Tandem Transformers for Inference Efficient LLMs

Aishwarya P S, Pranav Ajit Nair, Yashas Samaga, Toby Boyd, Sanjiv Kumar, Prateek Jain*, Praneeth Netrapalli*

Google DeepMind, Google Research

The Problem

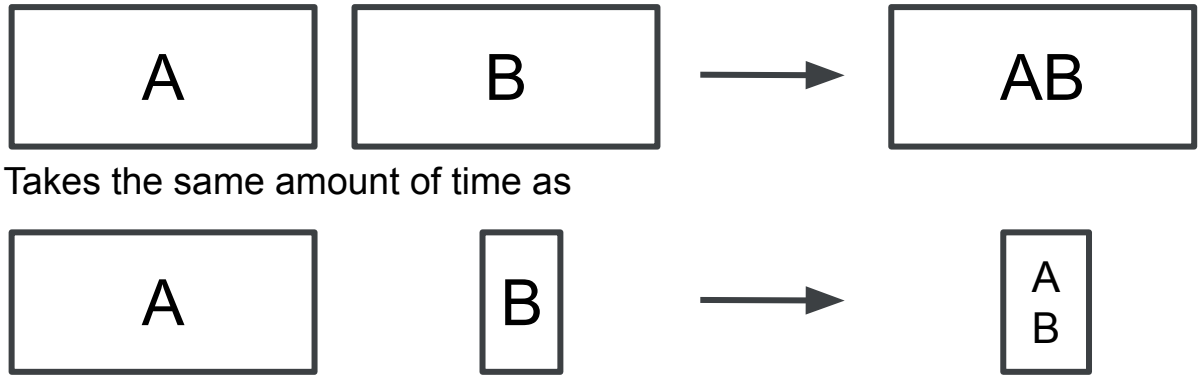
- A key factor contributing to high serving cost and latency of LLMs is their autoregressive nature where tokens are produced sequentially.

- This restricts the full utilization of ML accelerators which are optimized for matrix-matrix multiplications rather that matrix-vector and element-wise operations, and memory transfers.

Matrix-Matrix

vs

Matrix-Vector



Speculative Decoding [1] addresses the aforementioned problem by:

- Using a small drafter to autoregressively speculate blocks of tokens (thus, faster memory transfers).
- Using a larger primary model to verify (and correct) the output of the drafter, for the entire block in parallel (thus, matrix-matrix operations).

Can we improve the drafter?

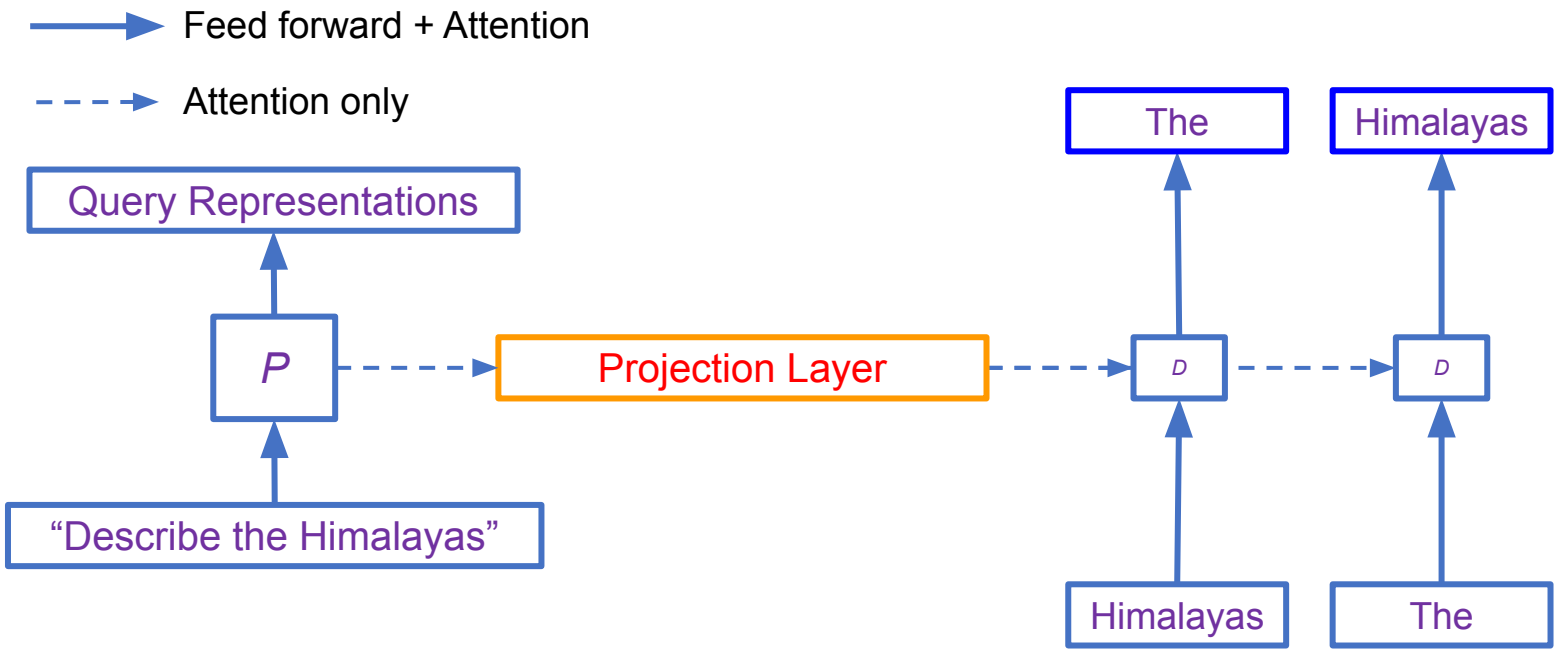
- Can we have a smaller/faster drafter? Block parallel decoding [2], heuristic drafter [3].
- Can we have a more aligned drafter? An obvious choice would be distillation, *but can we do better?*

