From Seq2Seq Recognition to Handwritten Word Embeddings

George Retsinas[1]
gretsinas@central.ntua.gr
Giorgos Sfikas[2]
sfikas@cs.uoi.gr
Christophoros Nikou[2]
cnikou@cs.uoi.gr
Petros Maragos[1]
maragos@cs.ntua.gr

[1] School of Electrical and Computer Engineering
National Technical University of Athens, Greece
[2] Department of Computer Science and Engineering
University of Ioannina, Greece

Abstract

In this work, we propose a system for automatically extracting handwritten word embeddings, using the encoding module of a Sequence-to-Sequence (Seq2Seq) recognition network. These embeddings are proven to be very discriminative, since they can be effectively used for Keyword Spotting, while they can also be fully decoded into the target string following the Seq2Seq rationale. Architecture-wise, the proposed system incorporates several novel modules (e.g. auto-encoder path or non-recurrent CTC-branch) that assist the training procedure and boost performance. Additionally, we also show how to further process these embeddings/representations with a binarization scheme to provide compact and highly efficient descriptors, suitable for Keyword Spotting. Numerical results validate the usefulness of the proposed architecture, as our method provides state-of-the-art results for both recognition and spotting.

1 Introduction

Handwritten Text Recognition (HTR) and Keyword Spotting (KWS) are two tasks of central importance in the literature of document image processing. HTR deals with automatic transcription of segmented lines of text [7] or isolated words [26, 35, 36], while in Keyword Spotting the goal is to detect instances of specific keyword in a given digitized document. Keyword spotting may be desirable over full text recognition, especially in cases where recognition of the whole text is unnecessary or would likely be suboptimal [9, 31]. The two problems are closely connected, and both have their analogous counterparts in speech and audiovisual signal processing [13, 24, 25]. Yet, very often they are faced with different families of techniques. As is the case with most, if not all, tasks in computer vision, both HTR and KWS are today dominated by neural network-based methods. In recognition, recurrent neural networks have become the baseline [21, 29], as they naturally fit with the sequential nature of handwriting, and especially after the introduction of a number of key elements to the standard recurrent network paradigm [6, 7]. Such key advances include the Long

Short-Term Memory model (LSTM) [12], that effectively dealt with the vanishing gradient problem, and the Connectionist Temporal Classification (CTC) method and corresponding output layer [10, 11], which allow simultaneous sequence alignment and recognition with a suitable decoding scheme. Research on decoding schemes is also active [4], with the beam search algorithm being a popular approach, capable of exploiting language models.

Regarding KWS, a number of recent deep learning methods have been inspired by the attribute-based model of Almazán et al. [1]. In this model, character-level attributes are learned as a Pyramidal-Histogram-of-Characters fixed-size vector (PHOC) and projected along with string representations to a common subspace, allowing Query-by-Example and Query-by-String word-level KWS. This base model has been further extended or adapted [16, 33, 38] using convolutional neural networks to replace the whole or part of the pipeline. These methods have been also used for word recognition, albeit lexicon-based, where the rationale is to compare the common image attribute / string representations and return the closest match in the lexicon. The attribute-based PHOC representation has been shown to be decodable without the use of a lexicon, with some limited success [36]. The Sequence-to-Sequence architecture, a component of the proposed method, has led to state-of-the-art results in Natural Language Processing [41], involving translating an input sequence to an output sequence of a different length in general. The use of the Seq2Seq architecture has recently been used in HTR and KWS as well [40, 44], with notable results.

In this work, we present a Deep Neural Network (DNN) architecture that can tackle both HTR and KWS, with the latter being in the spotlight of this paper. The main contribution of this work is the extraction of discriminative holistic representations of handwritten words, suitable for the KWS task. This is achieved through a Sequence-to-Sequence architecture, where the encoder sub-module generates a fixed-size embedding of the initial image. Contrary to typical KWS approaches, which use a handcrafted attribute-based representation as target, we translate input images into embeddings via *training a Seq2Seq based recognition system*. Moreover, the generated embedding can be fully translated into a character sequence through the decoder submodule, as opposed to attribute-based word representations, such as PHOC [1, 38]. To accelerate Seq2Seq training, we propose an extra non-recurrent CTC-based component, which acts in parallel to the Seq2Seq component and on top of a convolutional backbone. Concerning KWS, we distinguish two categories: 1) Query-by-Example (QbE), which can be addressed by simply comparing feature vectors, or 2) Query-by-String (QbS), which can be implemented either by employing an extra encoder module that translates query strings to the Seq2Seq intermediate representation space, or by forced aligning the query to the decoder. Furthermore, we show that the Seq2Seq-based representation can be refined by binarizing it with an efficient Straight-Through Estimator-based (STE) retraining scheme [2]. This binary representation, aside from being very compact and economical in terms of space, also allows for very fast KWS.

