CTR-BERT: Cost-effective knowledge distillation for billion-parameter teacher models

Aashiq Muhamed,* **Iman Keivanloo, Sujan Perera, James Mracek, Yi Xu, Qingjun Cui, Santosh Rajagopalan, Belinda Zeng, Trishul Chilimbi**
Amazon
{muhaaash, imankei, peresuja, jmracek, yxaamzn, qingjunc, syrgn, zengb, trishulc}@amazon.com

Abstract

While pre-trained large language models (LLM) like BERT have achieved state-of-the-art in several NLP tasks, their performance on tasks with additional grounding e.g. with numeric and categorical features is less studied. In this paper, we study the application of pre-trained LLM for Click-through-rate (CTR) prediction for product advertisement in e-commerce. This is challenging because the model needs to a) learn from language as well as tabular data features, b) maintain low-latency (<5 ms) at inference time, and c) adapt to constantly changing advertisement distribution. We first show that scaling the pre-trained language model to 1.5 billion parameters significantly improves performance over conventional CTR baselines. We then present CTR-BERT, a novel lightweight cache-friendly factorized model for CTR prediction that consists of twin-structured BERT-like encoders for text with a mechanism for late fusion for text and tabular features. We train the CTR-BERT model using cross-architecture knowledge distillation (KD) and empirically study the interaction between KD and distribution shift in this setting, by a) experimenting with pre-training, distillation pre-finetuning and fine-tuning strategies b) factorizing features based on their distribution shift time scales, that allows the model to readily adapt and be re-trained. Finally, we show that CTR-BERT significantly outperforms a traditional CTR baseline with a 2.3% relative ROC-AUC lift in offline experiments and a 2% CTR lift in an online experiment.

1 Introduction

The Transformer [1] enabled a paradigm shift in designing general-purpose scalable machine learning models for several natural language processing (NLP) and representation learning tasks. Transformer-based large language models (LLM) or foundation models [2] helped achieve state-of-the-art in several NLP tasks including question answering, natural language inference, sentiment analysis, etc. [3–5]. Traditional LLMs however rely on learning from text alone while several tasks require grounding in other modalities to learn robust representations [6]. CTR prediction for advertisement in e-commerce as an example task, requires strong grounding in contextual categorical and numeric data (e.g. historical sales data) in addition to natural language for learning generalizable representations. CTR prediction is challenging for additional reasons – the model must serve millions of requests per second at high throughput and low (<5 ms) latency. The feature and CTR distributions can also experience shifts due to special events, new campaigns, seasonality and other factors (e.g. a pandemic). To adapt to this shift, the model must be refreshed, often by retraining at a daily or hourly cadence.

While deep learning models have recently gained traction in CTR prediction [7], to meet latency requirements, the models are still relatively shallow by modern deep learning standards. The largest

*Corresponding author

35th Conference on Neural Information Processing Systems (NeurIPS 2021), Sydney, Australia.

model in DCN-V2 [8], for example uses <5 million parameters. In this paper, we show that scaling a single tower BERT-based language model helps in the CTR task: scaling to 1.5 billion parameters and larger data significantly improves model performance over conventional CTR baselines.

This large LLM, based on transformer blocks, however doesn't meet the low inference latency requirements for CTR prediction where the model must serve millions of requests per second. To address this challenge, we introduce CTR-BERT - a novel lightweight factorized model (70 million parameters) for CTR prediction that consists of twin-structured BERT-like encoders with a late mechanism fusion for text and tabular features. Factorizing the text and tabular features allows text features to be pre-computed offline and cached in memory. The inference-time cost is therefore from the late fusion layer only. As this fusion layer is a lightweight multi-layer perceptron (MLP), we can meet the strict latency and throughput requirements. We show that our factorized CTR-BERT (student) can be trained using *cross-architecture* distillation from a single tower LLM (teacher).