Two key conceptual tasks of an LLM: **Understanding** and **Generation**.

- Classically, encoder-decoder models decouple these and often use **larger encoder** models (for understanding) with **smaller decoder** models (for generation).
- Decoder-only architecture couples understanding and generation.
- *Can we decouple capacity required for these?*

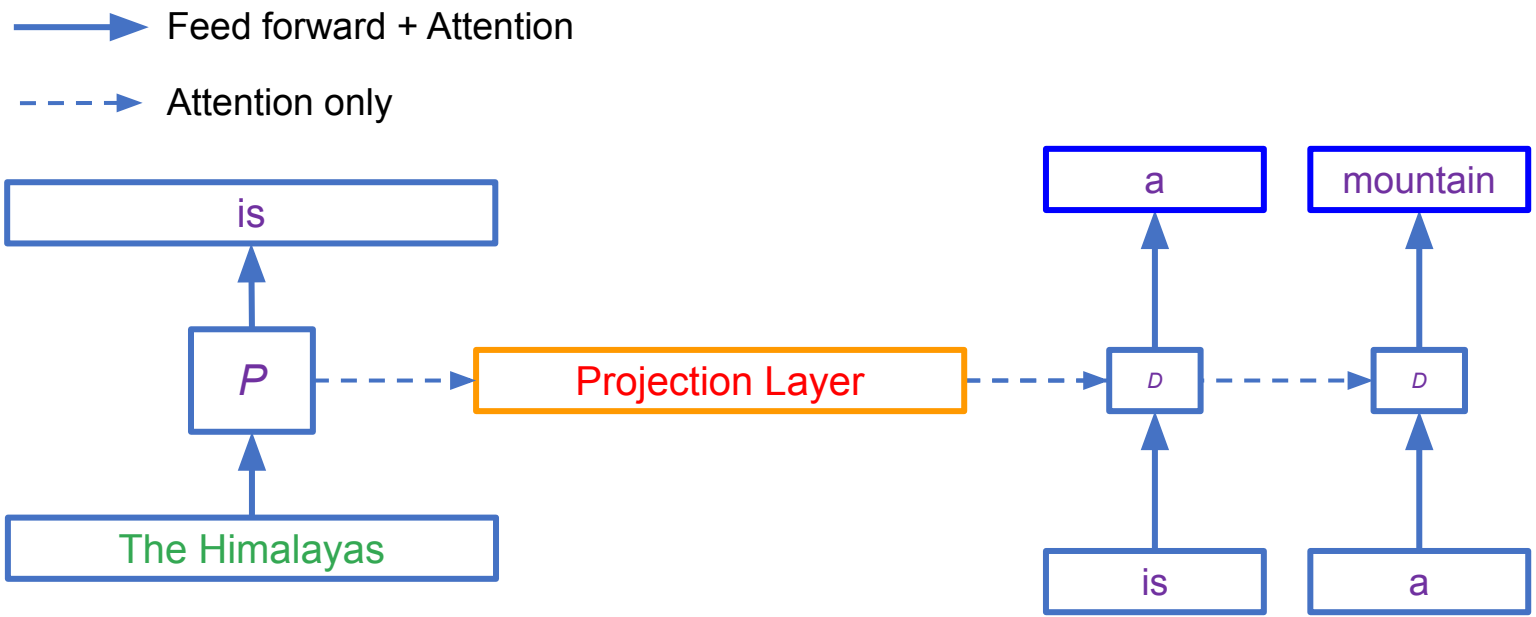
[1] Fast inference from Transformers via Speculative Decoding by Leviathan et al. ICML 2023
[2] MEDUSA: Simple LLM Inference Acceleration Framework with Multiple Decoding Heads by Cai et al. ICML 2024
[3] Rest: Retrieval-based speculative decoding by He et al. 2023

