LadaBERT: Lightweight Adaptation of BERT through Hybrid Model Compression

Abstract

BERT is a cutting-edge language representation model pre-trained by a large corpus, which achieves superior performances on various natural language understanding tasks. However, a major blocking issue of applying BERT to online services is that it is memory-intensive and leads to unsatisfactory latency of user requests, raising the necessity of model compression. Existing solutions leverage the knowledge distillation framework to learn a smaller model that imitates the behaviors of BERT. However, the training procedure of knowledge distillation is expensive itself as it requires sufficient training data to imitate the teacher model. In this paper, we address this issue by proposing a hybrid solution named LadaBERT (Lightweight adaptation of BERT through hybrid model compression), which combines the advantages of different model compression methods, including weight pruning, matrix factorization and knowledge distillation. LadaBERT achieves state-of-the-art accuracy on various public datasets while the training overheads can be reduced by an order of magnitude.

Introduction

The pre-trained language model, BERT BIBREF0 has led to a big breakthrough in various kinds of natural language understanding tasks. Ideally, people can start from a pre-trained BERT checkpoint and fine-tune it on a specific downstream task. However, the original BERT models are memory-exhaustive and latency-prohibitive to be served in embedded devices or CPU-based online environments. As the memory and latency constraints vary in different scenarios, the pre-trained BERT model should be adaptive to different requirements with accuracy retained to the largest extent. Existing BERT-oriented model compression solutions largely depend on knowledge distillation BIBREF1, which is inefficient and resource-consuming because a large training corpus is required to learn the behaviors of a teacher. For

example, DistilBERT BIBREF2 is re-trained on the same corpus as pre-training a vanilla BERT from scratch; and TinyBERT BIBREF3 utilizes expensive data augmentation to fit the distillation target. The costs of these model compression methods are as large as pre-training and unaffordable for low-resource settings. Therefore, it is straight-forward to ask, can we design a lightweight method to generate adaptive models with comparable accuracy using significantly less time and resource consumption? In this paper, we propose LadaBERT (Lightweight adaptation of BERT through hybrid model compression) to tackle the raised questions. Specifically, LadaBERT is based on an iterative hybrid model compression framework consisting of weighting pruning, matrix factorization and knowledge distillation. Initially, the architecture and weights of student model are inherited from the BERT teacher. In each iteration, the student model is first compressed by a small ratio based on weight pruning and matrix factorization, and is then fine-tuned under the guidance of teacher model through knowledge distillation. Because weight pruning and matrix factorization help to generate better initial and intermediate status

in the knowledge distillation iterations, the accuracy and efficiency of model compression can be greatly improved.

We conduct extensive experiments on five public datasets of natural language understanding. As an example, the performance comparison of LadaBERT and state-of-the-art models on MNLI-m dataset is illustrated in Figure FIGREF1. We can see that LadaBERT outperforms other BERT-oriented model compression baselines at various model compression ratios. Especially, LadaBERT-1 outperforms BERT-PKD significantly under $2.5\times $ compression ratio, and LadaBERT-3 outperforms TinyBERT under $7.5\times $ compression ratio while the training speed is accelerated by an order of magnitude.

The rest of this paper is organized as follows. First, we summarizes the related works of model compression and their applications to BERT in Section SECREF2. Then, the methodology of LadaBERT is introduced in Section SECREF3, and experimental results are presented in Section SECREF4. At last,

we conclude this work and discuss future works in Section SECREF5.

Related Work

Deep Neural Networks (DNNs) have achieved great success in many areas in recent years, but the memory consumption and computational cost expand greatly with the growing complexity of models. Therefore, model compression has become an indispensable technique for practice, especially in low-resource settings. In this section, we review the current progresses of model compression techniques briefly, which can be divided into four categories, namely weight pruning, matrix factorization, weight quantization and knowledge distillation. We also present hybrid approaches and the applications of model compression to pre-trained BERT models.

