International Journal on Document Analysis and Recognition (IJDAR) (2023) 26:401–417
https://doi.org/10.1007/s10032-022-00423-6

ORIGINAL PAPER

HWNet v3: a joint embedding framework for recognition and retrieval of handwritten text

Praveen Krishnan[1] · **Kartik Dutta**[1] · **C. V. Jawahar**[1]

Received: 28 February 2021 / Revised: 16 July 2022 / Accepted: 9 December 2022 / Published online: 28 January 2023

Abstract
Learning an efficient label embedding framework for word images enables effective word spotting of handwritten documents. In this work, we propose different schemes of label embedding for word images using deep neural architectures and their representations. We refer to our first scheme as the two-stage label embedding technique which projects both word images and their corresponding textual strings into a common subspace. We further introduce an end-to-end label embedding scheme using deep neural architecture which simplifies the embedding process and reports state-of-the-art performance for the task of word spotting and recognition. We also validate the role of synthetic data as a complementary modality to further enhance the embedding process. On the challenging IAM handwritten dataset, we report an mAP of 0.9753 for query-by-string-based word spotting, while under lexicon-based word recognition, our proposed method reports 1.67 and 3.62 character and word error rates, respectively. We also present the detailed ablation study on various variants of our end-to-end embedding architecture and perform analysis under varying embedding sizes. We further validate the embedding scheme on degraded printed document datasets from both Latin and Indic scripts.

Keywords Word image embedding · Word spotting · Word recognition · Label embedding

1 Introduction

Accurate recognition and retrieval of handwritten text from scanned document images have remained as the prime problem of importance for many decades. In recent years, there have been significant advancements in the field of computer vision and machine learning. Especially in deep learning, interesting methods with competitive performances on the standard handwritten benchmarks have evolved. Given the inherent challenges in handwritten documents, the problem is either posed in a recognition-free manner which is popularly referred to as word spotting [36] or in a recognition-based manner where the input image (line/word) is transcribed into a valid textual string. One can consider these two paradigms as complementary where the former tackles the problem of finding a holistic representation for word images, optimum for various downstream tasks, including retrieval and recognition of words, while in the latter, the goal is explicitly set for transcription rather than representation learning. In this work, we follow the quest of learning the appropriate representation for word images and their corresponding text in a label embedding framework [4,38], which is useful for downstream tasks such as word spotting and recognition.

The basic idea of label embedding is to embed both images and their labels (text) into a common subspace that respects the lexical similarity across modalities (image and text). The key question for optimum label embedding for a textual image lies in three parts: (i) finding a good representation of images, (ii) deriving a similar representation for text and (iii) finding the common subspace and a similarity metric that respects the lexical similarity across modality. In the domain of word spotting, the concept of word attributes [4,38] using the pyramidal histogram of characters (PHOC) has been a key contribution to the community, which enabled label embedding techniques to effectively apply to word images. The textual PHOC attributes are calculated by dividing the word into multiple pyramid levels, and at each level, the histogram of characters and bi-grams is computed and later concatenated for the final representation. Here each attribute or dimension denotes the presence or absence of a character at a particular spatial position. Taking inspiration from PHOC representation, recent works [33,48–50,55] use deep convo-

✉ Praveen Krishnan
praveen.krishnan@research.iiit.ac.in

[1] IIIT Hyderabad, Hyderabad, India

lutional networks to project a word image onto a PHOC-based attribute space. Embedding word images, and the textual strings onto a common space, enables comparisons using a suitable similarity metric. In our work, we also utilize the PHOC representation as a way to encode textual modality. However, we also demonstrate unique ways of using a synthetic image as a complementary representation to embed textual modality effectively.

1.1 Contributions

The major contributions of this work are:

- We propose different schemes of label embedding in the domain of document images.
- We present a novel end-to-end embedding framework for learning the embedding space using a multi-task loss function.
- We validate the role of synthetic data as a complementary modality to enhance the embedding process.
- We perform extensive experimentation on different variants of the proposed label embedding schemes, embedding sizes and also explore the reduced need for real data with effective pre-training using synthetic dataset.

The embedding architectures presented in this work first appeared in our previous works [21,22]. In [21] we first proposed the two-stage label embedding framework, and in [22], we extended it to an end-to-end embedding framework. All our architectures including the variants proposed in this work use HWNet v2 [25] as the trunk model for computing features before the actual embedding process. In the current work, we further present our insights in designing the end-to-end architecture and also simplify the existing architecture and propose a newer variant titled as HWNet v3. We also demonstrate the effectiveness of the synthetic modality toward learning better features for both of our embedding schemes. The proposed representation reports a state-of-the-art performance for both word spotting and constrained word recognition on major handwritten datasets. In addition to handwritten documents, we also present the results on printed degraded datasets from Indic scripts to validate the generality of the embedding scheme.

Figure 1 presents the sample results on the task of constrained word recognition using a test lexicon. Here, we use the learned representation of word images and its corresponding textual strings of our proposed end-to-end embedding scheme to associate a word image to their correct transcription. The transcribed words are shown in green color on top of its word image. Note that, in this work, we assume the availability of the word-level segmentation from document images, similar to other related works such as [4,12,33,48,49,55].

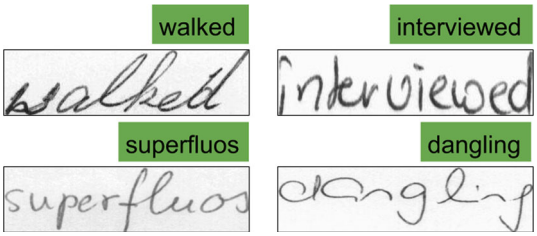

Fig. 1 Sample results of constrained word recognition using test lexicon. Here, we use the learned representation of word images and their corresponding textual strings of our proposed end-two-end embedding scheme to associate the word image to its correct transcription. The transcribed words are shown in green color on top of its word image

The rest of the paper is organized as follows. In Sect. 2, we present the related works in this domain, and in Sect. 3, we present the deep HWNet v2 architecture, which is used as the trunk model for computing the word-level representation. In Sects. 4 and 5 , we introduce the proposed label embedding schemes and its variants. In Sects. 6 and 7 , we demonstrate the various experiments using the proposed representation. In Sect. 8, we present the visualization on embedding space along with some qualitative results, and finally, in Sect. 9, we present our conclusion.

2 Related works

This section discusses the prominent works in word image representation learning for handwritten and printed documents. We split our discussion majorly into two parts; firstly, we overview the classical methods and move on to the recent methods based on deep architectures. Much of our discussion will be limited to the methods involving segmentation-based word spotting and label embedding frameworks proposed for word images and their text. For a much more detailed survey on word spotting methods, readers are recommended to refer [11].

2.1 Classical methods

Word spotting for document images was the first introduced by Manmatha et al. in [29] for indexing handwritten word images, and the idea was quickly popularized to degraded machine-printed documents and languages that do not have robust optical character recognizers (OCR). Initial attempts [27,29,37,53], mostly focused on variable-length representations of word images by considering it as a temporal sequence. Dynamic time warping (DTW)-based algorithms are found to be useful for matching variable-length representations (features) and are quite popular in speech [32,41] and other sequence matching problems. Most of these methods used profile features [30,35] which are computed at each

column of the word image and modeled in a temporal fashion. In [37,53] the local gradient features such as SIFT [28], HOG [8] were adapted for word spotting. For matching, [53] used a continuous DTW algorithm for partial word matches from line images, while [37] used hidden Markov model (HMM)-based classification method. Most of the former features discussed above are not robust to different fonts and writing styles, and are language-specific (Latin manuscripts). Moreover, the DTW, HMM-based scheme of matching does not scale to large datasets due to the higher time complexity. Hence the newer methods focused more on fixed length representation built on highly engineered features proposed in computer vision.

The popularity of the Bag Of Words (BOW) [46] framework using local gradient features such as SIFT and HOG led to its successful proliferation into the domain of document images [2,40,42,56]. In [42,56], BOW-based representation for word image retrieval of machine-printed documents was proposed using standard keypoint detectors such as Harris [16], FAST [39] corner detectors. The local features at those keypoints are computed using SIFT. Due to the fixed length and sparse nature of the representation, the matching was done using cosine distance and an inverted index was used for faster retrieval. To improve the precision of the search, Yalniz et al. [56] proposed the Longest Common Subsequence (LCS)-based spatial verification scheme between the top-k retrieved results and the original query representation. In [1,2] the BOW representation is further improved by projecting it on a topic space computed using latent semantic analysis [9], while the indexing is done through product quantization [19]. Aldavert et al. [2] present a detailed survey of BOW-based representation for handwritten word spotting with its analysis on the effect of codebook size, choice of encoding and normalization of the performance of word spotting. In [3], Almazán et al. use an exemplar-SVM for representing a query and performing the initial scoring of candidate words using a sliding window. Given the initial matches, the list is re-ranked using a Fisher vector-based representation. The key advantages of all these approaches were: (i) scalable representation and (ii) due to the unsupervised nature of learning, the methods were directly applicable to historical databases where the annotation is hard and expensive. However, its applicability was limited to single writer handwritten datasets where the variations in handwriting styles are less.

2.2 Word attributes

In the domain of word spotting, the concept of word attributes [4,38] using the pyramidal histogram of characters (PHOC) has been a key contribution to the community to formulate label embedding techniques. An attribute representation such as PHOC is the common link that connects a word image to its text. PHOC attributes are calculated by dividing the word into multiple pyramid levels and computing the histogram of characters and bi-grams at each level. The final representation is the concatenation of individual-level representations. The intuition behind the attribute is that it denotes the presence/absence of a character at a particular spatial position.

