Soft Target-Enhanced Matching Framework for Deep Entity Matching

Wenzhou Dou\textsuperscript{1}, Derong Shen\textsuperscript{1*}, Xiangmin Zhou\textsuperscript{2}, Tiezheng Nie\textsuperscript{1}, Yue Kou\textsuperscript{1}, Hang Cui\textsuperscript{3}, Ge Yu\textsuperscript{1}

\textsuperscript{1}Northeastern University, China  
\textsuperscript{2}RMIT University, Australia  
\textsuperscript{3}University of Illinois at Urbana-Champaign, USA

wenzhoudou@gmail.com, \{shenderong, nietiezheng, kouyue, yuge\}@cse.neu.edu.cn  
xiangmin.zhou@rmit.edu.au, hangcui2@illinois.edu

Abstract

Deep Entity Matching (EM) is one of the core research topics in data integration. Typical existing works construct EM models by training deep neural networks (DNNs) based on the training samples with onehot labels. However, these sharp supervision signals of onehot labels harm the generalization of EM models, causing them to overfit the training samples and perform badly in unseen datasets. To solve this problem, we first propose that the challenge of training a well-generalized EM model lies in achieving the compromise between fitting the training samples and imposing regularization, i.e., the bias-variance tradeoff. Then, we propose a novel Soft Target-Enhanced Matching (TEAM) framework, which exploits the automatically generated soft targets as label-wise regularizers to constrain the model training. Specifically, TEAM regards the EM model trained in previous iteration as a virtual teacher and takes its softened output as the extra regularizer to train the EM model in the current iteration. As such, TEAM effectively calibrates the obtained EM model, achieving the bias-variance tradeoff without any additional computational cost. We conduct extensive experiments over open datasets and the results show that our proposed TEAM outperforms the state-of-the-art EM approaches in terms of effectiveness and label efficiency.

Introduction

Entity Matching (EM) aims to identify the matching record pair that refer to the same real-world entity. As a fundamental research in data cleaning and integration, EM has been widely applied in many application scenarios including E-commerce and population census etc. Considering two data sources in Figure 1, EM determines whether the candidate record pairs (i.e., $a_1$ and $b_1$, $a_1$ and $b_2$) are matching or not, such that data integration and management tasks can be subsequently conducted based on the matching results. The comprehensive understanding of record semantics has become increasingly important, which requires deep EM as solutions in real scenarios.

Existing deep EM can be categorized into two types: deep learning-based (DL-based) (Mudgal et al. 2018; Fu et al. 2019; Li et al. 2020a; Fu et al. 2020) and pretrained language model-based (PLM-based) (Brunner and Stockinger 2020; Li et al. 2020b; Yao et al. 2022). DL-based approaches treat the elements of each record as a token sequence or a graph topology, which are then fed into deep learning models to predict the matching probability of the record pair from two given sources. And PLM-based approaches concatenate the record pair as a sentence in the format of $[\text{cls}] \ record_1 \ [\text{sep}] \ record_2 \ [\text{sep}]$, and feed it into the PLM to obtain the highly contextualized embeddings which improve the effectiveness of EM models. Although these approaches have good performance on training pairs, both types of EM models face the common problem of poor performance on unseen testing pairs. Due to the diversity of record distribution caused by unreliable sampling strategy (Goodfellow, Bengio, and Courville 2016), the information in training data can be far from that in testing data, causing the low quality of matching for testing data using the trained EM model. Thus, there is a requirement for designing an advanced method to well identify the matches over unseen record pairs.

