High-efficiency Device-Cloud Collaborative Transformer Model

Penghao Jiang, Ke Xin, Chunxi Li, Yinsi Zhou
University of Technology Sydney
Australia

Abstract

Natural Language Processing (NLP) experts have had significant success with unsupervised language pre-training techniques. However, compared to typical NLP models, modern self-attention models require far more computational and memory resources than conventional NLP models, making pre-training or even fine-tuning them quite costly. It drastically restricts their success and uses in a variety of fields. To improve the efficiency, we propose Device-Cloud Collaborative Transformer for an efficient language model, which is a framework across cloud and device, and is designed to encourage learning of representations that generalize better to many different tasks. Specifically, we design Device-Cloud Collaborative Transformer architecture of large language models that benefits both cloud modeling and device modeling. Experimental results demonstrate the effectiveness of our proposed method.

1. Introduction

Unsupervised pre-training approaches [1–16] with self-attention models (a.k.a. Transformer) [17] have achieved great success in the field of Natural Language Processing (NLP). Recent works [6, 15, 18–20] point out that with more computational and memory resources invested in longer pretraining and/or larger models, the performance of pre-trained Transformer models consistently improves. For each downstream task, the parameters of pre-trained model are fine-tuned on each task-specific data independently. If there are N downstream tasks, a standard solution would produce N BERT models, each of which corresponds to a specific task. Even the smallest BERT model still has hundreds of millions of parameters, training one multi-task model to serve numerous downstream tasks is an efficient method to avoid deploying multiple copies of large models in practice. The typical multi-task method [14] appends different task-specific layers on top of some shared Transformer layers. All layers are optimized jointly with all tasks in the training stage. Intuitively, the Transformer layer learns the generic feature representations, whereas each task-specific layer learns to accomplish a particular task. However the Transformer layers still cost a lot of money to pre-train or even merely fine-tune since they need a lot more computational and memory resources than more conventional NLP models. Their potential for use and success in new domains is mostly limited by this.

Recent research [21–24] studied a split deployment between cloud and device, which might lower the inference cost and memory resources, in order to reduce the computational and memory resources for cloud centralized models. Such publications on mobile computing and the Internet of Things (IoTs) are accelerating the spread of computing [25]. The increasing capacity of mobile devices makes it possible to consider the intelligence services, such as online machine translation and online dialogue modeling, from cloud to device modeling. Recent research has examined the advantages of ubiquitous computing from a variety of perspectives, including privacy [22, 23, 26, 27], efficiency [28, 29], and applications [21, 30–32]. There have been several initiatives to condense BERT onto mobile devices with constrained resources. Unsupervised language pre-training models still have a challenge in figuring out how to combine the advantages of device modeling with cloud modeling to benefit both parties.

To overcome the challenges mentioned above, we propose Device-Cloud Collaborative Transformer framework, which is one general framework across cloud and device. Previous unsupervised language pre-training methods learn a centralized cloud model, models designed for resource-limited mobile devices learn a task specific device model. Different from these method, our Device-Cloud Collaborative Transformer method share parameters in cloud and learn task specific parameters in device.

In summary, the contributions of this paper are:

- We develop a Device-Cloud Collaborative Transformer architecture that benefits both cloud modeling and device modeling, in contrast to prior efforts that either solely take into account cloud modeling or on-device modeling.

- We propose Device-Cloud Collaborative Transformer

method to learn a better representation between device and cloud.

- Experiments demonstrate that language models in many tasks can be significantly improved by our proposed Device-Cloud Collaborative Transformer framework.

2. Related Works

Transformer [2, 3, 5–8, 10, 12–15, 17, 33] have achieved state-of-the-art performance in natural language processing domain, such as sequence-to-sequence modeling [17], BERT [2], GPT [33], XLNet [3], RoBERTa [5], ALBERT [7], T5 [10] and other recent works show that transformers pre-trained on large corpora learn language representations that can be transferred to a number of downstream tasks through fine-tuning.

