SUPER TINY LANGUAGE MODELS RESEARCH SERIES - KNOWLEDGE DISTILLATION

Bobby Cheng
A*STAR, Institute of Infocomm Research (I2R)
bobby_cheng@i2r.a-star.edu.sg

Leon Guertler
A*STAR and Nanyang Technological University
leon002@e.ntu.edu.sg

Dylan Hillier
A*STAR and Singapore Management University
das.hillier.2023@phdcs.smu.edu.sg

Cheston Tan
A*STAR, Institute of High Performance Computing (IHPC)
cheston-tan@i2r.a-star.edu.sg

1 Motivation

In our earlier work, titled 'Super Tiny Language Models' [1], we outlined our ambitions and approach for developing Super Tiny Language Models (STLMs) that are high performing in spite having significantly smaller parameter counts. One of our listed strategies is the use of knowledge distillation.

Popularised by Hinton et al. (2015), knowledge distillation (KD) [2] involves distilling the knowledge of a larger neural network (teacher model) into a smaller neural network (student model). Compared to smaller models, larger models tend to have a deeper contextual knowledge of human language which makes them higher performing in a variety of tasks. Efficiently transferring this knowledge to smaller models can help to bridge the performance gaps between small and large models. A notable example is DistilBERT [3] which is 40% smaller and 60% quicker than BERT, yet retains 97% of it's knowledge thanks to knowledge distillation.

For STLMs, KD is particularly relevant. Reducing the parameter count of language models results in significant performance degradation. However, by leveraging the richer representations typically found in larger models through KD, we hypothesize that STLMs can achieve performance levels closer to their larger counterparts, whilst maintaining their compact size and efficiency. This approach aligns with our goals of creating highly efficient yet high performing models suitable for resource constrained environments.

In this preregistration paper, we will highlight our approach to study the effectiveness of transferring the decision-making abilities of a large model to our STLM through KD.

2 Method

2.1 Overview

Our research will focus on two areas of knowledge distillation. The first is the knowledge component, and the second is the student-teacher architecture.

2.2 Knowledge Component

Specifically, we will explore two areas of the knowledge component - response-based and feature-based.

2.2.1 Response-based Knowledge Distillation

The core concept of response-based knowledge is to allow the student model to learn from the teacher model's final predictions. These final predictions are the probability distributions of the model's final output layers, which Hinton et

al. (2015) calls soft targets. These soft targets are generated when the logits z of the model's final output layer undergo softmax activation with a temperature parameter T.

The softmax function with temperature is defined as:

$$p_i = \frac{\exp(z_i/T)}{\sum_j \exp(z_j/T)}$$

where p_i is the probability of class i, z_i is the logit for class i, and T is the temperature.

To measure the loss, also called divergence, between the probability distributions of the student and teacher models, we will explore two types of divergence calculations.

Forward KL Divergence The first is the forward KL divergence [4]. It is expressed between the teacher's soft targets P_T and the student's soft targets P_S as:

$$\mathcal{L}_{\text{response}} = Forward\ KL(P_T||P_S) = \sum_i P_T(i) \log\left(\frac{P_T(i)}{P_S(i)}\right)$$

where $P_T(i)$ and $P_S(i)$ are the probabilities for class i from the teacher and student models, respectively, after applying the softmax function with temperature T:

$$P_T(i) = \frac{\exp(z_{T,i}/T)}{\sum_j \exp(z_{T,j}/T)}, \quad P_S(i) = \frac{\exp(z_{S,i}/T)}{\sum_j \exp(z_{S,j}/T)}$$

Here, $z_{T,i}$ and $z_{S,i}$ are the logits for class i from the teacher and student models, respectively.

The temperature T is a critical hyper parameter. It softens the probability distributions which supposedly enhances the efficacy of the distillation process. Following Hinton et al. (2015), we will adopt an intermediate temperature in our study which is suitable when the student model is significantly smaller than the teacher.

Reverse KL Divergence The second is known as the reverse KL divergence function. This is expressed as:

$$\mathcal{L}_{\text{response}} = Reverse\ KL(P_S||P_T) = \sum_i P_S(i) \log\left(\frac{P_S(i)}{P_T(i)}\right)$$

Using reverse KL divergence can provide different gradients that will also be suitable for our training. It has been reported by Gu et al. (2024) [5] that the forward KL divergence tends to model after the teacher's low-probability regions, whereas reverse KL divergence models after the teacher's mode regions (peaks in the probability), which is more suitable for text generation. Although Wu et al (2024) [6] found no significant difference between forward or reverse KL divergence after multiple epochs, we find it will be interesting to observe any differences for STLMs.