In addition to meeting latency requirements, a CTR model in e-commerce must adapt to distribution shift in features and CTR. This shift happens at different time scales due to seasonality, changes to ad campaigns and other factors. The predominant strategy to keep up with these shifts is to train predictive models continuously or at a regular cadence (e.g., hourly) on fresh data, in order to prevent them from becoming stale, which for a transformer-based model is quite expensive and infeasible. For example, re-training a 1.5 billion parameter model over a 1-month window requires at least 256 A100 GPUs for over 1 week and costs $100,000. In this work, we study the interaction of KD and distribution shift in the setting where the LLM teacher is trained once and then frozen. To do so, we define an in-distribution (ID) dataset composed of recent data and an out-of-distribution (OOD) dataset composed of stale data. In practice, we have abundant OOD data and limited ID data. The teacher is trained on a billion OOD datapoints and we study the student pretraining, distillation and fine-tuning strategies to maximize ID performance. Our proposed strategy reduces the training cost by 1/100. Our major contributions include:

First, we show that a single but very large language model (1.5 billion parameters) in the form of a single tower BERT can significantly boost CTR prediction performance in e-commerce advertisement.

Second, we introduce CTR-BERT, a novel lightweight factorized model for CTR prediction that consists of twin-structured BERT-like encoders with a mechanism for late fusion for text and tabular features. This architecture allows text based features to be pre-computed offline and cached in memory to achieve low-latency. CTR-BERT is trained from the teacher LLM via cross-architecture knowledge distillation and those factorizes input features whose distribution shifts over short time scales (encoded with twin-BERT encoder), separately from the tabular features that shift over long time scales. This enables the model to adapt to distribution shift without refreshing the BERT encoder.

Third, we study the interaction between KD and distribution shift by performing an empirical study to identify the best combination of student pretraining, distillation and finetuning strategies to adapt to data distribution shift. We find that using Masked Language Modeling (MLM) pretraining, distilling from the OOD teacher labels on OOD data followed by self-training on ID data achieves the best performance. In an online experiment, we show that CTR-BERT distilled using our strategy (despite a $> 12+$ months gap with the teacher) outperforms the baseline by 2% lift in CTR with no latency degradation. A CTR lift of 0.1% is considered significant improvement [8].

2 Related Work

Cross-architecture distillation: Cross-architecture distillation has recently become popular in the information retrieval literature. The goal is for the student model to achieve comparable effectiveness on a particular task but more efficiently (e.g., lower inference latencies, fewer model parameters, etc.). While knowledge distillation is traditionally model agnostic and researchers have explored this approach for many years, this has recently gained traction in modern Transformer architectures [9–12]. Our work is closest to [12] in that we distill from a cross-attention teacher to a twin-tower student. We differ from other works in that we focus on a task grounded with additional numeric and categorical tabular data features. Our student model therefore has multiple arms to encode text and tabular features, which to the best of our knowledge, has not been studied in prior work.

Knowledge distillation for domain adaptation: The interaction between knowledge distillation and domain adaptation is underexplored in literature. [13] performs unsupervised knowledge distillation

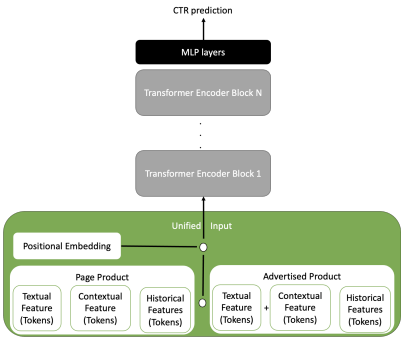

Figure 1: Pre-trained teacher LLM model architecture for CTR fine-tuning in e-commerce.

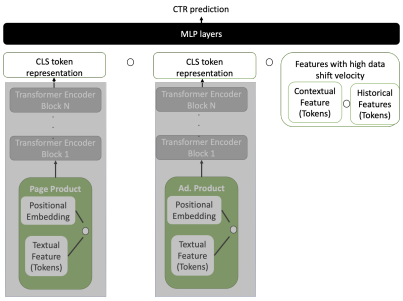

Figure 2: CTR-BERT: model has two BERT arms for the two inputs (Page and Ad products) and a late fusion layer for numeric/categorical features. Embeddings from each arm (gray) are now cache-able.

Table 1: Summary of CTR features for product advertisement in e-commerce

Feature group	Sample Feature	Feature Count	Distr Shift Time Scales
Textual features	Product title and description	4	Long (∼1 month)
Contextual features	Application name, Advertiser name	3	Medium (∼1 week)
Historical and sales features	Total sale value (for each product)	13	Short (∼ 1 hour)

from a teacher model to a student model, by leveraging both source and target unlabeled data. [14] leverages meta-learning to capture transferable knowledge across several unlabeled domains that can then be distilled. Our work differs in that we a) experiment with a massive 1.5B parameter BERT based teacher that is subsequently frozen, b) study the recommended pretraining, pre-finetuning and finetuning strategies when abundant *labeled* OOD data is available. [15] is the closest to our work in that they study the interaction between distillation and domain shift for Neural Machine Translation (NMT) using a 70M parameter teacher. Their work recommends adapting the teacher to the new domain for a second round of distillation which is infeasible in our setting.