To address this challenge, we need to generalize the obtained EM model to fit the unseen record pairs that belong to the item categories different from those for any training data. The popular approaches designed for generalizing EM models include model-oriented and data-oriented. Model-oriented approaches (e.g., model redesigning (Zhang et al. 2020; Fu et al. 2020)) introduce higher parameterized EM models that have higher capacity to learn universal features. Data-oriented approaches apply data augmentation (Li et al. 2020b, 2021b) or inject domain knowledge (Li et al. 2020b) to improve the data efficiency and avoid the model overfitting against the limited training data. However, both ap-
Deep entity matching adopts deep neural networks (DNNs) as encoders to encode the similarity features into fixed length vectors. DeepER (Ebraheem et al. 2018) and DeepMatcher (Mudgal et al. 2018) are the earliest attempts to apply deep learning technologies to entity matching tasks. DeepER adopts LSTM to convert the record to a distributed representation to capture the similarity between them. And DeepMatcher is a design space for EM tasks, and reveals the advantages of DL technologies for EM, especially in dirty and textual scenarios. Later, a series of improved methods (Fu et al. 2019; Zhang et al. 2020; Fu et al. 2020) are proposed. MPM (Fu et al. 2019) designs various similarity measures and adaptively selects the optimal measure for heterogeneous attributes in an end-to-end manner. MCA (Zhang et al. 2020) fully exploits the semantic context of embeddings for the record pairs, which takes into account the multi-context attention like self-attention, pair-attention, and global-attention for three types of context. And HierMatcher (Fu et al. 2020) is an end-to-end hierarchical matching network for deep entity matching, which jointly matches the entities in three levels—token, attribute, and entity. Some other works (Li et al. 2020a; Cappuzzo, Papotti, and Thirumuruganathan 2020; Chen, Shen, and Zhang 2021) consider representing the record hierarchy as graph topology and adopt graph representation learning technologies (e.g., GCNs (Kipf and Welling 2016)) to obtain embeddings for record pairs. For example, GraphER (Li et al. 2020a) is a graph-based EM model, which adopts an Entity Record Graph Convolutional Network (ER-GCN) to embed the semantic and structural information of record pairs. EMBDI (Cappuzzo, Papotti, and Thirumuruganathan 2020) designs a compact tripartite graph which effectively represents the syntactic and semantic relationship between the cell values, and uses random walk to obtain the local embeddings for data integration tasks such as entity matching and schema matching.

With the development of pretrained language models, the pretraining-then-finetuning procedure has become a new paradigm for deep EM (Brunner and Stockinger 2020; Li et al. 2020b; Peeters and Bizer 2021; Li et al. 2021a; Dou et al. 2022; Yao et al. 2022). Ditto (Li et al. 2020b) improves the matching capability of PLM-based EM model by developing three optimization techniques including domain knowledge injection, text summarization, and data augmentation. JointBERT (Peeters and Bizer 2021) combines the tasks of entity matching and classification, and proves that this dual-object training paradigm effectively improves the matching quality. GTA (Dou et al. 2022) enhances PLMs for relational data representation by injecting additional hybrid matching knowledge from a hybrid matching graph. HierGAT (Yao et al. 2022) proposes a Hierarchical Graph Attention Transformer Network to capture multiple relationships among entities to improve matching quality.

This paper has a different starting point from the above. We focus on the challenge in training a well-generalized deep EM model with the label-wise perspective. To the best of our knowledge, this is the first effort to realize it in a label-wise way, which avoids the model redesigning and heavy data preprocessing.

### Problem and Preliminary

Given a candidate record pair set $C$, entity matching aims to identify the matching pair set $M \subseteq C$, where each matching record pair refers to the same real-world entity (software, person, shop, and so on). Each record is a set of key-value pairs $r = \{(name_i, val_i)\}_{1 \leq i \leq k}$, where $name_i$ and $val_i$ refer to the attribute name and the attribute value with textual or numerical type, respectively. The EM model $f(\cdot)$ includes a training phase and a testing phase. This paper aims to generalize EM model in the training phase via label-wise regularization.

We formulate EM as a binary classification task. In the training phase, the EM model $f(\cdot)$ is fed with a training set...
Entity Matching Pipeline in STEAM

This section proposes the soft target-enhanced matching framework, STEAM, for the deep entity matching tasks.