In both cases, training the student model to minimize the soft target's to match the teacher would allow the student to acquire an understanding of the relative relationships between classes. For instance, in a sentiment analysis task, soft targets might reveal that a sentence classified as positive also carries slight negative connotations. Such nuanced information would be lost if only considering the final class prediction. Through this process, we hope this is one way to effectively distill the sophisticated understanding of large models into our STLMs.

2.2.2 Feature-based Knowledge Distillation

While response-based KD extracts knowledge from the output logit probabilities, feature-based KD extracts knowledge from the intermediate layers of the network, e.g. embedding space, transformer layer. While response-based KD gets the student to learn the result of the teacher, feature-based gets the student to learn the processes leading to the results. This would allow the student to learn additional or other aspects of the teacher.

The loss between the student and teacher features is defined as:

$$\mathcal{L}_{\text{hint}} = \mathcal{H}(F^s, F^t) = \|F^t - \phi(F^s)\|^2$$

where F^s, F^t are the outputs of the relevant layers for student and teacher respectively, and ϕ is the linear layer that projects the student's layer output into the same dimension as the teacher. $\mathcal{H}$ represents the metric function of mean square error (MSE).

Compared to DistilBERT [3], which uses only response-based KD, TinyBERT [7] found that features like the embedding and transformer layers have allowed it to achieve the same level of performance as DistilBERt, whilst compressing BERT by 7x as compared to DistilBERT's compression of 2x. As such, we are also interested to investigate the effects of feature-based KD for STLMs.

2.3 Loss Function

Building on the insights from feature-based KD, our approach to knowledge distillation integrates both feature-based and response-based methods. Hinton et al. (2015) found that distillation is most effective when the response-based KD loss is combined with the cross-entropy loss between the ground truth labels and the soft targets predicted by the student model. This dual loss approach allows the student to learn not only from the teacher's nuanced output but also from the true labels, thereby improving the model's robustness, particularly when the student struggles to replicate the teacher's soft targets exactly.

Consequently, our total loss function is formulated as:

$$L = \begin{cases} T^2 \, \mathcal{L}_{\text{hint}} \\ \mathcal{H}(y, \sigma(z_s)) \\ \mathcal{L}_{\text{hint}} \end{cases}$$

where $\mathcal{H}$ is the cross-entropy loss for the true labels and student predictions, $\mathcal{L}_{\text{response}}$ is the response-based loss, and $\mathcal{L}_{\text{hint}}$ is the feature-based loss. The term T is the temperature scaling factor, and as the gradient magnitudes of the soft targets scale as $1/T^2$, we multiply by T^2 [2] to normalize them appropriately.

2.4 Teacher Model Selection

As part of the student-teacher architecture component, the selection of appropriate teacher models are another crucial step in the KD process. Student models tend to be a simplified version of their teacher model's network with lesser layers and smaller hidden dimension. However, our model is unique in that it has no 'teacher' model to distil information from right away. Hence, our study will uncover the impact of distilling teacher models of reasonably different architectures.

We will base our teacher model selection criteria on these three factors: 1) **architecture similarities**, which ensures our teacher models have a similar adoption of our baseline model, 2) **computational resources**, which aligns with the low resource requirements suited for STLMs, and 3) **task alignment**, which ensures the model's capabilities are also suited to our student model. This narrowed our selection to Qwen 2 (0.5B, 1.5B) [8], MobiLlama (0.5B) [9], Phi 3 (3B) [10] and GPT2-Large (0.8B) [11].

2.4.1 Adaptation Strategy

A crucial pre-processing step would be to adapt the logits of the teacher model with the student model as each of them have different vocabulary sizes - Qwen 2: 152K, MobiLlama and Phi3: 32K, and our student model: 50K.

Our proposed adaptation strategy involves training a new set of embedding layer and language model head for our teacher model, while freezing it's transformer architecture. This ensures that the output logits will share the same vocabulary size as our student. We anticipate that this adaptation strategy will complicate and impact the effectiveness of the distillation process. To quantify this impact, we will take the baseline benchmarks of these teacher models and compare against the new benchmark scores of the adapted teacher models. This allows us to assess the effectiveness and implications of our adaptation strategy on the knowledge distillation process.

2.5 Evaluation

After assessing the effects of our teacher model adaptation strategies, we will use these adapted teacher models to perform knowledge distillation on our baseline model architecture, which is 47M in parameter size. The effectiveness of KD will be compared against our baseline model as reported in our earlier work [1].

