Pattern Recognition 107 (2020) 107488

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

Zero-shot Handwritten Chinese Character Recognition with hierarchical decomposition embedding

Zhong Cao, Jiang Lu, Sen Cui, Changshui Zhang*

Institute for Artificial Intelligence, Tsinghua University (THUAI); State Key Lab of Intelligent Technologies and Systems; Beijing National Research Center for Information Science and Technology (BNRist); Department of Automation, Tsinghua University, Beijing, P.R. China

ARTICLE INFO

Article history:
Received 5 November 2019
Revised 21 May 2020
Accepted 3 June 2020
Available online 6 June 2020

Keywords:
Chinese character recognition
Radical analysis
Zero-shot learning
Label embedding

ABSTRACT

Handwritten Chinese Character Recognition (HCCR) is a challenging topic in the field of pattern recognition due to large-scale character vocabulary, complex hierarchical structure, various writing styles, and scarce training samples. In this paper, we explored the hierarchical knowledge of Chinese characters and presented a novel zero-shot HCCR method. First, we handled the relations between the characters and their primitives, such as radicals and structures, to obtain a tree layout of primitives. Then, we presented a novel zero-shot hierarchical decomposition embedding method to encode the tree layout into a semantic vector. Next, we devised a Convolutional Neural Network (CNN) based framework to learn both radicals and structures of characters via the semantic vector. As different Chinese characters share some common radicals and structures, our method is able to recognize new categories without any labeled samples from them. Moreover, our method is effective in both traditional HCCR and zero-shot HCCR tasks. It achieves competitive performance on the traditional experiment setting and significantly surpasses the state-of-the-art methods on the zero-shot experiment setting.

1. Introduction

HCCR has been studied for decades. However, it remains challenging due to large-scale Chinese character vocabulary,[1] complex structure, various writing styles, and scarce training samples of uncommon characters, etc. Additionally, collecting and annotating the huge amounts of handwritten training samples for each character is cumbersome, expensive, and even impracticable for some uncommon Chinese characters. Consequently, exploiting a learning system capable of recognizing handwritten Chinese characters based on scarce training samples is a challenging task with practical significance. Fortunately, the basic primitives (radicals and spatial structures) that compose all Chinese characters are predefined and limited in quantity. It provides a promising route for transfer learning from the seen Chinese characters to the unseen Chinese characters for which very few, if any, training samples are available.

Recently, some attempts and efforts have been devoted to the zero-/few-shot HCCR, such as DenseRAN [1–3] and FewshotRAN [4]. Concretely, DenseRAN [1,3] defined some artificial rules for Chinese character description and performed zero-shot recognition via an encoder-decoder architecture. FewshotRAN [4] coped with unseen Chinese characters given few support samples by modeling each character as a sequence of radicals with prototypes. Nevertheless, there are three noticeable drawbacks or constraints in the above methods. (1) It is complicated and costly to design specific radical-level description rules for each character. (2) The test scenarios are limited to the requirement that the test set's radicals must appear in the training set. (3) The poor performances are far behind the expectation of humans under the zero-shot setting.

In this paper, our goal is to develop an efficient handwritten Chinese character learning system as shown in Fig. 1 based on the knowledge of the hierarchical decomposition of Chinese characters, which can circumvent the drawbacks above and be compatible with all unseen characters without intentional constraints or human rules. To achieve this goal, we propose a novel *Hierarchical Decomposition Embedding* (HDE) method to represent the Chinese character in an interpretable vector space. Specifically, HDE is built by a tree-based decomposition of Chinese characters. For each Chinese character, the corresponding HDE provides a strong basis for the representation of its primitives, including the radicals and structures. HDE narrows the gaps between training and test

* Correspoding author.
E-mail addresses: caozhong14@mails.tsinghua.edu.cn (Z. Cao), lu-j13@mails.tsinghua.edu.cn (J. Lu), cuis19@mails.tsinghua.edu.cn (S. Cui), zcs@mail.tsinghua.edu.cn (C. Zhang).

[1] The vocabulary of Chinese character in UNICODE 12.0 contains more than 80,000 different characters.

https://doi.org/10.1016/j.patcog.2020.107488
0031-3203/

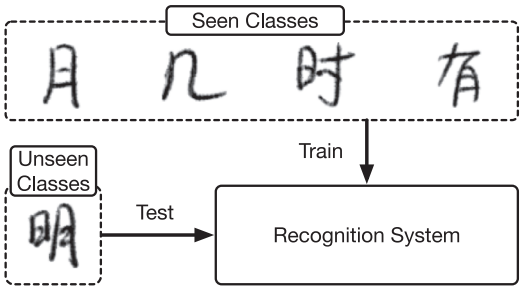

Fig. 1. Zero-shot HCCR task.

domains for zero-shot recognition. Then, we propose a zero-shot HCCR framework to learn the compatibility or similarity between the input handwritten character image and the knowledge-based HDE representation. And compatibility or similarity is also the criterion for classifying the unseen characters. Experiments show that our HDE and our HCCR framework have greatly improved the classification accuracy of unseen Chinese character categories. Our method is also applicable to the seen Chinese character categories and outperforms other radical-based methods on the ICDAR2013 competition database.

Our main contributions are summarized as follows:

1. We propose a novel HDE method to transform the hierarchical structural knowledge into a vector representation, which has the advantages of learning shared structural information between seen categories and unseen categories.
2. We introduce a zero-shot HCCR framework, HDE-Net. It regresses the primitives of Chinese characters, such as radicals and structures, and then classifies zero-shot characters via trainable compatibility between samples and labels.
3. We adopt loss functions REL, MBL, MCEL, and MCPL respectively to learn the similarity or compatibility between samples and labels, then compare their convergence, robustness, and effectiveness, which provides a reference for the follow-up researches on zero-shot learning.
4. Experiments show that our method outperforms state-of-the-art methods under the zero-shot setting while maintaining competitive performance under the traditional HCCR setting.

The rest of this paper is organized as follows: Section 2 introduces the related works. Section 3 introduces the hierarchical decomposition of Chinese characters. Section 4 describes how to encode the hierarchical decomposition knowledge to vector representations via HDE. Section 5 describes the details of our HCCR framework with vector representations. Section 6 presents the experimental results, and the paper is concluded in Section 7.

2. Related works

Radical-based Chinese character recognition Chinese character recognition based on radical knowledge is still a challenging problem due to the difficulty in detecting and identifying radicals. In the last few decades, many researchers have tried to solve this challenging problem. The existing solutions are roughly divided into three types: skeleton matching, stroke matching, and radical matching. The skeleton matching methods [2,5] firstly converted the character image into a skeleton image that can be utilized to find radicals, their spatial relations, and their character variations. The stroke matching methods [6,7] decomposed the radical into some primary strokes, thereby identifying the radical information from strokes and further analyzing the category of a character. The methods of stroke matching had performed well on the standard printed characters but badly on handwritten characters. The radical matching methods [3,4,8] judged the radical content at a specific position of a character and inferred the category of Chinese characters in combination with the knowledge of radicals. The radicals are composed in a variety of ways, which promotes the establishment of Chinese characters. Our method performs character matching on an interpretable vector space by representing each character as an HDE with each dimension representing an underlying primitive.

Fig. 2. Radical example. The radical, *tree* (in black color), is composed of four strokes (in red color). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Zero-shot learning In zero-shot learning, seen classes refer to the classes covered by the labeled training instances, while unseen classes refer to the other classes covered by unlabeled testing instances. The seen classes and unseen classes are disjoint. Given labeled training instances belonging to the seen classes, zero-shot learning aims to learn a classifier that can classify testing instances belonging to the unseen classes. From the definition, the general problem of zero-shot learning [9] is to apply the knowledge contained in the training sets to the task of classifying testing instances. Zero-shot learning is a sub-field of transfer learning [10]. Attribute-based methods [11,12] regarded each unseen class as an indicator vector with each dimension representing whether the category contains such an attribute. Embedding-based methods [13,14] encoded both images and labels into a shared space and then learned a compatibility function to expand to unseen classes. Some works [15,16] aimed to obtain labeled instances for unseen classes, then learned the zero-shot classifier with these generated instances. Our paper proposes the HDE for each unseen class. It has more structural information than attribute-based methods and is more explanatory than the existing embedding-based methods.