The remainder of this paper is organized as follows. In section 2 we present the proposed architecture and outline its use for word recognition. In section 3 we discuss how to use the proposed architecture for QbE and QbS Keyword Spotting. Finally, we present numerical experiments on both tasks in section 4 and conclude the paper with section 5.

2 Proposed Architecture and Word Recognition

As stated in the introduction, the main novelty of this work is the extraction of fixed-sized embeddings by training a recognition architecture. In this section, we describe in detail the proposed architecture for word recognition, while its subsequent extension to handle KWS

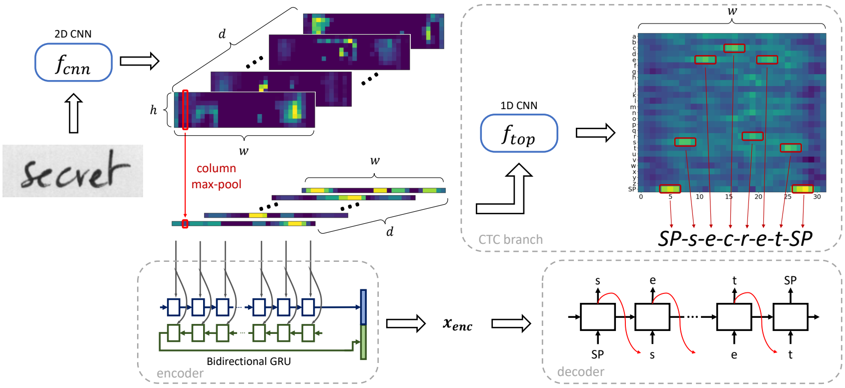


Figure 1: Proposed architecture. The word image is processed through a 2D convolutional backbone, leading to $w \times d$-sized feature map. The model then branches to two components, a CTC branch and a Seq2Seq (encoder/decoder) branch, combined through a multi-task loss.

will be described in the upcoming section.

Our architecture comprises three basic components, namely a *convolutional backbone* which feeds a *CTC-based branch* and a *Seq2Seq branch*. The Seq2Seq module [41], i.e. a Encoder/Decoder Recurrent network pair, which takes as input the output of the aforementioned CNN module, encodes the word information into a fixed-sized vector and consequently decodes it into a sequence of characters. The simultaneous use of these branches was motivated by the difficulty of training a Seq2Seq architecture, i.e. slow convergence, compared to CTC-based one. The proposed model architecture can be examined in Figure 1. In what follows, we describe our architecture modules and their functionality in detail.

Convolutional backbone: The visual feature extraction task is performed by a CNN, dubbed as the convolutional backbone. This part of the network produces a feature map based on a word image input, that will be subsequently processed by two "heads", the CTC and Seq2Seq branches. A total of four convolutional stacks, separated by max pooling operations, are used, where each stack consists of multiple residual blocks topped by ReLU (Rectified Linear Unit) non-linearities preceded by Batch Normalization (BN) and Dropout layers. To further promote simplicity we transform the output of the CNN backbone of size $h \times w \times d$ into a feature sequence of size $w \times d$ by column-wise max-pooling (see Fig. 1). The reasoning behind max-pooling is that we care only about the existence of features related to a character and not their spatial position [42].

CTC component: Contrary to the majority of existing CTC-based approaches, the proposed CTC branch does not comprise any recurrent layers. Instead, a stack of three 1D convolutional layers with kernel size equal to 7, along with BN, ReLU and Dropout are used. Note that multiple 1D convolutional layers are capable of encoding context-wise information, which is the major goal of recurrent networks. The gain of replacing recurrent networks with 1D CNNs is two-fold: First, LSTMs are known to exhibit convergence difficulties, while CNNs with BN can converge very fast. Second, convolutions can be fully parallelized and thus considerably improve training and inference time, as opposed to LSTMs. The output of the 1D CNN is of size $w \times n_{classes}$, where $n_{classes}$ is the number of possible character classes. Applying the softmax function on the final output, we form a sequence of probability distributions over the possible characters which is then propagated into the CTC loss L_{CTC}. Given the trained system and an input image, the recognized character sequence can be generated by a CTC decoding procedure [10]. Note that we strive for simplicity for this component, since its aim is to assist the training of the Seq2Seq branch. Specifically, overall convergence is assisted by quickly generating discriminative features at the top of the CNN backbone, simplifying the Seq2Seq task. Due to its training-oriented assisting nature, CTC head is omitted during evaluation.