Most of these previous works of transformers train or fine-tune a specific model for each of the tasks. For example, a pretrained transformer model BERT is fine-tuned separately on multiple downstream language tasks [2], which is extremely expensive on large computational and memory resources. Multi-task learning, which shares some parameters and learns task-specific parameters, is a good way to reduce the computational and memory resources. For example, a text-to-text transformer is jointly pretrained on different language tasks in T5 [10]. By combining the bottom layers of a transformer to create a multi-task language understanding model (MT-DNN) [14], it is suggested that the top layer be task-specific. In VILBERT-MT [34], 12 vision-and-language tasks were jointly learned with a multi-task transformer model based on VILBERT [35]. However, these methods are designed for cloud centralized model, which could not split deployment across cloud and device to reduce the computational resources.

Device Modeling is a key component of Edge AI [36], which reduces the onerous latency and incorporates rich features in a variety fileds. These works critically depend on the device capacity [37], efficient neural network architectures [38], the model compression technique [28], the distributed learning framework [22] and some split deployment strategies. Many hardware-efficient topologies, such MobileNets [39], have been suggested to minimize the computational budget and model size in device modeling. Model compression based on the network pruning or quantization [28, 40] is also useful to reduce the units, the channels or the value accuracy of the parameters. Federated Learning [22], is a distributed learning system that keeps the data in the local and only shares communicate gradients or parameters. Device modeling is used to compute the temporal training components and send to the cloud for averaging in Federated Learning [41]. Moreover, Lee *et al.* [42] describe how to use the mobile GPU to conduct deep neural network tasks, which considers the special limited of computing power, thermal constraints, and energy consumption in mobile. Some other works [21] also explore the divide-and-conquer deployment to reduce the device running time by moving computational prohibitive components to the cloud. Niu *et al.* [43] use a similar split in gradient computation and designed a tunable privacy to the local submodel. Different from these works, we focus on exploring the collaboration between the cloud modeling and the device modeling for mutual benefit of two sides in language models.

3. Device-Cloud Collaborative Transformer

Transformer Transformer layers [17], a highly modularized neural network, have attained cutting-edge performance across several tasks. The position-wise feed-forward network (P-FFN) and multi-head self-attention (S-Attn) sub-modules make up each Trans- former layer. Both sub-modules are encapsulated by residual connections and layer normalization. The computation of a single Transformer layer with a length T sequence of hidden states $\mathbf{h} = [h_1, \ldots, h_T]$ can be expressed as

$$\mathbf{h} \leftarrow \text{LayerNorm}(\mathbf{h} + \text{S-Attn}(\text{Q,K,V} = \mathbf{h})), \tag{1}$$

$$h_i \leftarrow \text{LayerNorm}(h_i + \text{P-FFN}(h_i)), \quad \forall i = 1, \cdots, T. \tag{2}$$

Masked Language Modeling The most often utilized pretraining target is the masked language modeling (MLM) suggested by BERT [2]. The masked language modeling objective constructs a corrupted sequence $\hat{\mathbf{x}}$ by randomly replacing 15% of the tokens of $\mathbf{x}$ with a special token `[mask]` , nd then trains a Transformer model [2] to reconstruct the original $\mathbf{x}$ based on $\hat{\mathbf{x}}$, where $\mathbf{x}$ is a length-T natural language sequence sampled from a large unlabeled set $\mathcal{D}$.

$$\begin{aligned} \max_{\theta} \ \mathcal{J}_{\text{MLM}}(\theta) &= \mathbb{E}_{\mathbf{x}\sim\mathcal{D}}\mathbb{E}_{\mathcal{I}} \sum_{i\in\mathcal{I}} \log P_\theta(x_i \mid \hat{\mathbf{x}}_{\mathcal{I}}) \\ &= \mathbb{E}_{\mathbf{x}\sim\mathcal{D}}\mathbb{E}_{\mathcal{I}} \sum_{i\in\mathcal{I}} \log \frac{\exp\left(e(x_i)^\top h_i(\hat{\mathbf{x}}_{\mathcal{I}})\right)}{\sum_{x'} \exp\left(e(x')^\top h_i(\hat{\mathbf{x}}_{\mathcal{I}})\right)}, \end{aligned} \tag{3}$$

where $\mathcal{I}$ is the positions of masked tokens, the subscript in $\hat{\mathbf{x}}_{\mathcal{I}}$ emphasizes its dependence on $\mathcal{I}$, $e(x)$ denotes the embedding of the token x, and $h_i(\hat{\mathbf{x}}_{\mathcal{I}})$ the last-layer hidden state at position i produced by the Transformer model. The entire model is finetuned in downstream activities after pretraining.

