ProtoUDA: Prototype-based Unsupervised Adaptation for Cross-Domain Text Recognition

Xiao-Qian Liu, Xue-Ying Ding, Xin Luo, and Xin-Shun Xu*, *Senior Member*, *IEEE*

Abstract—Text recognition reads from real scene text or handwritten text, facilitating many real-world applications such as driverless cars, visual Q&A, and image-based machine translation. Although impressive results have been achieved in single-domain text recognition, it still suffers from great challenges in cross-domain due to the domain gaps among the synthetic text, the real scene text, and the handwritten text. Existing standard unsupervised domain adaptation (UDA) methods struggle to solve the text recognition task since they view a domain or a text image (containing a character sequence) as a whole, ignoring the subunits that make up the sequence. In the paper, we present a Prototyped-based Unsupervised Domain Adaptation method for text recognition (ProtoUDA), where the class prototypes are computed from the source domain, target domain, and the mixed (source-target) domain, respectively. Technically, ProtoUDA initially extracts pseudo-labeled character features under word-level supervised information. Further, based on these character features, we propose two parallel and complementary modules to perform class-level and instance-level alignment, which explicitly transfer the knowledge learned in the source domain to the target domain. Among them, class-level alignment is to close the distance between the similar source prototypes and target prototypes. The instance-level alignment is based on contrastive learning, making the character instances of the mixed domain close to the corresponding class mixed prototype while staying away from other class mixed prototypes. To our knowledge, we are the first to adopt contrastive learning in UDA-based text recognition tasks. Extensive experiments on several benchmark datasets show the superiority of our method over state-of-the-art methods.

Index Terms—Unsupervised learning, Prototype, Text recognition, Contrastive learning, Domain adaptation.

1 Introduction

SINGLE-domain text recognition has made great progress in recent years [1], [2], [3]. However, it is still much more challenging in cross-domain text recognition [4], [5], [6], in which the source and target domains are different. One of the main reasons is that there are various inter-domain gaps, such as strokes, structure, and background; consequently, a model trained in a source domain may degrade in performance significantly when applied directly to a target domain, which is known as the domain drift problem [7]. In addition, in many scenarios, people usually lack sufficient labeled target data; as a result, fine-tuning a model may lead to overfitting problems. Therefore, it is an important and challenging problem to transfer the knowledge learned in a source domain to a target domain to mitigate the domain gaps in cross-domain text recognition.

Unsupervised domain adaptation [8] is an effective way to address the domain drift problem. It aims to mitigate the domain gaps by exploiting unlabeled target data. Therefore, the domain drift problem may be mitigated when unsupervised domain adaptation is applied to cross-domain text recognition. Many UDA methods have been proposed, *e.g.*, statistical based methods [9], [10], [11], adversarial learning based ones [12], and self-training based ones [13], [14]. However, applying these methods directly to text recognition tasks is not appropriate. For example, as shown in Fig. 1 (a), it treats the entire domain as a class, even if a domain contains different instances. These methods ignore that text sequences are composed of individual characters that can be considered subclasses. The characters are the minimal units that make up the text sequence. The correct recognition of a sequence depends on the accurate recognition of each subclass character.

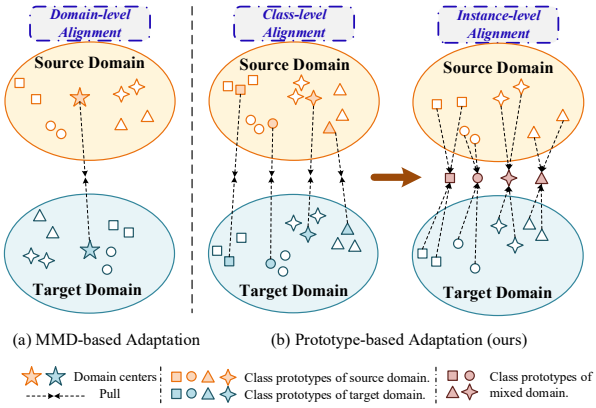


Fig. 1. Different alignment for domain adaptation: (a) MMD-based adaptation which treats a domain as a class. (b) Our proposed method which implements feature alignment at class and instance levels.

More recently, the UDA-based method ASSDA [4] is proposed for cross-domain text recognition. Although it has considered the character classes, it views all character classes in the source or target domain as one class and does not distinguish them in different character classes. In other words, ASSDA essentially treats the entire domain as a class and makes two classes indistinguishable, *i.e.*, the source class and the target class. Therefore, ASSDA

• *X.-Q. Liu, X.-Y. Ding, X. Luo, and X.-S. Xu are with the School of Software, Shandong University, Jinan 250101, China.*
E-mail: xuxinshun@sdu.edu.cn

Manuscript received 2 Feb. 2023; revised 11 Oct. 2023 and 5 Dec. 2023; accepted 13 Dec. 2023.
(Corresponding author: Xin-Shun Xu.)

can extract domain-invariant visual representations but not fine-grained domain-invariant visual representations, *i.e.*, character-level features.

In this work, we explore the distribution alignment of the fine-grained character features, where identical characters are a class, to extract fine-grained domain-invariant visual representations at the character level. Moreover, inspired by TPN [14], we assume the existence of three types of class prototypes in an embedding space, *i.e.*, source prototypes, target prototypes, and mixed prototypes, respectively. Intuitively, on the one hand, the prototypes of the same class originating from the source and target domains should be close in the embedding space; on the other hand, similar characters in the mixed domain generated by mixing the source and target data should be as close as possible to their class mixed prototypes while staying away from other mixed prototypes.

With the above motivation, we propose a prototype-based unsupervised domain adaptation method tailored for cross-domain text recognition (ProtoUDA). As shown in Fig. 1 (b), it implements fine-grained character feature alignment at the class and instance levels. Specifically, based on the attention decoding, a fine-grained representation module is introduced to extract pseudo-labeled character features. Then, two parallel and complementary modules are designed to perform class-level and instance-level alignment, respectively. The former is based on source and target prototypes, formulated to reduce the Euclidean distance of similar class prototypes in the embedding space. Since there is no explicit character-level supervised information, inspired by the superiority of contrastive learning in self-supervised visual representations, the latter is based on a contrastive paradigm in the mixed domain. Characters in the mixed domain should be close to their mixed prototype while staying away from other class prototypes. With the dual alignment of character features, ProtoUDA can extract fine-grained domain-invariant feature representations. In the inference phase, the class-level alignment and instance-level alignment modules are removed. The test text is decoded directly; thus, inference efficiency is guaranteed.

In summary, our main contributions are as follows:

- We propose a novel prototype-based UDA method tailored for text recognition, ProtoUDA, which can learn fine-grained and domain-invariant representations to transfer the knowledge learned in the source domain to target domains.
- Two parallel and complementary modules are designed to achieve class-level and instance-level alignment, where the latter is enabled with contrastive learning. To the best of our knowledge, we are the first to adopt contrastive learning in UDA-based text recognition tasks.
- Extensive experiment results demonstrate the superiority of our ProtoUDA. It significantly boosts the accuracy of cross-domain text recognition and achieves state-of-the-art (SOTA) performance on benchmark datasets.

2 Related Work

This section reviews the literature on unsupervised domain adaptation, deep learning-based text recognition, and domain adaptation for text recognition.

2.1 Unsupervised Domain Adaptation

Unsupervised domain adaptation is a subtopic of semi-supervised learning, which aims to transfer the knowledge learned in labeled source data to unlabeled target data [15], [16], [17]. These UDA methods can be categorized into three branches. The first branch is to align feature distributions across domains based on statistical characteristics, *e.g.*, Maximum mean discrepancy (MMD) [9] and correlation alignment distance (CORAL) [10], [11]. The second branch is adversarial learning methods based on a zero-sum two-player game. For example, the pioneering work DANN [18] learns domain invariant representations by formulating the problem as a minimax game. Coupled GANs [19] directly applies GANs to domain adaptation to explicitly reduce the domain drift by learning a joint distribution of multi-domain images. The third branch is self-training-based methods, which iteratively retrain a model through the pseudo-labeled target samples. For instance, Huang *et al.* [20] constructed a categorical domain-mixed dictionary from the labels of the source data and the pseudo labels of the target data and proposed a novel category contrast technique (CaCo) introducing semantic priors on top of instance discrimination for visual UDA tasks.

2.2 Deep Learning-based Text Recognition

With the development of deep learning, text recognition has attracted attention over the past years [21], [22], [23]. Researchers have viewed text recognition as a sequence recognition [24], which can be broadly divided into three categories: CTC-based [25], [26], attention-based [27], [28], [29], [30], and transformer-based [31], [32] methods. Connectionist Temporal Classification (CTC) layer [33] makes end-to-end sequence discriminative learning possible. Nevertheless, CTC-based methods [34] cannot handle complicated two-dimensional structures, such as irregular curve text. In this case, attention-based methods show superiority, which can attend to relevant information for each character during the decoding stage. For example, Baek *et al.* [35] proposed a four-stage model, usually utilized as a baseline model. However, since attention-based methods are autoregressive decoding, many transformer-based methods are proposed, which can decode in parallel and thus improve the decoding efficiency.

Recently, contrastive learning-based self-supervised methods have shown promising results in learning visual representations [36], [37]. The key idea of contrastive learning is to keep positive pairs close while negative pairs are far away. Inspired by SimCLR [38], SeqCLR [39] firstly apply contrastive learning to text recognition, where patches from different visual augmented images are considered as positive pairs. In addition, considering the contextual constraints, ConCLR [2] generates characters with different contexts via simple image concatenation operations and then optimizes the model by contrastive loss. Motivated by these methods, we expand the contrastive learning to UDA-based text recognition tasks. Rather than using visual augmentations to generate positive pairs, we consider a character to be a positive pair with its class prototype and a negative pair with other class prototypes.