Related Work ::: Weight pruning

Numerous researches have shown that removing a large portion of connections or neurons does not cause significant performance drop in deep neural network models BIBREF4, BIBREF5, BIBREF6, BIBREF7. For example, Han et al. BIBREF4 proposed a method to reduce the storage and computation of neural networks by removing unimportant connections, resulting in sparse networks without affecting the model accuracy. Li et al. BIBREF5 presented an acceleration method for convolution neural network by pruning whole filters together with their connecting filter maps. This approach does not generate sparse connectivity patterns and brings much larger acceleration ratio with existing BLAS libraries for dense matrix multiplications. Ye et al. BIBREF8 argued that small weights are in fact important for preserving the performance of a model, and Hu et al. BIBREF6 alleviated this problem by a data-driven approach that pruned zero-activation neurons iteratively based on intermediate feature maps. Zhu and Gupta BIBREF7 empirically compared large-sparse models with smaller dense models of similar parameter sizes and found that large sparse models performed better consistently. In addition,

sparsity-induced models BIBREF9, BIBREF10, BIBREF11 can be regarded as similar methods as pruning. For example, Wen et al. BIBREF9 applied group lasso as a regularizer at training time, and Louizos et al. BIBREF10 learned sparse neural networks through l_0 regularization.

Related Work ::: Matrix factorization

The goal of matrix factorization is to decompose a matrix into the product of two matrices in lower dimensions, and Singular Value Decomposition (SVD) is a popular way of matrix factorization that generalizes the eigendecomposition of a square normal matrix to a $m \times n$ matrix. It has been proved that SVD is the best approximation of a matrix given the rank r under Frobenius norm BIBREF12. Matrix factorization was widely studied in the deep learning domain for model compression and acceleration BIBREF13, BIBREF14, BIBREF15. Sainath et al BIBREF13 explored a low-rank matrix factorization method of DNN layers for acoustic modeling. Xu et al. BIBREF14, BIBREF15 applied singular value decomposition to deep neural network acoustic models and achieved comparable performances with state-of-the-art models through much fewer parameters. GroupReduce BIBREF16 focused on the compression of neural language models and applied low-rank matrix approximation to vocabulary-partition. Acharya et al. BIBREF17 compressed the word embedding layer via matrix factorization and achieved promising results in text classification. Winata et al. BIBREF18 carried out experiments for low-rank matrix factorization on different NLP tasks and demonstrated that it was more effective in general than weight pruning.

Related Work ::: Weight quantization

Weight quantization is a common technique for compressing deep neural networks, which aims to reduce the number of bits to represent every weight in the model. In a neural network, parameters are stacked into clusters, and the parameters in the same cluster share the same value. With weight quantization, the

weights can be reduced to at most 1-bit binary value from 32-bits floating point numbers. Zhou et al. BIBREF19 showed that quantizing weights to 8-bits does not hurt the performance, and Binarized Neural Networks BIBREF20 contained binary weights and activations of only one bit. Incremental Network Quantization BIBREF21 converted a pre-trained full-precision neural network into low-precision counterpart through three interdependent operations: weight partition, groupwise quantization and re-training. Variational Network Quantization BIBREF22 formulated the problem of network quantization as a variational inference problem. Moreover, Choi et al. BIBREF23 investigated the drawbacks of conventional quantization methods based on k-means and proposed a Hessian-weighted k-means clustering algorithm as the solution.

Related Work ::: Knowledge distillation

Knowledge distillation is first proposed by BIBREF1, which trains a compact or smaller model to approximate the function learned by a large and complex model. A preliminary step of knowledge distillation is to train a deep network (the teacher model) that automatically generates soft labels for training instances. This “synthetic" label is then used to train a smaller network (the student model), which assimilates the function that is learned by the teacher model. Chen et al. BIBREF24 successfully applied knowledge distillation to object detection tasks by introducing several modifications, including a weighted cross-entropy loss, a teacher bounded loss, and adaptation layers to model intermediate teacher distributions. Li et al. BIBREF25 developed a framework to learn from noisy labels, where the knowledge learned from a clean dataset and semantic knowledge graph were leveraged to correct the wrong labels. Anil et al. BIBREF26 proposed online distillation, a variant of knowledge distillation which enabled extra parallelism for training large-scale data. In addition, knowledge distillation is also useful for aggregating model ensembles into a single model by treating the ensemble model as a teacher.