Similar to the textual representation, the PHOC representation for word images can be derived from the scores of attribute-level binary classifiers, which are learned from training word images that possess the particular attribute. In the original work of PHOC [4], Almazán et al. use Fisher-based holistic features of word images to build attribute classifiers that represent each word image as a vectorial representation computed from normalized attribute classifier scores. The choice of the number of levels and the character set is fixed during the training which in turn decides the number of attributes. Given the attribute-level representation for both word images and text, one can compare both representations seamlessly. Further, the same work proposes a common subspace regression formulation with a closed-form solution to project both representations onto a common subspace which captures the correlation among the attributes and leads to better representation.

2.3 Deep embedding

The concept of learning word attributes using PHOC has been recently demonstrated successfully using convolutional neural networks [33,48,49], which validates the generality of this representation. Poznanski et al. [33] adapted VGGNet [45] for the recognition of PHOC attributes by having multiple fully connected layers in parallel, each one predicting PHOC attributes at a particular level. In a similar direction, different architectures [48,55] were proposed using CNN networks which embeds the features into textual embedding spaces. Sudholt et al. [48] proposed an architecture to embed directly image features to PHOC attributes by having sigmoid activation in the final layer and avoiding multiple fully connected layers. It is referred to as PHOCNet, which uses the final layer activation to derive a holistic representation for word spotting. In their subsequent works [49,50], the authors extended the PHOCNet network by adding a temporal pooling layer (TPP-PHOCNet) and evaluated the different textual embedding and loss functions.

In [55], the authors proposed a two-stage architecture where a triplet CNN network is trained using SoftPN loss function in the first stage, and the learned image representation is embedded into a word embedding space (either PHOC, DCTOW, semantic, etc.) using a fully connected neural network where the loss is defined using a cosine embedding function. Here DCTOW refers to discrete cosine transform of Words, which is an alternative textual representation scheme proposed by the same group. Given a textual string, the

characters are converted into a one-hot representation and represented as a matrix. Further, a discrete cosine transform is taken for each row of this matrix and cropped to retain only the first R components. The resulting matrix is flattened and provides a fixed dimensional representation. The work also explores a semantic textual representation obtained using LSTM Char–Large model [20].

A more recently published work in the same space is from Gomez et al. [12], which learns an embedding space that respects Levenshtein distance between the samples. In [6], the authors proposed a convolutional siamese network for learning embedding for word images. Most of the above works use the output activation from the penultimate layer of the CNN network as the word features to perform spotting or retrieval.

In this work, we present two significant ways of label embedding for word images and their corresponding text, which enables both retrieval and recognition of handwritten word images. Our first approach extends the framework proposed by Almazán et al. [4] using deep features and improvises the framework with an additional synthetic modality. Our second approach introduces a novel end-to-end embedding scheme that simplifies the learning process and provides better representation. In both of the approaches, we bank upon the HWNet v2 architecture [25], which provides a robust holistic representation for word images. In the next section, we briefly overview the HWNet v2 architecture and later, we present the different embedding techniques proposed in this work. For a more detailed explanation of HWNet v2 architecture, interested readers are suggested to refer [25].

3 HWNet v2 architecture

We use the improved HWNet architecture referred to as HWNet v2 [25] for computing the holistic representation of word images toward the task of label embedding. Figure 2 presents the HWNet v2 architecture which consists of a ResNet34 network with four blocks where each block contains multiple resnet modules, a temporal TPP pooling layer and two fully connected layers. Here each ResNet module consists of two convolutional layers and a shortcut connection to enable residual learning. We also use two layers of fully connected networks toward the end instead of global

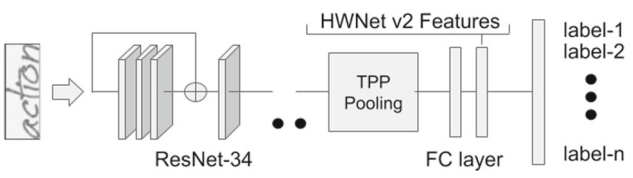

Fig. 2 HWNet v2 architecture [25]

average pooling (as originally proposed for ResNets) in order to capture better features in the penultimate layer of the network. To improve generalization and also converge faster, we use batch normalization after each convolutional and fully connected layer (FC) except the last one. The input word images are resized at multiple random scales on a padded image of size 128×384. While placing the image, we tend to avoid distorting the aspect ratio of word images except in a few rare cases where the image width is greater than 384. This scheme enables the ability to train at multiple scales, which is typically observed for a multiple writers scenario and thereby the network remains invariant to different scales while testing. The use of TPP pooling layer [with the valid region of interest (ROI) information] after the last convolutional network preserves only the valid activations coming from the region where the input word image is rendered and extracts a fixed dimensional representation which is further given to the FC layers. The training objective is formulated as a word classification problem using multinomial logistic regression loss and the weights are updated using a mini-batch gradient descent with momentum. To improve generalization, we train the network from scratch using IIIT-HWS synthetic dataset [23] and fine-tune it on a real dataset to reduce the domain gap between synthetic and real-world data. The activations from the penultimate layer of the network are taken as word image embedding after performing the L_2 normalization. For simplicity, we would refer to HWNet v2 as HWNet in the subsequent sections.

4 HWNet embedding

Taking inspiration from the HWNet-based word image representation, we now apply these features onto the word attributes framework [4] which enables the joint feature embedding for images and text. We refer to this scheme as the two-stage feature embedding, and it will be described in detail in Sect. 4.1. We will further enhance the two-stage scheme by incorporating the synthetic modality into the joint feature embedding process, and it will be presented in Sect. 4.2. However, before delving into the joint feature embedding process, we first motivate the need for embedding HWNet features.

A fundamental assumption used while training the HWNet architecture using the classification loss is that each class (vocabulary word) is assumed to be independent. In reality, different classes of word images share a considerable amount of visual information. For example, the words "School" and "Schooling" differ by just an inflection of "ing" in the suffix part of the second word, and we would prefer the feature space to obey the corresponding lexical ordering. Note that in this work, we only focus on lexical similarity and not on the actual semantics, for e.g., the words "car" and "cat," both are

perceived to have a lexical similarity, although they differ a lot in the semantic space. One can argue that such sharing of information is implicitly learned in the convolutional layers of the network. However, we believe that there is a need to make such relationships more explicit to aid learning better representation. In this section, we exploit the fine-grained relationships present among the word images using the word attribute framework [4] along with embedding the label information onto a common reduced subspace where both the text and image representations lie close to each other. Given such a reduced space, we have the following advantages: (i) The reduced space is of much lower dimensions as compared to the original dimensions with no loss in accuracy, (ii) seamlessly enables both query-by-string and query-by-example-based retrieval and (iii) less memory footprint for representation, which enables large-scale retrieval and recognition.

4.1 Two-stage joint feature embedding

The two-stage joint feature embedding, also referred as DeepEmbed, first converts the HWNet-based image features and the corresponding textual strings onto PHOC-based representations. Given the parallel representation from both image and text modality, we further embed it into a common subspace. This process is referred to as two-stage since the training of HWNet model and attributes model (for PHOC-based image representation) happens independently of each other. The computation of PHOC features for text and learning of the attribute models for PHOC-based image representation is done similarly as presented in [4].

Let $\mathcal{I} = \{\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_n\}$ be the set of n images from the training data and $\mathcal{Y} = \{\mathcal{Y}_1, \mathcal{Y}_2, \ldots, \mathcal{Y}_n\}$ be the corresponding text labels. Let $\phi_{\mathcal{Y}} : \mathcal{Y} \rightarrow \mathbb{R}^d$ be the text label embedding function which gives the textual PHOC representation. Here d is the number of attributes which is equivalent to PHOC dimension. Let $\mathcal{X} = \{\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_n\}$ where $\mathcal{X}_i$ is the HWNet representation for $\mathcal{I}_i$ word image. We can learn a similar attribute space for HWNet features by training d attribute classifiers which predict the probability of a particular attribute given its image representation. This would result in having both text and images in an $\mathbb{R}^d$ space and mutually comparable. We train the attribute classifiers using linear SVMs since the CNN (HWNet) features act as explicit feature maps that encode the nonlinearities present in feature space. Here each attribute classifier is trained discriminatively to predict a particular attribute such as "the presence of character 'x' in the first half of the word image" and so on. Given d attribute classifiers, the attribute embedding function is given as $\phi_{\mathcal{X}} : \mathcal{X} \rightarrow \mathbb{R}^d$, which encodes the classifier scores in predicting each attribute. Figure 3 presents the computation of $\phi_{\mathcal{Y}}$ and $\phi_{\mathcal{X}}$, respectively.

