Harmless Transfer Learning for Item Embeddings

Chengyue Gong[1], **Xiaocong Du**[2], **Dhruv Choudhary**[2],
Bhargav Bhushanam[2], **Qiang Liu**[1], **Arun Kejariwal**[2]
[1] University of Texas at Austin, [2] Meta Platforms, Inc.
{cygong, lqiang}@cs.utexas.edu
{xiaocongdu, choudharydhruv, bbhushanam, akejariwal}@fb.com

Abstract

Learning embedding layers (for classes, words, items, etc.) is a key component of lots of applications, ranging from natural language processing, recommendation systems to electronic health records, etc. However, the frequency of real-world items follows a long-tail distribution in these applications, causing naive training methods perform poorly on the rare items. A line of previous works address this problem by transferring the knowledge from the frequent items to rare items by introducing an auxiliary transfer loss. However, when defined improperly, the transfer loss may introduce harmful biases and deteriorate the performance.

In this work, we propose a harmless transfer learning framework that limits the impact of the potential biases in both the definition and optimization of the transfer loss. On the definition side, we reduce the bias in transfer loss by focusing on the items to which information from high-frequency items can be efficiently transferred. On the optimization side, we leverage a lexicographic optimization framework to efficiently incorporate the information of the transfer loss without hurting the minimization of the main prediction loss function. Our method serves as a plug-in module and significantly boosts the performance on a variety of NLP and recommendation system tasks.

1 Introduction

Since the advent of the deep learning era, learnable embedding layers for categorical and discrete features are basic modules for neural networks in many fields, such as natural language processing (NLP) (Vaswani et al., 2017; Devlin et al., 2018; Bahdanau et al., 2014, e.g.), recommendation systems (Zhang et al., 2019; Guo et al., 2017, e.g.), electronic health record (e.g. Kalyan and Sangeetha, 2020; Qian et al., 2017; Choi et al., 2018). Discrete items are first mapped to continuous representations in the feature space through the embedding layers and then processed by other neural modules.

However, a key challenge in learning item embeddings is due to the long tail phenomenon (Clauset et al., 2009): the frequencies of different items are usually extremely imbalanced and there often exists a large number of rare items that appear only a small number of times. As a result, it is very difficult to properly estimate the embeddings of rare items (e.g. Yin et al., 2020; Peng et al., 2019; Gao et al., 2019; Gong et al., 2018; Li et al., 2020a). The rare items often suffer from the under-fitting issue because they appears infrequently in the data (Sennrich et al., 2015; Provilkov et al., 2019).

To solve this long tail problem, transfer learning techniques are proposed (e.g. Gong et al., 2018; Li et al., 2020b; Yin et al., 2020; Chen et al., 2020) by propagating useful information from high-frequency items to rare items. The idea of these methods is to minimize the linear combination of the main prediction loss ℓ_{main} with an auxiliary transfer loss ℓ_{transfer} which encourages that the embeddings of the high-frequency and rare items follows a similar distribution, and hence allows us to the regularize the training of the rare items using the information from the common items. The basic assumption of the existing transfer learning techniques is that the distribution of the underlying embedding of the high-frequency and rare items are similar, which is necessarily false in practice. As a result, the transfer loss ℓ_{transfer} introduces potentially harmful bias and impacts the overall performance.

The goal of this work is to limit the potential harmful information in the transfer learning pipeline. We achieve this following two complementary directions:

1) Improving the definition of the transfer loss ℓ_{transfer} so that it inherently introduces less bias.

2) Adaptively balancing the combination coefficient of ℓ_{main} and ℓ_{transfer}, so that the transferring

information is efficiently incorporated without being conflicting with the main loss ℓ_{main}.

To improve the definition of ℓ_{transfer}, we leverage the co-appearance information to identify the rare items that are expected to be embedded similarly to the high-frequency items; by applying the transfer loss only between them, we significantly reduce the potential harmful bias. To adaptively balance ℓ_{main} and ℓ_{transfer}, we formulate the transfer learning as a *lexicographic (lexico) optimization*, in which we minimize ℓ_{transfer} subject to that the minimum of ℓ_{main} is achieved. This allows us to incorporate the information of ℓ_{transfer}, without interfering the minimization of the main loss ℓ_{main}. We provide a simple algorithm for solving the lexico optimization by extending the dynamic barrier algorithm of (Gong et al., 2021) to incorporate the item-wise structure in our problem.

We verify the performance of our method in multiple applications, from language model, machine translation, named entity recognition to click-through rate (CTR) prediction. Compared to different baseline transfer learning methods, our method achieves better or comparable results without tuning the hyper-parameters in all the applications.

2 Problem Set: Learning with Long Tail

In the following, we first formulate the problem and then introduce the details about our method. We start by introducing the framework of transfer learning for rare item embeddings. Assume we have a dataset $\mathcal{D}$ regarding a set of discrete items $\mathcal{I}$, such as the words in NLP and the users and products in recommendation systems. A typical deep learning model consists of two following parts:

1) a set of continuous embedding vectors $e = \{e_i \in \mathbb{R}^m : i \in \mathcal{I}\}$, which map the discrete items in $\mathcal{I}$ into the $\mathbb{R}^m$ Euclidean space;

2) a deep neural network f_θ, which takes the continuous embedding of the inputs and makes the desirable predictions. Assume f is indexed by a trainable parameter $\theta \in \mathbb{R}^{m'}$.

Both e and θ are parameters that we learn using the training data. In practice, the items $\mathcal{I}$ tend to exhibit a *long tail* distribution, consisting of both high-frequency, informative items that appear many times in the dataset, and also a large set of rare items that only appear a small number of times. As a result, the embedding vectors of the rare items may not be well estimated and hence deteriorate the performance.