Tandem Transformers (Block length: 2)



- A novel architecture where the drafter *attends* to the projected higher quality representations from the primary model to produce i) higher quality ii) further aligned output.
- Tandem Transformers decouple query understanding and response Generation.

Tandem Transformers (Block length: 2)



Training Tandem Transformers

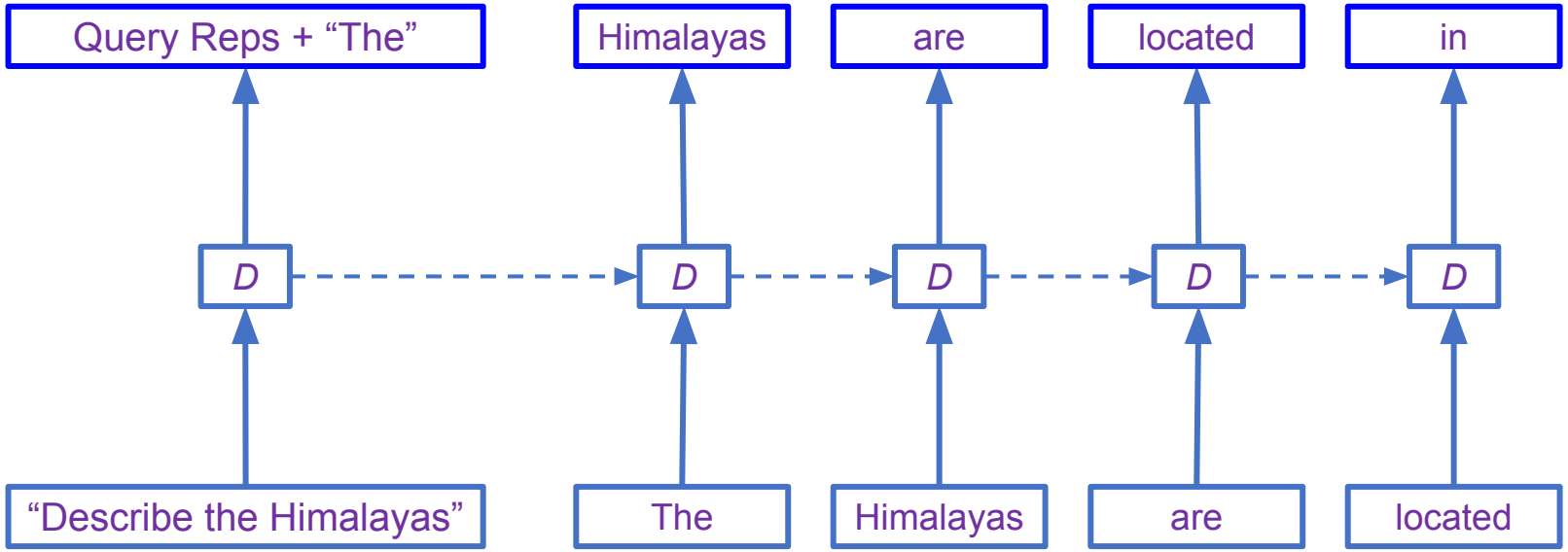
- We train a tandem model with P = PaLM2-Bison and D = PaLM2-Gecko
- In terms of size, PaLM2-Gecko < PaLM2-Otter < PaLM2-Bison
- After initializing with pretrained models, freeze P and train only the projection layers and D.
- Tandem-CE: trained with CE loss wrt ground truth labels
- Tandem-Distill: continue training Tandem-CE also with CE wrt PaLM2-Bison output logits.
- PaLM2-Gecko-Distill: continue training PaLM2-Gecko with distillation loss as above.