Framework Overview

Figure 2 depicts the proposed STEAM framework which consists of an entity matching pipeline and a model training pipeline. The entity matching pipeline contains an EM model to identify the matching pairs. It takes the unseen candidate set as input, and outputs the matching set. The model training pipeline takes the labeled samples as training set and optimizes the EM model in the manner of soft supervised training. The goal of STEAM is to obtain a well-generalized EM model to effectively identify the matching pairs from the unseen candidate set.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Definition and description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C$</td>
<td>The candidate record pair set</td>
</tr>
<tr>
<td>$D$</td>
<td>The training record pair set</td>
</tr>
<tr>
<td>$M$</td>
<td>The matching record pair set</td>
</tr>
<tr>
<td>$x$</td>
<td>The input record pair $x = {r_1, r_2}$, where $x \in C$</td>
</tr>
<tr>
<td>$L$</td>
<td>The label set $L = {\text{match, unmatch}}$</td>
</tr>
<tr>
<td>$z^1_i, z^t_i$</td>
<td>The output logit of teacher and student</td>
</tr>
<tr>
<td>$\tau$</td>
<td>The softening temperature</td>
</tr>
<tr>
<td>$p_1, p^t_i$</td>
<td>The original and softened output probability</td>
</tr>
<tr>
<td>$y_i, y^t_i$</td>
<td>The oneshot label and soft label</td>
</tr>
<tr>
<td>$p_{ce}, p_{ce}$</td>
<td>The output matching probability under original supervised training and its average value</td>
</tr>
<tr>
<td>$p_{st}, p_{st}$</td>
<td>The output matching probability under soft supervised training and its average value</td>
</tr>
<tr>
<td>$\lambda, \alpha$</td>
<td>The weights of soft training loss</td>
</tr>
</tbody>
</table>

Table 1: Summary of the main notations.

$D$ consisting of many record pairs and labels, where each record pair $x = \{r_1, r_2\}$ has a corresponding label from $L = \{\text{match, unmatch}\}$. In the testing phase, the trained EM model is expected to output the reliable matching probabilities of unseen record pairs. The notations used in this paper are summarized in Table 1.

Figure 2: The overview of STEAM framework.

Input Serialization Layer. For each candidate pair $x = \{r_1, r_2\}$ as the input of EM model, we serialize it as $[\text{cls}] r_1 \ [\text{sep}] r_2 \ [\text{sep}]$. The special tokens $[\text{cls}]$ and $[\text{sep}]$ act as the separators to split the record pair. We further perform a finer-grained separation. For each record, we split the attributes in the format of $[\text{nam}] \ [\text{name}] \ [\text{val}] \ [\text{value}]$, where the special tokens $[\text{nam}]$ and $[\text{val}]$ are the additional attribute-level separators to split each attribute, which indicates the start of attribute names and values.

Feature Extraction Layer. The feature extraction layer is initialized with a PLM such as RoBERTa. It is based on the Transformer architecture and has been pretrained with the large corpus. Therefore it contains the universal language knowledge to understand the semantic features of record pairs. The attention mechanism of Transformer conducts token-level comparison and aggregation, and generates highly contextualized embeddings for record pairs.

Entity Matching Layer. We employ a mean pooling layer as a feature aggregator. It aggregates all the contextualized token embeddings into a similarity vector. We then feed it into a two-layer MLP with softmax to output the matching probability.

Model Training Pipeline in STEAM

The model training pipeline trains highly generalized EM models via our proposed soft supervised training, which consists of three parts—label-wise regularization, training objectives construction, and soft training loss weighting.

Label-wise Regularization. As is shown in Figure 3, we design two training targets for the EM model in STEAM. They are (1) the manual oneshot labels $y_i$ for the original supervised training, which is commonly used in most existing EM approaches, and (2) the automatically generated soft labels $y^t_i$ for the label-wise regularization, which calibrates the model training and balances the bias and variance. It is inspired by the idea of knowledge distillation (KD) that the outputs of the cumbersome model (i.e., teacher) provide rich inter-class relationship, which can be naturally considered as soft target to regularize the lightweight model (i.e., student) label-wisely (Hinton, Vinyals, and Dean 2015). Unlike the original KD, we address the distillation procedure in a teacher-free mode, which takes the previous iteration’s EM model as a virtual teacher to regularize the EM model in current iteration. Therefore, for the EM model in current iteration, it takes two ways of supervision, one is the oneshot labels $y_i$ from the training set and the other is the soft labels $y^t_i$ from its virtual teacher. And the soft labels $y^t_i$ inherently take into account all the previously accumulated information to improve the generalization of obtained EM model.