3 Methodology

3.1 CTR prediction task

When a customer lands on a product page on the e-commerce website, advertisers compete in an auction to display their product on that page. We refer to the product on the page as the **Page Product** and each similar/relevant product displayed by advertisers at the bottom of the page as an **Ad product**. If y is binary random variable representing a customer click, the CTR prediction task is to predict the conditional click probability $P(y|x)$, given features x from a pair of Page and Ad products. Table 1 provides a high level summary of the various covariates. The covariates can also be categorized based on their distribution shift time scales. Textual features like product title change slowly, often remaining unchanged for periods longer than a month while features like sale value change rapidly as people purchase products every hour. We estimated the time shift scale by measuring the average time difference between such feature variations per feature averaged across all products in the training set.

3.2 CTR-BERT - A Factorized Cache-friendly Distilled Model

The teacher LLM model (Fig.1) is a single large BERT tower with an additional layer norm before the MLP layers. It is first pretrained using MLM on the train corpus and then finetuned for CTR prediction. During pretraining and finetuning, all features (textual, contextual, historical) are transformed into their string representation and concatenated along with their feature names.

Figure 2 presents our proposed CTR-BERT student model architecture that is ideal for the cached online inference scenario. It is a cache-friendly CTR model with separate arms for the Page product page and Advertised product. The text embeddings corresponding to the [CLS] token in each arm (highlighted in gray) can be computed and cached in advance for every product in the e-commerce catalog. At inference time CTR prediction, these [CLS] representations are retrieved from the cache

and concatenated with the tabular features before a late-fusion layer. The cost at inference is thus from the late fusion layer only. We use a 3-layer MLP as late-fusion layer, that incurs <3 ms latency.

In the CTR prediction problem, the distribution of advertisement and users that visit the Detail Page changes with time. To maintain the same accuracy, the model must be frequently refreshed to adapt to this distribution shift. However, finetuning a 1.5 billion parameter teacher is prohibitively expensive (100K USD for one experiment), difficult to stabilize and infeasible for an application like CTR where the model has to repeatedly adapt to the changing joint distribution. To adapt to this distribution shift the model must incur low cost penalty while retraining and generating embeddings. CTR-BERT uses about 70 million parameters which can be trained on 8 A100 GPUs in less than a day (<1000 USD). To further minimize the frequency of Transformer refresh, we factorize features based on their distribution shift time scales in Table 1. The BERT backbone is used to process textual features that shift over long time scales so that the frequency of re-training BERT can be reduced to 1 month, while the fusion MLP layer processing other features can be retrained everyday in production.

We train the student using *cross-architecture* distillation from the cross-attention teacher LLM model (Figure 1) to the CTR-BERT student model (Figure 2). We seek a BERT-CTR student p^s_θ parametrized by θ that is close to the LLM teacher p^t. Let the temperature softened outputs for the student/teacher network denoted by f be given by $p^f(\tau) = \text{softmax}(z^f/\tau)$, where z^f is the logits vector. To learn the optimal student θ^*, we minimize a combination of the cross entropy (CE) loss and Kullback Leibler (KL) divergence loss with the teacher [16].

$$\theta^* = \text{argmin}_\theta \, \mathbb{E}_{(x,y)\sim P}[(1-\alpha)L_{CE}(p^s_\theta(1), y) + \alpha L_{KL}(p^s_\theta(\tau), p^t(\tau))] \tag{1}$$

3.3 Initialization, Distillation and Fine-tuning strategies for distribution shift adaptation

While both adapting to distribution shift and knowledge distillation are widely-used, their interaction in LLMs is less understood. Motivated by the cost considerations in repeatedly finetuning LLMs, we study the interaction between knowledge distillation and distribution shift in the setting where a large teacher model is trained once on OOD data and subsequently frozen. Given a frozen teacher trained on OOD data, we ask the question: *What is the best ID task-specific training strategy for the student where there is abundant OOD labeled data and limited ID labeled data?*

This problem formulation mimics the CTR deployment setting where we have an abundance of stale OOD historical CTR data along with a small amount of recent ID CTR data. Using an OOD dataset from 2020 and an ID dataset from 2021, we examine the influence of various combinations of pretraining, distillation and fine-tuning strategies on the ID performance of the student. The student model is first initialized with a selected initialization strategy, then pre-finetuned with knowledge distillation using a specified distillation strategy and finally fine-tuned using a fine-tuning strategy. In particular, we examine the following strategies applied in sequence:

(i) Initialization strategy: (a) Random initialization, (b) Masked Language Modeling (MLM) recently suggested in [17], (c) Pretraining by supervised learning on labeled data.