Pretraining Results

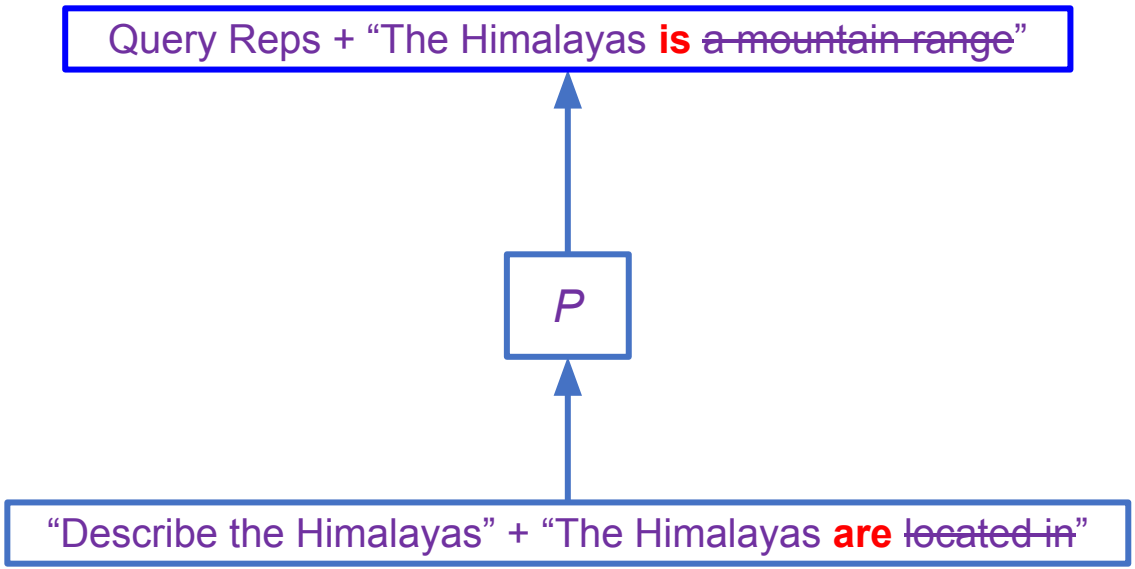
	PaLM2-Gecko	PaLM2-Gecko-Distil	Tandem-CE **(ours)**	Tandem-Distil **(ours)**
Accuracy (GT)	55.06	56.50	58.35	**58.61**
CE loss (GT)	2.14	2.12	**1.94**	1.99
Relative accuracy	74.64	75.30	80.00	**81.00**
Relative TV distance	0.391	0.318	0.178	**0.141**

Can we use Tandem Transformers without worrying about accuracy?