Given a language task, we aim to learn an encoder on the device side, and a decoder on the cloud side. Here, the output of device side is the input of cloud side. There aren't many works that take into account how to make device modeling and cloud modeling function for both sides simultaneously, as was indicated in earlier parts. However,

this is critical and meaningful, since the conventional centralized cloud model is fine-tuned on each downstream tasks separately, which is extremely expensive.

The key goal is to address the issue of computing efficiency and lower communication latencies in order to develop a comprehensive and effective framework between cloud and device for various tasks. Our model, which inherits the high capacity and optimization benefits of the Transformer architecture, uses a device encoder to reduce the sequence length of the hidden states while maintaining the overall skeleton of interleaved multi-head self-attention and position-wise feed-forward network.

The Device-Cloud Collaborative Transformer employs a powerful device Transformer encoder and a cloud Transformer decoder. It compresses a more meaningful representation to improve the performance and reduce communication latencies between devices and the cloud. The Device-Cloud Collaborative Transformer combines the device Transformer encoder with the cloud Transformer decoder. This combination provides a powerful and accurate learning system that can learn and make decisions based on the data it receives. The device transformer encoder first processes the raw data from the device and transforms it into a more meaningful representation.

The cloud Transformer decoder takes this representation and decodes it to make predictions on the data. The representation is compressed to reduce communication latencies between devices and cloud. The key to this compression lies in compressing the representation using a quantization technique called vector quantization. The vector quantization method involves dividing the data into groups called centroids, and each data point is then assigned to the nearest centroid. This process reduces the communication latency between devices and cloud, as instead of sending the raw data from the device to the cloud, we only send the compressed data. The compression process is:

$$Z_i = argmin_j ||x_i - c_j||^2 \tag{4}$$

where Z_i is the index of the nearest group or centroid, x_i is the i-th data point, and c_j is the j-th centroid. To decompress the data, we use the inverse of the vector quantization method, which is known as codebook mapping. Codebook mapping involves mapping the nearest centroid using a codebook table. This mapping is:

$$x_i = c_{Z_i} \tag{5}$$

The Device-Cloud Collaborative Transformer benefits from integrating the device Transformer encoder and the cloud Transformer decoder, in contrast to prior works that either exclusively consider the cloud modeling or on-device modeling. By employing vector quantization compression, the system is able to reduce communication latencies and learn meaningful representation between devices and the cloud.

4. Experiments

The proposed Device-Cloud Collaborative Transformer approach is empirically evaluated in this section by pretraining it and then finetuning it for downstream tasks. In keeping with earlier research, we analyze two typical pretraining settings:

- **Base scale**: The original BERT [2] utilized this option with 256-step batch sizes for pretraining models on Wikipedia and the Book Corpus. The Device-Cloud Collaborative Transformer and the traditional Transformer will be fairly compared in this environment, along with some ablation studies.

- **Large scale**: On the five datasets (Wikipedia + Book Corpus + ClueWeb + Gigaword + Common Crawl) utilized by XLNet [3] and ELECTRA [6], pretraining models for 500K steps with batch size 8K. We will compare Device-Cloud Collaborative Transformer trained at this scale with previous methods.

Datasets. We consider tasks include the GLUE benchmark for language understanding [44], 7 widely used text (sentiment / topic) classification tasks (IMDB, AD, DBpedia, Yelp-2, Yelp-5, Amazon-2, Amazon-5), SQuAD question answering task, and the RACE reading comprehension dataset [45]. We evaluate our method with these tasks:

- **CoLA:** [46] is the Corpus of Linguistic Acceptability. Identifying a sentence's grammar or lack thereof is the task at hand. The dataset includes 8.5k train instances taken from linguistic theory books and journal papers.

- **SST:** Stanford Sentiment Treebank [47]. Identifying if a statement has a favorable or negative attitude is the task. 67k train samples from movie reviews are included in the dataset.

- **MRPC:** Microsoft Research Paraphrase Corpus [48]. Predicting whether or not two statements are semantically comparable is the job at hand. 3.75 thousand instances from internet news sources make up the collection.

- **STS:** Semantic Textual Similarity [49]. On a scale from 1 to 5, predict how semantically similar two sentences are. The dataset includes 5.8 thousand train instances taken from recent headlines, captions for videos and images, and data from natural language inference.