Sequence-to-Sequence Component: The main branch of the proposed network architecture involves a Sequence-to-Sequence component [41]. It consists of two recurrent neural networks, the *encoder*, which projects a sequence into a fixed-sized vector, and the *decoder*, which is responsible for decoding the encoded fixed-sized feature vector into the target sequence. The encoder network generates the fixed-sized vector by extracting the last hidden vector of the recurrent operation as a holistic representation. The decoder network, given a hidden vector and the previous element of the sequence, predicts the next element. Concerning the problem of Handwriting Recognition, the input sequence is the sequence of visual features, generated by the backbone CNN, as described. The output sequence is, as expected, the target sequence of characters. For our system, both the starting and ending tokens (of a word) are selected to be the same space token (SP), which naturally separates words. Architecture-wise, the encoder is implemented by a multi-layered bidirectional GRU [3] module, while the decoder consists of a single-layered unidirectional GRU module. Implementation details as well as detailed formulation of the decoding process can be found in Section A.1 of supplementary material. State-of-the-art Seq2Seq models typically use an *attention module* [27], which directly propagates information from the input sequence to the output sequence, circumventing the intermediate representation. However, such an additional module would result in decreasing the significance of the intermediate vector between the encoder and the decoder and thus no such attention mechanisms are used in this work. Specifically, for attention-based approaches, the character encoding information is mostly propagated through the attention module while the intermediate feature vector usually assists the spatial correspondence of the attention module. As we mentioned in the introduction, our main goal it to generate unique word embeddings. The intermediate feature vector is ideal for this task and thus adding an attention path would greatly affect the ability of generating discriminative representations, despite the potential increase in recognition performance.

Training Scheme: The full model is then trained with a multi-task loss, defined as a weighted sum of the loss for the two branches:

$$L(w_{cnn}, w_{ctc}, w_{s2s}) = L_{CTC}(w_{cnn}, w_{ctc}) + \lambda L_{S2S}(w_{cnn}, w_{s2s}), \tag{1}$$

where L_{CTC} and L_{S2S} are the CTC and the Seq2Seq loss functions respectively and hyperparameter λ controls the contribution of each loss. L_{S2S} loss is the average cross-entropy loss across all the per-character predictions produced by the decoder. Since CTC head is only used for faster convergence, while Seq2Seq is responsible for generating the proposed embeddings, we expect a larger value of λ (we set $\lambda = 10$). The parameters of the backbone CNN w_{cnn} are jointly trained by both losses, while the parameters of the 1D CNN w_{ctc} and the parameters of the Seq2Seq w_{s2s} are optimized by their corresponding losses. In inference mode, each branch can provide a (different) predicted character sequence, following either the CTC or the Seq2Seq rationale. In practice, joint training offers an improved decoding for either of the two inference options, compared to having trained the two branches separately.

3 Keyword spotting using the Seq2Seq encoding

In this section, we focus on the KWS task. Specifically, we describe how we can use the proposed architecture, already trained for the recognition task, to tackle both QbE and QbS scenarios. Regarding QbS, we present two alternative schemes, one involving adding an autoencoder module to the main network, and the other involving using a forced alignment scheme on the Seq2Seq branch. Finally, we explore the effectiveness of training binarized holistic word embeddings with the use of Straight-Through Estimator [2], significantly compressing the generated representations.

Query-by-Example: The idea behind the proposed word spotting extension is to utilize the architecture to generate descriptive holistic representations for each word image and the query image. The intermediate feature vector of the Seq2Seq system, generated between the encoder and the decoder module, is ideal for this task, as it readily produces a fixed-sized descriptive word descriptor. Query-by-Example KWS is then straightforward, as it suffices to compare descriptors with a suitable distance measure (cosine distance).