Related Work ::: Hybrid approach

To improve the performance of model compression, there are many attempts to conduct hybrid model compression method that combines more than one category of algorithms. Han et al. BIBREF27 combined quantization, hamming coding and weight pruning to conduct model compression on image classification tasks. Yu et al. BIBREF28 proposed a unified framework for low-rank and sparse decomposition of weight matrices with feature map reconstructions. Polino et al. BIBREF29 advocated a combination of distillation and quantization techniques and proposed two hybrid models, i.e., quantified distillation and differentiable quantization to address this problem. Li et al., BIBREF30 compressed DNN-based acoustic model through knowledge distillation and pruning. NNCF BIBREF31 provided a neural network compression framework that supported an integration of various model compression methods to generate more lightweight networks and achieved state-of-the-art performances in terms of a trade-off between accuracy and efficiency. In BIBREF32, an AutoML pipeline was adopted for model compression. It leveraged reinforcement learning to search for the best model compression strategy among multiple combinatorial configurations.

Related Work ::: BERT model compression

In the natural language processing community, there is a growing interest recently to study BERT-oriented model compression for shipping its performance gain into latency-critical or low-resource scenarios. Most existing works focus on knowledge distillation. For instance, BERT-PKD BIBREF33 is a patient knowledge distillation approach that compresses the original BERT model into a lightweight shallow network. Different from traditional knowledge distillation methods, BERT-PKD enables an exploitation of rich information in the teacher's hidden layers by utilizing a layer-wise distillation constraint. DistillBERT BIBREF2 pre-trains a smaller general-purpose language model on the same corpus as vanilla BERT. Distilled BiLSTM BIBREF34 adopts a single-layer BiLSTM as the student model and achieves comparable results with ELMo BIBREF35 through much fewer parameters and less inference time. TinyBERT BIBREF3 reports the best-ever performance on BERT model compression, which

exploits a novel attention-based distillation schema that encourages the linguistic knowledge in teacher to be well transferred into the student model. It adopts a two-stage learning framework, including general distillation (pre-training from scratch via distillation loss) and task-specific distillation with data augmentation. Both procedures require huge resources and long training times (from several days to weeks), which is cumbersome for industrial applications. Therefore, we are aiming to explore more lightweight solutions in this paper.

Lightweight Adaptation of BERT ::: Overview

The overall pipeline of LadaBERT (Lightweight Adaptation of BERT) is illustrated in Figure FIGREF8. As shown in the figure, the pre-trained BERT model (e.g., BERT-Base) is served as the teacher as well as the initial status of the student model. Then, the student model is compressed towards smaller parameter size through a hybrid model compression framework in an iterative manner until the target compression ratio is reached. Concretely, in each iteration, the parameter size of student model is first reduced by $1-\Delta $ based on weight pruning and matrix factorization, and then the parameters are fine-tuned by the loss function of knowledge distillation. The motivation behind is that matrix factorization and weight pruning are complementary with each other. Matrix factorization calculates the optimal approximation under a certain rank, while weight pruning introduces additional sparsity to the decomposed matrices. Moreover, weight pruning and matrix factorization generates better initial and intermediate status of the student model, which improve the efficiency and effectiveness of knowledge distillation. In the following subsections, we will introduce the algorithms in detail.

Lightweight Adaptation of BERT ::: Overview ::: Matrix factorization

We use Singular Value Decomposition (SVD) for matrix factorization. Each parameter matrix, including the embedding layer are compressed by SVD. Without loss generality, we assume a matrix of parameters

${W} \in \mathbb {R}^{m\times n}$, the singular value decomposition of which can be written as:

where ${U} \in \mathbb {R}^{m \times p}$ and ${V} \in \mathbb {R}^{p \times n}$. ${\Sigma } =diag(\sigma _1,\sigma _2,\ldots ,\sigma _p)$ is a diagonal matrix composed of singular values and p is the full rank of W satisfying $p \le min(m, n)$.

To compress this weight matrix, we select a lower rank r. The diagonal matrix ${\Sigma }$ is truncated by selecting the top r singular values. i.e., ${\Sigma }_r =diag(\sigma _1, \sigma _2,\ldots ,\sigma _r)$, while ${U}$ and ${V}$ are also truncated by selecting the top r columns and rows respectively, resulting in ${U}_r \in \mathbb {R}^{m\times r}$ and ${V}_r \in \mathbb {R}^{r\times n}$.

Thus, low-rank matrix approximation of ${W}$ can be formulated as:

In this way, the original weight matrix W is decomposed by the multiplication of two smaller matrices, where ${A}={U}_r\sqrt{{\Sigma }_r} \in \mathbb {R}^{n\times r}$ and ${B}={V}_r\sqrt{{\Sigma }_r} \in \mathbb {R}^{m\times r}$. These two matrices are initialized by SVD and will be further tuned during training.