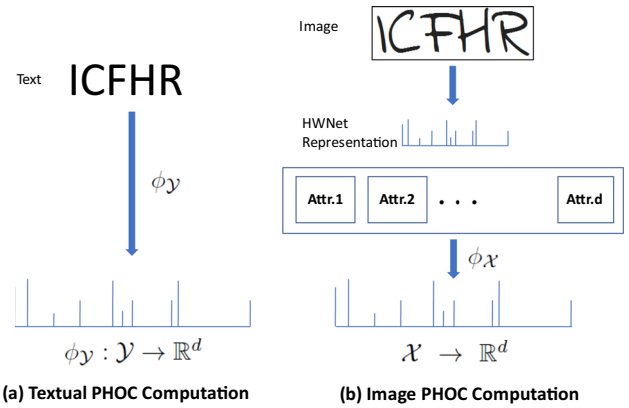

Fig. 3 Computation of PHOC representation using the embedding functions ($\phi_{\mathcal{Y}}, \phi_{\mathcal{X}}$) for text and image, respectively

Even though both image and text representation lie in $\mathbb{R}^d$ space, they are not optimally comparable because the scores lie in different ranges. To calibrate the scores along with maximizing the correlation between the multivariate vectors (text and images) we formulate the problem as a common subspace regression (CSR) as follows:

$$\begin{aligned} \underset{U,V}{\operatorname{argmin}} \; & \frac{1}{2} \parallel U^{\mathrm{T}} A - V^{\mathrm{T}} B \parallel_F^2 + \frac{\alpha}{2} \parallel U \parallel_F^2 + \frac{\alpha}{2} \parallel V \parallel_F^2 \\ \text{s.t. } & \psi_{\mathcal{X}}(\mathcal{X}) \psi_{\mathcal{X}}(\mathcal{X})^{\mathrm{T}} = 1 \\ & \psi_{\mathcal{Y}}(\mathcal{Y}) \psi_{\mathcal{Y}}(\mathcal{Y})^{\mathrm{T}} = 1 \end{aligned} \tag{1}$$

Here, $\psi_{\mathcal{X}}(\mathcal{X}) = U^{\mathrm{T}} A$, $\psi_{\mathcal{Y}}(\mathcal{Y}) = V^{\mathrm{T}} B$, $A = \phi_{\mathcal{X}}(\mathcal{X})$, and $B = \phi_{\mathcal{Y}}(\mathcal{Y})$. U, V are the projection matrices to be learned, and A, B are the features matrices storing the PHOC-based representation for image and text computed from the first stage. In the CSR formulation, the objective is to minimize the distance between an image and its corresponding text representation in a common subspace ($\mathbb{R}^{d'}$) found by the optimal projection matrices. In typical cases, $d' < d$, which is set empirically by tuning on a validation dataset. The above optimization problem could be treated as a generalized eigen vector problem, having a closed-form solution as presented in [4].

The learned subspace using projection matrices allows embedding both word images and text into joint feature space. This facilitates search using query-by-example and query-by-string word spotting seamlessly. It also enables constrained word level recognition using a finite lexicon. More details are presented in the experimental section. In the next subsection, we present an enhancement of the two-stage feature embedding framework which improves the projection of text modality into the common subspace.

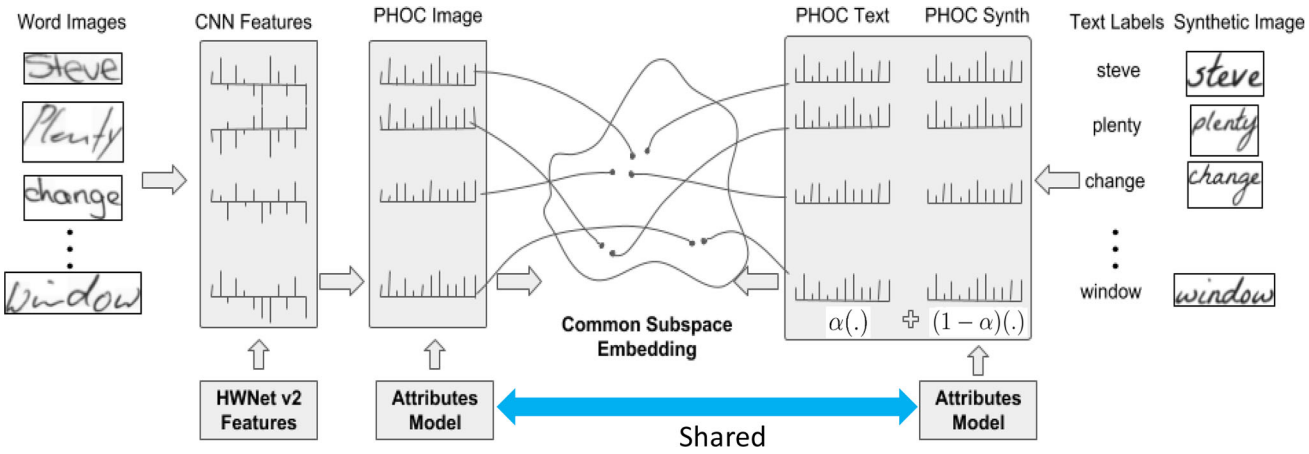

Fig. 4 Synthetic attribute embedding

4.2 Synthetic attribute embedding

In training the HWNet architecture, we demonstrated the role of synthetic data in pre-training deep neural networks which paved the way for efficient fine-tuning on real data. In this work, we present another interesting role of synthetic data, which could help to perform joint feature embedding more efficiently. Figure 4 presents the extended framework of the two-stage joint feature embedding, referred to as the synthetic attribute embedding or Synth + DeepEmbed, in results Sect. 6.5. We propose to use an additional synthetic modality to enrich the attribute embedding process of text labels and observed that it gives complementary information, which leads to effective learning of common subspace. Synthetic data for word images [23] are easy to generate and are available for most languages. For each textual label, we generate the corresponding synthetic image using a given font type. We use a shared attribute model to compute PHOC representation for both the real and synthetic images. Given the PHOC representation for both text label and synthetic image, we perform a weighted sum as follows:

$$\phi_{\mathcal{S}}(\mathcal{Y}, \mathcal{S}) = \alpha \times \phi_{\mathcal{Y}}(\mathcal{Y}) + (1-\alpha) \times \phi_{\mathcal{X}}(\mathcal{S}) \tag{2}$$

Here, $\mathcal{S}$ is the HWNet feature for the synthetic image corresponding to label $\mathcal{Y}$, and $\phi_{\mathcal{S}}(\mathcal{Y}, \mathcal{S})$ refers to the attribute embedding function on synthetic images and its corresponding label strings as presented in the above equation. Also, the CSR formulation remains the same as presented in Eq. 1; with the only difference being instead of $\phi_{\mathcal{Y}}$, we now use the newer representation given as $\phi_{\mathcal{Y},\mathcal{S}}$. The basic idea of composing the label representation using both text and visual (synthetic) modality allows us to exploit the complementary information during embedding. We refer to this as synthetic attribute embedding, which performs better for query-by-string retrieval. The value for α is set empirically. Note that, in comparison with our previous framework, as presented in Sect. 4.1, the synthetic attribute embedding framework does not add any complexity to the total number of learnable parameters of the system.

5 End-2-end embedding

The idea of synthetic attribute embedding discussed in the previous section is a two-stage approach where the image embedding is learned in the first pass, while the projection into the attribute space happens in the second pass. We now propose an end-to-end trainable deep word embedding network with three main objectives: (i) replacing the attribute classifiers using multi-layer fully connected networks, (ii) to learn a new embedding space rather than fixing it to be PHOC and (iii) to simplify both training and testing phases. Note that, although PHOC has shown the optimum attribute representation of word images and text, in this work, we investigate the possibility of automatic learning of such representation. Figure 5 presents the two variants of end-to-end deep convolutional network for simultaneous learning of both textual and image embeddings. Here, the Variant I architecture was proposed in our initial work [22]. Later as an extension, we simplified the architecture and the learning process for better sharing of features among channels, and this is represented as the Variant II architecture in the figure. In the following discussion, we first explain the Variant I architecture and later present the Variant II architecture which allows better sharing of information among real and synthetic image modalities.

5.1 Variant I

There are two streams in the proposed architecture, real and label stream, respectively, and each stream is passed through a set of layers of feature extraction and embedding. The real

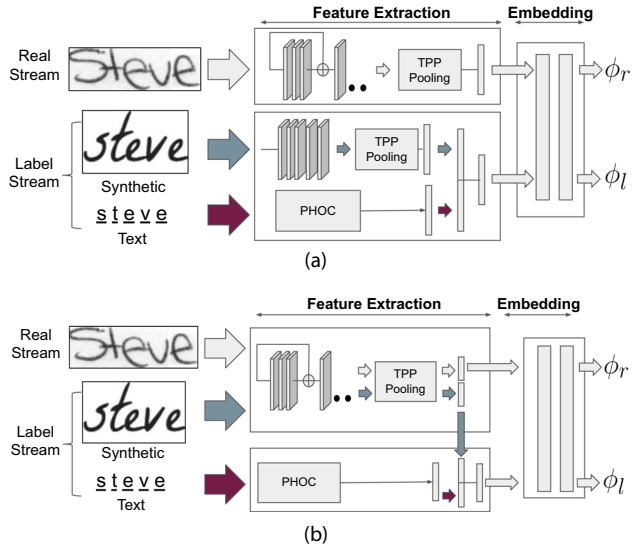

Fig. 5 The proposed variants of end-2-end embedding network for learning both image and textual embedding using multi-task loss function. **a** Variant I which was first proposed in [22], uses a separate feature extraction channel for real and synthetic images, while **b** presents the Variant II of the end-2-end embedding network which allows better sharing of features among the real and synthetic image

stream is associated with actual handwritten/printed word images, while the label stream is used for propagating the label information. We further split the label stream into two parts: the textual string, and the synthetic image rendering of the text. In the current set-up, we use PHOC as our textual representation and the synthetic image is rendered using one single font. Note that, the architecture is generic to support other textual representations such as DCToW [55] and the rendering of synthetic images could be done using multiple fonts. In comparison with the previous two-stage embedding scheme (synthetic attribute embedding), one could think of the real stream as the left part of Fig. 4 and the label stream as its right part.