Training Objectives Construction. We formulate the training objectives from two perspectives. The first is fitting the EM model to the ground truth (i.e., control the bias).
The second is regularizing the EM model (i.e., control the variance). The training objectives vary across different training iterations. At the beginning of the training iterations ($T = 0$), the EM model takes the labeled pairs as inputs and obtains the sharp supervision signals from the onehot labels $y_i$. The training objective is to minimize the standard cross-entropy loss:

$$L_{ce} = \mathbb{E}[-y_i \log p_i],$$  \hspace{1cm} (1)

where $p_i$ is the model’s output probability on the $i$th class. As the training iterations continue, the subsequent EM model ($1 \leq T \leq m$) additionally acquires soft labels $y_i^\tau$ from its previous one. The soft labels $y_i^\tau$ can be obtained via temperature softmax:

$$y_i^\tau = \frac{\exp(z_i^\tau / \tau)}{\sum_j \exp(z_j^\tau / \tau)},$$  \hspace{1cm} (2)

where $z_i^\tau$ is the output logit of the $i$th class (i.e., match or unmatch) from the virtual teacher. The temperature $\tau$ controls the softening strength. A higher $\tau$ produces a smoother probability distribution over these two classes. STEAM formulates its total training objectives as two parts—cross-entropy loss $L_{ce}$ and soft training loss $L_{st}$:

$$L_{total} = L_{ce} + \lambda L_{st},$$  \hspace{1cm} (3)

The former $L_{ce}$ is computed using the same method as equation (1). And $\lambda$ controls the weight of soft training loss $L_{st}$. The soft training loss $L_{st}$ is computed as follows:

$$L_{st} = \tau^2 \mathbb{E}[-y_i^\tau \log p_i^\tau],$$  \hspace{1cm} (4)

where $p_i^\tau$ is the softened output probability of EM model in the current iteration (i.e., student), which is computed using the same method in equation (2) with equal temperature $\tau$:

$$p_i^\tau = \frac{\exp(z_i^\tau / \tau)}{\sum_j \exp(z_j^\tau / \tau)},$$  \hspace{1cm} (5)

where $z_i^\tau$ is the student’s output logit on the $i$th class. The soft training loss $L_{st}$ is scaled with the parameter $\tau^2$ since the magnitudes of the gradients produced by the soft labels scale as $1/\tau^2$. This enables the contribution of onehot labels $y_i$ and soft labels $y_i^\tau$ to remain roughly unchanged when changing the temperature $\tau$ (Hinton, Vinyals, and Dean 2015). The hyperparameter $\lambda$ controls the scale of soft training loss $L_{st}$ in the total loss $L_{total}$.

The above method introduces the soft supervised training to impose label-wise regularization, which helps a lot in balancing the bias and variance and improving the generalization ability of EM model. The hyperparameter $\lambda$ controls the regularization strength, balancing the bias and variance, that a lower $\lambda$ leads to a higher variance, and vice versa. To go further, we additionally design a more flexible weighting method that the weight of soft training loss $L_{st}$ are assigned adaptively.

**Soft Training Loss Weighting.** The sample-wise soft weighting (SSW) method is motivated by the fact that the bias and variance vary across training samples and change dynamically during training iterations (Zhou and Song 2021). The hard-to-fit samples produce overly softened labels and soft labels $y_i^\tau$, thus a lower weight should be assigned for the soft training loss $L_{st}$ to alleviate this. To achieve it, STEAM framework quantifies the bias and variance from each training sample and adaptively reduces the weight of soft training loss $L_{st}$ for those hard-to-fit samples.

For the input pair $x$ in the training set $D$, we define the output under original supervised training as $p_{ce}$ and the output under soft supervised training as $p_{st}$. Then, we can decompose the cross entropy loss $L_{ce}$ and soft training loss $L_{st}$ as follows (Heskes 1998; Zhou and Song 2021):

$$L_{ce} = \mathbb{E}_x[-y_i \log y_i] + D_{kl}(y_i, \bar{p}_{ce}) + \mathbb{E}_D[D_{kl}(\bar{p}_{ce}, p_{ce})],$$

