9, 31, 6, 12, 17, 9, 7, 6, 12, 7, 9, 7, 6, 11, 7, 9, 7, 6, 11, 7, 9, 7, 6, 11, 7, 9, 7, 6, 11, 7, 9, 7, 6, 11, 7, 9, 7, 6, 11, 7, 9, 27, 7]

Table 11: Token-level early exit result and exit layer of Llama-2-Chat-7B(7B) with(-J) and without joint optimization on WMT22-DE2EN test set, under the constraint that early exit only after the 4-th token is generated. For copy kv operation, we represent directly copy as -d and recompute as -c.