- **QQP:** Quora Question Pairs [50]. Identifying semantic equivalentity between two queries is the problem

Hyperparameter	GLUE Value
Learning Rate	1e-4 for Base, 2e-4 for Large
Adam ϵ	1e-6
Adam β_1	0.9
Adam β_2	0.999
Learning rate decay	Linear
Warmup fraction	0.1
Attention Dropout	0.1
Dropout	0.1
Weight Decay	0.01

Table 1. Hyper-parameters for pretraining

Model size	IMDB	AG	DBpedia	Yelp2	Yelp5	Amazon2	Amazon5
BERT-Base	6.257	5.428	0.695	2.208	30.35	2.809	33.71
ELECTRA-Base	5.149	5.340	0.646	1.898	29.09	2.576	33.05
Ours	**4.354**	**5.254**	**0.603**	**1.784**	**28.43**	**2.543**	**32.35**

Table 2. Comparison of base models on the text classification dev set with MLM pretraining, *performances (the lower the better)*.

Model	CoLA	SST-2	MRPC	STS-B	QQP	MNLI	QNLI	RTE	GLUE-AVG
BERT-Base	55.2	91.5	91.1/87.8	88.1	87.2/90.6	82.7	90.0	64.6	81.3
ELECTRA-Base	60.5	93.6	92.4/89.2	89.4	88.2/91.3	86.4	92.5	75.0	84.7
Ours	**62.0**	**94.2**	**92.6/89.8**	**89.4**	**88.4/91.6**	**87.0**	**92.7**	**76.2**	**85.0**

Table 3. Comparison of base models on the GLUE dev set with MLM pretraining, *performances (the higher the better)*.

Model size	CoLA	SST-2	MRPC	STS-B	QQP	MNLI	QNLI	RTE	GLUE-AVG
BERT-Base	62.1	91.1	90.8/86.8	88.9	88.2/91.3	83.9	89.7	66.7	82.6
ELECTRA-Base	63.9	94.2	93.0/90.2	89.5	88.4/91.4	87.0	92.2	77.6	85.7
Ours	**64.0**	**94.3**	**93.3/90.4**	**90.0**	**88.7/91.6**	**87.2**	**92.3**	**78.1**	**86.0**

Table 4. Comparison of base models on the GLUE dev set with ELECTRA pretraining, *performances (the higher the better)*.

Model size	IMDB	AG	DBpedia	Yelp2	Yelp5	Amazon2	Amazon5
BERT-Base	6.220	5.395	0.674	2.287	30.16	2.759	33.57
ELECTRA-Base	4.924	5.342	0.671	1.913	29.00	2.523	32.85
Ours	**4.734**	**5.297**	**0.665**	**1.865**	**28.87**	**2.513**	**32.78**

Table 5. Comparison of base models on the text classification dev set with ELECTRA pretraining, *performances (the lower the better)*.

Model	**CoLA**	**SST**	**MRPC**	**STS**	**QQP**	**MNLI**	**QNLI**	**RTE**	**Avg.**
BERT	60.6	93.2	88.0	90.0	91.3	86.6	92.3	70.4	84.0
RoBERTa-100K	66.1	95.6	91.4	92.2	92.0	89.3	94.0	82.7	87.9
RoBERTa-500K	68.0	96.4	90.9	92.1	92.2	90.2	94.7	86.6	88.9
XLNet	69.0	97.0	90.8	92.2	92.3	90.8	94.9	85.9	89.1
ELECTRA-400K	69.3	96.0	90.6	92.1	92.4	90.5	94.5	86.8	89.0
ELECTRA-1.75M	69.1	96.9	90.8	92.6	92.4	90.9	95.0	88.0	89.5
Ours	**69.4**	**96.9**	**91.5**	**92.6**	**92.6**	**91.0**	**95.0**	**88.1**	**89.6**

Table 6. Comparison of large models on the GLUE dev set with ELECTRA pretraining, *performances (the higher the better)*.

Model	SQuAD2.0		SQuAD1.1	
	EM	F1	EM	F1
$\text{ROBERTA}_{\text{Large}}$ [5]	86.5	89.4	88.9	94.6
$\text{ELECTRA}_{\text{Large}}$ [6]	88.0	90.6	89.7	94.9
Ours	**88.2**	**90.8**	**90.1**	**95.0**
$\text{ROBERTA}_{\text{Base}}$ [5]	80.5	83.7	84.6	91.5
$\text{MPNet}_{\text{Base}}$	80.5	83.3	86.8	92.5
Ours	**83.2**	**85.7**	**87.1**	**93.0**

Table 7. SQuAD dev performance comparison.

at hand. The dataset includes 364k train instances from the Quora community website for question-and-answer exchange.