Query-by-String with an Autoencoder Module: As we have described, the existing system can perform QbE spotting straightforwardly. Nevertheless, the QbS variation cannot be executed with the current pipeline. To this end, we add an extra encoder module which translates the target sequence (i.e. the string query) into an intermediate representation space (following the rationale of [34]). This character encoder module along with the decoder module of Seq2Seq branch form an autoencoder path, aiming to encode and subsequently decode the exact same sequence. Training of the extra character encoder module is performed by extending the loss for the Seq2Seq branch (replacing the corresponding term in the full model multi-task loss of Eq. 1) in order to simultaneously train the decoder while imposing the same embedding space to both encoders. The detailed formulation of the updated loss is included in section A.3 of supplementary material.

An additional interesting consequence of adding this autoencoder path is that a word corpus (a collection of word transcripts) can be used to aid training. So far, we took into account only the words existing in the training set, even though Seq2Seq system is capable of learning an *implicit* language model of valid consecutive characters. The intuition behind this variation is to assist the underlying implicit language model of the Seq2Seq system, by feeding it with valid words that may not exist in the training set. Implementation-wise, at each optimization iteration and after updating weights by backpropagating the multi-task loss w.r.t. the standard training set, we fine-tune the autoencoder path with words drawn from the word corpus. Note that this approach is not equivalent to imposing an external n-gram language model at decoding time. Instead, we solely aim to improve the internal representations during training, refining the existing *implicit* language model, without adding any size or time inference overhead. Such implicit LM could assist the system to better generalize when considering out-of-vocabulary (OOV) words.

Query-by-String with Forced Alignment: Even though holistic representations of fixed size greatly simplify the upcoming matching step, we also consider the case of using the decoder model of the proposed Seq2Seq system as a scoring function for the representation x_{enc} constrained to produce a target word string s. This constrained decoding, usually referred to as forced alignment (FA), is a popular QbS alternative based on character lattices [43], where a specific sequence of character is scored according to a pre-computed character graph (consisted of the probability of the character at each node, as well as the transition probability from one node to another). The alignment term refers to the case of several possible alignments of the sequence of features to the desired sequence of characters.

However, considering the Seq2Seq rationale, query-constrained scoring can be efficiently performed according to $\sum_i L_{CE}(c'_i, c_i)$, where $s = \{c_i\}$ the query sequence and $s' = \{c'_i\}$ the predicted one. Specifically, we assume that the input of the decoder is the requested query s and thus it is straightforward to predict the next character c'_{i+1}, given the previous one c_i and the hidden vector computed this far. Consequently, the score is the average cross-entropy of the predictions and thus if the score is low, the query is similar to the representation x_{enc}. A more detailed description of the proposed force-alignment scheme is included in supplementary material (section A.4).

Binary Word Representation: One noteworthy variation is the binarization of the intermediate feature vector, which can greatly reduce storage requirements for storing large collections of documents, as well as time requirements of comparing such binary representations. Computational requirements and details about the binarization scheme can be found in Section A.4 of supplementary material. The binarization of the word representation can be simply performed by a *sign* operation on the already trained embeddings. Nevertheless, binarizing the already trained embeddings by using a sign operation may significantly affect performance (especially recognition). To address this problem we proposed a training scheme for binarized vectors based on the Straight-Through Estimator (STE) [2]. Specifically, we retain the exact same architecture and framework with the exception of a sign operator between the encoder and the decoder of the Seq2Seq branch. Back-propagating through the sign function is not feasible and thus we employ STE in order to effectively train the proposed binary-inducing framework.

The Straight-Through Estimator works with discontinuous (threshold) functions which are non-differentiable and uses them for the forward pass “as is”, while it allows the error to be propagated without change through the backward pass, effectively ignoring them as if they were the identity. Although STE has been used for training binary CNNs [30], it is rather crude to be straightforwardly applied on the generated representations. One main difference is that STE was applied on the network weights that are steadily updated, while we apply STE to a feature vector with considerable variations for each image/word at the same iteration. This problem was resolved by simulating a *tanh* activation based on the following observation: $x_{bin} = \text{sign}(x_{enc}) = \lim_{a\to\infty} \tanh(a x_{enc})$. We therefore distinguish two cases: In the forward pass, we simply use the binarized vector $\text{sign}(x_{enc})$ as decoder input; in the backward pass, we treat the signed vector as if a *tanh* operation was applied, namely $tanh(x_{enc})$, and compute gradients and backpropagation error accordingly. Following the STE reasoning, the a value is not important as long as we distinguish one hard operation at the forward pass and a corresponding soft differentiable one for the backward pass.