3. Hierarchical decomposition of Chinese characters

3.1. Primitives of Chinese characters

In our research, the term primitive includes both structures and generalized radicals (combinations of strokes that are repeated as a part of a Chinese character). The primitives are the minimum units of the decomposition list in the Ideographic Description Sequences (IDS) according to UNICODE standard 9.0. The hierarchical composition of Chinese characters is characterized by:

a) Characters are composed of one or more radicals.
b) Radicals appear in the specific locations that are unique to the character (e.g., the left, upper, upper-left, or full-surrounding areas of the character).
c) Radicals are composed of strokes which are the smallest inseparable unit of Chinese characters.

Here are examples of the strokes, the radical, and the character. As shown in Fig. 3, the Chinese character, *forest*, is composed of three same radicals, *tree*, which are composed of four basic strokes as shown in Fig. 2.

Fig. 3. Character example. The character, *forest* (in black color), is composed of three same radicals, *tree* (in red color). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

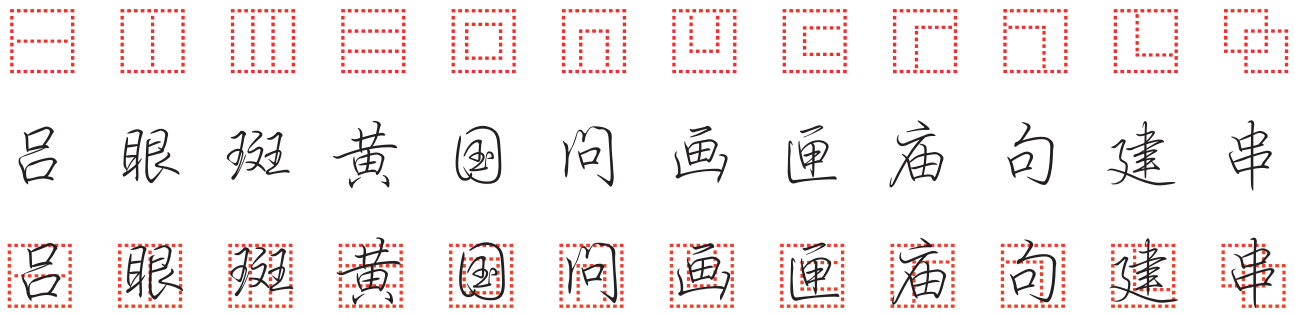

Fig. 4. Twelve Chinese character structures and corresponding character examples. Structures are in row 1, and corresponding Chinese character examples are in row 2. Characters can be decomposed to sub-characters as shown in row 3.

Unlike the horizontal arrangement of alphabets in western languages, Chinese characters have internal structures. All radicals of a character are placed in a square shape in a variety of ways. The arrangement styles, called structures, are defined as twelve categories in UNICODE as shown in Fig. 4. These structures are crucial in Chinese character, and they are the basis of decomposing characters into sub-characters.

3.2. Hierarchical layout of primitives

The hierarchical knowledge of Chinese characters can be conveniently expressed in the form of a tree as shown in Fig. 5. We process the existing IDS data, which contains Chinese character decomposing information in the form of an adjacency list, to a hierarchical decomposing layout. We set the structure of the character as a parent node and set the decomposed parts of the character as the child nodes. The child node will be a leaf node if it cannot be decomposed. Otherwise, we will search for IDS data to decompose the child node further. In this way, all characters can be represented as a tree with the nodes representing the basic primitives.

One key insight in this paper is that the relation between classes is obtained through the zero-shot (without image instances belonging to unseen classes) knowledge in the form of a hierarchical decomposing tree as shown in Fig. 5. A small set of nodes can be arranged to obtain almost all hierarchical decomposing trees of Chinese characters in the way. So we can classify the input image if given the knowledge of relation between candidate classes and seen classes. While most of researchers [4] generally defined a higher level radical set (such as '木,只' in Fig. 5), there are much more characters that could not be represented by their radicals because the radicals are not decomposed to basic primitives (such as '木,口,八', and so on). One might argue that other researchers could set the same basic radical set to represent as many characters as possible. In this case, the representation of each character – a sequence obtained by arranging the tree nodes in a certain order – will be too long, resulting in greatly reduced accuracy.

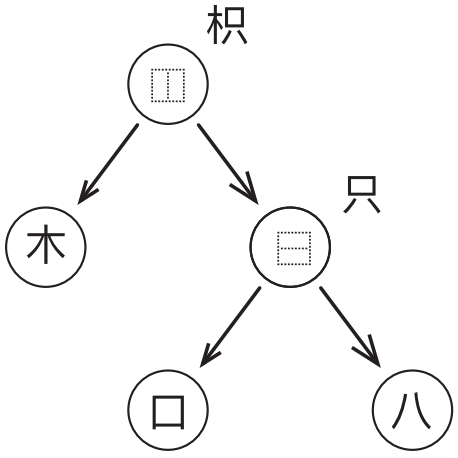

Fig. 5. Hierarchical decomposing of the Chinese character. The tree represents the decomposition of the character. The character is composed of five primitives as shown in the nodes.

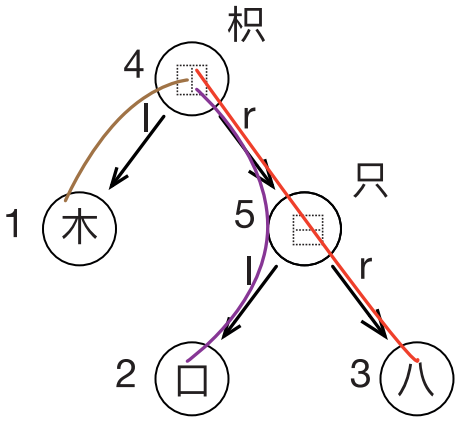

Fig. 6. Tree layout of the Chinese character. Each node has its node-path from root to itself, representing the layout of the node, e.g., the brown line shows that the left part of the character '枳' (orange) is the radical '木' (node 1). The leaf nodes 1,2,3 represent radicals and the parent nodes 4,5 represent structures. Best viewed in color.

4. Hierarchical decomposition embedding

We propose the HDE for each Chinese character in an interpretable vector space according to the corresponding tree layout which contains hierarchical primitive nodes as in Fig. 6.

4.1. Embedding method

We firstly adopt one-hot encoding y_{n_i} for each node n_i, then combine all vectors y_{n_i} in one tree to get the HDE $\phi(y)$. Specifically, we propose the concept of node-path, the shortest path from the root to the node as colorful lines in Fig. 6, to express the relation between the node and the tree. Thus we could take the pairs node/node-path to represent the decomposing tree. Fig. 7 is an example of node-path corresponding to the rightmost red line in Fig. 6.

Fig. 7. Node-path example in the form of structure-part sequences. The Nodes and parts along the node-path delimited by a backslash(\) represent hierarchical structure "the right-bottom part of the character". The impact value v for the node-path is $\alpha^2 - 2\beta_0(\alpha + \alpha^2)$.

Table 1
Provided that $\alpha = 0.5, \beta_0 = 0.001, \lambda = 0.5$, HDEs of the characters, tree, forest, stay, apricot, and orange respectively.

Dimensions [primitives]	64 [radical 八]	87 [radical 口]	141 [radical 木]	260 [left-right]	261 [up-down]	others
木 (tree)	0	0	1	0	0	0
森 (forest)	0	0	0.99675	0.2495	0.5	0
呆 (stay)	0	0.4995	0.499	0	0.5	0
杏 (apricot)	0	0.499	0.4995	0	0.5	0
枳 (orange)	0.2485	0.24875	0.4995	0.5	0.2495	0

The length l of node-path is defined as the number of nodes along this node-path. The l of the example is two, and the l would be zero if the node is the root node of the tree. The relation represented by node-path can be calculated as an impact value with the following two intuitive assumptions: The smaller the length l is, the higher the impact value will be. Different child nodes of the same parent node should have slightly different impact values. Based on the assumptions, we use $\alpha \in (0, A)$ as the attenuation coefficient of impacts when the length increases, and use $\beta \in (-B, +B)$ to express differences between child node impacts of the same parent node.