Given a rank $r \le min(m, n)$, the compression ratio of matrix factorization is defined as:

Therefore, for a target model compression ratio P_{svd}, the desired rank r can be calculated by:

Lightweight Adaptation of BERT ::: Overview ::: Weight pruning

Weight pruning BIBREF4 is an unstructured compression method that induces desirable sparsity for a neural network model. For a neural network $f({x; \theta })$ with parameters $\theta $, weight pruning

finds a binary mask ${M} \in \lbrace 0, 1\rbrace ^{|\theta |}$ subject to a given sparsity ratio, P_{weight}. The neural network after pruning will be $f({x; M \cdot \theta })$, where the non-zero parameter size is $||{M}||_1 = P_{weight}\cdot |\theta |$, where $|\theta |$ is the number of parameters in $\theta $. For example, when $P_m = 0.3$, there are 70% zeros and 30% ones in the mask ${m}$. We adopt a simple pruning strategy in our implementation: the binary mask is generated by setting the smallest weights to zeros BIBREF36.

To combine the benefits of weight pruning with matrix factorization, we leverage a hybrid approach that applies weight pruning on the basis of decomposed matrices generated by SVD. Following Equation (DISPLAY_FORM12), SVD-based matrix factorization for any weight matrix ${W}$ can be written as: ${W}_{svd}={A}_{m\times r}{B}_{n\times r}^T$. Then, weight pruning is applied on the decomposed matrices ${A} \in \mathbb {R}^{m \times r}$ and ${B} \in \mathbb {R}^{n \times r}$ separately. The weight matrix after hybrid compression is denoted by:

where ${M_A}$ and ${M_B}$ are binary masks derived by the weight pruning algorithm with compression ratio P_{weight}. The compression ratio of this hybrid approach can be calculated by:

In LadaBERT, the hybrid compression produce is applied to each layer of the pre-trained BERT model. Given an overall model compression target P, the following constraint should be satisfied:

where $|\theta |$ is the total number of model parameters and P is the target compression ratio; $|\theta _{embd}|$ denotes the parameter number of embedding layer, which has a relative compression ratio of P_embd, and $|\theta _{encd}|$ denotes the number of parameters of all layers in BERT encoder, which have a compression ratio of P_{hybrid}. The classification layer (often MLP layer with Softmax activation) has a small parameter size ($|\theta _{cls}|$), so it is not modified in the model compression procedure. In the experiments, these fine-grained compression ratios can be optimized by

random search on the validation data.

Lightweight Adaptation of BERT ::: Knowledge distillation

Knowledge distillation (KD) has been widely used to transfer knowledge from a large teacher model to a smaller student model. In other words, the student model mimics the behavior of the teacher model by minimize the knowledge distillation loss functions. Various types of knowledge distillation can be employed at different sub-layers. Generally, all types of knowledge distillation can be modeled as minimizing the following loss function:

Where x indicates a sample input and $\mathcal {X}$ is the training dataset. $f^{(s)}({x})$ and $f^{(t)}({x})$ represent intermediate outputs or weight matrices for the student model and teacher model correspondingly. $L(\cdot)$ represents for a loss function which can be carefully defined for different types of knowledge distillation. We follow the recent technique proposed by TinyBERT BIBREF3, which applies knowledge distillation constraints upon embedding, self-attention, hidden representation and prediction levels. Concretely, there are four types of knowledge distillation constraints as follows:

Embedding-layer distillation is performed upon the embedding layer. $f({x}) \in \mathbb {R}^{n \times d}$ represents for the word embedding output for input x, where n is the input word length and d is the dimension of word embedding. Mean Squared Error (MSE) is adopted as the loss function $L(\cdot)$.

Attention-layer distillation is performed upon the self-attention sub-layer. $f({x}) = \lbrace a_{ij}\rbrace \in \mathbb {R}^{n \times n}$ represents the attention output for each self-attention sub-layer, and $L(\cdot)$ denotes MSE loss function.

Hidden-layer Distillation is performed at each fully-connected sub-layer in the Transformer architectures.

$f({x})$ denotes the output representation of the corresponding sub-layer, and $L(\cdot)$ also adopts MSE loss function.