4 Experimental Evaluation

Experimental Setup: We have run numerical trials on four different datasets, namely the George Washington (GW) [6], Botany [28], Konzilsprotokolle [28] and IAM datasets [23]. The most widely used is the IAM dataset [23], consisting of a total of 13,353 handwritten lines or 115,320 words, written by 657 writers, and for which both line and word level segmentations are available. As IAM is a large and multi-writer dataset, it is very challenging and typically used as the standard benchmark of comparison for either Handwriting Recognition or Keyword Spotting methods alike [5, 9, 18]. As evaluation metrics, we use the standard metrics that are used in the related literature: Character Error Rate (CER) and Word Error Rate (WER) for recognition, and mean average precision (MAP) for KWS. All experiments follow the same setting: Training is performed assuming a word-level recognition system. We train the proposed architecture using the Adam optimizer [17] for 80 epochs along with a cosine annealing scheduler restarted every 20 epochs [22]. The pre-processing steps are: 1) All images attain a resolution of 64×256 pixels before used by the proposed framework. Arbitrary-sized original images are padded, preserving the existing aspect ratio, in order to

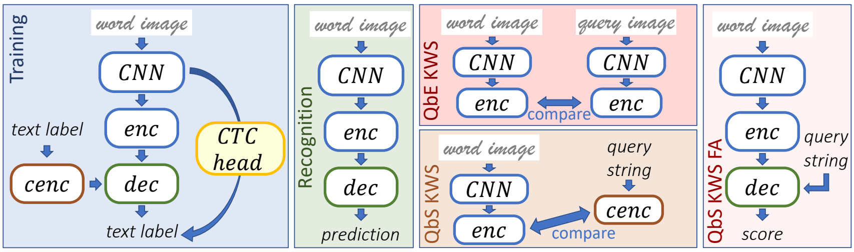


Figure 2: Overview of the different setups of the proposed system. Evaluation can be categorized to recognition and three spotting cases: QbE, QbS and QbS by force-alignment (FA).

attain the aforementioned fixed size. If the original image is greater than the predefined size, the image is rescaled. 2) A global affine augmentation is applied at every image as in [38]. 3) Each word transcription has spaces added before and after *only* during training. This operation aims to assist the system to predict the marginal spaces that exist in the majority of the images. Simultaneously, these spaces act as start and end tokens for the seq2seq approach. More detailed description of architectural/training settings and recognition/spotting protocols are provided in the supplementary material along with additional experimental results. For the upcoming experiments, the intermediate embedding dimension equals to 512. Note that no architecture exploration was conducted. We designed a compact DNN system, aiming to minimize the required resources, which performs adequately well. Therefore it is possible to observe further (minor) improvements, if a thorough architecture exploration is performed. Code is publicly available at `https://github.com/georgeretsi/Seq2Emb`.

Proposed System Overview: First, we present a brief recap of training and evaluation setups, focusing on clarifying which network modules are used at each case and clearly stating the different evaluation modes. As Figure 2 suggests, we can distinguish the training setup and four distinct evaluation scenarios. *Training* procedure, as expected, includes every proposed module, namely CNN backbone, CTC head, Seq2Seq head and the character encoder of the autoencoder path. As we have already mentioned, CTC head assists the faster convergence of the proposed framework and thus it is used only during training. Since the training setup assumes a recognition system, the *evaluation of recognition task* is straightforward: use the Seq2Seq head to predict an output sequence. We should highlight that the main focus of this paper lies on generating discriminative embeddings and thus maximizing recognition performance is sidelined. In fact, design choices of the proposed method do not favor recognition performance (no attention mechanism is used in order to attain discriminative embeddings/ convolutional-only CTC head aims to faster training).

Using the proposed embedding space, *keyword spotting* can be performed by straightforwardly comparing the extracted feature vectors. QbE relies on the CNN and the encoder modules for transforming both images into the embedding space, while QbS uses also the character encoder module to transform the query string into the same embedding space. Force-alignment QbS spotting mode requires the full Seq2Seq head (encoded/decoder) and thus is closely related to recognition mode. In fact, a well-performing force-alignment scheme is an indication of an effective recognition system.