Definition 1. Given a node-path with length l, the nodes and parts along the node-path are $p_1, p_2, \ldots, p_l$, we define the impact value v for the node-path as:

$$v = \alpha^l + \sum_{i=1}^{l} (\alpha^i \beta_{p_i}). \tag{1}$$

There are twelve kinds of nodes along node-path as in Fig. 4, two of which have three child nodes, and ten of which have two child nodes, thus the set of structure-part p_i has 26 elements. For simplicity in the experiments, we set $\beta = -\beta_0$ for left child nodes, $\beta = -2\beta_0$ for the second left child nodes, and $\beta = -3\beta_0$ for the third child nodes. From the Eq. (1), the impact value of the root node is 1 since the length of the root path is 0.

Definition 2. Given all pairs node/ node-path in the tree, we divide all nodes into two sets. One is leaf node n_i set $\mathcal{R}$ representing radicals, the other is the parent node n_j set $\mathcal{S}$ representing structures. We define the HDE as follows:

$$\phi(y) = \sum_{n_i \in \mathcal{R}} v_{n_i} y_{n_i} + \lambda \sum_{n_j \in \mathcal{S}} v_{n_j} y_{n_j}, \tag{2}$$

where λ balances the weight of leaf nodes and parent nodes.

Specifically, each HDE is a vector of the same dimensions as the number of node types from decomposing trees in the training phase, e.g., 272-dim vector in experiments 6.5. The first 260 dimensions represent different radicals, and the last 12 dimensions represent distinct structures. The value of each dimension is inversely related to the depth of corresponding primitive nodes in the decomposing tree. Provided that $\alpha = 0.5, \beta_0 = 0.001, \lambda = 0.5$, the embeddings of some characters are shown in Table 1. Different decomposing trees of characters correspond to different $\phi(y)$ if taking appropriate parameters. Since IDS data is not always effective, characters like '土,士' among all GB2312 characters share the identical decomposing tree. In this case, we take these two characters as different leaf nodes to get one-hot encoding rather than decomposing character to obtain HDE vectors. To sum up, the HDE method is a one-to-one mapping from the decomposing tree to an interpretable vector representation under our experiment settings, preserving the original hierarchical decomposition information and assigning each primitive to each dimension.

4.2. Differences from other label embedding methods

The attribute embedding[11,12] in zero-shot learning would lose hierarchical structure information if directly used in our HCCR task. The multi-label embedding [1] either limited the test scenarios or discarded the structures of Chinese characters. These methods worked when all pairs of the leaf nodes (radicals) and node-paths (radical-at-locations) corresponding to the test set appeared in the training set. Our HDE method takes node-path as the relation between the node and the tree, then encodes the radicals, the structures, and the relations separately to make full use of the hierarchical decomposing of characters. It works when the nodes rather than the pairs in the test set are consistent with the training set. Thus, HDE greatly reduces the complexity of embedding because the nodes are much less than the pairs node/node-path. Moreover, our method can still achieve quite good results even when a few kinds of nodes in our test set do not appear in the training set.

Some works [17,18] also proposed the so-called hierarchical embedding to deal with zero-shot learning problems, which explicitly used the expert knowledge to group the categories into a hierarchy. Hierarchical knowledge on the categories requires an ordering operation $\prec$ in $\mathcal{Y}$: $z \prec y$ means that z is an ancestor of y in the tree, and we can define $\xi_{y,z} = 1$ if $z \prec y$ or $z = y$. The hierarchy embedding $\varphi^{\mathcal{H}}(y)$ can be defined as the C dimensional vector, as showed in Fig. 8.

$$\varphi^{\mathcal{H}}(y) = \left[\xi_{y,1}, \ldots, \xi_{y,C}\right]. \tag{3}$$

The nodes represent the primitives (generalized radicals and structures) in our method, while the nodes represent classes themselves in other methods [17,18]. The compositions of primitives are so diverse that the relation between most categories cannot be described as the operation $\prec$ introduced in the previous methods. In contrast, it is appropriate to represent each character with the primitives since the number of primitives is much smaller. Besides, our method encodes the pairs node/node-path, which is consistent with the hierarchical structural knowledge.

5. Character recognition with HDE

5.1. Overview

The general image classification task is to annotate a given image with one label. The goal is to learn a function $f : \mathcal{X} \to \mathcal{Y}$, which maps the input x in the image space $\mathcal{X}$ to the output y in the category space $\mathcal{Y}$. In this work, we focus on the case where we do not have any samples for some classes, but classes are related. Our method works in the following four application scenarios. First, when the number of test classes is extensive, collecting sufficient labeled instances for such a large number of classes is challenging. Existing datasets such as CASIA [19,20] (3.9 million

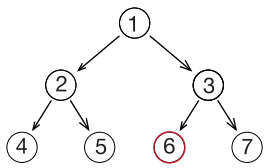

Fig. 8. Hierarchical label embedding method. In this example, given seven classes, class 6 has the node-path 1-3-6, so the class 6 can be encoded in a binary 7-dimensional space as [1, 0, 1, 0, 0, 1, 0].

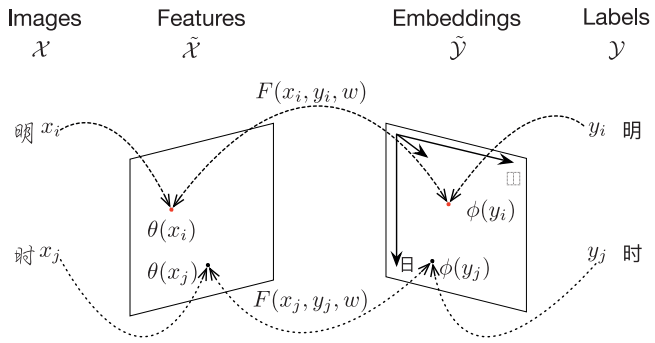

Fig. 9. Character recognition method with HDE. Most of the works mainly focus on how to extract effective features of the image. Our work mainly focuses on how to map the character categories into an embedding space, and learn the compatibility function F with the features $\theta(x)$ and the HDE $\phi(y)$. The two characters share the radical '日' and the left-right structure in embedding space, which enables us to recognition unseen classes by the radicals and the structures.

samples of 7356 categories) can only cover a small subset of these classes. Second, when test classes are rare due to infrequently used Chinese characters, we could not find all corresponding labeled instances. It costs a lot to make sufficient new instances. Third, when test classes change over time because linguists, sociologists, and government agencies keep Chinese characters up to date, other methods need to retrain to fit the new classes. Fourth, when in some particular tasks, it is expensive to obtain labeled instances, such as collecting and labeling as many characters as possible in the wild such as the CTW dataset [21].

We introduce a compatibility function $F : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ and define classification function f as following:

$$f(x; w) = \arg\max_{y \in \mathcal{Y}} F(x, y; w). \tag{4}$$

Suppose that we had gotten HDE $\phi(y)$ by function $\phi : \mathcal{Y} \to \tilde{\mathcal{Y}} = \mathbb{R}^E$ introduced in Section 4. The function ϕ has parameters α, β, λ that can be changed manually. Suppose that we could also get the features $\theta(x)$ using CNN as feature extractor $\theta : \mathcal{X} \to \tilde{\mathcal{X}} = \mathbb{R}^D$. We adopt multilayer perceptron g as a transfer from the image feature space $\tilde{\mathcal{X}}$ to label embedding space $\tilde{\mathcal{Y}}$. The compatibility function F could be written as similarity (τ is a learnable scale parameter):

$$F(x, y; w) = -\tau \cdot \mathcal{D}(g(\theta(x)), \phi(y)), \tag{5}$$

where $\mathcal{D}$ is distance defined as:

$$\mathcal{D}(x_1, x_2) = \begin{cases} \|x_1 - x_2\|_2, & \text{Euclidean distance,} \\ 1 - \frac{x_1^\top x_2}{\|x_1\| \times \|x_2\|}, & \text{Cosine distance.} \end{cases} \tag{6}$$

The compatibility function F can also be achieved by other methods. These methods could transfer the label embedding spaces $\tilde{\mathcal{Y}}$ to the image feature space $\tilde{\mathcal{X}}$ or transfer these two spaces to a shared space to calculate the distance or adopt a metric network to concatenate the pair of features together as input and directly output the compatibility. We propose the compatibility function as Eq. (5) because our HDE space is meaningful in the hierarchical decomposing of characters. It is convenient to analyze the embedding vectors in the space since each dimension represents each type of primitives. And it is flexible to change the network g to get an appropriate transfer from the feature space to the label embedding space. A graphic description of our framework can be seen in Fig. 9.