Background: SPEED - Speculative Decoding

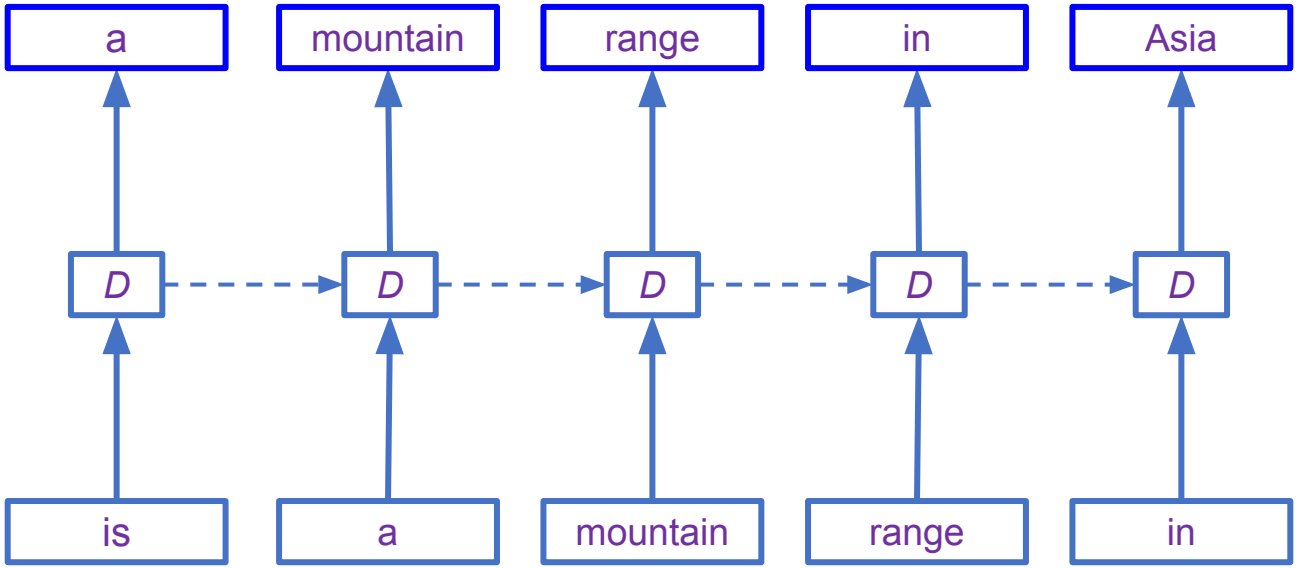


SPEED



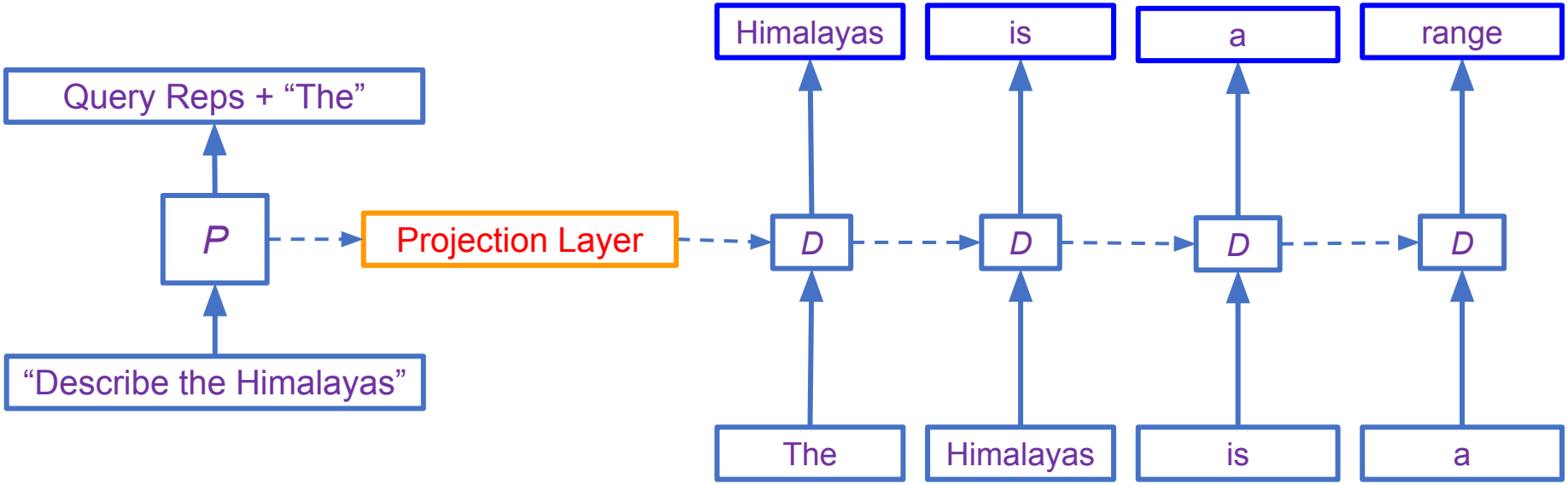