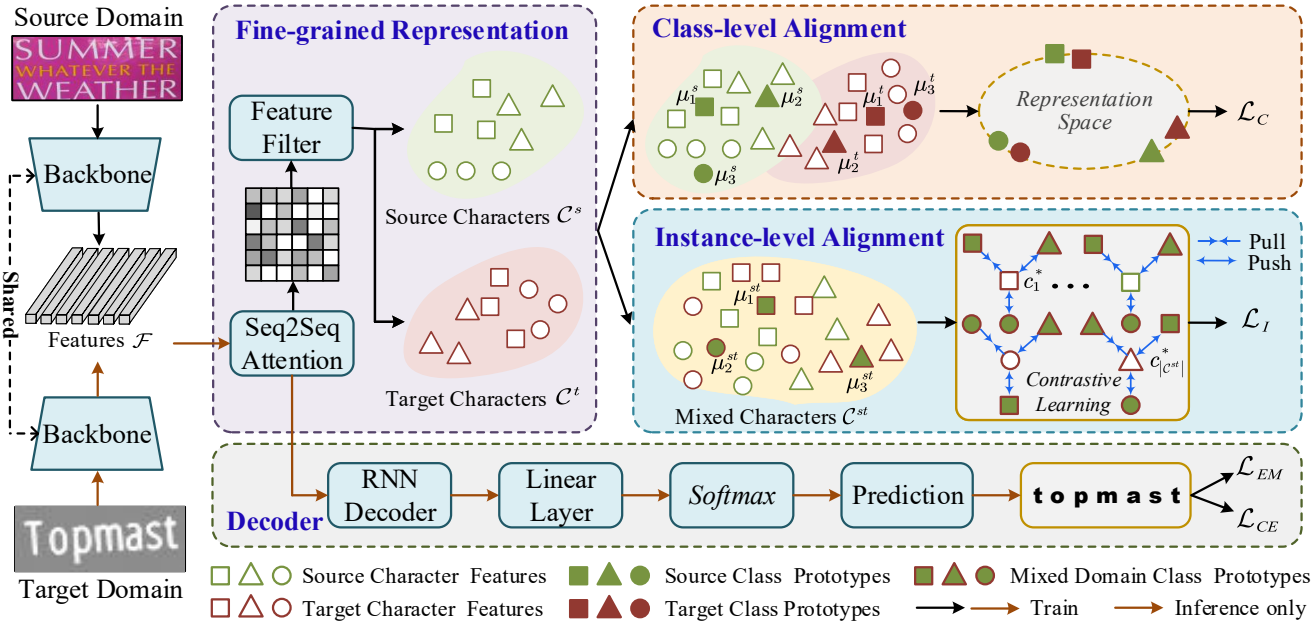


Fig. 2. The pipeline of ProtoUDA. First, the *Backbone* extracts global features $\mathcal{F}$ for source/target images. Then, on the one hand, the features $\mathcal{F}$ after *Seq2Seq Attention* is decoded autoregressively by *RNN Decoder*. Based on the predicted sequence, source data and target data are optimized by supervised cross-entropy $\mathcal{L}_{CE}$ and unsupervised entropy minimization $\mathcal{L}_{EM}$, respectively. On the other hand, two parallel and complementary modules are designed to perform *Class-level Alignment* and *Instance-level Alignment*. Since there is no explicit character-level label information in both the source and target domains, a *Fine-grained Representation* module is introduced to extract pseudo-labeled character features.

2.3 Domain Adaptation for Text Recognition

As single-domain text recognition has made great progress recently, cross-domain text recognition has gradually gained more attention. Domain adaptation for text recognition can be divided into two main categories. The first is the adaptation of writing style [40], [41], [42], where an author is considered as a domain. Owing to the differences in writing styles, handwritten text recognition (HTR) models cannot extract a general feature representation among multiple authors. Therefore, it is necessary to adapt the writing style for HTR. For example, MetaHTR [41] is a meta-learning-based framework that meta-learns instance-specific weights to exploit additional new-writer texts. Another category is the adaptation of multiple scenes [4], [5], [6], where a scene is viewed as a domain, such as synthetic text domain, real scene text domain, and handwritten text domain. For instance, SMILE [6] optimizes the source and target domains by supervised cross-entropy and unsupervised entropy minimization, respectively, ignoring the interaction between the two domains. ASSDA [4] is a sequence-to-sequence domain adaptation method for text recognition via adversarial loss based on global and local features.

3 Preliminaries

3.1 Problem Formulation

The UDA-based text recognition task generally focuses on the source and target domains, which have different low-level feature distributions but similar high-level feature distributions. Generally, the source domain is labeled synthetic text, and the target domain is unlabeled real scene or handwritten text. This task aims to mitigate the domain gaps by transferring the knowledge learned in the source domain to the target domain so that the model can perform well on real scene or handwritten text.

Formally, there are N^s annotated source samples $\mathcal{D}^S = \{(x_i^s, y_i^s)\}_{i=1}^{N^s}$ and N^t unlabeled target samples $\mathcal{D}^T = \{(x_i^t)\}_{i=1}^{N^t}$ ($N^s \gg N^t$). Unlike the standard UDA task, the source labels and prediction of text recognition are character sequences. Thus, the label of the source sample x_i^s is defined as $y_i^s = \{y_{i,1}^s, y_{i,2}^s, \cdots, y_{i,L}^s\}$, where L is the actual length of the text sequence. Correspondingly, our task aims to learn a model that performs well on $\mathcal{D}^T$.

3.2 Text Recognition Baseline Model

The baseline model used in our method contains backbone and decoder modules. In the decoder module, only the source data is optimized by supervised cross-entropy. For features $\mathcal{F} = [f_1, f_2, \cdots, f_T]$ extracted by the backbone, firstly, a Seq2Seq attention mechanism is introduced to locate the specific character features in features $\mathcal{F}$. Therefore, the representation of features $\mathcal{F}$ that are most relevant to the character y_t at time-step t is defined as a context vector g_t,

$$g_t = \sum_{i=1}^{T} \alpha_{t,i} f_i \,, \tag{1}$$

where f_i is the i-th subregion of features $\mathcal{F}$, and the attention weight $\alpha_{t,i} \in (0, 1)$ is calculated as follows,

$$\alpha_{t,i} = \frac{\exp\,(e_{t,i})}{\sum_{j=1}^{T} \exp\,(e_{t,j})} \,, \tag{2}$$

where the attention score $e_{t,i}$ indicates the correlation of the decoding character y_t with the feature $\mathcal{F}$. The attention score $e_{t,i}$ is defined as,

$$e_{t,i} = \omega^T \tanh\left(\mathcal{W}_s s_{t-1} + \mathcal{W}_f \mathcal{F} + b\right), \tag{3}$$

where ω, $\mathcal{W}_s$, $\mathcal{W}_f$, and b are trainable parameters, s_{t-1} is the hidden state of RNN at time-step t-1.

Next, the current hidden state s_t is updated as follows,

$$s_t = RNN\left(s_{t-1}, y_{t-1}, g_t\right), \tag{4}$$

where y_{t-1} is the one-hot encoding of the source label and the target prediction at time-step t-1.

Then, the probability of the character y_t is computed by,

$$p(y_t|x) = softmax\left(\mathcal{W}_o s_t + b_o\right), \tag{5}$$

where $\mathcal{W}_o$ and b_o are trainable parameters of a linear layer.

Lastly, a supervised cross-entropy is adopted to optimize the source data. For the sample $x_i^s \in \mathcal{D}^S$, the cross-entropy loss is defined as follows,

$$\mathcal{L}_{CE} = \frac{1}{N^s}\sum_{i=1}^{N^s}\sum_{t=1}^{T} -\log p(y_{i,t}^s|x_i^s), \tag{6}$$

where $p(y_{i,t}^s|x_i^s)$ is the predicted probability of each character after the *softmax* function, and N^s is the number of labeled source data.

4 Our Method

The overall architecture of ProtoUDA is depicted in Fig. 2, which consists of backbone, decoder, fine-grained representation module, and the proposed parallel alignment modules, *i.e.*, class-level and instance-level alignment. Given the input image x, the backbone first extracts the global features $\mathcal{F} = [f_1, f_2, \cdots, f_T] \in \mathbb{R}^{T\times D}$, where T is the maximum decoding length, and D is the feature dimension. Then, the features are decoded into a text sequence in the decoder module. During training, the predicted sequence is optimized by supervised cross-entropy and unsupervised entropy minimization. Our proposed parallel adaptation modules are implemented by class-level alignment and instance-level alignment. In this work, a character is referred to as an instance. In class-level alignment, character instances are aligned indirectly through class prototypes, which operate at the class level. In instance-level alignment, the optimization is done directly on each character instance and operates at the instance level. Specifically, the class-level alignment is performed by reducing the Euclidean distance between each source prototype and target prototype. The instance-level alignment is achieved to pull the characters close to its class mixed prototype while keeping other mixed prototypes far away. It is worth noting that both alignments are based on character features; therefore, a fine-grained representation module is introduced before them.

4.1 Fine-grained Representaion

Both class-level and instance-level alignment are based on fine-grained character features; therefore, the primary problem is extracting character features. Following [2], [4], we define the character feature c as the context vector g_t in RNN decoding process, *i.e.*, $c = g_t$. Since the supervised information of the source domain is word-level without explicit character-level annotation, the character features are based on pseudo labels. For $x_i^s \in \mathcal{D}^S$ and $x_i^t \in \mathcal{D}^T$, we can obtain $y_i^s = \{y_{i,1}^s, y_{i,t}^s, \cdots, y_{i,T}^s\}$ and $y_i^t = \{y_{i,1}^t, y_{i,k}^t, \cdots, y_{i,K}^t\}$ after decoding, where K is the pre-defined maximum decoding length for target domain. With word-level source supervised information, $y_{i,t}^s$ is the label of source character c^s, while in the target domain, the pseudo label of the target character c^t is the prediction $y_{i,k}^t$. Thus, we can obtain the character features of source domain $\tilde{\mathcal{C}}^s = \{(c_i^s, z_i^s)\}_{i=1}^{\tilde{M}}$, where $z_i^s = y_{i,t}^s$, and $\tilde{M}$ is the number of source character features before feature filter, and the character features of target domain $\tilde{\mathcal{C}}^t = \{(c_i^t, z_i^t)\}_{i=1}^{\tilde{N}}$, where $z_i^t = y_{i,k}^t$, and $\tilde{N}$ is the number of target character features before feature filter.