5.2. Feature extractor and transfer modules

ResNet [22] had been proved to be a useful feature extractor in visual tasks [23,24]. In this paper, we use four Resnet blocks with channels 64, 128, 256, and 512 sequentially as the feature extractor, denoted as θ. We also compare the feature extractor of papers [4,8] in ablation study.

Then we map the features $\theta(x) \in \mathbb{R}^D$ into the label embedding space with feature transfer module. Specifically, after obtaining the final convolutional layer output of the feature extractor and flattening it to a D-dimensional vector, we use two fully connected layers to get the E-dimensional vectors. So we could measure the compatibility between image x and label y in the embedding space $\mathbb{R}^E$.

5.3. Model inference

Given an input image x and the candidate category set $\mathcal{Y}$, we first get the features $\theta(x)$ by the CNN feature extractor and transfer the features into the label embedding space. Then we compare the similarity between the transformed feature corresponding to the input and any possible HDE $\phi(y)$, finally classify the input to the most compatible label as shown in Eq. (4).

To apply HDE to the scenario where test categories change over time, the basic primitives corresponding to every dimension are all from decomposing trees of training characters. Some nodes n_i (especially radicals) composed of test characters may never appear in the training data, and the encoding of these nodes y_{n_i} is manually set to be zero vectors. HDE-Net makes inferences based on the nodes that have appeared in the training data. For example, we could infer the character '晴' (sunny) by the radicals '日' (sun), '月' (moon) and the structures, left-right-structure of the whole character and up-down-structure of the right part. It still works when the remaining radical in the character has never been seen in the training phase. HDE-Net could recognize new characters without any samples or knowledge of these classes in the training phase if provided decomposing knowledge of new characters during the testing phase. Compared to most zero-shot learning methods [4,8] with the assumption that the knowledge of test categories is already given during the training phase, HDE-Net has more extensive application scenarios.

5.4. Model training

The trainable parameters in our framework are composed of three parts from the CNN feature extractor module θ, the feature transfer module g, and our compatibility function. The parameter τ controls the scale of compatibility within a range, simplifying the design of loss functions and the selection of hyperparameters, such as margin. These parameters are trained jointly in an end-to-end manner for improving the performance of classification. In the following paragraphs, we introduce four different loss functions designed for training.

5.4.1. Ranking error loss function (REL)

Inspired by Weston et al. [25], we adopt ranking error loss as follows:

$$\text{loss}(x, y; w) = \sum_{\bar{y} \in \mathcal{Y} \setminus \{y\}} \frac{L\left(\text{rank}_y^1(x, y)\right)}{\text{rank}_y^1(x, y)} |1 - F(x, y; w) + F(x, \bar{y}; w)|_+, \tag{7}$$

where

$$|u|_+ = \begin{cases} u, & u > 0 \\ 0, & u \le 0 \end{cases} \tag{8}$$

and

$$\text{rank}_y^1(x, y; w) = \sum_{\bar{y} \in \mathcal{Y} \setminus \{y\}} I(1 + F(x, \bar{y}; w) > F(x, y; w)), \tag{9}$$

is an margin-penalized rank of label y for image x. Loss would be set as zero if $\text{rank}_y^1 = 0$. I is an indicator function satisfying that $I(u) = 1$ if u is true and 0 otherwise. The ranking weight L is,

$$L(k) = \sum_{j=1}^{k} \sigma_j, \text{ with } \sigma_1 \geq \sigma_2 \geq \cdots \geq 0. \tag{10}$$

Hence, a decreasing sequence $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_C \geq 0$ implies the mistake on the top list bring higher loss than at the bottom list. We follow the original suggestions and choose $\sigma_k = 1/k$.

We adopt the margin-penalized rank because the Eq. (7) can be optimized effectively. If removed the margin 1 in Eqs. (7) and (9), the training with this loss would be unstable. The goal of REL is to make the compatibility of the positive pair (x, y) larger than the negative pairs $(x, \bar{y})$, especially those with the higher rank.

5.4.2. Margin-based loss function (MBL)

Sometimes it is hard to sort the negative categories. We are only concerned about the top one of the rank list. The top one should be a positive category. There may be no difference among rank 2, rank 3, or rank last. If we set all weights σ with the same value as $\sigma_k = 1$, then rank 2 or rank last would have the same weight in the loss function. Then, the special ranking error loss with $\sigma_k = 1$ will be as follows:

$$\text{loss}(x, y; w) = \sum_{\bar{y} \in \mathcal{Y} \setminus \{y\}} |1 - F(x, y; w) + F(x, \bar{y}; w)|_+. \tag{11}$$

It is similar to the idea of the margin-based loss function, which is to make the compatibility of pairs (x, y) with the right labels exceed that with the wrong labels by a margin. The idea of MBL is as follows.

If the sample is classified correctly, then we have $F(x, y; w) > F(x, \bar{y}; w)$ and we think the loss should be 0. If the sample is misclassified, then we have $F(x, y; w) < F(x, \bar{y}; w)$. Considering the above two situations, the loss with positive label y and negative label $\bar{y}$ should be naturally defined as:

$$l(x, y; w) = |F(x, \bar{y}; w) - F(x, y; w)|_+. \tag{12}$$

To improve the ability to classify, a margin m is added to the loss function, which is defined as:

$$l_m(x, y; w) = |F(x, \bar{y}; w) - F(x, y; w) + m|_+. \tag{13}$$

It penalizes the framework when it classifies correctly with a narrow margin. Thus, it can increase the robustness of the classification and improve the accuracy of the test set.

To apply margin-based loss successfully, we have to consider the selection of the margin m and the sample triples $(x, y, \bar{y})$ carefully. Benefit from our parameter τ, we could directly set the margin as 1. As for the sample triples, we could randomly select all possible negative labels with one image just as Eq. (11), or just select the most competitive negative label r. We suggest the latter selection strategy for efficiency when training our model,

$$r = \arg \max_{\bar{y} \in \mathcal{Y} \setminus \{y\}} F(x, \bar{y}; w). \tag{14}$$

Thus, our MBL is as follows:

$$\text{loss}(x, y; w) = |1 - F(x, y; w) + F(x, r; w)|_+. \tag{15}$$

5.4.3. Minimum classification error loss function (MCEL)

There are many ways to define a metric of misclassification error that is continuous with respect to the framework parameters. One reasonable function [26] is as follows:

$$l(x, y; w) = -F(x, y; w) + \left[\frac{1}{C-1} \sum_{\bar{y} \in \mathcal{Y} \setminus \{y\}} F(x, \bar{y}; w)^{\eta} \right]^{1/\eta}. \tag{16}$$

The measure in Eq. (16) is continuous and offers an amount of flexibility in the value of η taking all the potential categories into consideration. When η is 1, loss with all negative labels is equally weighted. When η approaches infinity, loss of the most competitive negative label r defined as in Eq. (14) is far larger than other negative labels. Thus the measure is:

$$l(x, y; w) = -F(x, y; w) + F(x, r; w). \tag{17}$$

To complete the loss function, the above measure is used in the logistic function or an exponential function. Both functions are smoothed zero-one differentiable functions.

a) Logistic function:

$$\ell = \frac{1}{1 + \exp(-\kappa_1 (l(x, y; w) + \kappa_2))}, \quad \kappa_1 > 0, \tag{18}$$

b) Exponential function:

$$\ell = \begin{cases} (l(x, y; w))^{\zeta}, & l(x, y; w) > 0 \\ 0, & l(x, y; w) \leq 0 \end{cases}, \quad \zeta > 0 \text{ and } \zeta \to 0. \tag{19}$$

From the definition of ℓ, we find the misclassification error $l(x, y; w)$ is minimized by minimizing ℓ. It means that the relative compatibility of true positive pairs (x, y) to the most competitive negative pairs (x, r) is increasing during loss decreasing. We finally define the MCEL by Eq. (18) logistic function with $\kappa_1 = 1, \kappa_2 = 0$ as:

$$\text{loss}(x, y; w) = \frac{1}{1 + \exp(F(x, y; w) - F(x, r; w))}. \tag{20}$$

The loss function MCEL pulls the sample closer to the correct label but pushes it away from the other categories. It might cause vanishing gradients and poor learning for deep networks that $\partial \ell / \partial l = \ell(1 - \ell)$.