are ≠ is

SPEED

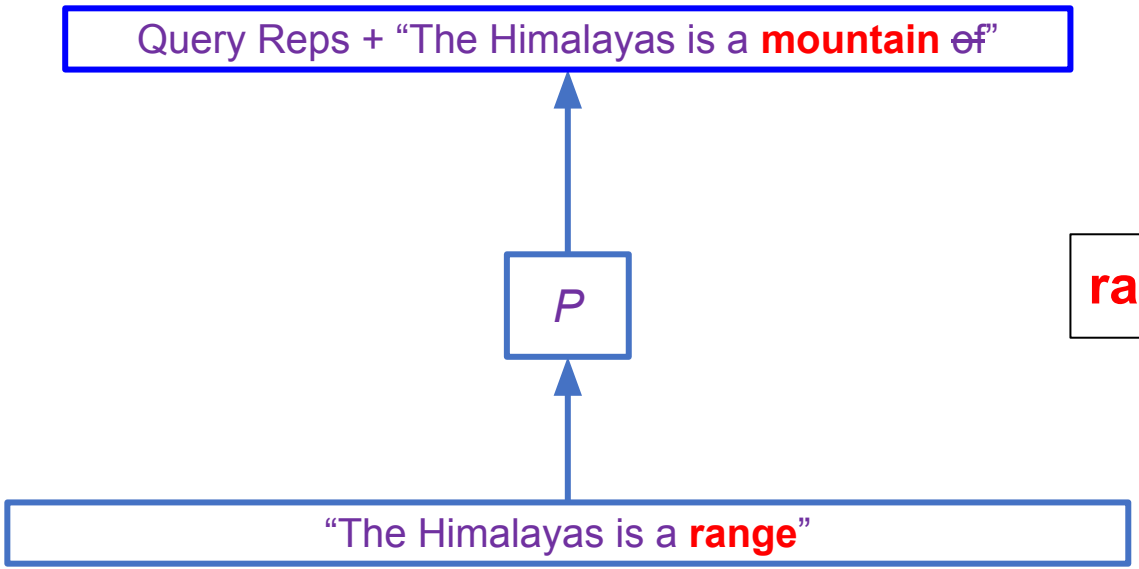


Less backtracking ➡ More speedup.