To address the long tail problem, a number of works have been proposed to train the parameters $\{e, \theta\}$ by regularizing the main prediction loss with a transfer loss that propagate information from the frequent items to the rare items:

$$\min_{\{e,\theta\}} \left\{\mathcal{L}(e,\theta) := \ell_{\text{main}}(e,\theta) + \lambda \ell_{\text{transfer}}(e)\right\}, \quad (1)$$

where $\lambda > 0$ is a combination coefficient; the ℓ_{main} is the main prediction loss on the training data based on model f_θ and the embedding e, which is usually cross-entropy loss, KL divergence, and ℓ_{transfer} is a *transfer loss*, designed to transfer the knowledge from the frequent items (the source domain) to the rare items (target domain). Denote by $\mathcal{S}$ and $\mathcal{T}$ the items in the source and target domain, respectively. The transfer loss is defined as

$$\ell_{\text{transfer}}(e) = Dist\left(\{e_i\}_{i\in\mathcal{S}},\ \{e_i\}_{i\in\mathcal{T}}\right), \quad (2)$$

where measures the discrepancy between the empirical distributions of the source and target items $\{e_i\}_{i\in\mathcal{S}}$ and $\{e_i\}_{i\in\mathcal{T}}$. The discrepancy measure $Dist(\cdot)$ can be defined in various ways, such as Maximum Mean Discrepancy (MMD) (Chen et al., 2020), adversarial networks (Gong et al., 2018; Yin et al., 2020), moment matching (Peng et al., 2019), Wasserstein distance (Xu et al., 2018) Additional techniques, such as meta learning (Zhu et al., 2021) and gradient alignment (Li et al., 2020b) have also been explored.

In practice, the source domain $\mathcal{S}$ and target domain $\mathcal{T}$ is chosen based on the item frequency with various heuristics. For example, Gong et al. (2018) and Li et al. (2020b) uses the Pareto Principle (Dunford et al., 2014) (or 80/20 rule), which places the top 20% high-frequency items in $\mathcal{S}$ and the rest in $\mathcal{T}$. Further, Yin et al. (2020) define the target domain as the items with frequency lower than a dataset-dependent threshold.

3 Harmless Transferring for Rare Items

The basic assumption that underpins the transfer loss ℓ_{transfer} in (2) is that the distribution of the high-frequency items $e_{\mathcal{S}}$ and rare items $e_{\mathcal{I}}$ is similar. However, this is not necessarily true in practice. There could be rare items that find no similar high-frequency items and should be embedded differently. When this happens, the transfer loss ℓ_{transfer} may provide biased and harmful information and hence deteriorate the overall performance.

In this work, we propose two techniques to limit the potential harmful information in the transfer

learning pipeline, by improving both the definition and the optimization of ℓ_{transfer}.
1) In Section 3.1, we improve the definition of ℓ_{transfer} in (2) to reduce potential harmful bias. This is made possible by filtering out the irrelevant items from the target domain $\mathcal{T}$ based on the co-appearance information between the items.
2) In Section 3.2, we further limits the potential harmful impact of ℓ_{transfer} by replacing the regularized loss in (1) with a *lexicographic (lexico) optimization* which minimizes ℓ_{transfer} subject to that the minimum of ℓ_{main} is achieved. This allows us to prioritize the optimization of ℓ_{main} when the gradient direction ℓ_{main} and ℓ_{transfer} are conflicting with each other. We provide a simple algorithm for solving the lexico optimization by extending the dynamic barrier algorithm of (Gong et al., 2021) to incorporate the item-wise structure in our problem.

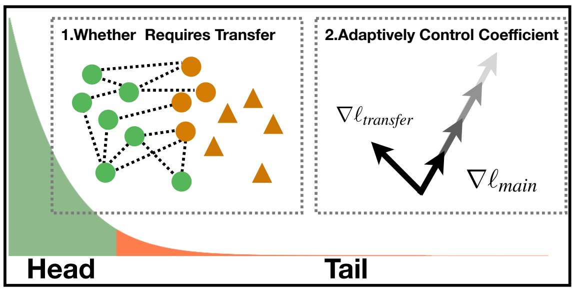

Figure 1: A demonstration of our method. 1) For tail items, we set the target domain as items which have similar (the dashed line denotes the large similarity) frequent items (the orange circles), and remove tail items with no similar frequent items (the orange triangle), 2) We dynamically control the coefficient between the main loss and transfer loss in a way that efficiently incorporates the transfer loss information without hurting the optimization of the main loss.

3.1 Selective Transfer Loss

In previous works, source $\mathcal{S}$ and target $\mathcal{T}$ domains are decided solely based on frequency information, which can not ensure that their embedding vectors $e_{\mathcal{S}}$ and $e_{\mathcal{T}}$ have a similar distribution. This may introduce harmful information into ℓ_{transfer}. We address this problem by incorporating co-appearance information between the items, so that target domain $\mathcal{T}$ only include the rare items that find some high-frequency items.

Specifically, starting with a pair of $\mathcal{S}$ and $\mathcal{T}$ using standard method (say, based on frequency cutoff), we refine $\mathcal{T}$ to only include rare items that find similar items in $\mathcal{S}$. To do so, we characterize each item $i \in \mathcal{I}$ by its co-appearance with the high-frequency items. Specifically, let $s_i = [s_{ij}]_{j\in\mathcal{S}}$, where s_{ij} denotes the number of times when item i and j co-appear in an n-gram window. Without further information, the co-appearance vector s_i provides a naive representation of item i. Then, we filter the target domain $\mathcal{T}$ by removing the rare items that has low similarity with every high frequency item, yielding a more selective target domain:

$$\mathcal{T}'_\eta = \mathcal{T} \cap \{i \in \mathcal{I} \colon \max_{j\in\mathcal{S}} \cos(s_i, s_j) \geq \eta\}, \quad (3)$$

$\cos(\cdot,\cdot)$ denotes the cosine similarity between two vectors, and η is a given threshold. We then replace $\mathcal{T}$ with $\mathcal{T}'_\eta$ in the definition of ℓ_{transfer}, that is,

$$\ell_{\text{transfer}}(e) = Dist\left(\{e_i\}_{i\in\mathcal{S}},\ \{e_i\}_{i\in\mathcal{T}'_\eta}\right).$$

In this way, we filter out the irrelevant items from the target domain, and hence reduce the potential biases in ℓ_{transfer}.