5.4.4. Maximum classification probability loss function (MCPL)

There are some works [27–29] producing a distribution over classes based on the softmax over distances to the prototypes in the embedding space. The probability satisfies the non-negative and sum-to-one properties. The probability of a sample x belonging to the category k can be measured by Eq. (21).

$$p_w(y = k|x) = \frac{\exp(F(x, k, w))}{\sum_{k' \in \mathcal{Y}} \exp(F(x, k', w)}. \tag{21}$$

If replacing the compatibility with the distance in Eq. (5) and we will get the following probability function:

$$p_w(y = k|x) = \frac{\exp(-\tau \cdot \mathcal{D}(g(\theta(x)), \phi(y)))}{\sum_{k' \in \mathcal{Y}} \exp(-\tau \cdot \mathcal{D}(g(\theta(x)), \phi(y)))}. \tag{22}$$

Based on the probability function, we can define the cross entropy loss by negative log-probability $\ell = -\log(p_w(y = k|x))$, thus our MCPL becomes:

$$\text{loss}(x, y; w) = \log \sum_{k' \in \mathcal{Y}} \exp(F(x, k'; w) - F(x, y; w)). \tag{23}$$

From the Eq. (23), we can observe that the goal is to maximize the compatibility of the positive pairs and minimize the compatibility of all the negative pairs. We randomly select the positive pair (x, y) and produce negative pairs with that sample x and all negative labels as training data. Our loss function is a bit different from the primary loss function in the prototype network. The previous loss function is based on the distance between the various input samples (called support samples and query samples) in the same feature space. Ours is based on the gap between the input samples and the label embeddings. We introduce the learnable scale parameter τ to control the hardness of the probability assignment as in the Eq. (22), and render the loss function more applicable to the different scales of distances.

We propose the losses mentioned above, i.e., REL, MBL, MCEL, and MCPL based on the compatibility function $F(x, y; w)$, which measures the similarity between the input feature and the label embedding. REL is based on the ranking of compatibility. The loss is weighted on different ranks. MBL introduces the margin so that the input with small changes will not be able to flip the predicted label. MCEL is a measure of misclassification error with a smoothing transformation function. MCPL calculates the probability of different categories with the purpose that the positive pairs should have a higher likelihood than the negative ones.

In terms of efficiency, REL and MCPL converge faster than MBL and MCEL, because the first two calculate the compatibility of the input samples with all categories every time, and then backward the gradient to update the network. But MBL and MCEL only calculate one negative category at a time. In terms of robustness, REL and MBL exceed others by introducing a margin 1 in the compatibility function. In terms of flexibility, MCEL and MCPL are better than the others. The former wins by defining various smoothing functions to obtain different loss functions. The advantage of the latter is that it could be directly applied to other classification loss functions due to the conversion of compatibility to the probability distribution. In terms of simplicity, MBL and MCPL are better. The REL needs to choose a suitable weight function L. The smoothing function ℓ in MCEL has various hyperparameters that are required to be defined. MBL and MCPL have no additional hyperparameters.

6. Experiments

In this section, we will introduce all experiments to verify the effectiveness of our method on seen categories and unseen categories. Our code[2] is written in Python, and the neural network model is implemented with Pytorch. The size of the input image is normalized to 64×64. All experiments are conducted with Adam optimization with $lr = 0.005, beta_1 = 0.5, beta_2 = 0.9$. We do gradient clipping for stable training if using Euclidean distance metrics.

6.1. Dataset

We use the hierarchical decomposing knowledge of Chinese characters from IDS defined in UNICODE standard 9.0 to provide each character description in terms of what sub-characters make it up and how they are laid out. IDS uses the full range of Chinese characters, so each character has its own corresponding ideographic description sequence.[3]

The handwritten Chinese characters are from the CASIA-HWDB 1.0–1.2 database [19] and ICDAR2013 database [20]. Previous radical matching methods filter characters with particular radicals or structures. We believe that a general zero-shot HCCR method should not limit the categories of test data. So we directly select all level-1 and level-2 6763 characters as our label set for HCCR tasks. And we conduct experiments on printed fonts. We select 20 fonts from [30] to generate a printed dataset (3755 classes) and compare the results of different methods. We also conduct experiments on characters in natural scenes. Chinese characters in the wild (CTW) [21] is a large Chinese character dataset in natural scenes, which contains rural, flat, raised, urban, poorly illuminated, distant, partially occluded characters. We have access to 805,182 instances, 99.1% of which are commonly used characters in GB2312. It contains 3650 character categories.

[2] Our code will be released if this paper is accepted.

[3] https://github.com/cjkvi/cjkvi-ids

Table 2
Performance comparisons on level-1 unseen handwritten characters with different methods.

train	test	DenseRAN [8]	FewshotRAN [4]	Ours
500	1000	1.70%	33.6%	***33.71%***
1000	1000	8.44%	41.5%	***53.91%***
1500	1000	14.71%	63.8%	***66.27%***
2000	1000	19.51%	70.6%	***73.42%***
2755	1000	30.68%	77.2%	***80.95%***

Table 3
Performance comparisons on level-1 unseen printed characters with different methods. FewshotRAN and ours(+1) are trained with another font for extra samples (one-shot experiment setting). DenseRAN and ours are trained under zero-shot experiment setting.

train	test	DenseRAN [8]	FewshotRAN [4]	Ours/Ours(+1)
500	1000	7.28%	20.81%	34.14%/89.51%
1000	1000	29.86%	42.61%	60.49%/94.05%
1500	1000	48.49%	61.23%	73.17%/95.92%
2000	1000	60.47%	70.08%	82.15%/96.56%
2755	1000	73.12%	79.68%	87.67%/97.31%

6.2. Experiments on unseen categories

The original selection in DenseRAN and FewshotRAN of test categories is based on the hypothesis that all radicals are covered in all training sets. However, a different selection of categories may have a great impact on the final result. We hope that the zero-shot HCCR task has no special constraints on the categories itself. So our basic experimental setup is as follows: Randomly select disjoint subsets $\mathcal{Y}_{trn}$ and $\mathcal{Y}_{tst}$ from all prepared classes, choose all samples with labels in $\mathcal{Y}_{trn}$ as a training set, choose all samples with labels in $\mathcal{Y}_{tst}$ as a test set. The accuracy on the test set is taken as the evaluation results.

We conduct experiments with different sizes of $\mathcal{Y}_{trn}$. This random division would increase the difficulty since some radicals in the test set would not be seen in the training set. It is necessary to achieve zero-shot recognition without limiting categories of the test sets, especially in an open-world scenario. Table 2 shows that our method is better than the zero-shot recognition method DenseRAN on the handwritten dataset, even better than the few-shot recognition method FewshotRAN which uses additional printed fonts as training samples. When provided one-shot (another font) as extra training data, our method could recognize almost all fonts in printed character dataset as in Table 3. HDE-Net also outperforms other methods on the CTW dataset as in Table 4.

6.3. Experiments on unseen categories with different radios

To further explore the effect of different ratios of training categories in all categories, we conduct experiments on level1 and level2 categories. The experiment setting is as follows: we randomly select a certain ratio of all 6763 Chinese character sets of GB2312-80 as $\mathcal{Y}_{trn}$, and the rest as test category $\mathcal{Y}_{tst}$. We collected all samples of CASIA-HWDB 1.0–1.2 and ICDAR2013, then used all

Table 4
Performance comparisons on CTW unseen wild characters with different methods.

train	test	DenseRAN [8]	FewshotRAN [4]	Ours
500	500	0.12%	2.36%	**23.53%**
1000	500	1.50%	10.49%	**38.47%**
1500	500	4.95%	16.59%	**44.17%**
2000	500	10.08%	22.03%	**49.79%**
3150	500	15.95%	28.45%	**57.42%**

Table 5
Performance comparisons on all GB2312-80 handwritten characters with different ratios (training set categories / total set categories).

training classes/total classes	10%	20%	30%	40%	50%	60%	70%	80%	90%
test classes/total classes	90%	80%	70%	60%	50%	40%	30%	20%	10%
accuracy%	23.92%	45.57%	57.51%	64.97%	73.29%	78.89%	83.91%	87.98%	91.93%

$\mathcal{Y}_{trn}$ category samples as training sets, and used all $\mathcal{Y}_{tst}$ category samples as test sets.