Model	Num. Params	BLiMP	HellaSwag	ARC_easy	WinoGrande	MMLU
Qwen2 [8]	571M	77%	36%	50%	54%	26%
Baseline [1]	50M	78%	30%	39%	50%	24%
GPT2-Qwen2 (0.5B)	50M	78%	30%	37%	52%	24%
Metric	–	Accuracy	Accuracy	Accuracy	Accuracy	Accuracy
Num. Choices	–	2	4	4	2	4
Chance Perf.	–	50%	25%	25%	50%	25%

Table 1: Initial results on chosen benchmarks. The table is divided into 4 sections: (1) Results of the adapted teacher models. (2) A baseline model for comparative purposes. (3) Results of the baseline model trained with the new experimental models (GPT2-Qwen2) as teacher models. (4) The metrics section includes the number of choices for each task and the chance performance for context. All results are obtained zero-shot with custom splits of the datasets.

3 Preregistration Disclosure

This paper is intended to act as a preregistered copy of the eventual research report. Ideally we would enter such a research report blind to partial results, however we have previously run some experiments while developing our framework. One such result was using a Qwen 2 0.5B adapted teacher model to train our baseline model. It gave the following scores in Table 1.

References

[1] Dylan Hillier, Leon Guertler, Cheston Tan, Palaash Agrawal, Chen Ruirui, and Bobby Cheng. Super tiny language models, 2024.

[2] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015.

[3] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, 2020.

[4] Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

[5] Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. Minillm: Knowledge distillation of large language models, 2024.

[6] Taiqiang Wu, Chaofan Tao, Jiahao Wang, Zhe Zhao, and Ngai Wong. Rethinking kullback-leibler divergence in knowledge distillation for large language models, 2024.

[7] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. Tinybert: Distilling bert for natural language understanding, 2020.

[8] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. Qwen technical report, 2023.

[9] Omkar Thawakar, Ashmal Vayani, Salman Khan, Hisham Cholakal, Rao M Anwer, Michael Felsberg, Tim Baldwin, Eric P Xing, and Fahad Shahbaz Khan. Mobillama: Towards accurate and lightweight fully transparent gpt. *arXiv preprint arXiv:2402.16840*, 2024.

[10] Marah Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Qin Cai, Martin Cai, Caio César Teodoro Mendes, Weizhu Chen, Vishrav Chaudhary, Dong Chen, Dongdong Chen, Yen-Chun Chen, Yi-Ling Chen, Parul Chopra, Xiyang Dai, Allie Del Giorno, Gustavo de Rosa, Matthew Dixon, Ronen Eldan, Victor Fragoso, Dan Iter, Mei Gao, Min Gao, Jianfeng Gao, Amit Garg, Abhishek Goswami, Suriya Gunasekar, Emman Haider, Junheng Hao, Russell J. Hewett, Jamie Huynh, Mojan Javaheripi, Xin Jin, Piero Kauffmann, Nikos Karampatziakis, Dongwoo Kim, Mahoud Khademi, Lev Kurilenko,

James R. Lee, Yin Tat Lee, Yuanzhi Li, Yunsheng Li, Chen Liang, Lars Liden, Ce Liu, Mengchen Liu, Weishung Liu, Eric Lin, Zeqi Lin, Chong Luo, Piyush Madan, Matt Mazzola, Arindam Mitra, Hardik Modi, Anh Nguyen, Brandon Norick, Barun Patra, Daniel Perez-Becker, Thomas Portet, Reid Pryzant, Heyang Qin, Marko Radmilac, Corby Rosset, Sambudha Roy, Olatunji Ruwase, Olli Saarikivi, Amin Saied, Adil Salim, Michael Santacroce, Shital Shah, Ning Shang, Hiteshi Sharma, Swadheen Shukla, Xia Song, Masahiro Tanaka, Andrea Tupini, Xin Wang, Lijuan Wang, Chunyu Wang, Yu Wang, Rachel Ward, Guanhua Wang, Philipp Witte, Haiping Wu, Michael Wyatt, Bin Xiao, Can Xu, Jiahang Xu, Weijian Xu, Sonali Yadav, Fan Yang, Jianwei Yang, Ziyi Yang, Yifan Yang, Donghan Yu, Lu Yuan, Chengruidong Zhang, Cyril Zhang, Jianwen Zhang, Li Lyna Zhang, Yi Zhang, Yue Zhang, Yunan Zhang, and Xiren Zhou. Phi-3 technical report: A highly capable language model locally on your phone, 2024.

[11] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.