3.2 Harmless Transfer via Lexico Optimization

Despite the improved definition above, the transfer loss ℓ_{transfer} may still contain harmful bias that will be inherited by the final result as we minimize the regularized loss function in (1). As a result, the choice of the regularization coefficient λ in (1) becomes critically important: a large λ may overemphasize the transfer loss and amplify the harmful biases, while a small λ may yield insufficient transferring. The optimal choice of λ depends on both ℓ_{main} and ℓ_{transfer} and need to be selected case by case for individual problems using grid search or other hyper-parameter optimization approaches, which yields high computational cost.

To incorporate the benefit of the transfer loss without hurting the optimization of the main loss, we propose to estimate the parameters $\{e,\theta\}$ by solving a lexicographic optimization problem (e.g., Dempe and Zemkoho, 2020; Gong et al., 2021):

$$\min_e \ell_{\text{transfer}}(e), \quad \text{s.t.} \ \ \{e,\theta\} \in \arg\min \ell_{\text{main}}, \quad (4)$$

where $\arg\min \ell_{\text{main}}$ denotes the set of (local) minimum points of ℓ_{main}. This means that we prioritize the optimization of the main loss, and minimize the ℓ_{transfer} only within the optimum set of ℓ_{main}.

A key fact that underpins the formulation in (4) is that the modern neural networks that we use in practice are almost always *over-parameterized*, and hence the optimum set of ℓ_{main} is not a set of isolated points, but connected manifolds consisting of an infinite number of points. Therefore, by searching in the solution manifold the point that minimizes ℓ_{transfer}, we gain improvement on the transfer loss *without* hurting the main loss.

Dynamic Barrier Gradient Descent We adopt the dynamic barrier gradient descent (DBGD) algorithm by (Gong et al., 2021) to solve the lexicographic optimization problem in (4). DBGD iteratively updates the parameters by

$$(e, \theta) \leftarrow (e, \theta) - \epsilon(\mu_e, \mu_\theta),$$

where ϵ is a step size and $\mu = (\mu_e, \mu_\theta)$ is the update direction for parameters (e, θ) given by solving the following optimization problem:

$$\begin{aligned} \mu = \operatorname*{argmin}_{\nu} \|\nabla_{(e,\theta)}\ell_{\text{transfer}} - \nu\|^2 \\ s.t. \quad \langle \nabla_{(e,\theta)}\ell_{\text{main}},\ \nu \rangle \geq \phi, \end{aligned} \tag{5}$$

where ϕ is a non-negative barrier function that equals zero only when we reach the minimum of ℓ_{main}. For example, for $\ell_{\text{main}} \geq 0$, we can take

$$\phi = \alpha \min(\|\nabla_{(e,\theta)}\ell_{\text{main}}\|^2, \ell_{\text{main}})$$

as suggested by (Gong et al., 2021), where $\alpha > 0$ is a positive coefficient. The idea of (5) is as follows:

1. By enforcing that $\langle \nabla_{(e,\theta)}\ell_{\text{main}},\ \mu \rangle \geq \phi \geq 0$, we ensure that we always monotonically minimize ℓ_{main} (when step size ϵ is sufficiently small), as ℓ_{main} is the main loss function.

2. By minimizing $\|\nabla_{(e,\theta)}\ell_{\text{transfer}} - \mu\|^2$ under the inner product constraint, we ℓ_{transfer} is minimized as much as possible without hurting the optimization of ℓ_{main}.

As shown in (Gong et al., 2021), the problem in Eq. 5 yields a simple closed form solution:

$$\mu = \nabla_{(e,\theta)}\ell_{\text{transfer}} + \lambda \nabla_{(e,\theta)}\ell_{\text{main}}, \tag{6}$$

where λ is a non-negative coefficient defined by

$$\lambda = \max\left(\frac{\phi - \nabla_{(e,\theta)}\ell_{\text{transfer}}^\top \nabla_{(e,\theta)}\ell_{\text{main}}}{\|\nabla_{(e,\theta)}\ell_{\text{main}}\|^2},\quad 0 \right). \tag{7}$$

Eq (6) can be viewed as the gradient of the regularized loss (2). The key difference is that the λ in (6) is decided adaptively by formula (7) in each iteration of the algorithm based on the inner product of the gradients, rather than pre-determined in the beginning.

3.2.1 Item-wise Dynamic Barrier Descent

To best exploit the special structure of our problem, we propose to modify the off-the-shelf algorithm above yield better results.

Update of θ Because the transfer loss $\ell_{\text{transfer}}(e)$ is independent with the neural network parameter θ, that is, $\nabla_\theta \ell_{\text{transfer}}(e) = 0$, the update on θ follows the standard gradient: $\mu_\theta = \lambda \nabla_\theta \ell_{\text{main}}$. It find its handy to modify it to be $\mu_\theta = \nabla_\theta \ell_{\text{main}}$, that is, we update θ by standard gradient descent on ℓ_{main}.

Update of Embedding Vectors e Another critical special property of our problem is that $e = \{e_i\}_{i \in \mathcal{I}}$ consists of the embedding vectors of all the items. It would be desirable to fine-grainedly control the co-efficient for different items; doing so can also decrease the computational cost since we only need to calculate the coefficient for the items that appear in the training mini-batch at each iteration.

Specifically, we update each embedding vector e_i by $e_i \leftarrow e_i - \epsilon\mu_{e_i}$, where μ_{e_i} is decided by

$$\begin{aligned} \mu_{e_i} = \operatorname*{argmin}_{\nu_i} ||\nabla_{e_i}\ell_{\text{transfer}} - \nu_i||^2 \\ s.t. \quad \langle \nabla_{e_i}\ell_{\text{main}},\ \nu_i \rangle \geq \phi_i, \end{aligned}$$

where ϕ_i is an item-wise control barrier associated with e_i: $\phi_i = \alpha \min(\|\nabla_{e_i}\ell_{\text{main}}\|^2, \ell_{\text{main}})$ (we simply take $\alpha = 2$ in practice). Similar to (6), we can show that μ_{e_i} is a linear combination of $\nabla_{e_i}\ell_{\text{main}}$ and $\nabla_{e_i}\ell_{\text{transfer}}$, with an item-wise coefficient λ_i.