$$L_{st} = \mathbb{E}_x[-y_i \log y_i] + D_{kl}(y_i, \bar{p}_{st}) + \mathbb{E}_x[y_i \log \frac{\bar{p}_{ce}}{p_{st}}] + \mathbb{E}_{D,\tau}[D_{kl}(\bar{p}_{st}, p_{st})],$$

where $\bar{p}_{ce}$ and $\bar{p}_{st}$ refer to the averages of $p_{ce}$ and $p_{st}$, and $D_{kl}(\cdot | \cdot)$ is the Kullback-Leibler divergence. Then the soft training loss $L_{st}$ can be rewritten as follows:

$$L_{st} = L_{ce} + L_{st} - L_{ce}$$

$$= L_{ce} + \mathbb{E}_x[y_i \log \frac{\bar{p}_{ce}}{p_{st}}] + \mathbb{E}_{D,\tau}[D_{kl}(\bar{p}_{st}, p_{st})] - \mathbb{E}_D[D_{kl}(\bar{p}_{ce}, p_{ce})].$$

It can be easily known that $L_{ce}$ leads to bias reduction that the EM model’s average output $\bar{p}_{ce}$ converges to corresponding onehot labels $y_i$, and $\bar{p}_{st}$ converges to soft labels $y_i^\tau$. Thus we can know, $\bar{p}_{ce}$ is closer to the ground truth labels $y_i$ than $\bar{p}_{st}$, which means $\mathbb{E}_x[y_i \log (\frac{\bar{p}_{ce}}{p_{st}})] \geq 0$. There is an assumption that the variance of soft supervised training is smaller than original supervised training under one-hot labels, thus $\mathbb{E}_{D,\tau}[D_{kl}(\bar{p}_{st}, p_{st})] - \mathbb{E}_D[D_{kl}(\bar{p}_{ce}, p_{ce})] \leq 0$. Therefore, $L_{st} - L_{ce}$ can be treated as an additional adversarial item which causes that the bias increases by $\mathbb{E}_x[y_i \log (\frac{\bar{p}_{ce}}{p_{st}})]$ and the variance decreases by $\mathbb{E}_D[D_{kl}(\bar{p}_{ce}, p_{ce})] - \mathbb{E}_{D,\tau}[D_{kl}(\bar{p}_{st}, p_{st})]$. Accordingly, for
soft training loss $L_{st}$, the bias-variance tradeoff reflects the change of $L_{ce}$ and $L_{st} - L_{ce}$. If the gradient $a = \frac{\partial L_{ce}}{\partial z_i}$ lower than $b = \frac{\partial (L_{st} - L_{ce})}{\partial z_i}$, the variance dominates the overall optimization of total loss $L_{total}$. We call them hard-to-fit samples, which imposes too much label-wise regularization via soft training loss $L_{st}$, and confuses a lot to the optimization direction. Thus, we should assign lower weights of $L_{st}$ for these samples, and vice versa.

To make the SSW method independent of temperature $\tau$, we set $\tau = 1$, so that $a = p_{i+1,1}^s - y_i$, and $b = y_i - p_{i+1,1}^b$, where $p_{i+1,1}^s$ and $p_{i+1,1}^b$ refer to the output of the student and corresponding virtual teacher. Then the comparison between $a$ and $b$ has been converted into the comparison between $p_{i+1,1}^s$ and $p_{i+1,1}^b$. If the student EM model performs better on a training sample than the corresponding virtual teacher which means $p_{i+1,1}^s > p_{i+1,1}^b$, a smaller weight should be assigned to the soft training loss $L_{st}$ of this training sample. The final total loss $L_{total}$ can be formulated as follows:

$$L_{total} = L_{ce} + \alpha L_{st}$$

$$= L_{ce} + \lambda (1 - \exp(-\frac{\log p_{i+1,1}^s}{\log p_{i+1,1}^b})) L_{st}$$

$$= L_{ce} + \lambda (1 - \exp(-\frac{\log L_{st}^s}{\log L_{ce}^s})) L_{st},$$

where $\alpha$ is the additional sample-wise soft weight of $L_{st}$ that changes dynamically across training samples and iterations. And $L_{st}^s$ and $L_{ce}^s$ refer to the standard cross-entropy loss of student EM model and its virtual teacher EM model respectively.