The features of both real and synthetic images are captured using convolutional layers, where we use ResNet34 architecture for real images, while a simpler network similar to AlexNet is used for extracting features from the synthetic image. The choice is made by understanding the complexity level of data variations of individual image domains. Here also, we use the TPP pooling layer after the last convolutional layer, similar to the one used for HWNet architecture which is presented in Sect. 3. The label stream consists of the network comprising of a PHOC extractor which is appended to the features obtained from the synthetic image. This is achieved by L_2 normalization of both features and concatenating the normalized features. Note that, the weights of the convolutional network (feature extraction) of individual streams are not shared in the Variant I scheme.

Given the pair of normalized features from real and label streams, we now perform label embedding by projecting both these features into a common subspace. We achieve this using the embedding layer as shown in the figure, which is a typical Siamese style network implemented as a multi-layer perceptron. Here, the weights are shared since we want to identify the common subspace where the correlation among similarly paired data is maximized.

5.2 Variant II

Variant II follows an overall architecture that is similar to Variant I; however, we now share the convolutional network for extracting features from the real and synthetic images as shown in Fig. 5b. This allows the features extracted from the synthetic image to be more robust and follows a similar distribution as learned from the real stream. The other obvious advantage is reduction of the number of parameters to be learned. Given the SYNTH features from the real stream, it is concatenated with the PHOC features coming from the label stream, and a fully connected layer is used to fuse both the features after L_2 normalization. The rest of the architecture, which includes the embedding layer, remains the same. We refer to this simpler variant as HWNet v3 in our experiments.

5.3 Training end-to-end network

To train the above variants of the end-to-end network which could exploit features from both the streams, we use a multi-task loss function as given below:

$$\mathcal{L}(\phi_r, \phi_l, y) = \mathcal{L}_1(\phi_r, y) + \mathcal{L}_2(\phi_l, y) + \mathcal{L}_3(\phi_r, \phi_l) \tag{3}$$

Here, ϕ_r, ϕ_l are the embeddings obtained from the real and label streams, respectively, as shown in the figure, while y is the ground truth label represented using one hot representation. The first two components $(\mathcal{L}_1, \mathcal{L}_2)$ of the loss function are cross-entropy-based classification loss functions computed on the soft-max scores for real and the label embeddings, respectively. The third component $(\mathcal{L}_3)$ is a similarity loss function, which is defined using the cosine similarity between the pairs of features belonging to the same label, and is given as:

$$\mathcal{L}_3(\phi_r, \phi_l) = 1 - \cos(\phi_r, \phi_l) \tag{4}$$

The choice of classification loss was done following our experience with training HWNet, which helped in learning efficient word image features useful for word spotting. Secondly, we chose a simpler cosine-based loss function instead of contrastive loss [7], because, in our experiments, we found that such a network is slow to train and the selection of pairs (positive and negative) is extremely crucial for optimum training. Note that using cosine loss, we achieved slightly better performance than contrastive loss; however, we believe

Table 1 The list of datasets used in this work

Dataset	Historical	#Words	#Writers
IAM	No	1,15,320	657
GW	Yes	4894	1∗
Botany	Yes	20,004	1∗
Konzilsprotokolle	Yes	12,993	1∗

Here GW, Botany and Konzilsprotokolle datasets are historical documents, written primarily by a single author and probably along with a few assistants(*)

that with careful selection of hard negative samples, one may achieve better learning using the contrastive loss. In the experiment section, we present a thorough ablation study of the importance of each variant of architecture, each loss function and type of label stream information. More details on the training strategy, such as pre-training, data augmentation and architectural details, will be discussed in Sect. 6.3 as part of the implementation details.

5.4 Image and text representation

Given the trained end-to-end embedding network, to compute embedding for the test images and textual strings, we extract the L_2 normalized activation from the penultimate layer of the network, which in our case are ϕ_r and ϕ_l, respectively. Note that for computing the representation of word images, we use only the real stream and embedding layer, whereas for computing the textual representation, we use label stream and the embedding layer. Note that the synthetic image being rendered in the label stream is of the same font used while training.

6 Experiments

In this section, we present the experiments conducted on the proposed embedding schemes, their variants and validate its effectiveness for the task of word spotting and recognition. We begin this section by mentioning the datasets used in this work, followed by an explanation of the evaluation metrics being chosen. Later we present the ablation study on the proposed variants of the end-to-end embedding scheme and choose the best variant. We then evaluate our two-stage approach and end-to-end embedding schemes with other state-of-the-art methods for the task of word spotting and recognition for both handwritten and printed word images. We also conduct various analyses on embedding size, percentage of training data required and present interesting visualizations on both embedding space and qualitative results.

6.1 Datasets

The IAM Handwriting Database [31]: It contains a total of 1539 handwritten forms written by 657 writers and is categorized as part of a modern collection. The database is labeled at the sentence, line and word levels. We use the official partition for writer independent text line recognition that splits the pages into training, validation and testing sets, which are writer-independent (Table 1).
George Washington (GW) [36]: It contains 20 pages of letters written by George Washington and his associates in 1755 and thereby categorized into a historical collection. The images are annotated at the word level and contain approximately 5000 words. Since there is no official partition, we use a random set (similar to [4]) of 75% for training and validation, and the remaining 25% for testing.
Botany and Konzilsprotokolle [34]: These two datasets are part of ICFHR 2016 Handwritten Keyword Spotting Competition [34]. We considered only the segmentation-based track data which contained cropped word images split into training and test sets. There were also three partitions of training sets: small, medium and large. Here we took only the largest partition, which contains 16,686 training images for Botany and 9102 for Konzilsprotokolle. And the test set contains 3318 word images for Botany and 3891 for Konzilsprotokolle. These two datasets are also categorized under historical document collection.

6.2 Evaluation measures

We evaluate the proposed embedding schemes for the task of word spotting and recognition. In word spotting, we design the protocol similar to [4] and also follow the train-val-test splits given along with each dataset. We perform our evaluation in a case-insensitive manner and also remove stopwords from the test query set; however, stopwords are kept in the retrieval set to act as outliers. For the query-by-example (QBE) setting, we test with each exemplar image from the test query set, while for query-by-string (QBS) scenario, we take the unique strings in the test set as queries. We evaluate word spotting using mean average precision (mAP), which is the standard evaluation method under retrieval problems.

The second major experiment is on word recognition, which is evaluated using the mean character error rate (CER) and mean word error rate (WER). Both CER and WER are computed using the Levenshtein distance between the predicted sequence of characters/words with the actual ground truth. Since in this work, we require word-level segmentation along with lexicon for recognition, our comparisons in this space will be limited to the methods in the literature that operate under this setting. Note that, as per the standard protocol, we remove only the punctuation from the test set while keeping all the other words, including stopwords for evaluation.

6.3 Implementation details

We train our end-to-end network using the stochastic gradient descent algorithm with momentum. We set the momentum factor to be 0.9 and the learning rate as $1e{-}2$ while training from scratch on synthetic data. While fine-tuning, we initialize the learning rate at $1e{-}3$ and reduce it by a factor of 2 once the loss does not change within a certain threshold in the last two epochs. The weights are initialized using He initialization [17]. We perform extensive data augmentation [25] while training the network, which includes elastic distortion, and affine transformations (scaling, translation, rotation and shear). The augmentations are done on-the-fly with a 50% probability of augmenting the current sample from the mini-batch. For elastic distortion, we set the hyper-parameters $\alpha = 0.8$ and $\sigma = 0.08$, denoted as scaling and smoothing parameters [44], respectively, which regulate the amount of distortion. For affine transformation, we randomly pick whether to rotate, shear or pad. The rotation and shear angles are sampled in the range of $(-5, 15)$ and $(-0.5, 0.5)$ degrees, respectively. For bringing translation in-variance, we randomly insert padding in the four boundaries within a range of 0–20 pixels.

In our experiments using PHOC, we extract unigrams at 10 levels, and the bigrams are extracted up to 6 levels. We use NVIDIA GeForce GTX 1080 Ti GPUs for all our experimentations, and the codes are written in PyTorch library.

6.4 Ablation study

In Sect. 5, we presented two variants of the end-to-end embedding network along with the multi-task loss function used while training. In Table 2, we present a detailed ablation study on the IAM dataset over these two variants (column 1), the choice of loss function (column 2) and the choice of modality in each stream (columns 3 and 4). We pick the valid combination of loss functions among the proposed three losses ($\mathcal{L}_1$, $\mathcal{L}_2$, $\mathcal{L}_3$). Note that the valid combination requires at least one loss function from the label stream for achieving common subspace embedding. Similarly, the possible modality through convolution layers of the real stream includes either REAL or both REAL + SYNTH images. The later setting is referred to as Variant II. Similarly, in the label stream, we got three possible settings: SYNTH, PHOC, SYNTH + PHOC. The first row of the table presents the setting with Variant I, training with all loss functions and uses a fusion of SYNTH + PHOC in the label stream. We treat this as our baseline performance where we obtain an mAP of QBE 0.9289 and 0.9650 for QBS, while the CER and WER are in the range of 3.76 and 6.77, respectively. The next three rows (row 2–4) show the results of using different combinations of these loss functions by excluding one loss function at a time. Here we observe that both ($\mathcal{L}_1$, $\mathcal{L}_3$) loss functions are quite important for embedding, while the $\mathcal{L}_2$ loss function acts as redundant and even deteriorates the performance of word recognition (last two columns) as compared to the baseline performance. We also notice the need for $\mathcal{L}_3$ loss since the exclusion of it (row 2) resulted in a huge drop in the performance of word recognition. One might observe a slight inconsistency in terms of the drop in performance among word spotting and recognition. Here, we see that there is only a slight drop in word spotting, while there is a huge drop in CER and WER values. This is because of the word images used for testing in word spotting and recognition. Under word spotting, the standard query set removes the stopwords which are typically shorter in length, while in word recognition, we have kept all the words except the punctuation. In our analysis, we observe that the majority of word recognition errors have occurred for shorter words and stopwords since the embedded features were not discriminative enough. Given the initial set of observations under different loss settings, we pick the combination of $\mathcal{L}_1$, $\mathcal{L}_3$ for our further experimentation.