See Algorithm 1 for details of the update rule.

Memory And Time Cost Compared with the standard gradient descent on the main loss ℓ_{main}, the main additional computational cost that we introduce is calculating the gradient of ℓ_{transfer}, which is much faster than that of ℓ_{main} because it only involves embedding vectors e, and does not need to backpropagate on the network f_θ. In practice, we find that we introduce an additional 5% to 10% time cost. For memory cost, we require to store the gradient for ℓ_{transfer} at each iteration. Consider that we store the gradient information for items in one given mini-batch instead of the whole embedding layer, the additional memory cost is light.

4 Related Works

Transfer Learning for Long-Tail Items Transfer learning techniques are proposed (e.g. Gong et al., 2018; Gao et al., 2019; Li et al., 2020b; Yin

Algorithm 1: Controllable Item-wise Optimization for Embeddings

```
Denote $\theta_t$, $e_t$ and $\{\epsilon_t\}$ as the parameters, embeddings and learning rate at $t$-th iteration.
for Iteration $t$ do
    $\theta_{t+1} \longleftarrow \theta_t - \epsilon_t \nabla_{\theta_t} \ell_{\text{main}}(e_t, \theta_t),$
    $e_{t+1,i} \longleftarrow e_{t,i} - \epsilon_t \left(\nabla_{e_{t,i}} \ell_{\text{transfer}}(e_t) + \lambda_{t,i} \nabla_{e_{t,i}} \ell_{\text{main}}(e_t, \theta_t)\right), \quad \forall \text{ items } i$
    where $\lambda_{t,i} = \max\left(\frac{\phi_{t,i} - \nabla_{e_{t,i}} \ell_{\text{transfer}}(e_t)^\top \nabla_{e_{t,i}} \ell_{\text{main}}(e_t, \theta_t)}{\|\nabla_{e_{t,i}} \ell_{\text{main}}(e_t, \theta_t)\|^2}, 0\right),$
          $\phi_{t,i} = \alpha \min(\|\nabla_{e_{t,i}} \ell_{\text{main}}(e_t)\|^2, \ell_{\text{main}}(e_t)).$
end for
```

et al., 2020; Chen et al., 2020) to leverage useful information from high-frequency items to low-frequency items. These methods use 1) different domain knowledge to split the $\mathcal{S}$ and $\mathcal{T}$ domain, 2) different distance functions to transfer the knowledge, and 3) different training pipelines. We have discussed the first two directions above. For training pipelines, some (e.g. Gong et al., 2018; Li et al., 2020b; Yin et al., 2020) jointly train the head and tail item embeddings while others first train the head item embeddings and then train the rare item embeddings (Kang et al., 2019). Improving long-tail item representations is not only an important topic for discrete input items but also useful for long-tail classification problems (e.g. Tang et al., 2020; Kang et al., 2019; Tan et al., 2020). The Softmax layer can be viewed as an embedding layer in which each dimension stands for an embedding vector for one class.

Lexicographic optimization Lexicographic optimization is a problem traditionally studied in operation research and economics (Ehrgott, 1998; Boggs and Tolle, 1995; Lewis and Gale, 1994). Its applications in deep learning tasks are demonstrated in Gong et al. (2021) with the dynamic barrier algorithm. Similar methods that trade-off multiple objective functions can be found in multi-objective optimization methods, e.g. PCGrad (Yu et al., 2020) and its variants (e.g. Lin et al., 2020; Javaloy and Valera, 2021; Liu et al., 2021a).

5 Experiments

We answer the following two key questions through empirical studies: 1) could our target domain selection strategy outperform the simple frequency-based methods? 2) could the lexicographic optimization approach outperform the simple linear combination method defined in (2)?

We test our method on a variety of tasks, including natural language modeling, named entity recognition, machine translation, and recommendation systems. In all experiments, we set the value of η to be the value of 0.2-quantile of all the similarity scores. It means that we remove 20% words in the rough target domain which only considered frequency. For algorithm 1, we always set $\alpha = 2$ and use the default optimizer (e.g., Adam) in each application to set the step size ϵ_t. In all experiments, we average the results over three trials. We do not report the variance in the table since the variance is small (e.g. < 0.1) in many cases. We refer readers to the appendix for full tables with the error bar.

For notation, we denote by 'select' our target domain selection strategy, and 'lexico' the lexicographic optimization algorithm 1, and 'ours' the combination of two techniques.

5.1 Neural Language Modeling

Settings We use Wikitext-2 (WT2) and Wikitext-103 (WT103) (Merity et al., 2016) to test our method. We adopt the method proposed in (Gong et al., 2018) and (Gao et al., 2019) as our baseline algorithm. Gong et al. (2018) split the rare words with 80/20 rule and then trains an adversarial network as the ℓ_{transfer}. Gao et al. (2019) add a regularization $\min_e \frac{1}{N}\sum_i^N \sum_{j,j\neq i}^N \frac{e_i^\top e_j}{\|e_i\|\|e_j\|}$, to all the words. This can be regarded as ℓ_{transfer} whose source domain is a spherical uniform distribution and the target domain is all words.

The practical difference of our method with the baselines are: 1) We replace the target domain $\mathcal{T}$ in the baselines with the more selective domain $\mathcal{T}'_\eta$ following (3). For (Gong et al., 2018), we regard words with the 20% highest frequency following

(Gong et al., 2018) as the source domain. For (Gao et al., 2019), we only apply Algorithm 1 and do not apply (3). 2) The update of each embedding vector e_i is based on an item-wise coefficient λ_i of the main and transfer loss that is adaptively decided as shown in Algorithm 1.