As shown in Table 5, we achieve an accuracy of 45.57% when only 20% categories appear in the training set ($1352/6763 = 0.2$). The accuracy increases to 91.93% when the ratio is 0.9. When the ratio of training sets to total categories is relatively small, such as from 10% to 20%, the accuracy on the test set increases by 21.65%. It is because the training set itself contains few radicals and structures, which are not enough to cover all radicals in the test set. When the ratio is relatively large, such as from 80% to 90%, since the training set has already been sufficient, the accuracy of the test set increases only by 3.95%.

6.4. Experiments on seen categories

To show the advantage and practicability of HDE-Net, we choose two challenging datasets named as CTW and ICDAR2013. The former is to classify Chinese characters in the wild, and the latter handwritten Chinese characters. We have compared with the state-of-the-art methods under the traditional experiment settings.

Our method achieves the state-of-the-art result on CTW. And it is quite competitive and outperforms other radical-based methods on ICDAR2013. We think it brings in advantage to infer characters by radicals and structures on CTW because there are many partially occluded or partially ambiguous characters in wild Chinese characters.

6.5. Qualitative and quantitative analyses

We conduct qualitative analysis of label embedding, feature spaces, and radical predictions to clarify why HDE-Net performs well on zero-shot setting or traditional setting. We also conduct an ablation study for quantitative analysis of different modules, loss functions, and parameters. We randomly choose 2755 classes as $\mathcal{Y}_{trn}$, and the other 1000 classes in the level-1 characters as $\mathcal{Y}_{tst}$ in this section.

6.5.1. Qualitative analysis

Since no samples of the test category are provided, the relation between the test category and the known category is particularly important. HDE is a novel label embedding to capture these relations with structural information, hierarchical weights, and fundamental radicals. For example, the radical (‘口’) and the sub-character (‘只’) are independent radicals in other methods but not in ours. HDEs of these two examples are similar in the 87-dim (representing ‘口’) and different in other dimensions. Most of the methods directly regard sub-characters as basic radicals to reduce the difficulty of predicting long representation sequences, resulting in fewer relations between characters, unfortunately. These relations are the key to transfer learning from the seen Chinese characters to the unseen Chinese characters.

Fig. 10 shows four t-SNE [39] visualizations of the 272-dim embeddings learned by HDE-Net. We visualize two random subsets of train and test characters, and two representative test subsets to gain better insight into embeddings. Most of the samples, even including misclassified samples of the same class, are closer than different classes. The clustering results show inspiring potential classifiable characteristic. But the embeddings of some classes are still

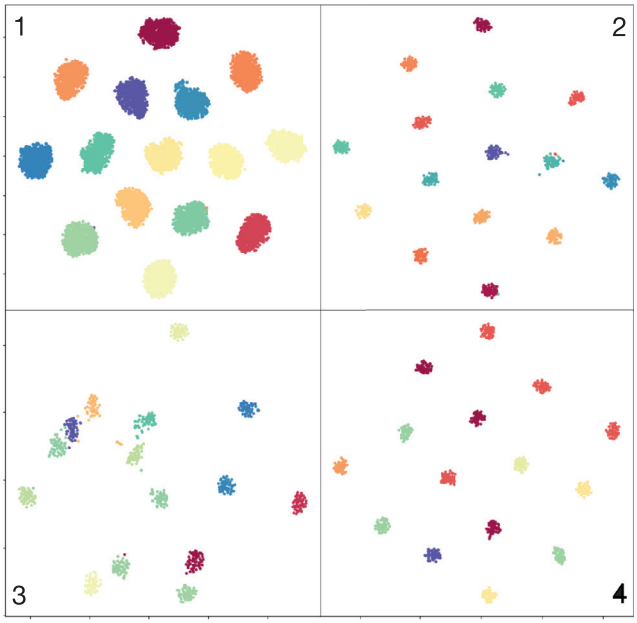

Fig. 10. t-SNE visualization of embeddings learned by HDE-Net. 1. random training subset, 2. random test subset, 3. the most incorrectly classified subset “年坯肉入事承手水瓦巡燕鼎曰臼四”, 4. the most correctly classified subset.

far from ground truths due to the rare and insufficiently trained radicals. The more frequently radicals appear during training, the easier to classify correctly. If the test characters are sorted according to the error rate, the average accuracy of the former half is 66.09% and the latter half 95.81%. Half of the categories contribute about 90% of the errors. Among them, 2% of Chinese characters were completely misidentified. Error often occurs when the significant radicals of new characters rarely seen in the training set. If the test character itself is a new radical, HDE-Net could not recognize correctly, (Tables 6 and 7).

To further illustrate the readable radical-level prediction of HDE-Net, we first predict the structure of samples. Base on the knowledge of the structure, we mask half part of the sample and then test whether HDE-Net will output the remaining part of the sample. For example, mask the left part of a character ‘明’ and then predict the remain radical ‘月’. We should extend the candidate

Table 6
Performance comparisons on CTW with different methods. Our method outperforms the-state-of-art methods.

Method	Accuracy
AlexNet [31]	76.43%
ResNet50 [22]	79.46%
ResNet152 [21]	80.94%
Inception-v4 [32]	82.28%
DenseNet [33]	79.88%
JSRAN [34]	87.57%
DenseRAN [8]	85.56%
FewshotRAN [4]	86.78%
Ours	**89.25%**

Fig. 11. Radical-level predictions of HDE-Net. The first row are input samples. The next five rows are radical-level predictions of the left or up parts with probabilities(%). The last five rows are those of the remaining right or down parts. Only top 5 results for each part of characters are shown in the figure. For example, our model infers that the character '明' (bright) is left-right structure, the left part is the radical '日' (sun) with probability 63.3%, and the right part is the radical '月' (moon) with probability 91.7%.

Table 7
Performance comparisons on ICDAR2013 with different methods. Our method outperforms the radical-based methods. Indeed, DenseRAN, FewshotRAN and our method use the prior decomposition knowledge of Chinese characters.

Method	Accuracy
Human Performance [20]	96.13%
HCCR-Gabor-GoogLeNet [35]	96.35%
CNN-Single [36]	96.58%
DirectMap+ConvNet+Adaptation [37]	**97.37%**
M-RBC+IR [38]	**97.37%**
DenseRAN [8]	96.66%
FewshotRAN [4]	96.97%
Ours	97.14%

Table 8
Performance comparison among different designs of ϕ and θ. Row 1,2,3,4 have the same modules but different HDE $\phi(y)$; Row 4,5,6 have the same HDE but different feature extractors θ.

Method	Accuracy
1. baseline	68.75%
2. - structure	71.36%
3. - radical	18.05%
4. ours	80.95%
5. θ/DenseNet [8]	77.21%
6. θ/ResNet [4]	78.62%

category set with all radicals to make radical-level predictions. The result in Fig. 11 shows that HDE-Net can indeed identify radicals of unseen Chinese characters.

6.5.2. Ablation study

Different modules. We mainly compare the differences among different strategies of HDE and different modules of the feature extraction. The baseline adopts attribute-based embedding, which represents each character as a binary vector, where each dimension corresponds to a particular radical. "2.−structure" represents hierarchical radicals by replacing the last 12-dim in HDE as zero. "3.−radical" represents hierarchical structures by replacing the value of the first 260-dim in HDE as zero. The meanings of these four strategies for the character '枳' (orange) are: the radicals are '木口八'; the half part is radical '木' and the quarter parts are radicals '口八'; the left part, right-up part, right-down part are three radicals; the left part is '木', and right-up, right-down parts are radicals '口八', respectively. We also use two other basic feature extraction modules 5. θ/DenseNet [8] and 6. θ/ResNet [4] to replace our module θ as comparisons.