We now experiment with the choice of modality in each stream under both variants with the fixed loss function. This is depicted from rows 5–8. Here at first, we use the Variant I architecture while keeping only one modality (SYNTH or

Table 2 Ablation study on IAM dataset under two variants (column 1) of end-to-end embedding architectures, the choice of loss functions (column 2) and the choice of modality in each stream (columns 3 and 4)

Variant	Loss	Real stream	Label stream	mAP-QBE	mAP-QBS	CER	WER
Variant I	$\mathcal{L}_1 + \mathcal{L}_2 + \mathcal{L}_3$	REAL	SYNTH + PHOC	0.9289	0.9650	3.76	6.77
	$\mathcal{L}_1 + \mathcal{L}_2$			0.9168	0.9631	31.50	28.71
	$\mathcal{L}_2 + \mathcal{L}_3$			0.9146	0.9289	7.90	12.36
	$\mathcal{L}_1 + \mathcal{L}_3$			**0.9306**	**0.9705**	**1.90**	**4.10**
Variant I	$\mathcal{L}_1 + \mathcal{L}_3$	REAL	SYNTH	**0.9333**	0.9695	1.98	4.18
			PHOC	0.9316	0.9711	1.98	4.31
Variant II		REAL + SYNTH	SYNTH	0.9297	0.9738	1.75	3.84
			SYNTH + PHOC	0.9322	**0.9753**	**1.67**	**3.62**

Bold values refer to the best performing method under each variant (column 1) against each evaluation criteria (last four columns)

PHOC) in the label stream. We observe the performance of the SYNTH stream to be either better or comparable with PHOC under all the tasks. Finally, we present the analysis on Variant II architecture where the real stream is shared among the real and synth images. Here we also evaluate two possible label stream settings which are SYNTH or SYNTH + PHOC. More specifically, the fused representation (SYNTH + PHOC) in the label stream was found to be better than using only one modality. In our further experiments, we refer to this Variant II architecture, trained with the loss function ($\mathcal{L}_1 + \mathcal{L}_3$), with real stream (REAL + SYNTH) and label stream (SYNTH + PHOC) as HWNet v3.

6.5 Word spotting results

Table 3 presents the results of word spotting under various proposed embedding methods, and comparisons are made with other recent state-of-the-art methods. We split the rows of the table into four blocks, where in the first block, we list out the state-of-the-art word spotting methods from literature. The next three blocks present and compare the results from our contributions evolved starting from HWNet architecture and various embedding schemes such as two-stage joint embedding, synthetic attribute embedding and finally to end-2-end embedding scheme. As mentioned in the dataset section, we will be evaluating against the four prominent datasets used in the community.

Most of the methods reported in the table are based on deep neural networks except for the first method KSCR [4], which uses Fisher-based representation for attribute embedding. As one can observe, the introduction of deep features gave a significant push to the performance of all handwritten datasets. Nevertheless, the PHOC-based representation originally proposed in this work inspired many of the deep learning methods [21,33,48] including ours due to its optimal representation of textual strings in the form of attributes. Rows 2–8 present the recent methods in word spotting which uses deep features for representation. Among these, the PHOCNet [48] architecture presented by Sudholt et al. uses PHOC representation as the output space and learns an embedding using a deep convolutional network along with spatial pyramid pooling (SPP) to accommodate variable sized input images. Later in the extension of this work [50], authors modified the SPP architecture to temporal pooling layer (TPP) and evaluated the method under different loss functions, embedding schemes and optimization functions. Here TPP-PHOCNet (BPA) stands for Binary Cross-Entropy Loss, PHOC embedding and Adam optimization (BPA), while TPP-PHOCNet (CPS) is Cosine Loss, PHOC embedding and standard SGD optimization. We report the best-performing methods in our

Table 3 Quantitative evaluation of various word spotting methods on standard handwritten datasets

Method	IAM		GW		Botany		Konzilsprotokolle	
	QBE	QBS	QBE	QBS	QBE	QBS	QBE	QBS
KCSR [4]	0.5573	0.7372	0.9304	0.9129	0.7577	0.6569	0.7791	0.8291
PHOCNet [48]	0.7251	0.8297	0.9671	0.9264	0.8969	0.7447	0.9605	0.9420
Triplet-CNN [55]	0.8158	0.8949	0.9800	0.9369	–	–	–	–
LSDE [12]	–	–	–	0.9131	–	–	–	–
Conv. Siamese Net. [6]	–	–	0.49	–	–	–	–	–
TPP-PHOCNet (BPA) [50]	0.8480	0.9297	0.9790	0.9673	0.9605	0.9738	**0.9811**	**0.9802**
TPP-PHOCNet (CPS) [50]	0.8274	0.9342	0.9796	0.9792	0.8081	0.9015	0.9642	0.9463
PHOCNet (BPA) [50]	0.8550	0.9238	0.9758	0.9558	0.9410	0.9543	0.9708	0.9622
HWNet [24]	0.8061	–	0.9484	–	0.8416	–	0.7913	–
DeepEmbed [21]	0.8425	0.9158	0.9441	0.9284	–	–	–	–
HWNet v2 (ROI) [25]	0.9065	–	0.9601	–	0.9401	–	0.9427	–
DeepEmbed (ROI) [22]	0.9038	0.9404	0.9801	0.9886	0.9546	0.9717	0.9411	0.9065
Synth + DeepEmbed (ROI) [22]	–	0.9509	–	0.9898	–	0.9718	–	0.9143
End-2-End Embed (ROI) [22]	0.9157	0.9621	0.9839	0.9894	0.9554	0.9372	0.9316	0.7105
HWNet v2 (TPP) [25]	0.9241	–	0.9824	–	0.9526	–	0.9347	–
DeepEmbed (TPP)	0.9180	0.9463	0.9848	0.9922	0.9617	**0.9804**	0.9435	0.8896
Synth + DeepEmbed (TPP)	–	0.9614	–	0.9922	–	**0.9804**	–	0.9024
HWNet v3	**0.9322**	**0.9753**	**0.9948**	**0.9980**	**0.9713**	0.9777	0.9624	0.9364

Here, the first block (rows 1–7) presents results from methods proposed in the literature. Note that in Triplet-CNN [55] we have taken the best results for IAM and GW across different word embedding used in the paper. The next three blocks of methods (rows 8–9, 10–13 and 14–17) present the different embedding schemes proposed in this chapter. Note, Conv. Siamese Net. [6] uses a different test set from GW, which is built using out-of-vocabulary words and is not directly comparable. Bold values refer to the best performing method for each dataset in QBE and QBS setting

table among all the ablation studies conducted in [50]. As one can observe, the TPP-PHOCNet (BPA) generally performs the best among other variations of the TPP-PHOCNet. Another significant architecture is the Triplet-CNN proposed by Wilkinson et al. [55] that uses a similar scheme as ours with a triplet CNN architecture for image embedding and a separate word embedding network for textual embedding. Given both image and text embedding, a shared fully connected 2-layer embedding network is used for common subspace embedding. The work uses a SoftPN loss function for image embedding and evaluates different textual embedding such as PHOC, DCToW, *n*-gram and semantic. In our table, we have taken the best results for IAM and GW across different textual word embeddings used in their paper. In comparison with PHOCNet variants, Triplet CNN performs slightly inferior on the IAM dataset and, however, report a good performance on GW dataset. Note that while training for the GW dataset, Triplet-CNN used an external real world CVL database for pre-training the network, while the other methods do not use any external real data. More recently, Gomez et al. [12] proposed a novel embedding scheme (LSDE), which projects both word images and the corresponding string into a space that respects the Levenshtein distance. The paper only evaluates the GW dataset and is slightly inferior to other methods in this space. The convolutional siamese network [6] uses a siamese network for learning the embedding, which is similar to the embedding stage of our end-2-end network. However, the method uses a different protocol to evaluate the GW dataset wherein only the out-of-vocabulary words are kept in the test set. Note that this is not directly comparable with the other set of methods in this table.