(ii) Distillation as a pre-finetuning strategy: (a) No distillation, (b) OOD distillation where we train on soft labels from the OOD teacher on OOD data , (c) ID distillation where we distill using soft labels from the OOD teacher on ID data, using ID soft-label validation loss for early stopping. We posit that ID distillation can help the model see the ID covariates during pre-finetuning, to further help downstream ID performance.

(iii) Finetuning strategy: (a) Vanilla fine-tuning on ID data, (b) Self-training where we use the finetuned ID student to further label both the OOD and ID data, then use these labels for a second round of fine-tuning. Recent work in transfer learning[18, 19] suggest that self-training provides gains orthogonal to supervised and self-supervised pretraining in the low transfer-data learning regime.

4 Experiments and results

4.1 Dataset

Our CTR dataset is sampled from online traffic and is different from existing public CTR datasets in that it comprises of text features in addition to numeric/categorical features.
Out-of-distribution (OOD) data We sample random train-test splits from 2020 online traffic. As

Table 2: Evaluation results (ROC-AUC delta) on the OOD test set of a single tower BERT-based LLM (top) and BERT student models (bottom) for CTR prediction.

Model Name	Model size (num params)	OOD test ROC-AUC delta
3-layer MLP Baseline model	65 million	0.00%
3-layer BERT model	30 million	-6.73%
12-layer BERT model	500 million	-2.39%
24-layer BERT model	1.5 billion	+2.59%
BERT student (factorized, no num/cat feats)	70 million	+1.93%
BERT student (unfactorized, yes num/cat feats)	70 million	+2.47%
CTR-BERT student (factorized, yes num/cat feats)	70 million	+2.27%

the proportion of clicks is much lesser than non-clicks in the dataset, the two classes are balanced in the train set by downsampling the dominant (non-clicks) class. We do not artificially balance the test set which remains skewed towards the non-clicks in a 95:5 ratio. The train set comprises of 1 billion data points and the test and validation sets comprise about 25 million points each.
In-distribution (ID) data The train-test splits are sampled from 2021 online traffic and balanced the same way as OOD data. The train set comprises 200 million data points and the test and validation sets comprise 25 million points each.
Preprocessing All text data is preprocessed using the Sentencepiece tokenizer [20]. A Byte Pair Encoding [21] subword vocabulary of 32000 tokens is constructed from the train corpus.
Metrics and evaluation All metrics are reported on the test sets of the ID and OOD datasets. As the task is binary classification, we report the ROC-AUC frequently used for imbalanced classification.

4.2 Findings: Teacher and CTR-BERT student model design

To study the effect of teacher model size on CTR prediction, we trained three versions of the single tower BERT encoder from 30 million to 1.5 billion parameters. Table 2 (Top) summarizes the performance achieved by the three trained BERT models. We compare against a 3 layer-MLP baseline with ReLU activations carefully designed to meet the latency for the CTR task. When the number of parameters is <500 million, the BERT teacher is unable to outperform the MLP baseline. However, as we increase the model size to 1.5 billion parameters, the proposed model (Figure 1) outperforms the baseline by a large margin (2.59% ROC-AUC increase).

Table 2 (Bottom) compares various factorized twin-structured CTR-BERT architecture choices on the OOD test dataset. All student models were initialized with MLM and pre-finetuned with OOD distillation. We compare three CTR-BERT students - (a) factorized BERT with no numeric/categorical features, (b) unfactorized BERT where the numeric/categorical features are treated as text. While unfactorized students learn cross-feature representations, they cannot be efficiently refreshed and (c) our proposed factorized CTR-BERT model with text and numeric/categorical feats. We observe that (a) Factorized BERT student that does not use numeric/categorical features suffers a 0.5% performance drop relative to CTR-BERT that does. (b) The factorized CTR-BERT model suffers a 0.2% performance drop relative to an unfactorized BERT student model of the same size. (c) The factorized CTR-BERT student model experiences only a 0.30% ROC-AUC drop relative to the 1.5 billion parameter teacher while being 25 times smaller.