The experimental results are shown in Table 8. Comparing 2 with 3, we find that the radicals play more important roles in character recognition than structures. From the comparison between 1 and 2, we can find that the hierarchical radical embedding with the impact factor of node-path has a 2.61% improvement than binary radical embedding. Form the comparison between 2 and 4, the accuracy increases by 9.59% through adding structures in HDE. Our method performs 12.20% higher accuracy than the baseline with multi-label embedding. As for different feature extractors, our method has a slight increase in accuracy compared with other modules in [4,8].

Different loss functions. We compare the accuracy in different loss functions. We adopt four loss functions with cosine distance and Euclidean distance, respectively. The results are as shown in Table 9.

The cosine distance metric is better than the Euclidean distance metric for its better performance and competitive convergence. There may be two reasons for this. One is the impact of our interpretable embedding space. In our method, each class is encoded as the HDE, where each dimension corresponds to a specific primitive. The separation of the HDE dimensions makes it more applicable to the inner product measure style. On the other hand, we introduce the scale parameter of τ. It is hard to get a uniform scale of compatibility in the Euclidean distance metric even with scale parameters.

We have previously pointed out that the margin in loss functions can make our method more robust, which has been verified by experiments. MCEL without margin performs much worse than REL and MBL which with margin 1 in loss functions. Besides, when we use REL but replacing the margin 1 with 0, the method does not converge regardless of the Euclidean distance and the cosine

Table 9
Performance comparisons among different loss functions. MCPL with cosine distance is the best in the table, and selected in other experiments.

Distance	REL	MBL	MCEL	MCPL
Cosine	79.30%	80.10%	74.15%	***80.95%***
Euclidean	75.09%	73.58%	57.00%	74.76%

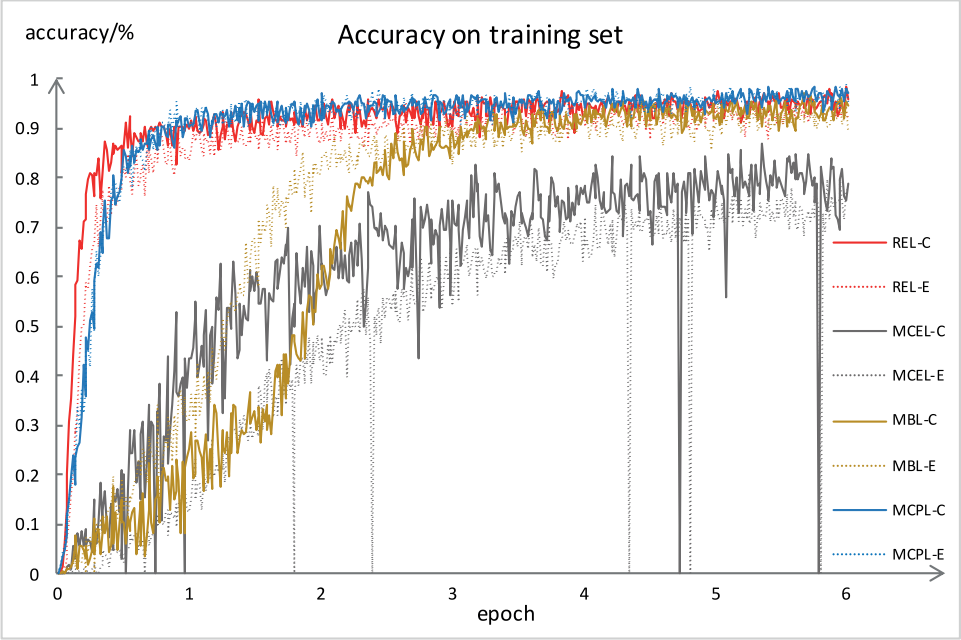

Fig. 12. Training accuracy on different loss functions in the first six epochs. REL-C represents the REL with cosine distance. It can be observed that REL and MCPL (in red and blue) converge faster than MBL and MCEL (in brown and gray). Training with MCEL is unstable due to zero margin and vanishing gradient. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

distance. However, MCPL performs well without margin. We guess that the softmax function is a normalization bringing the entire probability distributions of adjusted distances into alignment. Experiments also verify our conjecture about convergence. REL and MCPL converge faster than MBL, MCEL due to the weighted loss on different negative classes. And the training with MCEL is unstable due to zero margin and vanishing gradient. Details can be found in Fig. 12.

Different parameters α, β, λ. Different parameters may result in different HDE. We suggest $0 < \alpha < 1$ to make sure that nodes with shorter node-path (smaller depth) have greater impacts on the character and suggest $\beta \ll \alpha$ to ensure that the weight v_{n_i} is affected by the depth far more than the rank in brother nodes. For example, the HDE of '枳' as shown in Table 1 is composed by the following three radicals, '木' in the 141st dimension, '口' in the 64th dimension, '八' in the 87th dimension. And '木' is half of the character while '口' and '八' are the two quarters of the character '枳'. Intuitively, the value of the 141st dimension may be far larger than two roughly equal values of the 64th and the 87th dimensions. The trade-off λ balances the weights of radicals and structures. HDE would be totally hierarchical radicals if setting $\lambda = 0$. We have tried some α, β, λ as Tables 10 and 11. Experiments have shown $\alpha = 0.5$, $\beta_0 = 0.001$ and $\lambda = 0.5$ performs best. In fact, our hierarchical decomposition embedding method can achieve a quite good performance in many cases. We finally suggest that $\alpha \in [0.5, 0.8)$, $\beta_0 \in (0, 0.001]$, $\lambda \in (0.25, 0.5]$.

Table 10
Performance comparisons on level-1 unseen characters with different parameters α, β in HDE. The performance with $\beta_0 = 0$ is competitive due to the scarcity (0.212%) of those characters (i.e.,'呆杏') that have symmetric siblings.

$\alpha = 0.2$, $\beta_0 = 0.001$	$\alpha = 0.8$, $\beta_0 = 0.001$	$\alpha = 0.5$, $\beta_0 = 0.001$	$\alpha = 0.5$, $\beta_0 = 0.01$	$\alpha = 0.5$, $\beta_0 = 0.0001$	$\alpha = 0.5$, $\beta_0 = 0$
67.18%	79.02%	***80.95%***	79.71%	80.38%	80.64%

Table 11
Performance comparisons on level-1 unseen characters with different parameters λ in HDE.

λ	0.005	0.05	0.25	0.5	1	2.5	5	50
accuracy/%	73.91	79.01	80.77	**80.95**	79.00	71.71	65.58	37.32

7. Conclusion

In this work, we achieved zero-shot learning for HCCR tasks by building an HDE framework based on hierarchical knowledge of character labels. Based on the fact the different characters share the standard hierarchical primitives, we pursued a category-agnostic mapping from image to primitive. First, we introduced a novel HDE method to represent Chinese characters' decomposition knowledge with an embedding vector. It preserved the original hierarchical decomposition information and assigned each type of primitives to each dimension. Next, we developed HDE-Net to learn the transformation from the sample space to the embedding space and also designed four loss functions to train this framework. Qualitative and quantitative experiments are conducted to demonstrate the effectiveness in both the character classification and the radical prediction.

Compared with previous works, our HDE-Net has achieved state-of-the-art performance for zero-shot HCCR tasks. It even yields higher accuracy than several few-shot learning methods that used some training samples of test classes. Our HDE-Net is very suitable for the HCCR tasks when the number of test classes is extensive, or test classes are rare, or test classes change over time, or it is expensive to obtain labeled instances. On the one hand, our work contributes to finding the rules behind Chinese characters' decomposition knowledge and exploring the relevance of all Chinese characters. On the other hand, Our method can be extended to many areas with hierarchical structural knowledge and would be a potential inspiration for zero-shot learning. Because it transforms complex and diverse categories into basic primitive compositions that allow the underlying information to be shared between different categories. Since the performance is still worse than human performance on unseen classes, our framework has the potential to be improved further.

Declaration of Competing Interest

None declared.

Acknowledgements

This work is funded by the Natural Science Fundation of China (NSFC) and the German Research Foundation (DFG) in Project Crossmodal Learning, NSFC 61621136008 / DFG TRR-169. It is funded by Beijing Academy of Artificial Intelligence (BAAI). And I gratefully acknowledge the help of Tianwei Wang, who has offered me suggestions for some experiments about FewshotRAN.