We now sequentially bring the embedding schemes introduced in this work and compare them against the existing methods from the literature. In the second block (rows 8-9), we first report the performance of our baseline HWNet architecture which was proposed in [24]. The next row DeepEmbed [21] reports the performance of the two-stage joint feature embedding scheme on top of the original HWNet features. Note that we refer to the two-stage embedding scheme as DeepEmbed since we introduced this notation in our previous work [21]. As one can observe, the joint embedding scheme enables both QBE and QBS-based word spotting. It also improves the QBE performance. In the next block (rows 10–13) we first demonstrate the effectiveness of HWNet v2 (using ROI pooling) features and the corresponding DeepEmbed scheme, which utilizes these features. Note that at this stage, we report the state-of-the-art results for IAM and GW datasets across other methods proposed in the literature. The enhancement of DeepEmbed features using synthetic modality as presented in Sect. 4.2 shows a minor improvement for all the datasets under the QBS setting. Note that the QBE values remain the same for the Synth + DeepEmbed setting. The last row in the third block (row 13) presents the end-to-end embedding architecture Variant I, which was introduced in [22]. In comparison with DeepEmbed and Synth + DeepEmbed, the end-to-end Variant I result shows improvements in the IAM and GW datasets; however, for Botany, it is just comparable and for Konzilsprotokolle it is quite inferior under the QBS setting. We found that while training for the Konzilsprotokolle dataset in [22] we used an English font instead of using a German one which caused nonoptimal learning of the label stream. The choice of font was rectified in our later experiments.

In the final block (rows 14–17), we present the extended results while using an improved HWNet v2 architecture with TPP layer and the Variant II of end-to-end embedding scheme referred to as HWNet v3. One can notice that using HWNet v2 (TPP) has improved the overall results for word spotting under different embedding schemes. The HWNet v3 (Variant II) end-to-end architecture gives the best performance on IAM, GW and Botany datasets, among other methods in the literature. Considering the Konzilsprotokolle dataset, the QBE performance of HWNet v3 (Variant II) end-2-end architecture is comparable to that of the TPP-PHOCNet architecture, while the QBS performance is slightly inferior to that of the TPP-PHOCNet architecture.

6.6 Word recognition results

In this experiment, we evaluate the proposed embedding schemes for the task of word recognition. Note that, the current framework only supports recognition through lexicon; thereby, we compare our method with other methods in the literature that follow this convention. Table 4 presents the quantitative results on lexicon-based word recognition on the IAM dataset under three different lexicon settings: (i) the text lexicon which contains all words from the evaluation or test set, (ii) the train + test lexicon includes all the unique words from the training set along with the test set and (iii) a large lexicon setting that contains 90K words taken from the Hunspell dictionary. The large lexicon also includes train+test unique words. In general, the performance gets better with a reduced lexicon size provided there are no out-of-vocabulary words. This trend is visible where the error rates for test lexicon setting are low compared to train + test lexicon scenario.

One of the classical methods in this space is again from Almazán el al. [4] that represents a word image in terms of learned attribute representation constructed from Fisher features. One of the first extensions of attribute-based representation using deep features is proposed by Poznanski et al. [33] that uses a shared set of convolutional layers and multiple fully connected layers toward the end to predict PHOC attributes at various spatial levels. Most of the recent related works in word recognition uses either BLSTMs [15], encoder–decoder style architectures, or CNN-RNN hybrid

Table 4 Word recognition results on the IAM dataset under different settings of lexicon-based evaluation for making the predictions

Method	Lexicon	WER	CER
Almazán et al. [4]	Test	20.01	11.27
Sueiras et al. [51]		12.7	6.2
Wigington et al. [54]		4.97	2.82
CRNN-STN$_{synth}$ [22]		5.10	2.66
Dutta et. al. [10]		4.07	2.17
DeepEmbed (HWNet) [21]		6.69	3.72
DeepEmbed (HWNet v2) [22]		5.46	3.00
HWNet v3		**3.62**	**1.67**
Sun et al. [52]	Train + Test	11.51	–
Wigington et al. [54]		5.71	3.03
Stuner et al. [47]		5.93	2.78
Poznanski et al. [33]		6.45	3.44
Dutta et. al. [10]		4.80	2.52
HWNet v3		**4.59**	**2.09**
HWNet v3	Large (90K)	7.80	3.42

Bold values refer to the best performing method in WER and CER criteria against different Lexicon settings

style architectures [43]. In [51], the authors propose an encoder–decoder style architecture with CNNs for feature extraction from input patches of the word image, while methods such as [10,22,54] use a CNN-RNN style architecture which comprises of a set of convolutional layers for feature extraction and RNN (more specifically LSTM/BLSTM) layers for character prediction through a CTC [14] loss function. Wigington et al. [54] use novel pre-processing and data augmentation schemes for better learning. CRNN-STN$_{synth}$ uses an additional spatial transformer layer [18] for making the network invariant to affine transformation. Later the authors extended their work in [10] by explicitly performing input pre-processing (image slant and slope correction), along with data augmentation techniques to train a better model. Sun et al. [52] use a convolutional multi-directional recurrent network to capture context from all the three directions (top, bottom and diagonal) during prediction, while Stuner et al. [47] use a cascade of LSTMs classifiers along with a lexicon verification operator to increase the reliability of word predictions. The table reports the performance of all these methods under different lexicon settings.

In test-based lexicon setting, we progressively observe an improvement over the proposed two-stage embedding schemes (DeepEmbed) using HWNet and HWNet v2 features; however, they are slightly inferior to other existing methods in this space. The performance of the proposed Variant II architecture, referred to as HWNet v3, is better than all the related methods in this space with a considerable margin. Here we report a CER and WER values of 1.67 and 3.62, respectively. A similar trend is observed in train+test lexicon setting where the HWNet v3 reports the best performance with a CER and WER values of 2.09 and 4.59, respectively. Note that these values are even better than most of the reported methods in the test lexicon setting. Finally, to observe the robustness of the HWNet v3-based word recognition under a large lexicon setting that imparts a huge amount of distractors, we extend our lexicon to contain 90K words from an external dictionary. We observe that under such a large lexicon, our method works fairly enough by giving a CER and WER value of 3.42 and 7.80, respectively. The last experiment also validates the robustness of these embeddings for recognition and opens an interesting direction in the future for lexicon-free word recognition using such representation schemes.

6.7 Analysis of embedding size

Given the choice of varying the embedding sizes or dimensions for the embedding layer in end-to-end variants, we now present the analysis on reducing its size by the power of 2 from the default embedding size of 2048. Note that from now onward, we will be using HWNet v3 representation and its corresponding architecture for our analysis. Figure 6 presents these analyses on the IAM dataset for the word spotting (top-part) and recognition (bottom-part) tasks. In both

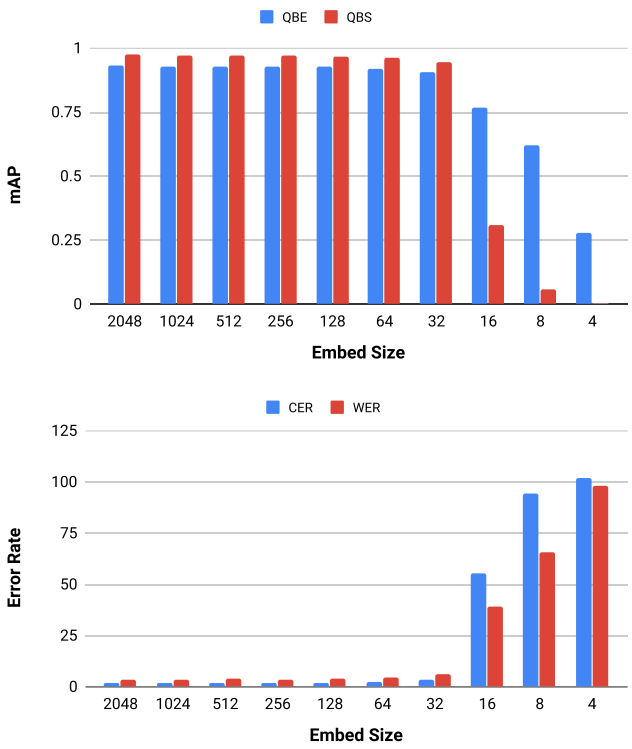

Fig. 6 Effect on performance after varying the HWNet v3 embedding size on the IAM dataset. The top figure shows analysis of word spotting under both QBE and QBS setting. The bottom figure presents the word recognition CER and WER values

Table 5 Evaluation of word spotting and recognition on the IAM dataset by learning HWNet v3 representation with the entire synthetic dataset while fine-tuning varying percentages of the IAM training data

Train %	100%	80%	60%	40%	20%	10%	0%
mAP-QBE	0.9322	0.9247	0.9173	0.8979	0.8837	0.8431	0.5411
mAP-QBS	0.9753	0.9703	0.9666	0.9427	0.9232	0.8614	0.6918
CER	1.67	1.89	2.04	3.36	4.01	6.65	26.40
WER	3.62	4.03	4.40	6.67	7.87	12.11	37.87

Here Train = 0% refers only to using synthetic data for training

the experiments, we vary the embedding size in the range of (2048-4). As one can observe the drop in performance is quite negligible in the range of (2048-32), while below 32 dimensions, there is a drop in the performance. This pattern is the same for both word spotting and recognition and also marks the similarity with the original HWNet representation as presented in [25]. Using mere 32 dimensions, we report an mAP of 0.9047 and 0.9438 for QBE and QBS, respectively, while CER, WER values are 3.61 and 5.96, respectively. Having a lower-dimensional robust representation helps in both the tasks to build compact search indexes and perform faster retrieval.

