6 Supplementary Material

6.1 Experimental Details

Table 2: Model configurations of the large and small models for each evaluation task. For comparison, the number of layers, hidden dimension, FFN dimension, and the number of decoder parameters (without embeddings) for each model are provided.

Task	Model	# Layers	dim	FFN dim	# Params
Machine Translation	mT5-large [75]	24	1024	2816	409M
	mT5-small [75]	8	512	1024	25M
Summarization	T5-large [44]	24	1024	4096	402M
	T5-small [44]	6	512	2048	25M

6.1.1 Training Details

For machine translation, we use IWSLT 2017 German-English [3] and WMT 2014 German-English [1] as target benchmarks, and mT5 [75] as a target model. We use the 8-layer mT5-small and the 24-layer mT5-large as the small and large models. For summarization, we use XSUM [40] and CNN/DailyMail [20] as target benchmarks, and T5 [44] as a target model. We use T5-small and T5-large with 6 and 24 layers, respectively, for the small and large models. Table 2 summarizes the size and configuration of each model. All the models are fine-tuned from the pre-trained checkpoints of the HuggingFace library [70] for 500k steps using a batch size of 16. We use Adafactor optimizer [50] with constant learning rate of $\{0.5, 1, 2, 5\}\text{e}{-4}$ for the small models and $\{0.5, 1\}\text{e}{-4}$ for the large models. We refer to the normally fine-tuned models on the validation datasets as the *baseline* small and large models.

When training *aligned* small models via the prediction alignment method described in Section 3.5.1, we first generate calibration datasets using the input sequences from the training datasets of each benchmark. We then use the fully trained large model to generate output sequences through greedy sampling with a beam size of 1. To ensure a fair comparison, we fine-tune pre-trained small models (rather than the baseline small models that are already fine-tuned on the training datasets) on the calibration datasets using the same training recipes and the number of training steps as described above. This decision is based on our observation that fine-tuning a baseline model using the calibration dataset tends to improve generation quality, likely due to the increased number of training examples and data augmentation effects, which makes it difficult to make a fair comparison between unaligned BiLD and aligned BiLD. However, in practice, one can obtain aligned models by applying the prediction alignment method directly to the fine-tuned baseline small models to achieve the best performance.

6.1.2 Evaluation Details

All inference evaluations including latency measurement are conducted on a single NVIDIA T4 GPU of a GCP n1-standard-4 instance with 4 vCPUs and 15GB memory. For inference, we use batch size 1, which is a common use case for online serving [48]. For the distance metric d in Equation 3 for the rollback policy, we use the cross-entropy loss between the small model's hard label and the large model's soft label. This measures the (negative log) likelihood of obtaining the small model's prediction from the large model's output. For BiLD inference, we sweep over different fallback and rollback thresholds to explore different trade-offs between generation quality and latency. For the machine translation tasks, we use fallback thresholds in [0.5, 0.9] and rollback thresholds in [1, 10]. For the summarization tasks, fallback thresholds in [0.2, 0.6] and rollback thresholds in [2, 6]. We keep the maximum generation length of the small model to 10 to avoid high rollback costs. In Appendix 6.3.3, we provide a detailed analysis of how varying the fallback and rollback thresholds impacts the trade-offs between generation quality and latency in the BiLD framework.

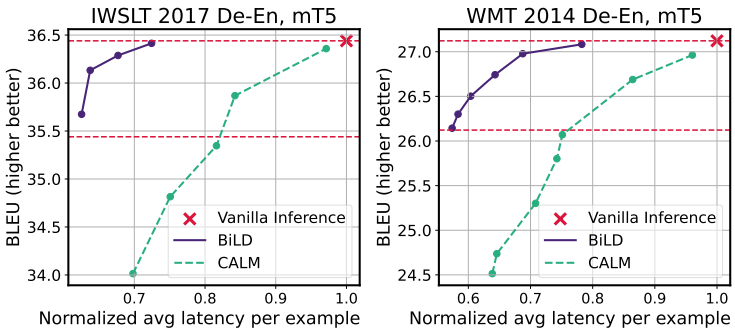

Figure 7: The trade-off curves between inference latency and BLEU score for BiLD and CALM in the early exiting setting for (Left) IWSLT 2017 De-En and (Right) WMT 2014 De-En. The $\times$ marks indicate the vanilla inference latency and BLEU score of the mT5-small models. The horizontal lines indicate the vanilla inference score and 1 point degradation from it. BiLD outperforms CALM across all speedup regimes by up to $2 \sim 2.5$ points better BLEU score, demonstrating the effectiveness of our approach for the early exiting strategy.