4.3 Findings: Pre-training, Distillation pre-finetuning, Finetuning strategy

Table 3 compares model performance for different pretraining, distillation and fine-tuning strategies. a) **Which pretraining strategy gives best results?** We compare the performance of the three pre-training strategies that correspond to a particular distillation strategy. When we examine approaches 1,2,3 for No distillation, approaches 4,5,6 for OOD distillation and approaches 7,8,9 for ID distillation, we see that the downstream performance of models pretrained with Random < Supervised learning < MLM. This trend is consistent across all approaches. Both Supervised learning and MLM improve model performance over random initialization which is unsurprising as both self-supervised pretraining and supervised training helps the model learn features that generalize to OOD data[5]. Our findings suggest that MLM is better than supervised learning even when *abundant* labeled data is available. While MLM as a self-supervised objective is popularly used to pretrain LLMs on unlabeled corpora, the performance gain over *supervised learning* is underexplored in literature.

Table 3: CTR-BERT student performance for the various choices of pretraining, pre-finetuning distillation and fine-tuning strategies. For distillation we report the OOD AUC delta which is the relative ROC-AUC on the OOD test set and ID soft AUC dekta which is the relative ROC-AUC on the ID softlabel test set. The ID softlabel set is created by generating ID softlabels using the OOD teacher. For finetuning we report the ROC-AUC relative to baseline on the ID test set.

Approach	Pre-training	Distillation	OOD AUC delta	ID soft AUC delta	Fine-tuning	ID AUC delta
3-layer MLP baseline	Random	No KD	0.00%	-	No fine-tuning	0.00%
Teacher	Random	No KD	+2.59%	-	No fine-tuning	-11.37%
1	Random	No KD	-	-	Vanilla	+0.52%
2	MLM	No KD	-	-	Vanilla	+1.03%
3	Supervised	No KD	-1.47%	-	Vanilla	+0.58%
4	Random	OOD KD	+2.02%	-	Vanilla	+1.83%
5	MLM	OOD KD	**+2.27%**	-	Vanilla	+2.02%
6	Supervised	OOD KD	+2.14%	-	Vanilla	+1.94%
7	Random	ID KD	-	0.00%	Vanilla	+0.70%
8	MLM	ID KD	-	+2.88%	Vanilla	+1.32%
9	Supervised	ID KD	-	+1.66%	Vanilla	+0.72%
10	MLM	OOD KD	**+2.27%**	-	Self-training	**+2.17%**

b) **Which distillation pre-finetuning strategy gives best results?** When we compare approaches 1,4,7; approaches 2,5,8 and approaches 3,6,9 we see across the different pre-training strategies that performance of models pre-finetuned with No distillation < ID distillation < OOD distillation. Both OOD distillation and ID distillation help learn from the OOD teacher, and boosts model performance over No distillation. This is consistent with recent literature [15]. Why does ID distillation perform worse than OOD distillation? We speculate that in the low ID data regime, to effectively learn from the OOD teacher, the student model needs a larger distribution support.

c) **Can we separate pretraining, pre-finetuning distillation and finetuning stages and train students in this three step sequence?** We observed that pretraining, distillation pre-finetuning and finetuning in general, do not interact. A model that had the highest performance in any one stage should be used in the next stage. Given a distillation strategy, say OOD distillation, we find (from approaches 4,5,6) that OOD ROC-AUC is a good proxy for final performance on ID ROC-AUC.

d) **Does self-training provide an additional boost?** We find that self-training after the best sequence of strategies (approach 5) boosts performance on the downstream ID task. Self-training is therefore an effective method to learn from labeled OOD data and the performance gain is complementary to gains from pre-training and distillation pre-finetuning. **When infeasible to adapt the teacher to distribution shift, we therefore recommend an MLM, OOD distillation, self-training strategy.**

e) **Does knowledge distillation boost performance over supervised learning on the OOD dataset?** If we had to pick between pre-training and distillation pre-finetuning, comparing approaches 2 (MLM), 3 (Supervised learning), and 4 (OOD distillation), we find that the best strategy to learn from the OOD dataset is to first train a large teacher (MLM, vanilla fine-tuning) and then distill this teacher to a small student model, which performs much better than MLM or supervised learning alone on the OOD dataset. This suggests that training or finetuning at least one LLM on OOD data can significantly boost performance on the ID task.