6.8 Effect of pre-training

In the implementation section, we mentioned the process of pre-training the network using synthetic data. In Table 5, we present an interesting outcome of pre-training which resulted in the reduced need for real training data. Here we experiment with the reduced need for real data for learning HWNet v3 representation by varying the amount of training data as compared to the previous experiments. Here we take the IAM dataset as our test bench and use different proportions of IAM real data for training and compare the performance with the architecture that uses full training data. Here, full is depicted as 100%, while 0% depicts the use of only synthetic data. Note that all these experiments are first trained on the entire synthetic data and later fine-tuned using varying real data proportions. As one can notice, the drop in performance with the reduced real training data starts very slowly and surprisingly, using a mere 20% of real data only drops the performance by around 5% in word spotting and by around 4% in WER. Although this suggests a lesser dependency on real data, we believe that this needs a thorough study (out of the scope of the current work) to evaluate the differences in domain gap between synthetic data and the target handwritten styles.

7 Applicability for printed documents

To validate the performance of the proposed representation on a different modality such as printed documents, we tested our architecture on a standard book in English and a few challenging books taken from DLI corpus [5] in Hindi and Telugu languages.

Table 6 The list of printed datasets used in this work. Here both Hindi and Telugu datasets are taken from the Digital Library of India [5] corpus

Dataset	#Pages	#Words
English-1601	310	1,13,008
DLI Hindi (HS1)	1533	4,20,100
DLI Telugu (TS1)	1005	1,61,265

English-1601 [56]: The dataset contains pages from the book in English titled "Adventures of Sherlock Holmes" written by Arthur Conan Doyle. It was first used in [56] for comparing OCR-based results with image search. (Table 6).

DLI Hindi and Telugu [26]: These two datasets, in Hindi and Telugu languages from Indic scripts, are part of the Digital Library of India (DLI) [5] project. DLI has one of the largest collections of document images in Indian scripts. Many of the pages that present in DLI contain serious forms of document degradation which restricts the present-day OCR's and text spotting systems from working efficiently. We take one such subset [26] which was annotated at the level of lines and words and referred to as HS1 and TS1 datasets.

Table 7 presents the quantitative results of word spotting on the printed datasets. Here are the methods proposed by Yalniz et al. [56] and Krishnan et al. [26] that uses an unsupervised bag of word framework for deriving a holistic word-level descriptor. These methods are not directly comparable with HWNet v2 and HWNet v3 which are learned in a supervised fashion. However, such comparisons highlight the superiority of the supervised methods. As reported in the table, we obtain better results using the HWNet v3 representation on these datasets. Note that the previous methods have not reported query-by-string results since those methods did not embed textual strings into the representation space.

8 Visualization and qualitative results

Figure 7 presents the t-SNE embedding of the HWNet v3 representation of IAM word images taken from its validation corpus. For visualization, we embed these representations

Table 7 Quantitative evaluation of word spotting on printed datasets

Method	Supervision	English mAP-QBE	English mAP-QBS	Hindi mAP-QBE	Hindi mAP-QBS	Telugu mAP-QBE	Telugu mAP-QBS
Yalniz et.al [56]	No	0.9300	–	–	–	–	–
Krishnan et.al [26]	No	–	–	0.6055	–	0.7438	–
HWNet v2 (TPP)	Yes	0.9570	–	0.9509	–	**0.9582**	–
HWNet v3	Yes	**0.9939**	**0.9921**	**0.9537**	**0.9659**	0.9499	**0.9474**

Bold values refer to the best performing method for each dataset in QBE and QBS setting

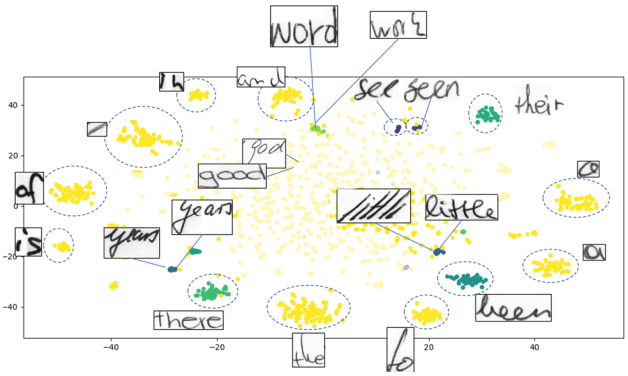

Fig. 7 tSNE image embedding

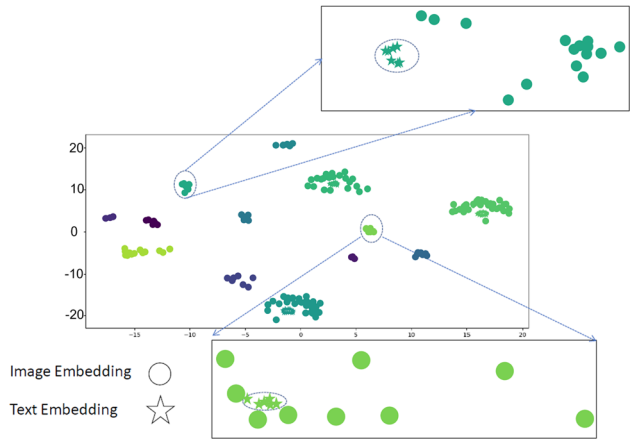

Fig. 8 tSNE common subspace embedding

onto the 2D space. The top frequent words in the dataset are usually the stopwords and the t-SNE embedding usually projects the representations corresponding to these words onto the outer periphery of the reduced space. In the figure, one can observe these clusters marked inside ellipses. The inner region contains the rest of the word images which have fewer occurrences in the corpus. Here we have zoomed out few instances along with their word image. Some of the examples are little, years, god, good, etc. To observe the common space embedding, we take a small snapshot of the reduced space and project both the image and their corresponding textual representation as shown in Fig. 8. To reduce the confusion, we have only plotted 13 unique words and their instances in the dataset. The image embedding vectors are depicted in a "circle," while the text embeddings are shown in a "star" shape. We observe nice clustering among different classes of word images, and we also see that within a particular cluster, the image and its corresponding text embedding lie close to each other.

Figure 9 presents the qualitative results of word spotting across all the datasets in the query-by-string setting. The query is shown in the left part of the column, while on the right, we arrange the ranked results obtained for the query. Notice the variation of styles among different writers, which the representation can handle.

Figure 10 presents the qualitative results of word recognition on the IAM dataset using the test lexicon. The top two rows show the successful cases, while the bottom row shows the failure scenarios. Here the green box shows the text where the prediction matches the actual text. In the failure cases, the red box shows the incorrect recognition, while the blue box shows the correct answer. Notice the robustness of the proposed representation to recognize some hard words such as "shuffle," "gilberto," while some cases, such as "rose," "took" resulted in incorrect recognition as they were quite ambiguous in the image space.

9 Conclusion and future works

In this work, we presented two major types of label embedding techniques for word images using deep architecture and their representations. This includes the two-stage embedding framework and an end-to-end deep neural architecture that embeds both word images and their corresponding textual strings into a common subspace. We also extensively validate the role of synthetic images for pre-training and also utilize this modality for learning better representation of the textual data. The proposed end-to-end framework and its representation referred to as HWNet v3 reports state-of-the-art performance for both word spotting and recognition tasks. These are evaluated for standard handwritten, historical and printed document datasets in both Latin and non-Latin scripts. We also bring out a detailed ablation study on dif-

Fig. 9 Qualitative results of query-by-string word spotting on **a** IAM, **b** GW, **c** Botany, **d** Konzilsprotokolle, **e** Hindi and **f** Telugu dataset, respectively. The query is shown in the left most column, and the word images shown on the right are the retrieved images in the ranked order

Fig. 10 Qualitative results of word recognition on the IAM dataset using the test lexicon. The top two rows show the successful cases, while the bottom row shows the failure scenarios. Here the green box shows the text where the prediction matches the actual text. In the failure cases, the red box shows the incorrect recognition, while the blue box shows the correct answer

ferent variants of the end-2-end embedding architecture and perform analysis of varying embedding sizes.

As a future plan, we would like to attempt the problem of unconstrained word recognition by learning representation at the level of character n-grams of a word. Another interesting direction would be to generate synthetic data on the principles of recent generative models such as GANs [13] which might bring more realism and variation to synthetic data.

References

1. Aldavert, D., Rusiñol, M., Toledo, R., Lladós, J.: Integrating visual and textual cues for query-by-string word spotting. In: International Conference on Document Analysis and Recognition, ICDAR, pp. 511–515 (2013)
2. Aldavert, D., Rusiñol, M., Toledo, R., Lladós, J.: A study of bag-of-visual-words representations for handwritten keyword spotting. Int. J. Doc. Anal. Recognit. **18**(3), 223–234 (2015)
3. Almazán, J., Gordo, A., Fornés, A., Valveny, E.: Segmentation-free word spotting with exemplar SVMs. Pattern Recognit. **47**(12), 3967–3978 (2014)
4. Almazán, J., Gordo, A., Fornés, A., Valveny, E.: Word spotting and recognition with embedded attributes. IEEE Trans. Pattern Anal. Mach. Intell. **36**(12), 2552–2566 (2014)
5. Ambati, V., Balakrishnan, N., Reddy, R., Pratha, L., Jawahar, C.: The digital library of India project: process, policies and architecture. In: International Conference on Digital Libraries, ICDL (2006)
6. Barakat, B.K., Alasam, R., El-Sana, J.: Word spotting using convolutional siamese network. In: 2018 13th IAPR International Workshop on Document Analysis Systems (DAS), pp. 229–234. IEEE (2018)
7. Chopra, S., Hadsell, R., LeCun, Y.: Learning a similarity metric discriminatively, with application to face verification. In: Computer Vision and Pattern Recognition, CVPR, pp. 539–546 (2005)
8. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: Computer Vision and Pattern Recognition, CVPR, pp. 886–893 (2005)
9. Deerwester, S.C., Dumais, S.T., Landauer, T.K., Furnas, G.W., Harshman, R.A.: Indexing by latent semantic analysis. J. Am. Soc. Inf. Sci. **41**(6), 391–407 (1990)