References

[1] T. Wang, F. Yin, C. Liu, Radical-based Chinese character recognition via multi-labeled learning of deep residual networks, in: International Conference on Document Analysis and Recognition, 2017, pp. 579–584.
[2] T. Wang, C. Liu, Fully convolutional network based skeletonization for handwritten Chinese characters, in: AAAI Conference on Artificial Intelligence, 2018.
[3] J. Zhang, J. Du, L. Dai, Radical analysis network for learning hierarchies of Chinese characters, Pattern Recognit. (2020) 107305.
[4] T. Wang, Z. Xie, Z. Li, L. Jin, X. Chen, Radical aggregation network for few-shot offline handwritten Chinese character recognition, Pattern Recognit. Lett. 125 (2019) 821–827.
[5] W.W. Ip, K.F. Chung, D.S. Yeung, Offline handwritten Chinese character recognition via radical extraction and recognition, in: International Conference on Document Analysis and Recognition, 1997, pp. 185–189.
[6] A. Wang, K. Fan, Optical recognition of handwritten Chinese characters by hierarchical radical matching method, Pattern Recognit. 34 (1) (2001) 15–35.
[7] A. Amin, C. Sammut, K. Sum, Learning to recognize hand-printed Chinese characters using inductive logic programming, Int. J. Pattern Recognit. Artif. Intell. 10 (7) (1996) 829–847.
[8] W. Wang, J. Zhang, J. Du, Z. Wang, Y. Zhu, Denseran for offline handwritten Chinese character recognition, in: International Conference on Frontiers in Handwriting Recognition, 2018, pp. 104–109.
[9] W. Wang, V.W. Zheng, H. Yu, C. Miao, A survey of zero-shot learning: settings, methods, and applications, ACM Trans. Intell. Syst. Technol. 10 (2) (2019) 13.
[10] K. Weiss, T.M. Khoshgoftaar, D. Wang, A survey of transfer learning, J. Big Data 3 (1) (2016) 9.
[11] F.X. Yu, L. Cao, R.S. Feris, J.R. Smith, S.-F. Chang, Designing category-level attributes for discriminative visual recognition, in: IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 771–778.
[12] C.H. Lampert, H. Nickisch, S. Harmeling, Attribute-based classification for zero-shot visual object categorization, IEEE Trans. Pattern Anal. Mach. Intell. 36 (3) (2014) 453–465.
[13] Z. Akata, F. Perronnin, Z. Harchaoui, C. Schmid, Label-embedding for image classification, IEEE Trans. Pattern Anal. Mach. Intell. 38 (7) (2016) 1425–1438.
[14] G. Xie, L. Liu, X. Jin, F. Zhu, Z. Zhang, J. Qin, Y. Yao, L. Shao, Attentive region embedding network for zero-shot learning, in: IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 9384–9393.
[15] J. Lu, J. Li, Z. Yan, F. Mei, C. Zhang, Attribute-based synthetic network (abs-net): learning more from pseudo feature representations, Pattern Recognit. 80 (2018) 129–142.
[16] E. Schonfeld, S. Ebrahimi, S. Sinha, T. Darrell, Z. Akata, Generalized zero-and few-shot learning via aligned variational autoencoders, in: IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 8247–8255.
[17] I. Tsochantaridis, T. Joachims, T. Hofmann, Y. Altun, Large margin methods for structured and interdependent output variables, J. Mach. Learn. Res. 6 (Sep) (2005) 1453–1484.
[18] M. Saerens, F. Fouss, L. Yen, P. Dupont, The principal components analysis of a graph, and its relationships to spectral clustering, in: European Conference on Machine Learning, Springer, 2004, pp. 371–383.
[19] C. Liu, F. Yin, D. Wang, Q. Wang, Casia online and offline Chinese handwriting databases, in: International Conference on Document Analysis and Recognition, 2011, pp. 37–41.
[20] F. Yin, Q. Wang, X. Zhang, C. Liu, Icdar 2013 Chinese handwriting recognition competition, in: International Conference on Document Analysis and Recognition, 2013, pp. 1464–1470.
[21] T. Yuan, Z. Zhu, K. Xu, C. Li, T. Mu, S. Hu, A large Chinese text dataset in the wild, J. Comput. Sci. Technol. 34 (3) (2019) 509–521.
[22] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[23] K. He, G. Gkioxari, P. Dollár, R. Girshick, Mask r-cnn, in: IEEE International Conference on Computer Vision, 2017, pp. 2961–2969.
[24] Z. Wu, C. Shen, A. Van Den Hengel, Wider or deeper: revisiting the resnet model for visual recognition, Pattern Recognit. 90 (2019) 119–133.
[25] J. Weston, S. Bengio, N. Usunier, Large scale image annotation: learning to rank with joint word-image embeddings, Mach. Learn. 81 (1) (2010) 21–35.
[26] B. Juang, S. Katagiri, Discriminative learning for minimum error classification, IEEE Trans. Signal Process. 40 (12) (1992) 3043–3054.
[27] J. Kim, T.-H. Oh, S. Lee, F. Pan, I.S. Kweon, Variational prototyping-encoder: pne-shot learning with prototypical images, in: IEEE Conference on Computer Vision and Pattern Recognition, 2019.
[28] C. Liu, M. Nakagawa, Evaluation of prototype learning algorithms for nearest-neighbor classifier in application to handwritten character recognition, Pattern Recognit. 34 (3) (2001) 601–615.
[29] J. Snell, K. Swersky, R. Zemel, Prototypical networks for few-shot learning, in: Advances in Neural Information Processing Systems, 2017, pp. 4077–4087.
[30] Z. Zhong, L. Jin, Z. Feng, Multi-font printed Chinese character recognition using multi-pooling convolutional neural network, in: International Conference on Document Analysis and Recognition, 2015, pp. 96–100.
[31] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, 2012, pp. 1097–1105.
[32] C. Szegedy, S. Ioffe, V. Vanhoucke, A.A. Alemi, Inception-v4, inception-resnet and the impact of residual connections on learning, AAAI conference on Artificial Intelligence, 2017.
[33] G. Huang, Z. Liu, L. Van Der Maaten, K.Q. Weinberger, Densely connected convolutional networks, in: IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 4700–4708.
[34] C. Wu, Z. Wang, J. Du, J. Zhang, J. Wang, Joint spatial and radical analysis network for distorted Chinese character recognition, in: International Conference on Document Analysis and Recognition, 5, 2019, pp. 122–127.
[35] Z. Zhong, L. Jin, Z. Xie, High performance offline handwritten chinese character recognition using googlenet and directional feature maps, in: International Conference on Document Analysis and Recognition, 2015, pp. 846–850.
[36] L. Chen, S. Wang, W. Fan, J. Sun, S. Naoi, Beyond human recognition: a cnn-based framework for handwritten character recognition, in: Asian Conference on Pattern Recognition, 2015, pp. 695–699.
[37] X. Zhang, Y. Bengio, C. Liu, Online and offline handwritten Chinese character recognition: a comprehensive study and new benchmark, Pattern Recognit. 61 (2017) 348–360.
[38] X. Yang, D. He, Z. Zhou, D. Kifer, C.L. Giles, Improving offline handwritten Chinese character recognition by iterative refinement, in: International Conference on Document Analysis and Recognition, 2017, pp. 5–10.
[39] L.v.d. Maaten, G. Hinton, Visualizing data using t-sne, J. Mach. Learn. Res. 9 (Nov) (2008) 2579–2605.

Zhong Cao received the B.S. degree from Tsinghua University, Beijing, China, in 2014, where he is currently working toward the Ph.D. degree in the Department of Automation. His research interests include machine learning, deep learning and computer vision.

Jiang Lu received the B.S. degree from Tsinghua University, Beijing, China, in 2013, where he is currently working toward the Ph.D. degree in the Department of Automation. His research interests include machine learning, deep learning and computer vision.

Sen Cui received the B.S. degree from Tsinghua University, Beijing, China, in 2019, where he is currently working toward the Ph.D. degree in the Department of Automation. His research interests include machine learning, deep learning and computer vision.

Changshui Zhang received the B.S. degree in mathematics from Peking University, Beijing, China, in 1986, and the Ph.D. degree from the Department of Automation, Tsinghua University, Beijing, in 1992. His current research interests include artificial intelligence, image processing, pattern recognition, machine learning, and evolutionary computation.