Model Description For WT2 datasets, we closely follow the regularization and optimization techniques introduced in AWD-LSTM (Merity et al., 2018b). For WT103, we use Quasi-Recurrent neural networks (QRNN)-based language models (Merity et al., 2018a; Bradbury et al., 2017) as our base model for efficiency.

	# Dataset	Baseline	+ CosReg	+ Ours
WT2	Eval	68.6	67.4	**65.5**
	Test	65.8	64.7	**63.1**
	# Dataset	Baseline	+ FRAGE	+ Ours
WT2	Eval	68.6	66.5	**64.3**
	Test	65.8	63.4	**61.6**
	# Dataset	Baseline	+ FRAGE	+ Ours
WT103	Eval	32.0	31.3	**30.5**
	Test	33.0	32.5	**31.4**

Table 1: Perplexities measured on validation and test sets on WT2 and WT103. 'CosReg' refers Gao et al. (2019) and FRAGE refers to Gong et al. (2018).

Cosine Similarity Calculation We construct the tri-gram co-occurrence matrix (co-occurrence counts with window size 3) with the top 1K frequent words, calculate the cosine similarity between rare and frequent words, rank the similarity and filter 20% items in the rare items.

Results Table 1 displays that for both FRAGE and CosReg loss, our method can improve the baseline performance on both evaluation and test sets. For example, on WT2 dataset, we boost the test perplexity with a large-marginal improvement, from 63.4 of FRAGE to 61.6. We refer readers to the appendix for sampled examples on WT103 test set.

	# Dataset	Baseline	Select	Lexico	Both
WT2	Eval	68.6	68.1	65.1	**64.3**
	Test	65.8	65.5	62.2	**61.6**
	# Dataset	Baseline	Select	Lexico	Both
WT103	Eval	32.0	31.4	30.9	**30.5**
	Test	33.0	32.5	32.1	**31.4**

Table 2: Ablation studies about our two techniques. Experiments are based on FRAGE.

Analyses We further construct ablation studies for understanding the benefits of each component in our method. We start from Table 2 in which we apply each technique in our method to the baseline. We notice that
1) for small-size vocabularies (e.g. WT2), 'select' only brings marginal improvements, while applying lexico optimization can improve the 63.4 test perplexity obtained by FRAGE to 62.2;
2) for WT103, whose vocabulary size is over 200K, both components of our method can gain substantial improvements over the baseline;
3) combining the two parts of our method together, we achieve the best performance on these two datasets.

Method	Head	Tail	Total
Baseline	21.7	313.2	33.0
+ FRAGE	21.8	297.2	32.5
+ Ours	**21.5**	**275.6**	**31.4**

Table 3: Perplexity of the high-frequency (head) and rare (tail) words on the test set of WT103 language modeling task.

As shown in Table 3, we substantially improve the performance on rare items without hurting the performance of high-frequency items.

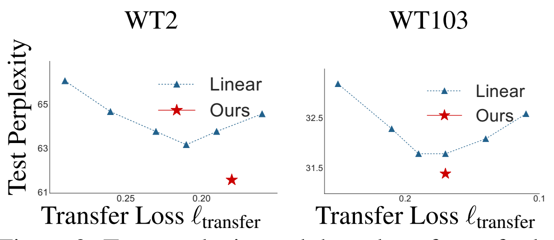

Figure 2: Test perplexity and the value of transfer loss.

We analyze whether the adaptive coefficient strategy in lexico optimization outperforms the standard method with a fixed coefficient λ in (2). We compare our approach with grid search the coefficient λ for ℓ_{transfer}. The blue curves in Figure 2 shows the performance as we vary λ. Each point on the curve denotes the ℓ_{transfer} value and the test perplexity for one λ value. We can see that our method yields strictly better results than any fixed λ in grid search.

5.2 Named Entity Recognition and Machine Translation

To verify whether our method can be useful for other NLP problems, we set up experiments on named entity recognition and machine translation.

Settings For name entity recognition, we follow Li et al. (2020b), which transfers knowledge to rare entities with GAN. They use GAN to define the discrepancy measure in ℓ_{transfer} and apply the method to the CoNLL-03 dataset (Sang and De Meulder, 2003). We follow the BERT + CRF model used in

Method	BLEU
Transformer Base (Vaswani et al., 2017)	27.8
+ FRAGE (Gong et al., 2018)	28.3
+ Ours	**28.6**

Table 4: BLEU scores for the WMT2014 EN-DE machine translation.

Li et al. (2020b), implement the method based on the codebase `Transformer` (Wolf et al., 2020) by ourselves and yield a slightly worse score than the score reported in Li et al. (2020b). For machine translation, we do experiments on neural machine translation on the WMT2014 EN-DE (Bojar et al., 2014) dataset and adopt the method FRAGE in Gong et al. (2018). We follow the settings in Gong et al. (2018) and use the Transformer-based machine translation model (Vaswani et al., 2017) as our base model with `Fairseq` (Ott et al., 2019) as the codebase. We follow all the other settings the same as the language model case.

Cosine Similarity Calculation We calculate the cosine similarity following the settings in language model experiments.

Method	F1
BERT (Devlin et al., 2018) + CRF	91.0
+ DEI (Li et al., 2020b)	91.8
+ Ours	**92.5**

Table 5: Named entity recognition results of baselines and the proposed model on CoNLL-03.

Results As displayed in Table 4 and Table 5, we notice that 1) the transfer learning can improve the baseline performance 2) we can further improve the transfer learning loss. We enhance the BLEU score [1] from 28.3 to 28.6 on WMT while boost the F1 score from 91.8 to 92.5 on CoNLL-03.

5.3 Recommendation Systems

In recommendation systems (RS), learning tail item embeddings is also an important topic (e.g. Yin et al., 2020; Zhang et al., 2021; Zhu et al., 2021). We extend our method to the click-through rate (CTR) prediction task to verify its generalizability.