The extracted character features may be inaccurate since no explicit character-level annotation exists. To mitigate this problem, we introduce a feature filter mechanism for filtering out character features with low confidence. We assume that the character feature is relatively distinguishable with a higher probability. Therefore, we can get the filtered character features $\mathcal{C}^s$ and $\mathcal{C}^t$ as follows,

$$\mathcal{C}^s = \tilde{\mathcal{C}}^s \odot \delta(p), \ \mathcal{C}^t = \tilde{\mathcal{C}}^t \odot \delta(p), \tag{7}$$

where $\odot$ denotes element-wise multiplication, and $\delta(p)$ is a threshold function defined as follows,

$$\delta(p) = \begin{cases} 1, & p(y_{i,t}^s|x_i^s) \geq \eta \ or \ p(y_{i,k}^t|x_i^t) \geq \eta \\ 0, & otherwise \end{cases}, \tag{8}$$

where $p(y_{i,t}^s|x_i^s)$ and $p(y_{i,k}^t|x_i^t)$ are the predicted probabilities of each character after the *softmax* function, and η is a threshold hyperparameter. It means that the character feature is retained if the probability exceeds the threshold η; otherwise, it is discarded.

For the brevity of the following description, we define the filtered source character features as $\mathcal{C}^s = \{(c_i^s, z_i^s)\}_{i=1}^{M}$ and the target character features as $\mathcal{C}^t = \{(c_i^t, z_i^t)\}_{i=1}^{N}$, where M and N are the numbers of source and target character features after the feature filter, respectively.

4.2 Class-level Alignment

As mentioned before, class-level alignment is based on the source and target prototypes. Inspired by TPN, we also consider there are three types of class prototypes in embedding space: the source prototype $\mu^s = \{(\mu_e^s)\}_{e=1}^{E}$, the target prototypes $\mu^t = \{(\mu_e^t)\}_{e=1}^{E}$, and the mixed prototypes $\mu^{st} = \{(\mu_e^{st})\}_{e=1}^{E}$, where E is the number of categories. The class-level alignment is performed between the source prototypes μ^s and the target prototypes μ^t.

Based on the filtered character features $\mathcal{C}^s$ and $\mathcal{C}^t$, the source and target prototypes are computed separately. For the e-th character class, the class source prototype μ_e^s and the class target prototype μ_e^t are updated as follows,

$$\mu_e^s \leftarrow \frac{1}{2}(\mu_e^s + \frac{1}{|\mathcal{C}_e^s|}\sum_{c^s\in\mathcal{C}_e^s} c^s), \ \mu_e^t \leftarrow \frac{1}{2}(\mu_e^t + \frac{1}{|\mathcal{C}_e^t|}\sum_{c^t\in\mathcal{C}_e^t} c^t), \tag{9}$$

where $\mathcal{C}_e^s$ and $\mathcal{C}_e^t$ denote the set of e-th class character features of source and target domains, and $|\mathcal{C}_e^s|$ and $|\mathcal{C}_e^t|$ are the numbers of e-th class characters of source and target domains, respectively.

We believe that the more similar the distribution of character features in the source and target domains, the more fine-grained domain-invariant features can be extracted. To reduce the domain discrepancies, we keep the same

source prototype and target prototype as closely as possible. Formally, we define class-level alignment as reducing the Euclidean distance between class prototypes,

$$\mathcal{L}_C = \frac{1}{E}\sum_{e=1}^{E} \| \mu_e^s - \mu_e^t \|^2, \tag{10}$$

where E is the number of categories. In our task, E is 38, which contains 10 numbers, 26 case-insensitive letters, a start symbol ['GO'], and a stop symbol ['S'].

Comparison with MMD. MMD [9] is a kernel method to compare distributions between the source and target domains. The empirical estimation of MMD is computed by,

$$\mu^s \leftarrow \frac{1}{|\mathcal{D}^s|}\sum_{x^s \in \mathcal{D}^s} \phi(x^s)\,,\ \mu^t \leftarrow \frac{1}{|\mathcal{D}^t|}\sum_{x^t \in \mathcal{D}^t} \phi(x^t)\,, \tag{11}$$

$$\mathcal{L}_{MMD} = \| \mu^s - \mu^t \|^2, \tag{12}$$

where x^s (x^t) is a source (target) sample, $|\mathcal{D}^s|$ ($|\mathcal{D}^t|$) is the number of the source (target) samples, and $\phi(\cdot)$ is a mapping function. We can see the difference by comparing Eq. 9 and Eq. 11. MMD treats a domain as a whole and no longer distinguishes instances within a domain. Thus, a domain could get a prototype. In contrast, we focus on distinguishing instances within a domain, *i.e.*, generating a prototype for each category within a domain.

Comparison with TPN. TPN [14] implements general-purpose domain adaptation based on three types of class prototypes in an embedding space. It brings three class prototypes close to each other. Differently, we merely consider the prototypes μ^s and μ^t on class-level alignment, whereas the mixed prototypes μ^{st} are employed separately for instance-level alignment. In addition, TPN addresses non-sequential adaptation, while ours is sequence-to-sequence fine-grained domain adaptation.

4.3 Instance-level Alignment

Class-level adaptation aligns from a global perspective, integrally measuring the source and target domains and ignoring the character features within a domain. In class-level adaptation, similar samples interact indirectly, *i.e.*, aligned via class prototypes. Therefore, we further achieve instance-level adaptation.

Compared to the indirect alignment between characters in class-level adaptation, we would like to learn discriminative fine-grained character representations more directly. In addition, there is no explicit character-level supervised information for characters in either the source or target domains because the supervised information of the source domain is word-level, and the target domain is without any supervised information. Therefore, inspired by the superiority of contrastive learning in self-supervised visual representation, we adopt the contrastive learning paradigm in instance-level adaptation.

To achieve character feature alignment across domains, a natural idea is that the source character is close to the corresponding target prototype; conversely, the target character is close to the corresponding source prototype. However, when the domain discrepancies between the source and target domains are more obvious, such as synthetic text and handwritten text, this contrastive way may interfere with the unique features of the target characters, such as the stroke characteristics of handwritten text. Thus, we design a mixed-domain contrastive learning way capable of extracting common features of the source and target domains that contribute to knowledge transfer without overly interfering with the unique features of the target characters. Detailed discussions of the contrastive ways are also provided in Section 5.3.3.

Formally, given the filtered source character features $\mathcal{C}^s$ and target character features $\mathcal{C}^t$, we combine them to obtain the mixed character features $\mathcal{C}^{st}$,

$$\mathcal{C}^{st} = \mathcal{C}^s \cup \mathcal{C}^t, \tag{13}$$

where $\cup$ denotes concatenation operation. For the brevity of the following description, we denote $\mathcal{C}^{st} = \{(c_i^*, z_i^*)\}_{i=1}^{M+N}$, where c_i^* represents a character feature from source or target domain. For a query character c^* with its class pseudo-label z^*, the positive key is its corresponding mixed prototype μ_e^{st}, and the negative keys are other mixed prototypes. Then, the instance-level alignment is performed by keeping the query character c^* close to its class prototype while away from other class prototypes, which can be formulated as follows,

$$\mathcal{L}_I = -\frac{1}{|\mathcal{C}^{st}|}\sum_{c^* \in \mathcal{C}^{st}} \log \frac{\mathbb{I}(z^* = e)\exp\left(c^* \cdot \mu_e^{st}/\tau\right)}{\sum_{e=1}^{E}\exp\left(c^* \cdot \mu_e^{st}/\tau\right)}, \tag{14}$$

where $|\mathcal{C}^{st}|$ is the number of character features of the mixed domain, and τ is a temperature hyperparameter. $\mathbb{I}(z^* = e)$ = 1 if z^* is e and $\mathbb{I}(z^* = e)$ = 0 otherwise. $\cdot$ denotes the dot production used for measuring the similarity between the character feature c^* and the mixed prototype μ_e^{st}.

Comparison with InfoNCE. InfoNCE [43] is a representative contrastive learning function. For an embedding c, the InfoNCE loss is defined as follows,

$$\mathcal{L}_{InfoNCE} = -\log \frac{\exp\left(c \cdot c^+/\tau\right)}{\exp\left(c \cdot c^+/\tau\right) + \sum_{c^- \in \mathcal{C}_{c^-}}\exp\left(c \cdot c^-/\tau\right)}, \tag{15}$$

where c^+ and c^- are positive and negative keys. Looking closely at Eq. 14 and Eq. 15, we can observe some connections. Concretely, the positive and negative pairs of InfoNCE are constructed between instances. Differently, our positive and negative pairs are between character instances and class prototypes, thus playing the role of clustering in the embedding space. In addition, the label information of InfoNCE is ground truth, while our class information is the pseudo label, which is related to our task itself.

Comparison with SeqCLR. SeqCLR [39] is the first to apply contrastive learning to scene text recognition (STR), but our method differs from it in two ways. On the one hand, SeqCLR applied it to STR under a single-domain to learn visual feature representations. However, our method extends it to a UDA task for learning domain-invariant feature representations under cross-domains. On the other hand, the positive pairs of SeqCLR are constructed based on the raw sample by strong and weak augmentations, while ours are the character instance and its class prototype.