- **MNLI:** Multi-genre Natural Language Inference [51]. Predict if a premise implies a hypothesis, contradicts a hypothesis, or does neither, given a premise and a hypothesis statement. Three hundred ninety three thousand train samples from 10 sources make up the dataset.
- **QNLI:** Question Natural Language Inference is created from SQuAD [52]. Predicting whether a context sentence contains the response to a question phrase is the task at hand. The dataset includes 108k Wikipedia train samples.
- **RTE:** Recognizing Textual Entailment [53]. The aim is to determine if a premise implies a hypothesis given a premise statement and a hypothesis sentence. From a series of yearly textual entailment challenges, the dataset includes 2.5k train samples.
- **SQuAD:** A span of text from the associated reading passage serves as the response to each question in the Stanford Question Answering Dataset [52], which is a reading comprehension dataset made up of questions put by crowdworkers on a collection of Wikipedia articles. The question itself may also be unanswerable.
- **Race:** An extensive reading comprehension dataset called Race [45] contains more than 28,000 texts and around 100,000 questions. The information is compiled from middle school and high school students' responses to English exams given in China.

4.1. Base-scale Results

First, we compare the performance of the device-cloud collaborative transformer to that of the traditional Transformer when subjected to equivalent amounts of computation (i.e., FLOPs). For this, we take into account the big and basic model sizes for the typical Transformer. The results on the GLUE benchmark and text classification are shown in Table 2 and 3 respectively, based on the MLM pretraining goal.

Here, we discover that our Device-Cloud Collaborative Transformer beats the traditional Transformer in the majority of workloads with equivalent or less FLOPs. We also take ELECTRA into consideration for pretraining in order to further evaluate the universality of Device-Cloud Collaborative Transformer. Table 4 and 5 provide a summary of the findings. Overall, we observe a similar pattern, even if the increase on the GLUE benchmark is a little lower.

4.2. Large-scale Results

Considering the positive outcomes of Device-Cloud Collaborative Transformer at base-scale, we now investigate training Device-Cloud Collaborative Transformer under the large-scale scenario and compare it to earlier models pretrained in comparable situations. We shall employ the ELECTRA goal for all large-scale trials due to ELECTRA's marginally greater performance over MLM.

We examine the finetuning performance on the GLUE benchmark in Table 6 using the pretrained Device-Cloud Collaborative Transformer model of various sizes. Our technique outperforms the equivalent baselines in the majority of jobs, mirroring the base-scale findings, indicating the strong scalability of our suggested Device-Cloud Collaborative Transformer.

On the SQuAD datasets, we hone our methodology, and we contrast it with earlier models in Table 7. Likewise, we may conclude that our Device-Cloud Collaborative Transformer performs better than benchmarks.

5. Conclusion

Natural language processing (NLP) experts have had significant success with unsupervised language pre-training techniques. Modern self-attention models demand far more FLOPs and memory resources than conventional NLP models, therefore pretraining or even just fine-tuning them is very costly. Device-Cloud Collaborative Transformer, which is intended to promote the learning of representations that generalize more effectively to a wide range of activities, is proposed for an efficient language model in order to

increase efficiency. In particular, we create a device architecture that not only has a reduced resource-to-performance ratio but also makes use of both combined device and cloud modeling. Additionally, we take into account a gradient normalization approach to automatically balance training in deep multitask models by dynamically adjusting gradient magnitudes in order to prevent task interference or negative transfer for language models. Experimental results show the effectiveness of proposed Device-Cloud Collaborative Transformer.