Settings We examine our method in CTR prediction tasks and create two different datasets, Movielens 1M (Harper and Konstan, 2015) and Criteo (Cri). In CTR prediction, the user-item pair is represented as a discrete feature vector, and each feature dimension is converted into a dense embedding. Passing a neural module, the combination of the dense embedding is finally converted into a dense feature and processed by other neural modules to give the final predictions. We apply transfer learning to the embedding of the features. For the simplicity of implementation, instead of following recent works (e.g. Zhang et al., 2021; Zhu et al., 2021), we follow our settings in the language model section and use an adversarial network (Tzeng et al., 2017) as the ℓ_{transfer}. It introduces an additional regularization term to the final training objective, which is the same as the language model experiments. We report AUC (Area Under the ROC Curve) to measure the performance of models.

Model Description We direct apply our method to DeepFM (Guo et al., 2017) and AutoInt (Song et al., 2019) models. DeepFM consists of an FM component and a deep component (e.g. MLP) which are integrated into a parallel structure. The input feature vectors are first converted into a dense embedding by a lookup table, and then pass the FM component and the deep component. Compared to DeepFM, instead of MLP, AutoInt (Song et al., 2019) uses a multi-head self-attentive neural network with residual connections to explicitly model the feature interactions.

Implementation Details We build our code upon the implementation given by Liu et al. (2021b) [2]. Liu et al. (2021b) automatically creates models with the different number of parameters with learnable embedding sizes. This allows us to examine our method with flexible model sizes. We additionally introduce exponential moving average (EMA) (Booth et al., 2006) to reproduce the results of (Liu et al., 2021b).

Dataset Description Movielens 1M dataset encodes each example into a 7-dimension discrete feature vector, which represents timestamp, age, gender, occupation, age, zip code, years, and gender. Among these dimensions, zip code is encoded into 3439 different classes, and has a long tail frequency. Criteo contains 39 feature dimensions. We apply transfer loss to the zip code embeddings for Movielens 1M, and for all the feature dimensions with more than 100 values for Criteo.

Cosine Similarity Calculation We construct the side-information matrix to calculate cosine similarity. For each typical item, we count the times this

[1] `https://github.com/moses-smt/mosesdecoder/blob/master/scripts/generic/multi-bleu.perl`

[2] `https://github.com/ssui-liu/learnable-embed-sizes-for-RecSys`

item co-occurred with one of the top 1K items in one record and construct the 1K dimension feature vector.

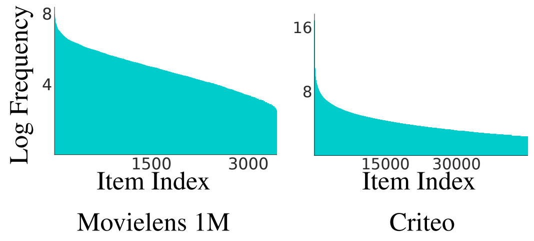

Figure 3: The frequency distribution of all the items in the Movielens 1M and Criteo training datasets.

	Deep FM			AutoInt		
# P	B	+ T	+ Ours	B	+ T	+ Ours
10k	84.9	84.9	**85.2**	84.5	84.6	**84.8**
20k	85.0	84.9	**85.2**	84.7	**84.8**	**84.8**
30k	85.1	85.0	**85.3**	84.7	84.8	**84.9**

Table 6: Test AUC on the Movielens 1M. '# P' denotes the number of parameters; 'B' the baseline only trained with ℓ_{main}; '+ T' denotes the transfer learning using the regularized loss (2).

	Deep FM			AutoInt		
# P	B	+ T	+ Ours	B	+ T	+ Ours
50k	79.7	79.8	**80.0**	79.3	79.5	**79.7**
20k	79.6	79.6	**79.9**	79.2	79.4	**79.6**

Table 7: Test AUC on Criteo with DeepFM and AutoInt.

Results The results of our method and baseline are shown in Table 6 and Table 7. In these two tables, we examine our method with different architectures (DeepFM and AutoInt) and different model sizes. For all these results, we report the performance of the baseline method (training only with ℓ_{main}), the transfer learning method (2) in which the coefficient is set to $\lambda = 10^{-2}$, and our method. From these results, we observe that:

1) our method never hurts the baseline performance, and sometimes can boost the results; simply applying transfer learning, on the other hand, requires a proper λ to avoid hurting the baseline performance;

2) compared to Movielens 1M, we can achieve more improvements on Criteo dataset. One reasonable explanation is that Criteo contains more long-tail items (see Figure 3) and therefore our method can be more helpful in this case.

Convergence Speed We empirically examine the convergence speed of both our method and the baseline method without ℓ_{transfer}. We demonstrate the test loss together with test AUC in Figure 4, in which we train the $10K$ parameters Deep FM model on the Movielens 1M dataset. We train the model with 90 epochs and plot the test log loss and test AUC every 5 epochs. We notice that compared to the baseline, our method has comparable convergence speed and finally boosts both test log loss (0.370 v.s. 0.368) and test AUC.

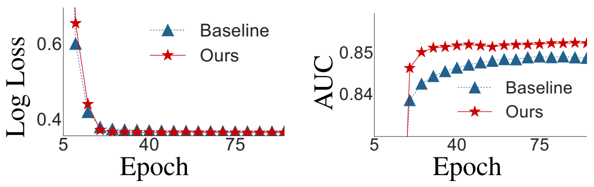

Figure 4: Using Moivelen 1M and $10K$ parameter model, we show the test log loss and test AUC at different epochs for the baseline method and our method.

	SGD + 0.9 Momentum			SGD		
# P	B	+ T	+ Ours	B	+ T	+ Ours
10k	84.9	84.7	**85.1**	84.2	84.2	**84.3**
20k	84.9	84.8	**85.1**	84.3	84.3	**84.4**

Table 8: Test AUC on Movielens 1M dataset with DeepFM model. The meanings of '# P', 'B', and '+ T' are the same as Table 6.

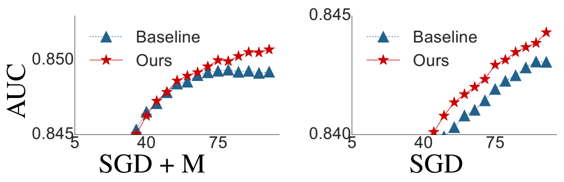

Figure 5: The test AUC vs. epochs when we use SGD and SGD+Momentum to train the $10K$-parameter DeepFM model on the Movielens 1M dataset.