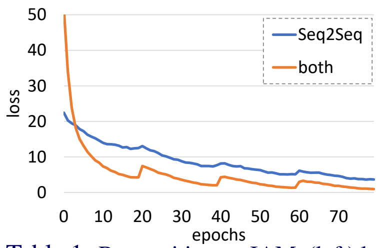
Recognition and CTC Branch Impact: We start by exploring the word recognition task with the full set of existing characters, i.e. letters of either, digits and punctuation, following the line-level recognition setting. Recognition results, using a greedy decoding procedure, are summarized at Table 1(upper-right). We distinguish the training scheme to *alone*, where each branch is trained separately, and *joint*, where both recognition flows are trained together as a multi-task problem (see Eq. 1). Experimental results suggest that the Seq2Seq approach has convergence difficulties when trained alone. Nevertheless, when used alongside the CTC branch, Seq2Seq has a notable increase in performance for the exact same number of epochs. This is in line with our initial suggestion of a fast converging CTC pipeline, which helps the generation of meaningful visual features at the CNN output in only a few epochs. We further validate this behavior through the results of Table 1(left), where joint training achieves faster convergence with consistently lower loss values despite using an additional loss term.

Moreover, we should highlight that Seq2Seq variation gives the best word error rate. This behavior can be attributed to the implicit language model learnt by the Seq2Seq approach, which can generate more plausible sequences of characters. To further examine this observation, we present examples of decoding errors from the two different recognition branches in Table 1(bottom-right). We can deduce from these errors that the two branches almost consistently lead to somewhat different decodings, while Seq2Seq errors show that this branch has learned a language model (character n-grams) to an extent, as we expected.

Keyword Spotting: Having established the functionality of the recognition system, we focus on exploring the proposed variations for the Keyword Spotting task. Spotting approaches usually use a different character set, consisted only of lowercase letters and digits, and thus we follow the same setting, both for word spotting as well as the reported recognition results from now on. Ablation studies are performed on the challenging IAM dataset.

A simple, yet interesting, preliminary spotting experiment is to use the recognition system in order to generate string predictions and then evaluate the QbS performance by comparing these predictions with the text queries (using the Levenshtein distance). The MAP performance of this setting is 89.28%, when the proposed embedding-based QbS is 95.50%. We can see that the "hard" decision of explicitly specifying a unique character sequence leads to notable precision decrease. Moreover, embedding-based spotting can be implemented more efficiently since no decoding procedure nor the time-consuming Levenshtein comparison are required. In a nutshell, this experiment highlights the importance of dedicated spotting methods instead of treating KWS as a by-product of the recognition output.

Table 2 (left) contains the experimental results for both QbE and QbS, along with the recognition metrics for the decoder (Seq2Seq branch), using the proposed system dubbed as *Seq2Emb*. Recognition results act as an extra indication of the effectiveness of the Seq2Seq branch. Along with vanilla trained *Seq2Emb* system, we also consider feeding word strings from a corpus to the autoencoder path and binarizing the intermediate embeddings. Specifically, we learn an implicit LM (*Seq2Emb+LM*), as described in the previous section, using a concatenation of the LOB [14] and BROWN [8] corpora, from which we sample words according to their occurrence frequency. This extra information feed has a positive im-



setting	branch	CER	WER
word (alone)	CTC	6.3	16.7
	Seq2Seq	9.8	21.6
word (joint)	CTC	6.3	16.6
	Seq2Seq	6.5	16.0

ground truth	*security*	*just*	*laughter*	*no*	*roads*
CTC	*seemrite*	*jurl*	*laughtes*	*n0*	*souds*
Seq2Seq	*securitte*	*jurst*	*laughters*	*no*	*souch*

Table 1: Recognition on IAM. (left) loss curves indicating the importance of the CTC assistance module. (upper-right) Recognition results reported w.r.t training each branch separately ("alone") versus training with the proposed multitask loss ("joint"). (bottom-right) Error decoding examples using the proposed joint model. Results using either model branch are compared (CTC/Seq2Seq).