### Experiments

In this section, we evaluate our proposed STEAM framework on two open EM benchmarks (eight datasets) to demonstrate its performance against existing SOTA methods.

### Benchmarks and Metrics

We evaluate STEAM framework on WDC benchmark (Primpeli, Peeters, and Bizer 2019) and DeepMatcher benchmark (Mudgal et al. 2018). The summaries of them are shown in Table 2 and Table 3.

For WDC benchmark, we split the training/validation sets with the ratio of 4:1, which is the same as Ditto (Li et al. 2020b). We evaluate STEAM on all the WDC subsets—computers, cameras, watches, and shoes. All the WDC subsets contain four attributes—title, description, brand, and specTableContent. For the sake of fairness, we only use the title attribute following Ditto (Li et al. 2020b), so that all the results can be fairly compared. We evaluate STEAM on all the versions (i.e., Small, Medium, Large, and xLarge) and comprehensively compare the results with Hybrid (Mudgal et al. 2018) and Ditto (Li et al. 2020b) to show the EM performance under different training scales.

For DeepMatcher benchmark, we split the training/validation/testing sets with the ratio of 3:1:1, which is the same as existing methods like DeepMatcher (Mudgal et al. 2018), Ditto (Li et al. 2020b), and HierGAT (Yao et al. 2022). In all the subsets of this benchmark, we choose the challenging and representative datasets—Amazon-Google (A-G), Walmart-Amazon (W-A) with its dirty version W-A*, and Abt-Buy (A-B). The A-G dataset contains three attributes—title, manufacturer, and price. And W-A and W-A* contain five attributes—title, category, brand, modelno, and price. The W-A* dataset is manually generated by randomly injecting other attribute values into title to simulate a common kind of dirty data seen in the wild. And the A-B dataset contains three attributes—name, description, and price, at least one of which is long text. These datasets are representative enough to evaluate STEAM in common EM scenarios.

Following previous researches, we report the F1 score to evaluate the EM performance of proposed STEAM. All the compared results are obtained from the original paper.

### Implementation and Training Details

We implement STEAM framework with PyTorch and HuggingFace libraries. All the BERT-like PLMs (e.g., BERT (Devlin et al. 2019), RoBERTa (Liu et al. 2019), DistilBERT (Sanh et al. 2019)) are supported in STEAM framework. We adopt RoBERTa-base as the default PLM and report its results in all the benchmarks. The size of mini-batch is 64, and the maximum length of the input is limited to 128 (256 for A-B dataset) that any tokens beyond that are truncated. We train STEAM using Adam optimizer and the learning rate is 3e-5. The maximum training epoch is 50, and we adopt the early stop strategy with the patients varying from 5 to 15 according to the datasets. We adopt data augmentation (e.g., dropping token, swapping record, and swapping attribute values) and dropout strategy with the probability of 0.5. For the part of soft supervised training, we set the temperature $\tau = 10$ to automatically generate the soft labels. And the hyperparameter $\lambda$ is set to be 1.

### Comparisons of Effectiveness and Label Efficiency

Table 4 and 5 show the experimental results of our proposed STEAM framework compared with existing baselines. As can be seen, for both WDC and DeepMatcher benchmarks, STEAM achieves the leading F1 score and becomes...
### Ablation Study

We design three versions of STEAM to comprehensively analyze the role of pretrained language model and soft supervised training. **STEAM-BASE** refers to the version that directly finetunes the PLM on onehot labels without any optimization. **STEAM-FSW** refers to the version that adopts fixed soft weighting (FSW) method to weight the soft training loss $L_{st}$. And **STEAM-SSW** is the complete version that adapts sample-wise soft weighting (SSW) method to compute the weight of soft training loss $L_{st}$ adaptively. As is shown in Table 5, the basic version **STEAM-BASE** shows superior performance against other non PLM-based methods, this demonstrates that PLMs are powerful encoders for relational records and have become the de facto best solution in deep EM tasks. And **STEAM-FSW** is competitive against baselines, which demonstrates that the label-wise regularization can also improve the generalization ability of EM model like existing model-oriented and data-oriented approaches. And the complete version **STEAM-SSW** achieves leading performance on all the datasets. The results show that the soft supervised training with sample-wise soft weighting can effectively improve the generalization ability of EM model to obtain the promising matching results for unseen record pairs.