Algorithm 1 The Proposed ProtoUDA Algorithm

Input: Labeled source data $\mathcal{D}^S$, unlabeled target data $\mathcal{D}^T$, coefficients α_1, α_2, and α_3, learning rate β, batchsize B, the number of categories E, a pre-trained model f.
Output: A text recognition model f parameterized by $\tilde{\theta}_f$.

Initialize the parameters θ_f by model f, and randomly initialize the mixed prototypes $\mu^{st} = \{(\mu_e^{st})\}_{e=1}^{E}$.
repeat
  Sample B minibatch samples to get $\mathcal{D}_B^S$ and $\mathcal{D}_B^T$.
  $\tilde{\mathcal{C}}^s \leftarrow f_{\theta_f}(\mathcal{D}_B^S)$, $\tilde{\mathcal{C}}^t \leftarrow f_{\theta_f}(\mathcal{D}_B^T)$.
  Feature filter to get $\mathcal{C}^s$ and $\mathcal{C}^t$ by Eq. 7 and Eq. 8.
  repeat
    Get prototype μ_e^s and μ_e^t by Eq. 9, respectively.
  until $(e = E)$
  Get character features $\mathcal{C}^{st}$ by Eq. 13.
  $\theta_f \leftarrow \theta_f - \beta\partial_{\theta_f}(\mathcal{L}_{CE} + \alpha_1\mathcal{L}_{EM} + \alpha_2\mathcal{L}_C + \alpha_3\mathcal{L}_I)$.
  $\mu^{st} \leftarrow \mu^{st} - \beta\partial_{\mu^{st}}\mathcal{L}_I$.
until The objective function in Eq. 17 converges.
Return the optimized model parameters $\tilde{\theta}_f$.

4.4 Overall Objective Function

After decoding, the target samples are optimized by unsupervised entropy minimization. For time step k of the predicted character $y_{i,k}^t$, an entropy value is obtained. Therefore, the entropy of a text sequence $y_i^t = \{y_{i,1}^t, y_{i,k}^t, \cdots, y_{i,K}^t\}$ is the sum of all time steps. Thus, the unsupervised entropy minimization loss of the target domain is defined as follows,

$$\mathcal{L}_{EM} = \frac{1}{N^t}\sum_{i=1}^{N^t}\sum_{k=1}^{K} -p(y_{i,k}^t|x_i^t)\log p(y_{i,k}^t|x_i^t)\,, \tag{16}$$

where $p(y_{i,k}^t|x_i^t)$ is the predicted probability of each character after the *softmax* function, and N^t is the number of target samples. In our setting, the maximum decoding length K of the target domain is equal to T of the source domain.

After that, the overall training objective integrates the supervised cross-entropy loss in Eq. 6, unsupervised entropy minimization loss in Eq. 16, class-level alignment loss in Eq. 10, and instance-level alignment loss in Eq. 14. Hence, we obtain the following optimization objective,

$$\mathcal{L} = \mathcal{L}_{CE} + \alpha_1\mathcal{L}_{EM} + \alpha_2\mathcal{L}_C + \alpha_3\mathcal{L}_I\,, \tag{17}$$

where α_1, α_2, and α_3 are trade-off parameters. With this overall loss objective, the fine-grained domain-invariant feature representations can be extracted. The proposed ProtoUDA is summarized in Algorithm 1.

5 Experiments

To evaluate the performance of our method, we conduct extensive experiments on benchmark datasets. In this section, we first describe the datasets and experimental settings. After that, we give the results of ProtoUDA on different tasks and compare them with some SOTA methods. To gain deep insight, we also give ablation results and parameter analysis. Finally, we further visualize some results.

5.1 Datasets and Experimental Settings

5.1.1 Datasets

We conduct extensive experiments to validate the proposed ProtoUDA on general benchmark datasets. **Synthetic Text (Syn):** Synth90k (MJ) [44] contains 8.9 million images generated from a set of 90k common English words. SynthText (ST) [45] contains 5.5 million images with English words. MJ and ST are generally used for source training set. **Real Scene Text (RST):** Seven benchmarks are tested, including four regular texts (RT), *i.e.*, IIIT5K [46], SVT [47], IC03 [48], and IC13 [49], and three irregular texts (IT), *i.e.*, SVTP [50], CUTE80 [51], and IC15 [52]. **Handwritten Text:** According to standard partition [53], IAM [54][1] is divided into 53841 training words, 8566 validation words, and 17616 test words, including not only capital and lowercase but also the common punctuation marks. CVL [55] contains 12289 training words and 84949 test words.

5.1.2 Implementation Details

For fairness of comparison, we adopt the same backbone and protocols as in [35]. The trade-off parameters, α_1, α_2 and α_3, are set to $\{1, 0.001, 0.0001\}$ on Syn→RT and Syn→IT tasks, and $\{0.1, 0.001, 0.0001\}$ on other tasks. The confidence threshold η is set to 0.3. The temperature parameter τ is set to 1. We train the model with *Adadelta* optimizer with a learning rate initialized to 0.1. The maximum training iteration is set as 300*k*. The maximum decoding length T and K are set to 25. All experiments are conducted on an NVIDIA 2080Ti GPU with batch size 48.

5.1.3 Evaluation Metric

To evaluate performance, we adopt the metric of word accuracy for STR. Moreover, to comprehensively assess the performance, we introduce an *Average* score, which is the accuracy over the union of samples in all test datasets. For the SOTA comparison in HTR, Word Error Rate (WER) and Character Error Rate (CER) are employed. CER is defined as the Levenstein distance between the predicted and real character sequences of the word.

5.2 Performance Comparison

We evaluate the performance of ProtoUDA on cross-domain adaptation tasks of synthetic text, real scene text, and handwritten text. Since the synthetic text is only used for the training set, the experiment that synthetic text is used as the target data is not conducted. Further, the real scene text is divided into regular and irregular. Thus, the tasks of synthetic text to regular and irregular scene text are conducted. The *Baseline* model is trained using only the labeled source data, serving as a pre-trained model for our method. The *Finetune* model is finetuned to the *Baseline* model using both the labeled source and target data. It should be noted that we focus on unconstrained text recognition without any language model or lexicon for fair comparison.

1. https://fki.tic.heia-fr.ch/databases/iam-handwriting-database

TABLE 1
Results of our ProtoUDA on domain adaptation from synthetic text to real scene text compared with SOTA methods, including both regular text (RT) and irregular text (IT). The 'Baseline' is only trained on labeled source data. The 'Finetune' represents fine-tuning the baseline model using labeled synthetic and real scene text.

Model	Reference	UDA	Labeled Train	Syn→RT				Syn→IT		
				IIIT5K	SVT	IC03	IC13	SVTP	CUTE80	IC15
STAR-Net [56]	BMCV2016	N	MJ	87.0	86.9	94.4	92.8	-	71.7	76.1
RARE [29]	CVPR2016	N	MJ	86.2	85.8	93.9	92.6	-	70.4	74.5
CRNN [25]	TPAMI2017	N	MJ	82.9	81.6	93.1	91.1	-	-	-
GRCNN [57]	NIPS2017	N	MJ+PRI	84.2	83.7	93.5	90.9	-	-	-
Char-Net [58]	AAAI2018	N	MJ	83.6	84.4	91.5	90.8	-	-	60.0
STR2019 [35]	ICCV2019	N	MJ+ST	87.9	87.5	94.9	93.6	79.2	74.0	77.6
SSDAN [5]	CVPR2019	Y	MJ+ST	87.6	88.1	94.6	93.8	-	73.9	78.7
ASSDA [4]	TIP2021	Y	MJ+ST	88.3	88.6	95.5	93.7	-	76.3	78.7
SMILE [6]	arXiv2022	Y	MJ+ST	89.3	87.6	96.0	94.9	-	75.6	78.9
Baseline	ours	-	MJ+ST	87.40	87.02	95.12	92.88	80.00	74.22	78.08
ProtoUDA	ours	Y	MJ+ST	**89.33**	**89.65**	**96.05**	**96.27**	**80.47**	**78.75**	**81.23**
Finetune	ours	N	MJ+ST	91.73	91.04	95.35	95.10	81.71	75.26	81.67

5.2.1 Synthetic Text to Regular Scene Text

A domain adaptation from synthetic text to regular scene text (Syn→RT) is conducted on four regular scene datasets, *i.e.*, IIIT5K, SVT, IC03, and IC13. The source data is the synthetic text, MJ and ST, and the target data is unlabeled regular scene text. Table 1 shows the comparison results with the SOTA methods. As can be seen:

- Compared with the *Baseline* model, ProtoUDA significantly improves all four datasets. This illustrates that our method can adequately explore the target samples without supervised information.
- Compared with the SOTA methods, ASSDA [4] and SMILE [6], our method achieves the best performance. It mainly benefits from character-level adaptation, which can learn fine-grained and domain-invariant features by class-level and instance-level alignment.
- Compared to *Finetune* model, ProtoUDA achieves supervised training performance on the IC03 and IC13 datasets, demonstrating the effectiveness of our method.

5.2.2 Synthetic Text to Irregular Scene Text

The adaptation from synthetic text to irregular scene text (Syn→IT) is conducted on three irregular datasets, *i.e.*, SVTP, CUTE80, and IC15. The experimental results are shown in Table 1 in the last three columns. We can observe:

- ProtoUDA is substantially improved compared to the *Baseline* model and outperforms the SOTA methods on three datasets. It performs better than the *Finetune* model (78.75% vs. 75.26%) on the CUTE80 dataset.
- The overall performance improvement of ProtoUDA on irregular scene text is more apparent than that on regular scene text since irregular text has more variance in appearance, which is hard to generate by synthetic engine [59]. This shows that benefiting the ability of instance-level alignment to focus on fine-grained character features, ProtoUDA can extract domain-invariant character features under complex background and different illumination.