6.2 Details of Early Exiting Strategy in the BiLD Framework

6.2.1 Training and Evaluation Details

BiLD. We use the mT5-small model as the large model and the first (out of 8) layer as the small model, and evaluate it on two machine translation benchmarks: IWSLT 2017 De-En and WMT 2014 De-En. To ensure consistency between the prediction made after the first layer and the one made after the last layer, we fine-tune the pre-trained mT5 model using the average loss of the first and the final layers, similar to [10, 48]. That is, $\mathcal{L} = \frac{1}{2}(\mathcal{L}_1 + \mathcal{L}_{-1})$ where $\mathcal{L}_1$ and $\mathcal{L}_{-1}$ are the negative log-likelihood loss after the first layer and the final layer. The prediction head is shared for these two layers. We fine the pre-trained mT5-small model on each benchmark for 500k steps using a batch size of 16. Similar to the main experiments, we use Adafactor optimizer [50] with constant learning rate of $\{0.5, 1, 2, 5\}$e-4. For evaluation, we use fallback thresholds in [0.2, 0.8] and rollback thresholds in [0.5, 1.5].

CALM. To reproduce CALM [48] in our experimental setup, we have fine-tuned the pre-trained mT5-small model on IWSLT 2017 De-En and WMT 2014 De-En datasets. We employ the averaged loss across all layers, i.e., $\mathcal{L} = \sum_{i=1}^{L} w_i \mathcal{L} i$, where $w_i = i / \sum_{j=1}^{L} j$, which was introduced in the paper to ensure the layer consistency. We use Adafactor optimizer [50] with constant learning rate of $\{0.5, 1, 2, 5\}$e-4 for 500k training steps. To make a fair comparison, we match the BLEU score of the fine-tuned model to that of BiLD's models Among the two training-free confidence measures introduced in the CALM paper, softmax-based and hidden-state saturation-based measures, we have chosen to use the latter approach as an early exiting criterion. That said, if the cosine similarity between the current layer's hidden states and the previous layer's hidden states exceeds a certain threshold, we perform early exiting. We have found that the softmax-based alternative is not applicable in our evaluation scenario due to the large output vocabulary (more than 200k for mT5, which is $\sim 10\times$ larger than T5), which significantly increases latency overhead. As described in the paper, when early exiting happens, the hidden states of the exited layer are propagated down to the remaining layers to compute the key and value caches. To achieve different trade-offs between latency and generation quality, we sweep over λ in [0.7, 0.98] and t in $\{0, 1, 2, 4, 8\}$ in the decaying threshold function.

6.2.2 Performance Comparison between BiLD and CALM

Figure 7 illustrates the BLEU score and latency curves of BiLD compared to CALM in the early exiting setting. In both tasks, our method achieves significantly better BLEU scores with the same latency speedup, yielding up to around 2 point better BLEU score in the $\sim 1.5\times$ speedup regime. This can be attributed to two factors. First, in BiLD, even if an early exited prediction (i.e., prediction made by the smaller model) is incorrect, it can be corrected and replaced using the rollback policy. Therefore, an error in the early exited layer is propagated less drastically to the future prediction.

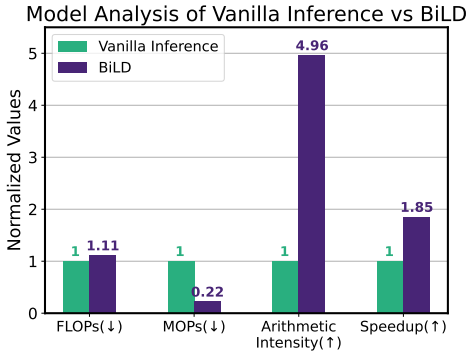

Figure 8: FLOPs, MOPs (memory operations), arithmetic intensity, and latency speedup comparison of vanilla inference and BiLD. BiLD approach results in a remarkable reduction in MOPs due to the improved token-level parallelism, resulting in significantly higher arithmetic intensity.

Second, the key and value caches for skipped layers are filled with actual values instead of being computed from the exiting layer's hidden states. This also leads to reduced error propagation and improved decoding stability.

6.3 Additional Analysis

6.3.1 Model Analysis of BiLD: FLOPs, MOPs, and Arithmetic Intensity

Figure 8 compares average FLOPs, MOPs (memory operations), arithmetic intensity, and the latency speedup of the vanilla inference and BiLD on the CNN/DailyMail benchmarks. For BiLD, we use the model with roughly the same ROUGE-L score as the vanilla inference, and all the numbers are normalized by the numbers of the vanilla inference. The figure illustrates that BiLD exhibits slightly higher FLOPs compared to the vanilla inference. This is due to the fact that the autoregressive and non-autoregressive executions have the same amount of FLOPs, and BiLD involves additional overhead of running the small model alongside. However, in the case of MOPs, BiLD demonstrates a significant $\sim 5\times$ reduction of memory operations. This can be attributed to the capability of BiLD to process multiple tokens with a single weight load, thereby enhancing token-level parallelism and maximizing data reuse. In contrast, this is not the case in the vanilla inference where a single weight load can only process a single token. Consequently, BiLD achieves a significantly higher arithmetic intensity, which is approximately 5 times larger than the vanilla inference. Arithmetic intensity [69] measures the number of arithmetic operations that can be performed per memory operation. Given that memory operations can contribute more to the overall inference latency than arithmetic operations in many Transformer decoding scenarios [30], decreasing memory operations and increasing arithmetic intensity can effectively alleviate the inference bottleneck. This leads to an overall latency speedup of $1.85\times$ on actual hardware.