### Detail Analysis

In this section, we conduct the deep insight of our proposed STEAM. To control the variables, we take our basic version **STEAM-BASE** as a baseline, and analyze the results of **STEAM-SSW** with it to explain the obtained improvements in soft supervised training.

#### Tradeoff between Bias and Variance

Figure 5 (a) shows the tradeoff between bias and variance in WDC-Computers dataset, where the bias is computed as

$$bias^2(x) = (\hat{f}(x) - y)^2,$$  
(10)

and the variance is computed as

$$var(x) = E_D[(f(x) - \hat{f}(x))^2].$$  
(11)

With the reduction of bias (deep and light green lines), the outputs of STEAM converge to ground truth. What we need to pay more attention is the change of variance (deep and

<table>
<thead>
<tr>
<th>A-G</th>
<th>W-A</th>
<th>W-A*</th>
<th>A-B</th>
</tr>
</thead>
<tbody>
<tr>
<td>49.1</td>
<td>71.9</td>
<td>37.4</td>
<td>43.6</td>
</tr>
<tr>
<td>Magellan (Konda et al. 2016)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RNN (Mudgal et al. 2018)</td>
<td>59.9</td>
<td>67.6</td>
<td>39.6</td>
</tr>
<tr>
<td>Attention (Mudgal et al. 2018)</td>
<td>61.1</td>
<td>50.0</td>
<td>53.8</td>
</tr>
<tr>
<td>Hybrid (Mudgal et al. 2018)</td>
<td>69.1</td>
<td>66.9</td>
<td>46.0</td>
</tr>
<tr>
<td>MPM (Fu et al. 2019)</td>
<td>70.7</td>
<td>73.6</td>
<td>-</td>
</tr>
<tr>
<td>Seq2SeqM (Nie et al. 2019)</td>
<td>-</td>
<td>78.2</td>
<td>68.3</td>
</tr>
<tr>
<td>GraphER (Li et al. 2020a)</td>
<td>68.08</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>MCA (Zhang et al. 2020)</td>
<td>70.3</td>
<td>73.4</td>
<td>-</td>
</tr>
<tr>
<td>HierMatcher (Fu et al. 2020)</td>
<td>74.9</td>
<td>81.6</td>
<td>68.5</td>
</tr>
<tr>
<td>BERT-ER (Li et al. 2021a)</td>
<td>75.3</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Ditto (Li et al. 2020b)</td>
<td>75.63</td>
<td>86.97</td>
<td>85.69</td>
</tr>
<tr>
<td>HierGAT (Yao et al. 2022)</td>
<td>76.4</td>
<td>88.2</td>
<td>86.3</td>
</tr>
<tr>
<td>STEAM-BASE</td>
<td>73.85</td>
<td>84.32</td>
<td>83.33</td>
</tr>
<tr>
<td>STEAM-FSW</td>
<td>75.10</td>
<td>87.07</td>
<td>85.42</td>
</tr>
<tr>
<td>STEAM-SSW</td>
<td>77.91</td>
<td>88.41</td>
<td>87.23</td>
</tr>
</tbody>
</table>

Table 5: F1 scores on the DeepMatcher benchmark.

Table 4: F1 scores on the WDC benchmark. We calculate $\Delta F1$ and $\Delta F1_{\text{ave}}$ against Ditto.
In this paper, we propose a novel Steam framework to improve the generalization of EM model. It leverages the teacher-free knowledge distillation to automatically generate soft targets to conduct a label-wise regularization. By taking the softened outputs from the previous iteration’s virtual teacher as additional regularizers to train the current iteration’s EM model, Steam effectively calibrates the EM model and balances the bias and variance without additional computational cost. Extensive experiments on two open EM benchmarks (eight datasets) show that our proposed Steam framework effectively improves the performance of obtained EM model on unseen testing data in terms of effectiveness and label efficiency.

Conclusions
Acknowledgments

This work is supported by National Natural Science Foundation of China (62172082, 62072084, 62072086), and the Fundamental Research Funds for the Central Universities (N2116008).

References