5.2.3 Synthetic Text to Handwritten Text

Synthetic text is generated based on some characteristics of real scene text, so there are some similarities between these two domains. In contrast, the handwritten text contains unique stroke and fluency characteristics. Thus, the differences between synthetic text and handwritten text are more obvious. To test the performance of ProtoUDA when the source and target domains are widely distributed differently, we experiment on the task from synthetic text to handwritten text IAM (Syn→IAM) and CVL (Syn→CVL) datasets. From Table 2, we can see that:

TABLE 2
Evaluation results on the task from synthetic text to handwritten text. † denotes the reproduced results based on its open-source code.

Model	Syn→IAM		Syn→CVL	
	WER↓	CER↓	WER↓	CER↓
Base [4], [5]	54.30	28.41	-	-
SSDAN [5]	53.65	27.26	-	-
ASSDA [4]	43.78	19.96	-	-
SMILE† [6]	45.57	19.35	64.63	30.34
Baseline	57.07	30.90	72.28	40.08
ProtoUDA	**40.20**	**16.34**	**63.27**	**29.58**
Finetune	15.96	6.02	18.41	7.23

- Compared to the *Baseline* model, ProtoUDA achieves substantial improvements. For instance, on the task of Syn→IAM, the WER is reduced by 16.87% (from 57.07% to 40.20%), and the CER is reduced by 14.56% (from 30.90% to 16.34%), respectively. This obvious enhancement demonstrates that domain gaps can be reduced through class-level and instance-level alignment.
- Compared with the SOTA methods, ProtoUDA achieves the best performance. We reproduce the SMILE method and supplement the experiments on IAM and CVL datasets according to its open-source code. It is observed that our method achieves better results than SMILE. One of the main reasons is that our class-level and instance-level alignment implement an explicit interaction between the source and target domains.
- There is still a significant performance gap compared to *Finetune* model. This illustrates the relatively wide difference between synthetic and handwritten text distribution and the necessity to explore a multi-domain text recognition model.

TABLE 3
Evaluation results on the task from handwritten text to real scene text (RST). Test‡ represents the result on IAM/CVL test set.

Domain	Model	Test‡	Regular Scene Text				Irregular Scene Text		
			IIIT5K	SVT	IC03	IC13	SVTP	CUTE80	IC15
IAM→RST	Baseline	80.53	10.63	0.62	10.58	9.80	0.47	1.74	0.83
	ProtoUDA	-	14.50	2.63	19.77	17.74	1.71	3.48	3.64
	Finetune	-	72.83	62.13	77.44	76.08	48.68	34.50	57.43
CVL→RST	Baseline	86.51	1.00	0	0.47	0.12	0	0.35	0
	ProtoUDA	-	11.73	1.55	17.67	16.80	1.09	2.79	2.49
	Finetune	-	67.83	58.89	70.00	68.73	46.36	32.40	55.00

TABLE 4
Evaluation results of different components on the task from synthetic text to real scene text, including regular scene text and irregular scene text.

Model	E_M	A_C	A_I	Syn→RT				Syn→IT			Average
				IIIT5K	SVT	IC03	IC13	SVTP	CUTE80	IC15	
1. Baseline	✗	✗	✗	87.40	87.02	95.12	92.88	80.00	74.22	78.08	85.63
2. +E_M	✓	✗	✗	89.17	88.90	95.70	94.70	80.59	76.40	80.62	87.20
3. +A_C	✗	✓	✗	88.67	88.10	95.81	93.82	80.47	75.61	79.07	86.67
4. +A_I	✗	✗	✓	88.73	87.94	94.77	94.63	80.16	78.75	78.80	86.68
5. +E_M+A_C	✓	✓	✗	89.27	89.65	96.16	94.87	81.09	78.75	81.23	87.80
6. +E_M+A_I	✓	✗	✓	89.50	89.34	96.05	95.22	80.08	75.96	81.39	87.83
7. ProtoUDA	✓	✓	✓	89.33	89.65	96.05	96.27	80.47	78.75	81.23	87.91

5.2.4 Handwritten Text to Real Scene Text

The adaptation from handwritten text to real scene text is conducted on IAM and CVL datasets. The *Baseline* model is trained only by the labeled training set of IAM/CVL datasets and serves as a pre-trained model for our method. The *Finetune* model is finetuned to *Baseline* model by the labeled training set of the IAM/CVL dataset and real scene text. The experimental results are shown in Table 3. It can be observed that:

- Compared with the performance of *Baseline* model on the IAM/CVL test set, its performance on real scene text decreases dramatically, which indicates that the *Baseline* model is unable to transfer the knowledge learned in the handwritten text to real scene text.
- Compared with *Baseline* on real scene text, ProtoUDA achieves a significant improvement. Although it can achieve knowledge transfer somewhat, the performance gap is pronounced compared to *Finetune* model.
- Jointly considering the ProtoUDA results in Table 2 and Table 3, we can find that the performance decline from handwritten text to real scene text is more apparent. This phenomenon is also demonstrated in [60]. The reason may be that synthetic text has more generalized features, and our model can extract useful features for handwritten text recognition from a large amount of synthetic text.

5.3 Ablation Study

5.3.1 Effect of Each Component

To validate the contributions of each component, *i.e.*, Entropy Minimization (E_M), Class-level Alignment (A_C), and Instance-level Alignment (A_I), we design several variants of our method, which are indexed from top to bottom in Table 4 and Table 5 as model 1-7. In the experiments, two tasks are conducted: synthetic text to real scene text and synthetic text to handwritten text. All the variants in the ablation study have the same configurations as used in ProtoUDA. Model-2, model-3, and model-4 all show improved performance in average results compared to model-1. Thereinto, entropy minimization improves the performance most significantly. Besides, model-5 and model-6 exhibit enhanced behavior compared to model-2, demonstrating that our proposed class-level and instance-level alignment can effectively extract fine-grained domain-invariant character features. Further, by simultaneously utilizing entropy minimization, class-level alignment, and instance-level alignment, the ProtoUDA consistently boosts the performance of the two tasks. These results illustrate the superiority of jointly mitigating domain discrepancies at the class and instance levels.

TABLE 5
Evaluation results of different components on the task from synthetic text to handwritten text.

Model	E_M	A_C	A_I	Syn→IAM		Syn→CVL	
				WER↓	CER↓	WER↓	CER↓
1. Baseline	✗	✗	✗	57.07	30.90	72.28	40.08
2. +E_M	✓	✗	✗	45.57	19.35	64.63	30.34
3. +A_C	✗	✓	✗	54.56	27.88	65.31	33.25
4. +A_I	✗	✗	✓	54.15	27.83	68.43	35.64
5. +E_M+A_C	✓	✓	✗	41.88	17.11	63.87	29.96
6. +E_M+A_I	✓	✗	✓	44.43	18.93	64.53	30.19
7. ProtoUDA	✓	✓	✓	40.20	16.34	63.27	29.58

5.3.2 Effect of Prototype Generation

The class prototype plays an essential role in the fine-grained alignment. To explore its effect, we design different ways of generating class prototypes for class-level and instance-level alignment. The first is based on Eq. 9, denoted by $update$, which preserves the class prototype of the previous minibatch with a certain probability. The second is that the class prototype is computed entirely from a minibatch, denoted by get. Formally, the class prototype

TABLE 6
Evaluation results of different generation ways of class prototypes.

Model	Class-level		Instance-level			Syn→RT				Syn→IT			Average
	update	get	random	update	get	IIIT5k	SVT	IC03	IC13	SVTP	CUTE80	IC15	
1. Baseline	-	-	-	-	-	87.40	87.02	95.12	92.88	80.00	74.22	78.08	85.63
2. $+A_C$	✓	-	-	-	-	88.67	88.10	95.81	93.82	80.47	75.61	79.07	86.67
	-	✓	-	-	-	88.20	89.03	96.05	93.82	80.16	77.70	78.74	86.57
3. $+A_I$	-	-	✓	-	-	88.73	87.94	94.77	94.63	80.16	78.75	78.80	86.68
	-	-	-	✓	-	88.10	88.87	95.47	94.87	80.78	76.31	78.30	86.47
	-	-	-	-	✓	88.37	88.10	95.47	93.93	81.55	76.66	78.30	86.48
4. ProtoUDA	✓	-	✓	-	-	89.33	89.65	96.05	96.27	80.47	78.75	81.23	87.91
	✓	-	-	✓	-	89.40	89.34	95.93	95.45	81.09	77.35	80.95	87.75
	✓	-	-	-	✓	89.37	88.87	95.70	94.98	80.31	78.05	81.23	87.65
	-	✓	✓	-	-	89.73	89.18	95.93	94.87	81.55	78.05	80.78	87.83
	-	✓	-	✓	-	89.53	89.34	95.70	94.98	81.24	78.40	81.01	87.79
	-	✓	-	-	✓	89.40	90.11	95.35	94.28	80.93	78.05	81.06	87.67

TABLE 7
Results of different contrastive ways of instance-level alignment.

Contrastive Way	Syn→RST	Syn→IAM	
	Average	WER↓	CER↓
intra-domain	86.31	56.98	29.79
inter-domain	86.49	56.59	29.09
mixed-domain	86.68	54.15	27.83

TABLE 8
Evaluation results of different UDA methods.