10. Dutta, K., Krishnan, P., Mathew, M., Jawahar, C.V.: Improving CNN-RNN hybrid networks for handwriting recognition. In: International Conference on Frontiers in Handwriting Recognition, ICFHR, pp. 80–85 (2018)
11. Giotis, A.P., Sfikas, G., Gatos, B., Nikou, C.: A survey of document image word spotting techniques. Pattern Recognit. **68**, 310–332 (2017)
12. Gomez-Bigorda, L., Rusiñol, M., Karatzas, D.: LSDE: Levenshtein space deep embedding for query-by-string word spotting. In: International Conference on Document Analysis and Recognition, ICDAR, pp. 499–504 (2017)
13. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Advances in Neural Information Processing Systems, NIPS, pp. 2672–2680 (2014)
14. Graves, A., Fernández, S., Gomez, F.J., Schmidhuber, J.: Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In: Proceedings of the Twenty-Third International Conference, ICML, vol. 148, pp. 369–376 (2006)
15. Graves, A., Schmidhuber, J.: Framewise phoneme classification with bidirectional LSTM and other neural network architectures. Neural Netw. **18**(5–6), 602–610 (2005)
16. Harris, C.G., Stephens, M.: A combined corner and edge detector. In: Taylor, C.J. (ed.) Alvey Vision Conference, AVC, pp. 1–6 (1988)
17. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: surpassing human-level performance on imagenet classification. In: IEEE International Conference on Computer Vision, ICCV, pp. 1026–1034 (2015)
18. Jaderberg, M., Simonyan, K., Zisserman, A., Kavukcuoglu, K.: Spatial transformer networks. In: Advances in Neural Information Processing Systems, NIPS, pp. 2017–2025 (2015)
19. Jégou, H., Douze, M., Schmid, C.: Product quantization for nearest neighbor search. IEEE Trans. Pattern Anal. Mach. Intell. **33**(1), 117–128 (2011)
20. Kim, Y., Jernite, Y., Sontag, D.A., Rush, A.M.: Character-aware neural language models. In: Schuurmans, D., Wellman, M.P. (eds.) Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, pp. 2741–2749 (2016)
21. Krishnan, P., Dutta, K., Jawahar, C.V.: Deep feature embedding for accurate recognition and retrieval of handwritten text. In: International Conference on Frontiers in Handwriting Recognition, ICFHR, pp. 289–294 (2016)
22. Krishnan, P., Dutta, K., Jawahar, C.V.: Word spotting and recognition using deep embedding. In: IAPR International Workshop on Document Analysis Systems, DAS, pp. 1–6 (2018)
23. Krishnan, P., Jawahar, C.V.: Generating synthetic data for text recognition (2016). CoRR arXiv:1608.04224
24. Krishnan, P., Jawahar, C.V.: Matching handwritten document images. In: European Conference on Computer Vision, ECCV, vol. 9905, 766–782 (2016)
25. Krishnan, P., Jawahar, C.V.: Hwnet v2: an efficient word image representation for handwritten documents. Int. J. Doc. Anal. Recognit. IJDAR **22**(4), 387–405 (2019)
26. Krishnan, P., Shekhar, R., Jawahar, C.V.: Content level access to digital library of India pages. In: Triggs, B., Bala, K., Chandran, S. (eds.) Indian Conference on Vision, Graphics and Image Processing, ICVGIP, p. 5 (2012)
27. Kumar, A., Jawahar, C.V., Manmatha, R.: Efficient search in document image collections. In: Asian Conference on Computer Vision, ACCV, vol. 4843, pp. 586–595 (2007)
28. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. IJCV **60**(2), 91–110 (2004)
29. Manmatha, R., Han, C., Riseman, E.M.: Word spotting: a new approach to indexing handwriting. In: Computer Vision and Pattern Recognition, CVPR, pp. 631–637 (1996)
30. Marti, U., Bunke, H.: Using a statistical language model to improve the performance of an hmm-based cursive handwriting recognition system. Int. J. Pattern Recognit. Artif. Intell. **15**(1), 65–90 (2001)
31. Marti, U., Bunke, H.: The IAM-database: an English sentence database for offline handwriting recognition. Int. J. Doc. Anal. Recognit. **5**(1), 39–46 (2002)
32. Myers, C., Rabiner, L., Rosenberg, A.: Performance tradeoffs in dynamic time warping algorithms for isolated word recognition. IEEE Trans. Acoust. Speech Signal Process. **28**(6), 623–635 (1980)
33. Poznanski, A., Wolf, L.: CNN-N-gram for handwriting word recognition. In: Computer Vision and Pattern Recognition, CVPR, pp. 2305–2314 (2016)
34. Pratikakis, I., Zagoris, K., Gatos, B., Puigcerver, J., Toselli, A.H., Vidal, E.: ICFHR2016 handwritten keyword spotting competition (H-KWS 2016). In: International Conference on Frontiers in Handwriting Recognition, ICFHR, pp. 613–618 (2016)
35. Rath, T.M., Manmatha, R.: Word image matching using dynamic time warping. In: Computer Vision and Pattern Recognition, CVPR, pp. 521–527 (2003)
36. Rath, T.M., Manmatha, R.: Word spotting for historical documents. Int. J. Doc. Anal. Recognit. **9**(2–4), 139–152 (2007)
37. Rodriguez, J.A., Perronnin, F.: Local gradient histogram features for word spotting in unconstrained handwritten documents. In: International Conference on Frontiers in Handwriting Recognition, ICFHR, pp. 7–12 (2008)
38. Rodriguez-Serrano, J.A., Gordo, A., Perronnin, F.: Label embedding: a frugal baseline for text recognition. Int. J. Comput. Vis. **113**(3), 193–207 (2015)
39. Rosten, E., Drummond, T.: Machine learning for high-speed corner detection. In: European Conference on Computer, ECCV, vol. 3951, pp. 430–443 (2006)
40. Rusiñol, M., Aldavert, D., Toledo, R., Lladós, J.: Efficient segmentation-free keyword spotting in historical document collections. Pattern Recognit. **48**(2), 545–555 (2015)
41. Sakoe, H., Chiba, S.: Dynamic programming algorithm optimization for spoken word recognition. IEEE Trans. Acoust. Speech Signal Process. **26**(1), 43–49 (1978)
42. Shekhar, R., Jawahar, C.V.: Word image retrieval using bag of visual words. In: IAPR International Workshop on Document Analysis Systems, DAS, pp. 297–301 (2012)
43. Shi, B., Bai, X., Yao, C.: 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. **39**(11), 2298–2304 (2016)
44. Simard, P.Y., Steinkraus, D., Platt, J.C.: Best practices for convolutional neural networks applied to visual document analysis. In: International Conference on Document Analysis and Recognition, ICDAR, pp. 958–962 (2003)
45. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: International Conference on Learning Representations, ICLR (2015)
46. Sivic, J., Zisserman, A.: Video google: a text retrieval approach to object matching in videos. In: IEEE International Conference on Computer Vision, ICCV, pp. 1470–1477 (2003)
47. Stuner, B., Chatelain, C., Paquet, T.: Cohort of LSTM and lexicon verification for handwriting recognition with gigantic lexicon (2016). CoRR arXiv:1612.07528
48. Sudholt, S., Fink, G.A.: PHOCNet: a deep convolutional neural network for word spotting in handwritten documents. In: International Conference on Frontiers in Handwriting Recognition, ICFHR, pp. 277–282 (2016)
49. Sudholt, S., Fink, G.A.: Evaluating word string embeddings and loss functions for CNN-based word spotting. In: International Conference on Document Analysis and Recognition, ICDAR, pp. 493–498 (2017)

50. Sudholt, S., Fink, G.A.: Attribute CNNs for word spotting in handwritten documents. Int. J. Doc. Anal. Recognit. **21**(3), 199–218 (2018)
51. Sueiras, J., Ruíz, V., Sánchez, Á., Vélez, J.F.: Offline continuous handwriting recognition using sequence to sequence neural networks. Neurocomputing **289**, 119–128 (2018)
52. Sun, Z., Jin, L., Xie, Z., Feng, Z., Zhang, S.: Convolutional multi-directional recurrent network for offline handwritten text recognition. In: International Conference on Frontiers in Handwriting Recognition, ICFHR, pp. 240–245 (2016)
53. Terasawa, K., Tanaka, Y.: Slit style HOG feature for document image word spotting. In: International Conference on Document Analysis and Recognition, ICDAR, pp. 116–120 (2009)
54. Wigington, C., Stewart, S., Davis, B.L., Barrett, B., Price, B.L., Cohen, S.: Data augmentation for recognition of handwritten words and lines using a CNN-LSTM network. In: IAPR International Conference on Document Analysis and Recognition, ICDAR, pp. 639–645 (2017)
55. Wilkinson, T., Brun, A.: Semantic and verbatim word spotting using deep neural networks. In: International Conference on Frontiers in Handwriting Recognition, ICFHR, pp. 307–312 (2016)
56. Yalniz, I.Z., Manmatha, R.: An efficient framework for searching text in noisy document images. In: IAPR International Workshop on Document Analysis Systems, DAS, pp. 48–52 (2012)