Prediction-layer distillation makes the student model to learns the predictions from a teacher model directly. It is identical to the vanilla form of knowledge distillation BIBREF1. It takes the soft cross-entropy loss function, which is formulated as:

where $f^t({x})$ and $f^s({x})$ are the predictive logits of teacher and student models respectively.

Experiments ::: Datasets & Baselines

We compare LadaBERT with state-of-the-art model compression approaches on five public datasets of different tasks of natural language understanding, including sentiment classification (SST-2), natural language inference (MNLI-m, MNLI-mm, QNLI) and pairwise semantic equivalence (QQP). The statistics of these datasets are described in Table TABREF27.

The baseline approaches are summarized below.

Weight pruning and matrix factorization are two simple baselines described in Section SECREF2. We evaluate both pruning methods in an iterative manner until the target compression ratio is reached.

Hybrid pruning is a combination of matrix factorization and weight pruning, which conducts iterative weight pruning on the basis of SVD-based matrix factorization. It is performed iteratively until the desired compression ratio is achieved.

BERT-FT, BERT-KD and BERT-PKD are reported in BIBREF33, where BERT-FT directly fine-tunes the

model via supervision labels, BERT-KD is the vanilla knowledge distillation algorithm BIBREF1, and BERT-PKD stands for Patient Knowledge Distillation proposed in BIBREF33. The student model is composed of 3 Transformer layers, resulting in a $2.5\times $ compression ratio. Each layer has the same hidden size as the pre-trained teacher, so the initial parameters of student model can be inherited from the corresponding teacher.

TinyBERT BIBREF3 instantiates a tiny student model, which has totally 14.5M parameters ($7.5\times $ compression ratio) composed of 4 layers, 312 hidden units, 1200 intermediate size and 12 heads. For a fair comparison, we reproduce the TinyBERT pipeline without general distillation and data augmentation, which is time-exhaustive and resource-consuming.

BERT-SMALL has the same model architecture as TinyBERT, but is directly pre-trained by the official BERT pipeline. The performance values are inherited from BIBREF3 for reference.

Distilled-BiLSTM BIBREF34 leverages a single-layer bidirectional-LSTM as the student model, where the hidden units and intermediate size are set to be 300 and 400 respectively, resulting in a $10.8 \times $ compression ratio. This model requires a expensive pre-training process using the knowledge distillation constraints.

Experiments ::: Setup

We leverage the pre-trained checkpoint of base-bert-uncased as the initial model for compression, which contains 12 layers, 12 heads, 110M parameters, and 768 hidden units per layer. Hyper-parameter selection is conducted on the validation data for each dataset. After training, the prediction results are submitted to the GLUE-benchmark evaluation platform to get the evaluation performance on test data.

For a comprehensive evaluation, we experiment with four settings of LadaBERT, namely LadaBERT-1, -2, -3 and -4, which reduce the model parameters of BERT-Base by 2.5, 5, 7.5 and 10 times respectively. In our experiment, we take the batch size as 32, learning rate as 2e-5. The optimizer is BertAdam with default setting. Fine-grained compression ratios are optimized by random search and shown in Table TABREF38.

Experiments ::: Performance Comparison

The evaluation results of LadaBERT and state-of-the-art approaches are listed in Table TABREF40, where the models are ranked by parameter sizes for feasible comparison. As shown in the table, LadaBERT consistently outperforms the strongest baselines under similar model sizes. In addition, the performance of LadaBERT demonstrates the superiority of hybrid combination of SVD-based matrix factorization, weight pruning and knowledge distillation.

With model size of $2.5\times $ reduction, LadaBERT-1 performs significantly better than BERT-PKD, boosting the performance by relative 8.9, 8.1, 6.1, 3.8 and 5.8 percentages on MNLI-m, MNLI-mm, SST-2, QQP and QNLI datasets respectively. Recall that BERT-PKD initializes the student model by selecting 3 of 12 layers in the pre-trained BERT-Base model. It turns out that the discarded layers have huge impact on the model performance, which is hard to be recovered by knowledge distillation. On the other hand, LadaBERT generates the student model by iterative pruning on the pre-trained teacher. In this way, the original knowledge in the teacher model can be preserved to the largest extent, and the benefit of which is complementary to knowledge distillation.