UDA Methods	Syn→RST	Syn→IAM	
	Average	WER↓	CER↓
CMMD-TR	86.92	41.88	17.37
CaCo-TR	86.96	41.30	17.54
ProtoUDA	87.91	40.20	16.34

is the mean value of the same character class features of the minibatch. The third is to randomly generate class prototypes of each class to be optimized as model parameters, denoted as $random$, which is designed separately for instance-level alignment. We conduct experiments on four variants separately. From Table 6, we can draw that:

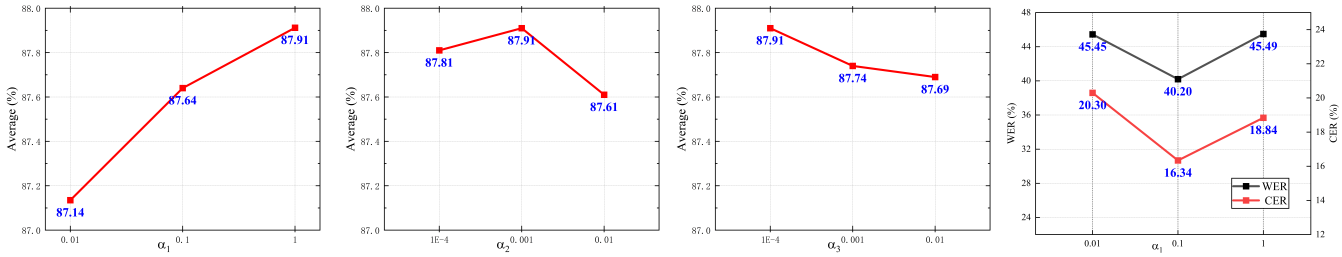
- Regardless of which generation way is taken, the performance of all variants improves compared to the *Baseline* model. This illustrates that the class prototype plays the role of clustering, which allows the aggregation of similar characters from the source and target domains.
- Slightly better results are obtained by the combination of *update* and *random*. This mitigates data perturbations to a certain extent and makes it easier to achieve the global optimum. Therefore, our method combines *update* and *random* to get the class prototypes.

5.3.3 Effect of Contrastive Way

In the instance-level alignment, the mixed prototypes are used to perform contrastive learning. In addition, we design two other ways of source prototypes and target prototypes for contrastive learning. The first one is intra-domain contrastive learning: for source characters, their positive and negative cases are source prototypes, and for target characters, their positive and negative cases are target prototypes. The second is inter-domain contrastive learning: for source characters, their positive and negative cases are target prototypes; conversely, for target characters, their positive and negative cases are source prototypes. In order to exclude the effects of other modules, the experiments are conducted on model-4 in Tables 4 and 5. From the experimental results in Table 7, it is clear that the mixed-domain way gives the best results. This is because the first way does not perform cross-domain feature alignment. Although the second way aligns features across domains, it affects the unique features of the target characters when the domain discrepancies are more pronounced. In contrast, our way is not only able to integrate the general features of the source and target data but also to not interfere unduly with the unique features of the target data.

5.3.4 Comparison with UDA Methods

To further validate the effectiveness of ProtoUDA, we compare other UDA methods that also consider category information, CMMD [61] and CaCo [20]. Direct comparison is not appropriate due to the different tasks. We replicate these methods and adapt them for deployment on the text recognition (TR) task, notated as CMMD-TR and CaCo-TR, respectively. As seen from Table 8, our proposed ProtoUDA for the text recognition task is optimal. The results are reasonable. On the one hand, the character features of these two methods may contain noise. In the original task of these two methods, explicit category supervised information exists in the source domain. Whereas in the text recognition task, there is no explicit character-level category supervised information in either the source or target domain. Therefore, we design a low-confidence character filtering mechanism in the fine-grained representation module, laying the foundation for class-level and instance-level alignment. On the other hand, the fixed-size category dictionary, the main contribution of CaCo, is not applicable to the text recognition task. In text recognition, the sampling units are images containing words, not characters, which does not guarantee a fixed size for each class key in the dictionary. This can lead to the underutilization of many character classes, which prevents effective contrastive learning.



(a) α_1 (Syn→RST) (b) α_2 (Syn→RST) (c) α_3 (Syn→RST) (d) α_1 (Syn→IAM)

Fig. 3. Effect of trade-off parameters on Syn→RST and Syn→IAM tasks.

TABLE 9
Results of proportions on Syn→RT and Syn→IT tasks.

L: U	Syn→RT				Syn→IT			Average
	IIIT5k	SVT	IC03	IC13	SVTP	CUTE80	IC15	
3:1	89.13	89.80	95.81	95.33	82.33	75.96	80.95	87.71
2:1	89.33	89.65	96.05	96.27	80.47	78.75	81.23	87.91
1:1	89.17	88.72	95.12	94.87	82.02	76.66	80.73	87.47
1:2	89.40	88.72	95.47	94.75	80.16	77.70	81.06	87.54
1:3	89.23	88.56	95.12	94.52	80.47	76.31	81.61	87.51

TABLE 10
Results of proportions on Syn→IAM.

L: U	Syn→IAM	
	WER↓	CER↓
3:1	42.03	17.11
2:1	41.49	17.12
1:1	40.20	16.34
1:2	42.11	17.34
1:3	40.75	16.90

TABLE 11
Results of different τ on Syn→RT and Syn→IT tasks.

Parameter τ	Syn→RT				Syn→IT			Average
	IIIT5k	SVT	IC03	IC13	SVTP	CUTE80	IC15	
0.2	88.40	89.18	95.35	94.75	80.78	76.66	78.36	86.60
0.4	88.13	88.56	95.35	94.28	80.16	76.66	78.74	86.44
0.8	87.43	88.56	95.70	94.98	81.09	77.35	79.46	86.56
1	88.73	87.94	94.77	94.63	80.16	78.75	78.80	86.68
2	88.40	87.94	95.58	94.28	79.85	75.61	78.41	86.38
4	88.27	88.72	95.23	94.52	80.78	75.61	79.24	86.64

TABLE 12
Results of different τ on Syn→IAM task.

Parameter τ	Syn→IAM	
	WER↓	CER↓
0.2	56.10	29.14
0.4	56.98	29.35
0.8	57.27	30.52
1	54.15	27.83
2	57.35	30.40
4	57.34	30.40

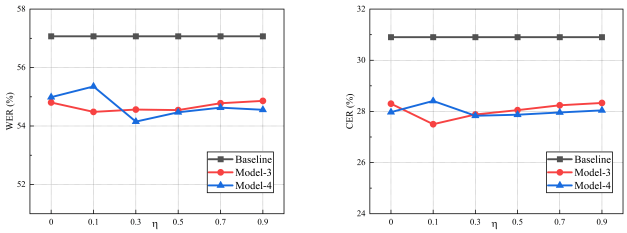
5.4 Algorithm Analysis

5.4.1 Trade-off Parameter Sensitive Analysis

In this section, we perform sensitive analysis of trade-off parameters, α_1, α_2, and α_3, on Syn→RST and Syn→IAM tasks. Specifically, we vary $\alpha_1 \in \{0.01, 0.1, 1\}$, $\alpha_2 \in \{0.0001, 0.001, 0.01\}$, and $\alpha_3 \in \{0.0001, 0.001, 0.01\}$. When studying one parameter, we fix all others with the default setting mentioned in Section 5.1. As shown in Fig. 3(a), the performance gradually improves as α_1 increases on the Syn→RST task. Under α_1=1, the model achieves better results when α_2=0.001 and α_3=0.0001. We apply this optimal combination of parameters to the Syn→IAM task. However, as can be seen from Fig. 3(d), the same parameter variation shows different results on the Syn→IAM task. This illustrates that when the feature distributions in the source and target domains differ significantly, unsupervised entropy minimization can produce overconfident results, affecting the stability of the model.

5.4.2 Proportion Parameter of Minibatch

The UDA task aims to transfer the knowledge learned in labeled to unlabeled data. Thus, each minibatch contains both labeled and unlabeled images during the training. We explore different proportions of the number of labeled and unlabeled images (L: U) in a minibatch from {3:1, 2:1, 1:1, 1:2, 1:3} on Syn→RST and Syn→IAM tasks. From Table 9 and Table 10, it can be seen that when the ratio is 2:1 on Syn→RST task and 1:1 on Syn→IAM task, better results can be obtained. It can be inferred that configuring the number of labeled and unlabeled images in this ratio allows for sufficient information exploration of the source data, which is beneficial for knowledge transfer.



(a) WER Results (b) CER Results

Fig. 4. Effect of threshold η on WER and CER on Syn→IAM task.

5.4.3 Temperature Parameter Sensitive Analysis

In instance-level alignment, we adopt contrastive learning to keep similar characters in the mixed domain close to the class mixed prototype. Thereinto, a dot production is used to represent the similarity of a character to its mixed prototype, where the temperature hyperparameter τ controls the probability distribution of the similarity. We explore the effect of

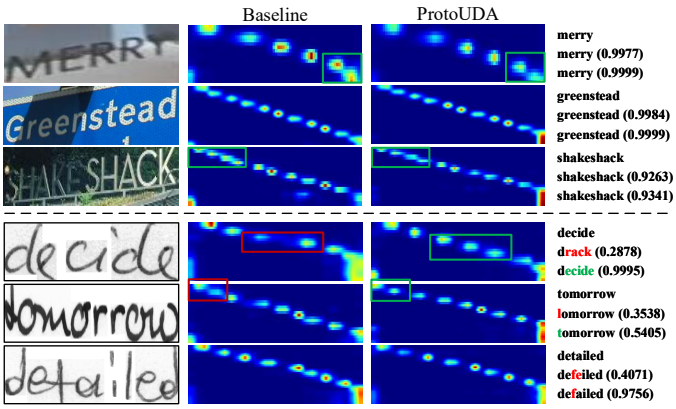


Fig. 5. Visualization of attention results and prediction results for Baseline and ProtoUDA. Above the dotted line are the correct cases, and below are the false ones. The prediction results are on the right side, and the numbers in parentheses indicate the confidence: the first row is the label, the second is the Baseline result, and the third is the ProtoUDA result. All green indicates true, and red indicates false.

different temperature hyperparameters on the instance-level alignment. As can be observed from Table 11 and Tabel 12, better results can be achieved when τ=1.