SPEED+Tandem



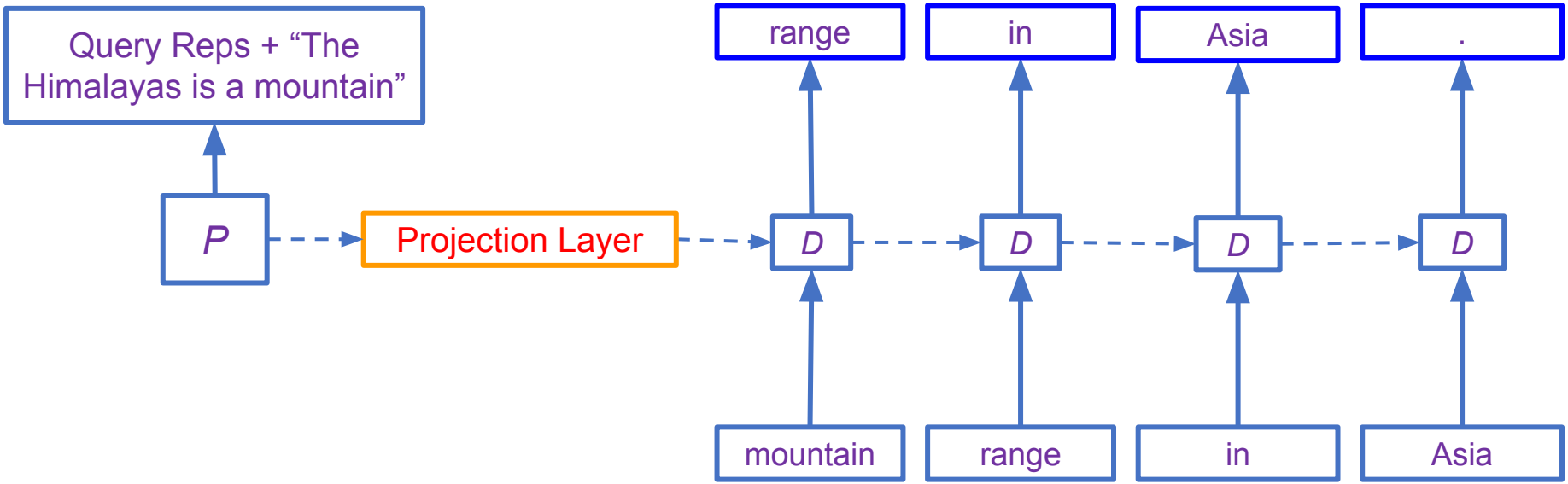
SPEED+Tandem



range ≠ mountain

SPEED+Tandem

P = PaLM2-Bison
D = PaLM2-Gecko



Results

Tandem Transformer's improvements over SPEED + Distillation

Dataset	num-samples	PaLM2-Gecko-Distil (baseline)	Tandem-Distil (**ours**)	Tandem-Distil (**ours**; relative gain)
Reddit	1	2.169× ($\gamma = 7$)	2.471× ($\gamma = 7$)	**1.139**×
	4	1.919× ($\gamma = 5$)	2.234× ($\gamma = 7$)	**1.164**×
CNN/DailyMail	1	2.219× ($\gamma = 7$)	2.473× ($\gamma = 7$)	**1.115**×
	4	1.940× ($\gamma = 5$)	2.190× ($\gamma = 7$)	**1.129**×
LM1B	1	2.348× ($\gamma = 7$)	2.610× ($\gamma = 7$)	**1.112**×
	4	2.011× ($\gamma = 5$)	2.359× ($\gamma = 7$)	**1.173**×

γ optimized for each setting – Tandem can use larger more effectively.

Tandem Transformers + Adaptive block length's improvements over the baseline, and Tandem-Distil + SPEED (i.e., penultimate column from the table above).

Dataset	PaLM-Bison	Tandem-Distil + SPEED
Reddit	2.582× ($\gamma_{max} = 17$)	**1.045**×
CNN/ DailyMail	2.599× ($\gamma_{max} = 17$)	**1.051**×
LM1B	2.853× ($\gamma_{max} = 27$)	**1.093**×

Adaptive Block Length: After every token predicted by the drafter model, we use a small, inexpensive router to determine whether to continue predicting with the secondary model, or verify the tokens generated so far with the primary model.

Smaller Drafter: Replacing PaLM2-Gecko with PaLM2-XXXS for Tandem training. Tandem Transformers give **larger gains with a smaller drafter**.

Dataset	num-samples	PaLM2-XXXS-Distil (baseline)	Tandem-Distil (**ours**)	Tandem-Distil (**ours**; relative gain)
LM1B	1	2.445× ($\gamma = 5$)	3.040× ($\gamma = 5$)	**1.243**×
	4	1.821× ($\gamma = 5$)	2.488× ($\gamma = 5$)	**1.366**×

Tandem Transformers as a standalone model: Tandem Transformers can be used a standalone model (without SPEED) to exploit the full spectrum of latency vs quality. Please refer to the paper for model details.

Dataset	PaLM2-Gecko	Tandem-Distil (**ours**)	PaLM2-Otter	PaLM2-Bison
Generative-tasks	28.8	44.0	51.1	57.5
MBPP	4.8	21.2	20.8	30.4
WMT22-1shot-to-nonenglish	35.1	44.1	48.4	50.5
TydiQA-GoldP	55.0	69.0	69.7	73.4
SuperGLUE	62.8	78.8	79.0	81.5
Speedup over PaLM2-Bison	6.40×	2.75×	2.36×	1×

Conclusion

- Inference latency is a big bottleneck with very large room for improvement.
- Tandem Transformers can be used a standalone model (without SPEED) to exploit the full spectrum of latency vs quality. It can produce 1.16x speedup over PaLM2-Otter with similar quality.
- Tandem Transformers within the SPEED framework can give upto **3x** speedup over PaLM2-Bison and **1.36x** speedup over SPEED + Distillation.

Future Directions

- Other variants of Tandem Transformers, e.g., using LLM for plan generation and SLM for autoregressive token generation using plan.
- Tandem Transformers as an alternative to LoRA for fine-tuning.