LadaBERT-3 has a comparable size as TinyBERT with a $7.5 \times $ compression ratio. As shown in the results, TinyBERT does not work well without expensive data augmentation and general distillation, hindering its application to low-resource settings. The reason is that the student model of TinyBERT is

distilled from scratch, so it requires much more data to mimic the teacher's behaviors. Instead, LadaBERT has better initial and intermediate status calculated by hybrid model compression, which is much more light-weighted and achieves competitive performances with much faster learning speed (learning curve comparison is shown in Section SECREF41). Moreover, LadaBERT-3 also outperforms BERT-SMALL on most of the datasets, which is pre-trained from scratch by the official BERT pipeline on a $7.5 \times $ smaller architecture. This indicates that LadaBERT can quickly adapt to a smaller model size and achieve competitive performance without expansive re-training on a large corpus.

Moreover, Distilled-BiLSTM performs well on SST-2 dataset with more than $10 \times $ compression ratio, perhaps owing to its advantage of generalization on small datasets. Nevertheless, the performance of LadaBERT-4 is competitive on larger datasets such as MNLI and QQP. This is impressive as LadaBERT is much more efficient without exhaustive re-training on a large corpus. In addition, the inference speed of BiLSTM is usually slower than transformer-based models with similar parameter sizes.

Experiments ::: Learning curve comparison

To further demonstrate the efficiency of LadaBERT, we visualize the learning curves on MNLI-m and QQP datasets in Figure FIGREF42 and FIGREF42, where LadaBERT-3 is compared to the strongest baseline, TinyBERT, under $7.5 \times $ compression ratio. As shown in the figures, LadaBERT-3 achieves good performances much faster and results in a better convergence point. After training 2×10^4 steps (batches) on MNLI-m dataset, the performance of LadaBERT-3 is already comparable to TinyBERT after convergence (approximately 2×10^5 steps), achieving nearly $10 \times $ acceleration. And on QQP dataset, both performance improvement and training speed acceleration is very significant. This clearly shows the superiority of combining matrix factorization, weight pruning and knowledge distillation in a reinforce manner. Instead, TinyBERT is based on pure knowledge distillation, so the learning speed is much slower.

Experiments ::: Effect of low-rank + sparsity

In this paper, we demonstrate that a combination of matrix factorization and weight pruning is better than single solutions for BERT-oriented model compression. Similar phenomena has been reported in the computer vision scenarios BIBREF28, which shows that low-rank and sparsity are complementary to each other. Here we provide another explanation to support this observation.

In Figure FIGREF44, we visualize the distribution of errors for a weight matrix in the neural network after pruning to 20% of its original parameter size. The errors can be calculated by $\mathop {Error}=||\hat{{M}}-{M}||_1$, where $\hat{{M}}$ denotes the weight matrix after pruning.

The yellow line in Figure FIGREF44 shows the distribution of errors generated by pure weight pruning, which has a sudden drop at the pruning threshold. The orange line represents for pure SVD pruning, which turns out to be smoother and aligned with Gaussian distribution. The blue line shows the result of hybrid pruning, which conducts weight pruning on the decomposed matrices. First, we apply SVD-based matrix factorization to reduce 60% of total parameters. Then, weight pruning is applied on the decomposed matrices by 50%, resulting in only 20% parameters while the error distribution changes slightly. As a result, it has smaller mean and deviation than pure matrix factorization. In addition, a smoother distribution is more appropriate for the knowledge distillation procedure to fine-tune the weights, so it is advantageous than pure weight pruning.

Conclusion

Model compression is a common way to deal with latency-critical or memory-intensive scenarios. Existing model compression methods for BERT need to be re-trained on a large corpus to reserve its original performance, which is inapplicable in low-resource settings. In this paper, we propose LadaBERT to

address this problem. LadaBERT is a lightweight model compression pipeline that generates adaptive BERT model efficiently based on a given task and specific constraint. It is based on a hybrid solution, which conducts matrix factorization, weight pruning and knowledge distillation in a reinforce manner. The experimental results verify that EAdaBERT is able to achieve comparable performance with other state-of-the-art solutions using much less training data and time budget. Therefore, LadaBERT can be easily plugged into various applications with competitive performances and little training overheads. In the future, we would like to apply LadaBERT to large-scale industrial applications, such as search relevance and query recommendation.