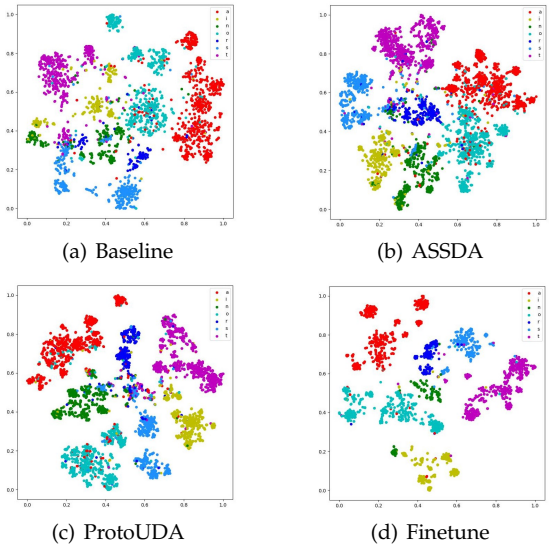
5.4.4 Threshold Parameter Sensitive Analysis

This section explores the sensitivity of hyperparameter η in Eq. 8. Parameter η is a threshold that determines which character features are involved in the alignment. Specifically, if the confidence is greater than η, the character feature will participate in class-level and instance-level alignment; conversely, it is ignored. Overall, if η is larger, only the character features with higher confidence are involved in adaptation; conversely, the number of characters used for adaptation is enlarged. To exclude the effect of entropy minimization, we conduct experiments on three variants (model-1, model-3, and model-4 in Table 5) on Syn→IAM task. Here, we explore the different η from {0, 0.1, 0.3, 0.5, 0.7, 0.9}. The experimental results are shown in Fig. 4. On the one hand, model-3 and model-4 outperform the *Baseline* regardless of the threshold η setting, even for the most stringent feature selection condition of η=0.9. On the other hand, the relatively best WER and CER are achieved when η=0.3. Therefore, we finally set the threshold η to 0.3.

5.5 Results Visualization

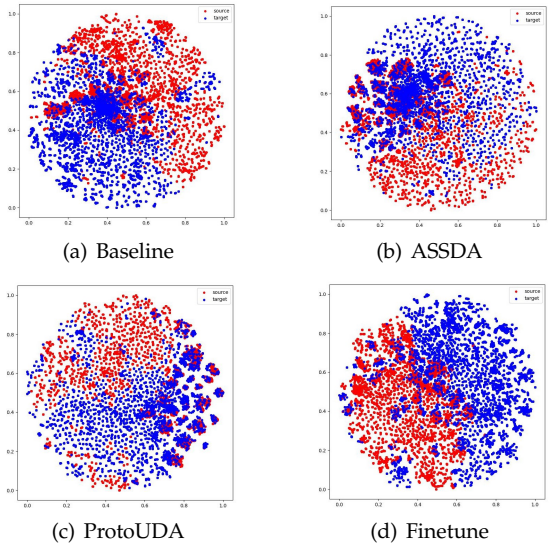
5.5.1 Visualization of Attention and Prediction Results

We randomly select several text images to visualize the attention and prediction results. During attention decoding, the decoding results after the stop symbol ['S'] are ignored. As can be observed from Fig. 5, on the one hand, our ProtoUDA can recognize the number of characters more accurately. For example, *Baseline* and ProtoUDA predict 'drack' and 'decide'. The former error prediction 'drack' is most intuitively reflected in the attention visualization as a missing hotspot. On the other hand, ProtoUDA can locate character features more precisely. Thanks to the ability to extract high-quality character features, ProtoUDA obtains more accurate attention results, thus a more precise location. For example, *Baseline* and ProtoUDA recognize 'lomorrow' and 'tomorrow'. The size of hotspots in the former attention visualization is slightly larger, along with a lower prediction probability (0.3538 vs. 0.5405).



(a) Baseline (b) ASSDA (c) ProtoUDA (d) Finetune

Fig. 6. Visualization of target character features on Syn→IAM task.



(a) Baseline (b) ASSDA (c) ProtoUDA (d) Finetune

Fig. 7. Visualization of source and target character features on Syn→IAM task.

5.5.2 Visualization of Feature Distribution

To demonstrate the effectiveness of class-level and instance-level alignment, we perform two visualizations using a t-SNE tool on the Syn→IAM task. The first is the visualization of character features. Specifically, a few classes are randomly selected to visualize the character features in the embedding space. As shown in Fig. 6, our ProtoUDA can better cluster similar character features compared to ASSDA [4] and *Baseline* model. This is attributed to intra-class aggregation based on contrastive learning, which draws similar characters closer together and pulls the different class characters as far away as possible. The second is the feature visualization of source and target domains. Concretely, the source and target domains are viewed as a class separately. As shown in Fig.

7, our ProtoUDA can blend the character features of the two domains to mitigate the domain drift problem.

6 Conclusion

This paper proposes a novel prototype-based domain adaptation method for cross-domain text recognition. Based on the implicit character features, we implement class-level and instance-level alignment. This way, the knowledge learned in the source data can be partly transferred to the target data, allowing the learning of fine-grained domain-invariant character features. Extensive cross-domain experiments demonstrate the superiority of ProtoUDA, achieving SOTA performance on benchmark datasets.

Acknowledgments

We are very grateful to the editors and reviewers for their professional comments and constructive suggestions on this paper, which are very helpful in improving it.

This work was supported in part by: (1) National Natural Science Foundation of China (Grant No. 62172256, No. 62202278, and No. 62202272); (2) Natural Science Foundation of Shandong Province (Grant No. ZR2019ZD06, No. ZR2020QF036, and No. ZR2021ZD15).

References

[1] Y. Wang, H. Xie, S. Fang, J. Wang, S. Zhu, and Y. Zhang, "From two to one: A new scene text recognizer with visual language modeling network," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021, pp. 14 174–14 183.

[2] X. Zhang, B. Zhu, X. Yao, Q. Sun, R. Li, and B. Yu, "Context-based contrastive learning for scene text recognition," in *Proc. AAAI Conf. Artif. Intell.*, 2022, pp. 3353–3361.

[3] Y. He, C. Chen, J. Zhang, J. Liu, F. He, C. Wang, and B. Du, "Visual semantics allow for textual reasoning better in scene text recognition," in *Proc. AAAI Conf. Artif. Intell.*, 2022, pp. 888–896.

[4] Y. Zhang, S. Nie, S. Liang, and W. Liu, "Robust text image recognition via adversarial sequence-to-sequence domain adaptation," *IEEE Trans. Image Process.*, vol. 30, pp. 3922–3933, 2021.

[5] Y. Zhang, S. Nie, W. Liu, X. Xu, D. Zhang, and H. T. Shen, "Sequence-to-sequence domain adaptation network for robust text image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 2740–2749.

[6] Y. Chang, Y. Chen, Y. Chang, and Y. Yeh, "Smile: Sequence-to-sequence domain adaptation with minimizing latent entropy for text image recognition," in *Proc. IEEE Int. Conf. Image Process.*, 2022, pp. 431–435.

[7] Z. Cheng, F. Bai, Y. Xu, G. Zheng, S. Pu, and S. Zhou, "Focusing attention: Towards accurate text recognition in natural images," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 5086–5094.

[8] Y. Yan, H. Wu, Y. Ye, C. Bi, M. Lu, D. Liu, Q. Wu, and M. K. Ng, "Transferable feature selection for unsupervised domain adaptation," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 11, pp. 5536–5551, 2022.

[9] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. J. Smola, "A kernel approach to comparing distributions," in *Proc. AAAI Conf. Artif. Intell.*, 2007, pp. 1637–1641.

[10] B. Sun and K. Saenko, "Deep CORAL: Correlation alignment for deep domain adaptation," in *Proc. Eur. Conf. Comput. Vis. Workshops (3)*, 2016, pp. 443–450.

[11] J. Zhuo, S. Wang, W. Zhang, and Q. Huang, "Deep unsupervised convolutional domain adaptation," in *Proc. ACM Multimedia Conf.*, 2017, pp. 261–269.

[12] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2962–2971.

[13] Y. Zou, Z. Y. an B. V. K. Vijaya Kumar, and J. Wang, "Unsupervised domain adaptation for semantic segmentation via class-balanced self-training," in *Proc. Eur. Conf. Comput. Vis. (3)*, vol. 11207, 2018, pp. 297–313.

[14] Y. Pan, T. Yao, Y. Li, Y. Wang, C. Ngo, and T. Mei, "Transferrable prototypical networks for unsupervised domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 2239–2247.

[15] M. Pilanci and E. Vural, "Domain adaptation on graphs by learning aligned graph bases," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 2, pp. 587–600, 2022.

[16] P. Wei, Y. Ke, and C. K. Goh, "A general domain specific feature transfer framework for hybrid domain adaptation," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 8, pp. 1440–1451, 2019.

[17] J. Li, M. Jing, H. Su, K. Lu, L. Zhu, and H. T. Shen, "Faster domain adaptation networks," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 12, pp. 5770–5783, 2022.

[18] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. S. Lempitsky, "Domain-adversarial training of neural networks," *J. Mach. Learn. Res.*, vol. 17, pp. 59:1–59:35, 2016.

[19] M. Liu and O. Tuzel, "Coupled generative adversarial networks," in *Proc. Neural Inf. Process. Syst.*, 2016, pp. 469–477.

[20] J. Huang, D. Guan, A. Xiao, S. Lu, and L. Shao, "Category contrast for unsupervised domain adaptation in visual tasks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 1193–1204.

[21] C. Luo, L. Jin, and J. Chen, "SimAN: Exploring self-supervised representation learning of scene text via similarity-aware normalization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 1029–1038.

[22] C. Liu, C. Yang, and X. Yin, "Open-set text recognition via character-context decoupling," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 4513–4522.

[23] Y. Liu, C. Shen, L. Jin, T. He, P. Chen, C. Liu, and H. Chen, "ABCNet v2: Adaptive bezier-curve network for real-time end-to-end text spotting," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 11, pp. 8048–8064, 2022.

[24] W. Hu, X. Cai, J. Hou, S. Yi, and Z. Lin, "GTC: Guided training of CTC towards efficient and accurate scene text recognition," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 11 005–11 012.