References

[1] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," *arXiv preprint arXiv:1802.05365*, 2018. 1

[2] 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. 1, 2, 3

[3] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," in *Advances in neural information processing systems*, 2019, pp. 5754–5764. 1, 2, 3

[4] T. Xiao, Z. Chen, Z. Guo, Z. Zhuang, and S. Wang, "Decoupled self-supervised learning for graphs," in *Advances in Neural Information Processing Systems*. 1

[5] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019. 1, 2, 5

[6] K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning, "Electra: Pre-training text encoders as discriminators rather than generators," *arXiv preprint arXiv:2003.10555*, 2020. 1, 2, 3, 5

[7] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "Albert: A lite bert for self-supervised learning of language representations," *arXiv preprint arXiv:1909.11942*, 2019. 1, 2

[8] L. Kong, C. d. M. d'Autume, W. Ling, L. Yu, Z. Dai, and D. Yogatama, "A mutual information maximization perspective of language representation learning," *arXiv preprint arXiv:1910.08350*, 2019. 1, 2

[9] T. Xiao, Z. Chen, D. Wang, and S. Wang, "Learning how to propagate messages in graph neural networks," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021, pp. 1894–1903. 1

[10] 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. 1, 2

[11] Z. Chen and D. Wang, "Multi-initialization meta-learning with domain adaptation," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 1390–1394. 1

[12] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," *arXiv preprint arXiv:1910.13461*, 2019. 1, 2

[13] K. Song, X. Tan, T. Qin, J. Lu, and T.-Y. Liu, "Mass: Masked sequence to sequence pre-training for language generation," *arXiv preprint arXiv:1905.02450*, 2019. 1, 2

[14] X. Liu, P. He, W. Chen, and J. Gao, "Multi-task deep neural networks for natural language understanding," *arXiv preprint arXiv:1901.11504*, 2019. 1, 2

[15] K. Song, X. Tan, T. Qin, J. Lu, and T.-Y. Liu, "Mpnet: Masked and permuted pre-training for language understanding," *arXiv preprint arXiv:2004.09297*, 2020. 1, 2

[16] Z. Chen, S. Gai, and D. Wang, "Deep tensor factorization for multi-criteria recommender systems," in *2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, December 9-12, 2019*, 2019, pp. 1046–1051. 1

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017. 1, 2

[18] Z. Dai, G. Lai, Y. Yang, and Q. Le, "Funnel-transformer: Filtering out sequential redundancy for efficient language processing," *Advances in neural information processing systems*, vol. 33, pp. 4271–4282, 2020. 1

[19] T. Xiao, Z. Chen, and S. Wang, "Representation matters when learning from biased feedback in recommendation," in *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, 2022, pp. 2220–2229. 1

[20] Z. Chen, D. Wang, and S. Yin, "Improving cold-start recommendation via multi-prior meta-learning," in *43rd European Conference on Information Retrieval, ECIR 2021*, 2021. 1

[21] Y. Gong, Z. Jiang, Y. Feng, B. Hu, K. Zhao, Q. Liu, and W. Ou, "Edgerec: Recommender system on edge in mobile taobao," in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 2020, pp. 2477–2484. 1, 2

[22] J. Konečný, H. B. McMahan, F. X. Yu, P. Richtarik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," in *ICLR*, 2016. 1, 2

[23] S. P. Karimireddy, S. Kale, M. Mohri, S. J. Reddi, S. U. Stich, and A. T. Suresh, "Scaffold: Stochastic controlled averaging for on-device federated learning," *ICML*, 2020. 1

[24] Z. Chen, J. Ge, H. Zhan, S. Huang, and D. Wang, "Pareto self-supervised training for few-shot learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 13 663–13 672. 1

[25] M. Satyanarayanan, "The emergence of edge computing," *Computer*, vol. 50, no. 1, pp. 30–39, 2017. 1

[26] I. Bistritz, A. Mann, and N. Bambos, "Distributed distillation for on-device learning," *Advances in Neural Information Processing Systems*, vol. 33, 2020. 1

[27] Z. Chen, T. Xiao, and K. Kuang, "Ba-gnn: On learning bias-aware graph neural network," in *2022 IEEE 38th International Conference on Data Engineering (ICDE)*. IEEE, 2022, pp. 3012–3024. 1

[28] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," *ICLR*, 2016. 1, 2

[29] H. Cai, C. Gan, L. Zhu, and S. Han, "Tinytl: Reduce memory, not parameters for efficient on-device learning," *Advances in Neural Information Processing Systems*, vol. 33, 2020. 1

[30] P. Sundaramoorthy, G. K. Gudur, M. R. Moorthy, R. N. Bhandari, and V. Vijayaraghavan, "Harnet: Towards on-device incremental learning using deep ensembles on constrained devices," in *Proceedings of the 2nd International Workshop on Embedded and Mobile Deep Learning*, 2018, pp. 31–36. 1