Different Optimizers The default optimizer in RS is Adam, and we test whether our method can be applied for different optimizers. We report the results in Table 8 and Figure 5. We observe that 1) SGD with momentum performs worse than Adam while SGD performs the worst, and 2) our method can improve the baseline performance with all different optimizers. It is possible to combine our method with the other recent advanced optimizers.

6 Conclusion

This work improves transfer learning for rare items embeddings in multiple tasks. In the future, we plan to 1) extend our method for more general methods and settings in the embedding learning problems, 2) apply our method to datasets with larger-scale vocabulary, and 3) develop an online version algorithm for selecting the target and source domain.

Acknowledgement The work is conducted in the statistical learning and AI group in the computer science department at UT Austin, which is supported in part by CAREER-1846421, SenSE-2037267, EAGER-2041327, and Office of Navy Research, and NSF AI Institute for Foundations of Machine Learning (IFML). We would like to thank the anonymous reviewers and the area chair for their thoughtful comments and efforts towards improving our manuscript.

References

Criteo. http://labs.criteo.com/downloads/.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Paul T Boggs and Jon W Tolle. 1995. Sequential quadratic programming. *Acta numerica*, 4:1–51.

Ondřej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, et al. 2014. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the ninth workshop on statistical machine translation*, pages 12–58.

Eric Booth, Jeff Mount, and Joshua H Viers. 2006. Hydrologic variability of the cosumnes river floodplain. *San Francisco Estuary and Watershed Science*, 4(2).

James Bradbury, Stephen Merity, Caiming Xiong, and Richard Socher. 2017. Quasi-recurrent neural networks. *International Conference on Learning Representations, ICLR*.

Zhihong Chen, Rong Xiao, Chenliang Li, Gangfeng Ye, Haochuan Sun, and Hongbo Deng. 2020. Esam: Discriminative domain adaptation with non-displayed items to improve long-tail performance. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 579–588.

Edward Choi, Cao Xiao, Walter F Stewart, and Jimeng Sun. 2018. Mime: Multilevel medical embedding of electronic health records for predictive healthcare. *arXiv preprint arXiv:1810.09593*.

Aaron Clauset, Cosma Rohilla Shalizi, and Mark EJ Newman. 2009. Power-law distributions in empirical data. *SIAM review*, 51(4):661–703.

Stephan Dempe and Alain Zemkoho. 2020. *Bilevel optimization*. Springer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Rosie Dunford, Quanrong Su, and Ekraj Tamang. 2014. The pareto principle.

Matthias Ehrgott. 1998. Discrete decision problems, multiple criteria optimization classes and lexicographic max-ordering. *Trends in multicriteria decision making*, pages 31–44.

Jun Gao, Di He, Xu Tan, Tao Qin, Liwei Wang, and Tie-Yan Liu. 2019. Representation degeneration problem in training natural language generation models. *arXiv preprint arXiv:1907.12009*.

Chengyue Gong, Di He, Xu Tan, Tao Qin, Liwei Wang, and Tie-Yan Liu. 2018. Frage: Frequency-agnostic word representation. *arXiv preprint arXiv:1809.06858*.

Chengyue Gong, Xingchao Liu, and Qiang Liu. 2021. Automatic and harmless regularization with constrained and lexicographic optimization: A dynamic barrier approach. *Advances in Neural Information Processing Systems*, 34.

Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. Deepfm: a factorization-machine based neural network for ctr prediction. *arXiv preprint arXiv:1703.04247*.

F Maxwell Harper and Joseph A Konstan. 2015. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):1–19.

Adrián Javaloy and Isabel Valera. 2021. Rotograd: Gradient homogenization in multi-task learning.

Katikapalli Subramanyam Kalyan and Sivanesan Sangeetha. 2020. Secnlp: A survey of embeddings in clinical natural language processing. *Journal of biomedical informatics*, 101:103323.

Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. 2019. Decoupling representation and classifier for long-tailed recognition. *arXiv preprint arXiv:1910.09217*.

David D Lewis and William A Gale. 1994. A sequential algorithm for training text classifiers. In *SIGIR'94*, pages 3–12. Springer.

Minchen Li, Zachary Ferguson, Teseo Schneider, Timothy Langlois, Denis Zorin, Daniele Panozzo, Chenfanfu Jiang, and Danny M Kaufman. 2020a. Incremental potential contact: Intersection-and inversion-free, large-deformation dynamics. *ACM transactions on graphics*.

Yangming Li, Han Li, Kaisheng Yao, and Xiaolong Li. 2020b. Handling rare entities for neural sequence labeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6441–6451.

Xi Lin, Zhiyuan Yang, Qingfu Zhang, and Sam Kwong. 2020. Controllable pareto multi-task learning. *arXiv preprint arXiv:2010.06313*.

Bo Liu, Xingchao Liu, Xiaojie Jin, Peter Stone, and Qiang Liu. 2021a. Conflict-averse gradient descent for multi-task learning. *Advances in Neural Information Processing Systems*, 34.

Siyi Liu, Chen Gao, Yihong Chen, Depeng Jin, and Yong Li. 2021b. Learnable embedding sizes for recommender systems. *arXiv preprint arXiv:2101.07577*.

Stephen Merity, Nitish Shirish Keskar, and Richard Socher. 2018a. An analysis of neural language modeling at multiple scales. *arXiv preprint arXiv:1803.08240*.

Stephen Merity, Nitish Shirish Keskar, and Richard Socher. 2018b. Regularizing and optimizing lstm language models. *ICLR*.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. 2019. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1406–1415.

Ivan Provilkov, Dmitrii Emelianenko, and Elena Voita. 2019. Bpe-dropout: Simple and effective subword regularization. *arXiv preprint arXiv:1910.13267*.