[25] B. Shi, X. Bai, and C. Yao, "An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 11, pp. 2298–2304, 2017.

[26] J. Wang and X. Hu, "Gated recurrent convolution neural network for OCR," in *Proc. Neural Inf. Process. Syst.*, 2017, pp. 335–344.

[27] D. Yu, X. Li, C. Zhang, T. Liu, J. Han, J. Liu, and E. Ding, "Towards accurate scene text recognition with semantic reasoning networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 12 110–12 119.

[28] M. Liao, J. Zhang, Z. Wan, F. Xie, J. Liang, P. Lyu, C. Yao, and X. Bai, "Scene text recognition from two-dimensional perspective," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 8714–8721.

[29] B. Shi, X. Wang, P. Lyu, C. Yao, and X. Bai, "Robust scene text recognition with automatic rectification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 4168–4176.

[30] A. K. Bhunia, A. Sain, A. Kumar, S. Ghose, P. N. Chowdhury, and Y. Song, "Joint visual semantic reasoning: Multi-stage decoder for text recognition," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021, pp. 14 920–14 929.

[31] Z. Qiao, Y. Zhou, J. Wei, W. Wang, Y. Zhang, N. Jiang, H. Wang, and W. Wang, "PIMNet: A parallel, iterative and mimicking network for scene text recognition," in *Proc. ACM Multimedia Conf.*, 2021, pp. 2046–2055.

[32] M. Yang, M. Liao, P. Lu, J. Wang, S. Zhu, H. Luo, Q. Tian, and X. Bai, "Reading and writing: Discriminative and generative modeling for self-supervised text recognition," in *ACM Multimedia*, 2022, pp. 4214–4223.

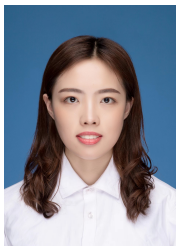
[33] A. Graves, S. Fernández, F. J. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proc. Int. Conf. Mach. Learn.*, vol. 148, 2006, pp. 369–376.

[34] P. He, W. Huang, Y. Qiao, C. C. Loy, and X. Tang, "Reading scene text in deep convolutional sequences," in *Proc. AAAI Conf. Artif. Intell.*, 2016, pp. 3501–3508.

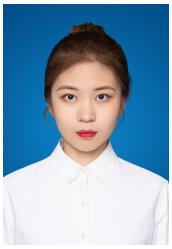
[35] J. Baek, G. Kim, J. Lee, S. Park, D. Han, S. Yun, S. J. Oh, and H. Lee, "What is wrong with scene text recognition model comparisons? dataset and model analysis," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 4714–4722.

[36] K. He, H. Fan, Y. Wu, S. Xie, and R. B. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 9726–9735.

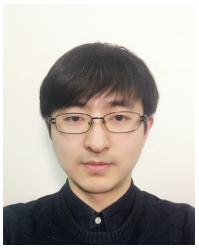
[37] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. E. Hinton, "Big self-supervised models are strong semi-supervised learners," in *Proc. Neural Inf. Process. Syst.*, 2020.
[38] T. Chen, S. Kornblith, M. Norouzi, and G. E. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. Int. Conf. on Mach. Learn.*, vol. 119, 2020, pp. 1597–1607.
[39] A. Aberdam, R. Litman, S. Tsiper, O. Anschel, R. Slossberg, S. Mazor, R. Manmatha, and P. Perona, "Sequence-to-sequence contrastive learning for text recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 15 302–15 312.
[40] S. Azadi, M. Fisher, V. G. Kim, Z. Wang, E. Shechtman, and T. Darrell, "Multi-content GAN for few-shot font style transfer," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7564–7573.
[41] A. K. Bhunia, S. Ghose, A. Kumar, P. N. Chowdhury, A. Sain, and Y. Song, "MetaHTR: Towards writer-adaptive handwritten text recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 15 830–15 839.
[42] L. Kang, M. Rusinol, A. Fornés, P. Riba, and M. Villegas, "Unsupervised writer adaptation for synthetic-to-real handwritten word recognition," in *Proc. IEEE Winter Conf. App. Comput. Vis.*, 2020, pp. 3502–3511.
[43] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *CoRR*, vol. abs/1807.03748, 2018.
[44] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman, "Synthetic data and artificial neural networks for natural scene text recognition," *CoRR*, vol. abs/1406.2227, 2014.
[45] A. Gupta, A. Vedaldi, and A. Zisserman, "Synthetic data for text localisation in natural images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2315–2324.
[46] A. Mishra, K. Alahari, and C. V. Jawahar, "Scene text recognition using higher order language priors," in *Proc. Brit. Mach. Vis. Conf.*, 2012, pp. 1–11.
[47] K. Wang, B. Babenko, and S. J. Belongie, "End-to-end scene text recognition," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2011, pp. 1457–1464.
[48] S. M. Lucas, A. Panaretos, L. Sosa, and A. T. et al., "ICDAR 2003 robust reading competitions: entries, results, and future directions," *Int. J. Document Anal. Recognit.*, vol. 7, no. 2-3, pp. 105–122, 2005.
[49] D. Karatzas, F. Shafait, S. Uchida, and M. I. et al., "ICDAR 2013 robust reading competition," in *Proc. Int. Conf. Document Anal. Recog.*, 2013, pp. 1484–1493.
[50] T. Q. Phan, P. Shivakumara, S. Tian, and C. L. Tan, "Recognizing text with perspective distortion in natural scenes," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2013, pp. 569–576.
[51] A. Risnumawan, P. Shivakumara, C. S. Chan, and C. L. Tan, "A robust arbitrary text detection system for natural scene images," *Expert Syst. Appl.*, vol. 41, no. 18, pp. 8027–8048, 2014.
[52] D. Karatzas, L. Gomez-Bigorda, and A. N. et al., "ICDAR 2015 competition on robust reading," in *Proc. Int. Conf. Document Anal. Recog.*, 2015, pp. 1156–1160.
[53] J. Sueiras, "Continuous offline handwriting recognition using deep learning models," *CoRR*, vol. abs/2112.13328, 2021.
[54] U. Marti and H. Bunke, "The IAM-database: an english sentence database for offline handwriting recognition," *Int. J. Document Anal. Recognit.*, vol. 5, no. 1, pp. 39–46, 2002.
[55] F. Kleber, S. Fiel, M. Diem, and R. Sablatnig, "CVL-Database: An off-line database for writer retrieval, writer identification and word spotting," in *Proc. Int. Conf. Document Anal. Recog.*, 2013, pp. 560–564.
[56] W. Liu, C. Chen, K. K. Wong, Z. Su, and J. Han, "STAR-Net: A spatial attention residue network for scene text recognition," in *Proc. Brit. Mach. Vis. Conf.*, 2016.
[57] J. Wang and X. Hu, "Gated recurrent convolution neural network for OCR," in *Proc. Neural Inf. Process. Syst.*, 2017, pp. 335–344.
[58] W. Liu, C. Chen, and K. K. Wong, "Char-Net: A character-aware neural network for distorted scene text recognition," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 7154–7161.
[59] C. Zheng, H. Li, S. Rhee, S. Han, J. Han, and P. Wang, "Pushing the performance limit of scene text recognizer without human annotation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 14 096–14 105.
[60] A. K. Bhunia, A. Sain, P. N. Chowdhury, and Y. Song, "Text is text, no matter what: Unifying text recognition using knowledge distillation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021, pp. 963–972.
[61] H. Yan, Z. Li, Q. Wang, P. Li, Y. Xu, and W. Zuo, "Weighted and class-specific maximum mean discrepancy for unsupervised domain adaptation," *IEEE Trans. Multim.*, vol. 22, no. 9, pp. 2420–2433, 2020.



Xiao-Qian Liu received the M.S. degree in control engineering in 2020 from Shandong University, China. She is pursuing a Ph.D. in artificial intelligence at the School of Software, Shandong University, Jinan, China. Her research interests include deep learning, computer vision, domain adaptation, and OCR.



Xue-Ying Ding received the M.S. degree in software engineering in 2023 from Shandong University, China. She is currently an engineer at the Nanjing Research Institute of Huawei. Her research interests include network communications, machine learning, computer vision, and scene text recognition.



Xin Luo received the Ph.D. degree in computer science from Shandong University, Jinan, China, in 2019. He is currently an assistant professor with the School of Software, Shandong University, Jinan, China. His research interests mainly include machine learning, multimedia retrieval, and computer vision. He has published over 20 papers on TIP, TKDE, ACM MM, SIGIR, WWW, IJCAI, et al. He serves as a Reviewer for ACM International Conference on Multimedia, International Joint Conference on Artificial Intelligence, AAAI Conference on Artificial Intelligence, the IEEE Transactions on Cybernetics, Pattern Recognition, and other prestigious conferences and journals.

Xin-Shun Xu is currently a professor with the School of Software, Shandong University. He received his M.S. and Ph.D. degrees in computer science from Shandong University, China, in 2002, and Toyama University, Japan, in 2005, respectively. He joined the School of Computer Science and Technology at Shandong University as an associate professor in 2005, and joined the LAMDA group of Nanjing University, China, as a postdoctoral fellow in 2009. From 2010 to 2017, he was a professor at the School of Computer Science and Technology, Shandong University. He is the founder and the leader of MIMA (Machine Intelligence and Media Analysis) group of Shandong University. His research interests include machine learning, information retrieval, data mining, and image/video analysis and retrieval. He has published in TIP, TKDE, TMM, TCSVT, AAAI, CIKM, IJCAI, MM, SIGIR, WWW, and other venues. He also serves as an SPC/PC member or a reviewer for various international conferences and journals, *e.g.*, AAAI, CIKM, CVPR, ICCV, IJCAI, MM, TCSVT, TIP, TKDE, TMM, and TPAMI.