6.3.2 Examples of Generated Sequences

Figure 9 provides examples of text sequences generated by BiLD on the validation set of IWSLT 2017 De-En, along with the ground truths (i.e., labels) and outputs of the pure large and small baseline models. The tokens generated from the large model of BiLD are highlighted in green, while all the other tokens are generated by the small model. The results illustrate that the small model often produces low-quality texts, by predicting inaccurate tokens which can alter the meaning of the entire sentence. To contrast, it is observed from the examples that BiLD is able to improve the text generation quality by letting the large model interrupt when the small model generates incorrect tokens. Particularly, in the examples provided, BiLD tends to be as strong as the large model at predicting terminologies. Overall, the large model's engagement in BiLD decoding not only improves the prediction accuracy but also prevents incorrect predictions from impacting the future ones.

Ground Truth	And Siftables are an example of a new ecosystem of tools for manipulating digital information.
Large	And the Siftables are an example of a new generation of manipulation tools for digital data.
Small	And the if you look at the ifleses are an example of a new generation of technologies for manipulation of digital data.
BiLD (ours)	And the **Siftables** are an example of a new generation of **manipulation of** digital data.
Ground Truth	Which is great, because the Romans did not actually think that a genius was a particularly clever individual.
Large	That's great. The Romans didn't really think that a genius was a particularly smart individual.
Small	That's great. The tube didn't really think that a genius was a particularly lonely individual.
BiLD (ours)	That's great. The **Romans** didn't really think that **a** genius was a particularly smart individual.
Ground Truth	The viral particles then were released from the cells and came back and killed the E. coli.
Large	The viral particles then were released by the cells and came back and killed E. coli.
Small	The viral particles were then released by the cells and came back and killed E. Coke.
BiLD (ours)	The viral particles **then were** released **by the** cells and came back and killed E. **coli**.

Figure 9: Example text sequences that BiLD generates with the validation set of IWSLT 2017 De-En, compared to the ground truths and the outputs of the large and small baselines. For BiLD, tokens generated by the large model are highlighted in red, while all the other tokens are generated by the small model. This illustrates that with a small engagement of the large model, BiLD can correct not only inaccurate vocabulary but also wrong semantics of the text that the small model would have otherwise generated.

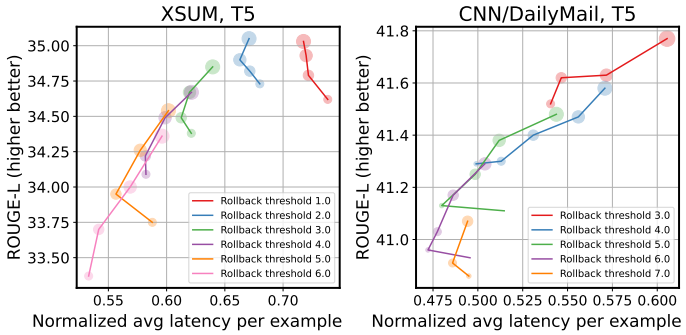

Figure 10: The trade-off between latency and generation quality (ROUGE-L) for the aligned BiLD model on two summarization tasks: (Left) XSUM and (Right) CNN/DailyMail. Each curve represents a different rollback threshold, with smaller thresholds indicating more rollbacks. The trade-off can be further obtained within each curve with different fallback thresholds, where larger scatter sizes indicate larger fallback thresholds. A larger fallback threshold implies more fallbacks.

6.3.3 Impact of Fallback and Rollback on Performance

We have explored how the BiLD framework can achieve different trade-offs between latency and generation quality by adjusting fallback and rollback thresholds. In this section, we present a detailed analysis of how these thresholds affect the performance using the aligned BiLD model on two different summarization tasks, XSUM and CNN/DailyMail, as illustrated in Figure 10. Different curves in the plot represent different rollback thresholds, and each scatter point within the curve represents different fallback thresholds. Note that a small rollback threshold implies more rollback, while a larger fallback threshold implies more fallback.

We observe a general trend where smaller rollback thresholds (i.e., more rollbacks) result in better generation quality but longer latency. This trend is expected because, with more rollback, we preempt more small model's predictions that can be potentially inaccurate by sacrificing the latency. Similarly, there is also a general trend that smaller fallback thresholds (i.e., fewer fallbacks) result in faster latency but a worse generation quality. However, we observed that lowering the fallback rates beyond a certain point can actually hurt both the latency and generation quality. This is because inaccurate predictions that the small model should have fallen back are later rolled back, incurring an extra 'flush' cost for the tokens that follow.