Feng Qian, Chengyue Gong, Lu-chen Liu, Lei Sha, and Ming Zhang. 2017. Topic medical concept embedding: Multi-sense representation learning for medical concept. In *2017 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 404–409. IEEE.

Erik F Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *arXiv preprint cs/0306050*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Weiping Song, Chence Shi, Zhiping Xiao, Zhijian Duan, Yewen Xu, Ming Zhang, and Jian Tang. 2019. Autoint: Automatic feature interaction learning via self-attentive neural networks. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 1161–1170.

Jingru Tan, Changbao Wang, Buyu Li, Quanquan Li, Wanli Ouyang, Changqing Yin, and Junjie Yan. 2020. Equalization loss for long-tailed object recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11662–11671.

Kaihua Tang, Jianqiang Huang, and Hanwang Zhang. 2020. Long-tailed classification by keeping the good and removing the bad momentum causal effect. *arXiv preprint arXiv:2009.12991*.

Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. 2017. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7167–7176.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Hongteng Xu, Wenlin Wang, Wei Liu, and Lawrence Carin. 2018. Distilled wasserstein learning for word embedding and topic modeling. *arXiv preprint arXiv:1809.04705*.

Jianwen Yin, Chenghao Liu, Weiqing Wang, Jianling Sun, and Steven CH Hoi. 2020. Learning transferrable parameters for long-tailed sequential user behavior modeling. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 359–367.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. 2020. Gradient surgery for multi-task learning. *arXiv preprint arXiv:2001.06782*.

Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. 2019. Deep learning based recommender system: A survey and new perspectives. *ACM Computing Surveys (CSUR)*, 52(1):1–38.

Yin Zhang, Derek Zhiyuan Cheng, Tiansheng Yao, Xinyang Yi, Lichan Hong, and Ed H Chi. 2021. A model of two tales: Dual transfer learning framework for improved long-tail item recommendation. In *Proceedings of the Web Conference 2021*, pages 2220–2231.

Yongchun Zhu, Ruobing Xie, Fuzhen Zhuang, Kaikai Ge, Ying Sun, Xu Zhang, Leyu Lin, and Juan Cao. 2021. Learning to warm up cold item embeddings for cold-start recommendation with meta scaling and shifting networks. *arXiv preprint arXiv:2105.04790*.

.1 Qualitative Analysis

#2 Context	A former basketball player , he grew up in <unk> , Indiana , where he starred on the <unk> Community High School basketball team , setting four school records . After high school , he attended DePauw University , where he played basketball and earned a degree in [MASK]				
Baseline top-5	the	.	;	that	engineer
Ours top-5	the	.	engineer	science	law
Reference	he attended DePauw University , where he played basketball and earned a degree in economics .				
#2 Context	Bradley Kent " Brad " Stevens (born October 22 , 1976) is an American professional basketball head coach for the Boston Celtics of the [MASK]				
Baseline top-5	.	team	;	the	season
Ours top-5	.	the	team	year	NCAA
Reference	an American professional basketball head coach for the Boston Celtics of the NBA .				
#2 Context	several of the tanks destined to be deployed to the Eighth Army in the Middle East for the North African Campaign were left in Britain when their cooling systems were determined to be unable to cope with the intense North [MASK]				
Baseline top-5	.	wind	sea	;	winds
Ours top-5	;	.	Britain	its	Africa
Reference	their cooling systems were determined to be unable to cope with the intense North African heat .				

Table 9: Comparison of next-token prediction on WT103 data. '[MASK]' denotes the location to make prediction, while the reference displays the sentence in the corpus.

	# Dataset	Baseline	+ CosReg	+ Ours
WT2	Eval	68.6	67.4	**65.5±0.2**
	Test	65.8	64.7	**63.1±0.3**
	# Dataset	Baseline	+ FRAGE	+ Ours
WT2	Eval	68.6	66.5	**64.3±0.2**
	Test	65.8	63.4	**61.6±0.1**
	# Dataset	Baseline	+ FRAGE	+ Ours
WT103	Eval	32.0	31.3	**30.5±0.1**
	Test	33.0	32.5	**31.4±0.1**

Table 10: Perplexities measured on validation and test sets on WT2 and WT103.

In Table 9, we sample some examples from WT103 test set. 'Baseline' denotes the results of the model without transfer loss, and 'ours' denotes our method results. 'Reference' shows the ground-truth sentence sampled from the dataset.

.2 Full Results

Method	BLEU
Transformer Base (Vaswani et al., 2017)	27.8±0.2
+ FRAGE (Gong et al., 2018)	28.3±0.1
+ Ours	**28.6±0.2**

Table 11: BLEU scores for the WMT2014 Ee→De machine translation.

Method	F1
BERT (Devlin et al., 2018) + CRF	91.0±0.1
+ DEI (Li et al., 2020b)	91.8±0.1
+ Ours	**92.5±0.0**

Table 12: Named entity recognition results of baselines and the proposed model on CoNLL-03.

	Deep FM			AutoInt		
# P	B	+ T	+ Ours	B	+ T	+ Ours
10k	84.9±0.1	84.9±0.1	**85.2±0.1**	84.5±0.1	84.6±0.2	**84.8±0.1**
20k	85.0±0.1	84.9±0.1	**85.2±0.1**	84.7±0.1	**84.8±0.2**	**84.8±0.1**
30k	85.1±0.0	85.0±0.0	**85.3±0.0**	84.7±0.0	84.8±0.0	**84.9±0.0**

Table 13: Test AUC on the Movielens 1M. '# P' denotes the number of parameters, 'B' the baseline only trained with ℓ_{main}, while '+ T' denotes with ℓ_{transfer}.

	Deep FM			AutoInt		
# P	B	+ T	+ Ours	B	+ T	+ Ours
50k	79.7±0.2	79.8±0.1	**80.0±0.1**	79.3±0.1	79.5±0.2	**79.7±0.1**
20k	79.6±0.1	79.6±0.1	**79.9±0.1**	79.2±0.2	79.4±0.1	**79.6±0.0**

Table 14: Test AUC on the Criteo dataset with DeepFM and AutoInt.