[31] X. Dai, I. Spasić, B. Meyer, S. Chapman, and F. Andres, "Machine learning on mobile: An on-device inference app for skin cancer detection," in *2019 Fourth International Conference on Fog and Mobile Edge Computing (FMEC)*. IEEE, 2019, pp. 301–305. 1

[32] Z. Chen, Z. Xu, and D. Wang, "Deep transfer tensor decomposition with orthogonal constraint for recommender systems," in *The Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI*, vol. 2021, 2021, p. 3. 1

[33] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020. 2

[34] J. Lu, V. Goswami, M. Rohrbach, D. Parikh, and S. Lee, "12-in-1: Multi-task vision and language representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 10 437–10 446. 2

[35] J. Lu, D. Batra, D. Parikh, and S. Lee, "Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks," *Advances in neural information processing systems*, vol. 32, 2019. 2

[36] I. Stoica, D. Song, R. A. Popa, D. Patterson, M. W. Mahoney, R. Katz, A. D. Joseph, M. Jordan, J. M. Hellerstein, J. E. Gonzalez *et al.*, "A berkeley view of systems challenges for ai," *arXiv preprint arXiv:1712.05855*, 2017. 2

[37] G. Bedi, G. K. Venayagamoorthy, R. Singh, R. R. Brooks, and K.-C. Wang, "Review of internet of things (iot) in electric power and energy systems," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 847–870, 2018. 2

[38] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," *Advances in neural information processing systems*, vol. 28, pp. 1135–1143, 2015. 2

[39] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *ICLR*, 2017. 2

[40] H. Cai, C. Gan, T. Wang, Z. Zhang, and S. Han, "Once for all: Train one network and specialize it for efficient deployment," in *ICLR*, 2020. 2

[41] A. E. Eshratifar, M. S. Abrishami, and M. Pedram, "Jointdnn: an efficient training and inference engine for intelligent mobile cloud computing services," *IEEE Transactions on Mobile Computing*, 2019. 2

[42] J. Lee, N. Chirkov, E. Ignasheva, Y. Pisarchyk, M. Shieh, F. Riccardi, R. Sarokin, A. Kulik, and M. Grundmann, "On-device neural net inference with mobile gpus," *arXiv preprint arXiv:1907.01989*, 2019. 2

[43] C. Niu, F. Wu, S. Tang, L. Hua, R. Jia, C. Lv, Z. Wu, and G. Chen, "Billion-scale federated learning on mobile clients: A submodel design with tunable privacy," in *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*, 2020, pp. 1–14. 2

[44] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, "Glue: A multi-task benchmark and analysis platform for natural language understanding," *arXiv preprint arXiv:1804.07461*, 2018. 3

[45] G. Lai, Q. Xie, H. Liu, Y. Yang, and E. Hovy, "Race: Large-scale reading comprehension dataset from examinations," *arXiv preprint arXiv:1704.04683*, 2017. 3, 5

[46] A. Warstadt, A. Singh, and S. R. Bowman, "Neural network acceptability judgments," *arXiv preprint arXiv:1805.12471*, 2018. 3

[47] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," in *EMNLP*, 2013. 3

[48] W. B. Dolan and C. Brockett, "Automatically constructing a corpus of sentential paraphrases," in *IWP@IJCNLP*, 2005. 3

[49] D. M. Cer, M. T. Diab, E. Agirre, I. Lopez-Gazpio, and L. Specia, "Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation," in *SemEval@ACL*, 2017. 3

[50] S. Iyer, N. Dandekar, and K. Csernai, "First Quora dataset release: Question pairs," 2017. [Online]. Available: https://data.quora.com/First-Quora-Dataset-Release-Question-Pairs 3

[51] A. Williams, N. Nangia, and S. R. Bowman, "A broad-coverage challenge corpus for sentence understanding through inference," in *NAACL-HLT*, 2018. 5

[52] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. S. Liang, "Squad: 100, 000+ questions for machine comprehension of text," in *EMNLP*, 2016. 5

[53] D. Giampiccolo, B. Magnini, I. Dagan, and W. B. Dolan, "The third pascal recognizing textual entailment challenge," in *ACL-PASCAL@ACL*, 2007. 5