method	QbE	QbS	CER	WER
Seq2Emb	91.62	95.50	5.1	15.1
Seq2Emb+LM	92.04	95.91	5.0	14.7
Seq2Emb+LM+B (w/o train)	86.12	90.80	79.62	98.53
Seq2Emb+LM+B (w/ train)	90.89	93.51	5.0	15.2

method	binarization	MAP
Autoencoder Module	No	95.91
	Yes	93.51
Forced Alignment	No	96.53
	Yes	96.38

Table 2: Keyword Spotting on IAM. (left) Comparison of recognition and KWS results when incorporating a language model (+LM) and binarizing the representation (+B). (right) Comparison of different approaches for QbS KWS with or without the binarization scheme

pact on both recognition and KWS, supporting the observation drawn from Table 1(bottom-right). Finally, we also report the effect of the proposed binarization scheme, dubbed as *Seq2Emb+B*, for two different settings: straightforwardly apply the sign operation on the embeddings generated by an already trained system or re-train the whole system according to the STE rationale. The first case produces decent KWS results, even though it deteriorates performance, while it completely ruins the decoding ability of the generated embeddings. Nonetheless, using the STE scheme, recognition performance is close to the initial non-binary case, while KWS results are also considerably improved. Overall, we get a well-performing system over all tasks despite the information loss due to the binarization step, while at the same time obtaining a very compact descriptor.

We also compare the two proposed QbS/KWS strategies, presented in Table 2 (right). Specifically, we compare QbS using the character encoder module versus the forced alignment approach. Notably, the Forced Alignment approach increases spotting performance, regardless if the word representation is binarized or not. Of course, the improvement is achieved at the cost of computational effort, since encoder-based QbS relies on comparing fixed-length representations with a simple distance metric, and is thus very efficient.

SoA Comparisons: Finally, we compare the proposed method to state-of-the-art approaches for both the recognition and the spotting task. Even though we focused on generating effective embeddings and expect sub-optimal recognition performance, we report recognition results in Table 3 for the sake of a thorough exploration. We considered the two recognition-oriented datasets for this scenario: IAM and GW. Since we follow the setup of the reduced character set (as in [1]) which is also used in KWS, we included related works with the same setup and the same task, i.e. unconstrained lexicon-free word recognition. Detailed information about the setup is included in the supplementary material. Compared methods include Seq2Seq approaches [15, 40, 47] and CTC-based approaches [5, 20, 45], while [15, 40, 47] do not state explicitly the character set used (if the full character set is used as in Table 1, a reduced performance is expected). Results show competitive performance of the proposed method even when binarized embeddings were considered, outperformed only by [5] for IAM. Thus, the numerical results support our claim of extracting discriminative binary word representations which can be faithfully decoded with negligible degradation of accuracy.

Method	IAM		GW	
	CER	WER	CER	WER
Sueiras et al. [40]	8.8	23.8	-	-
Wigington et al. [45]	6.07	19.07	-	-
Krishnan et al. [20]	6.34	16.19	-	-
Dutta et al. [5]	4.88	12.61	4.29	12.98
Zhang et al. [47]	8.50	22.20	-	-
Kang et al. [15]	6.43	16.39	-	-
Proposed Models				
Seq2Emb+LM	5.01	14.73	4.23	10.82
Seq2Emb+LM+B	5.04	15.21	4.65	12.27

Table 3: Comparison of state of the art word recognition approaches for IAM and GW. Binarized variation is also included.

A comparison of our method versus state-of-the-art methods for KWS is presented in Table 4, including all four aforementioned KWS datasets. The proposed variations outperform or are in par with existing SoA approaches, even when we extract binary embeddings. Interestingly enough, the binarization scheme has almost non-existent degrading impact on the "easier" datasets GW, Botany and Konzilsprotokolle, as opposed to the deteriorated results of IAM. Contrary to the reported methods of Table 2, we successfully use a recognition-oriented architecture to extract word embeddings. Specifically, most of the works employ PHOC-based representations for training [1, 16, 19, 38, 39]. Others rely to training according to an explicit word classification task, including of all the possible words [17, 18], while the triplet loss for generating distinct embeddings is employed in [46]. Also, [19, 46] make use of semantic information to increase performance. Note that Krishnan et al. [16, 17, 18, 19, 20] use several performance-enhancing techniques, which are orthogonal to our approach (and thus potentially beneficial if added to the proposed pipeline), such as pretraining with a very large synthetic dataset and extensive data augmentation with local deformations. Also note that our network, including all its sub-components, is considerably smaller compared to vast VGG-based models such as PHOCNet [38, 39].