5 Online evaluation and conclusion

We performed an online experiment where CTR-BERT was tested within a major e-commerce platform. When compared against the MLP-based baseline, CTR-BERT improves CTR by 2% on average. Additionally, we observe a 5+% CTR lift on tail traffic which indicates that the CTR-BERT model generalizes better than the conventional model. In this work we address some significant challenges [7] in the application of LLMs to CTR by demonstrating that a cross-architecture distillation approach with the right pretraining strategy can make LLMs work for CTR prediction. In the future, we plan to further improve representation learning by grounding the LLM with image data.

References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in Advances in neural information processing systems, 2017, pp. 5998–6008.

[2] R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill et al., "On the opportunities and risks of foundation models," arXiv preprint arXiv:2108.07258, 2021.

[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," arXiv preprint arXiv:1810.04805, 2018.

[4] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," arXiv preprint arXiv:1910.10683, 2019.

[5] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell et al., "Language models are few-shot learners," arXiv preprint arXiv:2005.14165, 2020.

[6] E. M. Bender and A. Koller, "Climbing towards NLU: On meaning, form, and understanding in the age of data," in Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. Online: Association for Computational Linguistics, Jul. 2020, pp. 5185–5198. [Online]. Available: https://aclanthology.org/2020.acl-main.463

[7] W. Zhang, J. Qin, W. Guo, R. Tang, and X. He, "Deep learning for click-through rate estimation," arXiv preprint arXiv:2104.10584, 2021.

[8] R. Wang, R. Shivanna, D. Cheng, S. Jain, D. Lin, L. Hong, and E. Chi, "Dcn v2: Improved deep & cross network and practical lessons for web-scale learning to rank systems," in Proceedings of the Web Conference 2021, 2021, pp. 1785–1797.

[9] S. Sun, Y. Cheng, Z. Gan, and J. Liu, "Patient knowledge distillation for BERT model compression," in Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 4323–4332. [Online]. Available: https://aclanthology.org/D19-1441

[10] J. Lin, R. Nogueira, and A. Yates, "Pretrained transformers for text ranking: Bert and beyond," arXiv preprint arXiv:2010.06467, 2020.

[11] S. Hofstätter, S. Althammer, M. Schröder, M. Sertkan, and A. Hanbury, "Improving efficient neural ranking models with cross-architecture knowledge distillation," arXiv preprint arXiv:2010.02666, 2020.

[12] W. Lu, J. Jiao, and R. Zhang, "Twinbert: Distilling knowledge to twin-structured bert models for efficient retrieval," arXiv preprint arXiv:2002.06275, 2020.

[13] L. T. Nguyen-Meidine, E. Granger, M. Kiran, J. Dolz, and L.-A. Blais-Morin, "Joint progressive knowledge distillation and unsupervised domain adaptation," in 2020 International Joint Conference on Neural Networks (IJCNN), 2020, pp. 1–8.

[14] H. Pan, C. Wang, M. Qiu, Y. Zhang, Y. Li, and J. Huang, "Meta-kd: A meta knowledge distillation framework for language model compression across domains," arXiv preprint arXiv:2012.01266, 2020.

[15] M. A. Gordon and K. Duh, "Distill, adapt, distill: Training small, in-domain models for neural machine translation," arXiv preprint arXiv:2003.02877, 2020.

[16] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," arXiv preprint arXiv:1503.02531, 2015.

[17] I. Turc, M.-W. Chang, K. Lee, and K. Toutanova, "Well-read students learn better: On the importance of pre-training compact models," arXiv preprint arXiv:1908.08962, 2019.

[18] B. Zoph, G. Ghiasi, T.-Y. Lin, Y. Cui, H. Liu, E. D. Cubuk, and Q. V. Le, "Rethinking pre-training and self-training," arXiv preprint arXiv:2006.06882, 2020.

[19] J. Du, E. Grave, B. Gunel, V. Chaudhary, O. Celebi, M. Auli, V. Stoyanov, and A. Conneau, "Self-training improves pre-training for natural language understanding," arXiv preprint arXiv:2010.02194, 2020.

[20] T. Kudo and J. Richardson, "Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing," arXiv preprint arXiv:1808.06226, 2018.

[21] C. Wang, K. Cho, and J. Gu, "Neural machine translation with byte-level subwords," in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, no. 05, 2020, pp. 9154–9160.