Method	IAM		GW		Botany		Konzils	
	QbE	QbS	QbE	QbS	QbE	QbS	QbE	QbS
Attributes+KCSR [1]	55.73	73.72	93.04	91.29	75.77	65.69	77.91	82.91
PHOCNet [38]	72.51	82.97	96.71	92.64	89.69	74.47	96.05	94.20
HWNet [17]	80.61	-	94.84	-	84.16	-	79.13	-
Triplet-CNN [46]	81.58	89.49	98.00	93.69	54.95	3.40	82.15	12.19
PHOCNet-TPP [39]	82.74	93.42	97.78	98.02	91.23	95.06	97.70	97.28
DeepEmbed [16]	84.25	91.58	94.41	92.84	-	-	-	-
Deep Descriptors [32]	84.68	-	-	-	-	-	-	-
Zoning Ensemble PHOCNet [33]	87.48	-	-	-	-	-	-	-
End2End Embed [20]	89.07	91.26	98.14	97.42	94.82	88.60	92.96	71.00
DeepEmbed [20]	90.38	94.04	98.01	98.86	95.46	97.17	94.11	90.65
Synth+DeepEmbed [20]	-	95.09	-	98.98	-	97.18	-	91.43
HWNetV2[18]	92.41	-	98.24	-	95.26	-	93.47	-
NormSpot[19]	92.54	96.54	99.37	99.46	-	-	-	-
Proposed Variations								
Seq2Emb	92.04	95.91	97.86	98.41	95.73	98.23	97.51	98.66
Seq2Emb+B	90.89	93.51	97.83	98.38	94.89	98.01	96.79	98.26
Seq2Emb+FA	-	96.53	-	98.77	-	99.00	-	99.21
Seq2Emb+B+FA	-	96.38	-	98.52	-	98.78	-	99.25

Table 4: Comparison of the state of the art for Keyword Spotting versus variations of the proposed method for four KWS datasets: IAM, GW, Botany and Konzilsprotokolle (Konzils). The proposed method uses the implicit LM training module (+LM) only for IAM and GW.

5 Conclusions

We have proposed a novel DNN-based system that can generate discriminative word embeddings, ideal for Keyword Spotting. The core idea is to make use of the encoding output of a Sequence-to-Sequence architecture, trained for performing word recognition. A number of extensions and variants of the base architecture have been also proposed and discussed, including a retraining scheme that can produce binarized, compact descriptors as well as two alternative ways to handle QbS KWS with our model. In almost all of the variations of the tasks considered, the proposed model was shown to provide state-of-the-art results.

Acknowledgements

This research has been partially co-financed by the EU and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call OPEN INNOVATION IN CULTURE, project *Bessarion* (T6YBΠ-00214).

References

[1] J. Almazán, A. Gordo, A. Fornés, and E. Valveny. Word spotting and recognition with embedded attributes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(12):2552–2566, Dec 2014.

[2] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

[3] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[4] Ronan Collobert, Awni Hannun, and Gabriel Synnaeve. A fully differentiable beam search decoder. *arXiv preprint arXiv:1902.06022*, 2019.

[5] Kartik Dutta, Praveen Krishnan, Minesh Mathew, and C.V. Jawahar. Improving CNN-RNN hybrid networks for handwriting recognition. In *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pages 80–85. IEEE, 2018.

[6] A. Fischer, A. Keller, V. Frinken, and H. Bunke. Lexicon-free handwritten word spotting using character HMMs. *Pattern Recognition Letters*, 33(7):934–942, 2012.

[7] Andreas Fischer. *Handwriting recognition in historical documents*. PhD thesis, Verlag nicht ermittelbar, 2012.

[8] W Nelson Francis and Henry Kucera. Brown corpus. *Department of Linguistics, Brown University, Providence, Rhode Island*, 1, 1964.

[9] A. P. Giotis, G. Sfikas, B. Gatos, and C. Nikou. A survey of document image word spotting techniques. *Pattern Recognition*, 68:310 – 332, 2017.

[10] Alex Graves. Connectionist temporal classification. In *Supervised Sequence Labelling with Recurrent Neural Networks*, pages 61–93. Springer, 2012.

[11] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376. ACM, 2006.

[12] Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. LSTM: A search space odyssey. *IEEE Transactions on Neural Networks and Learning Systems*, 28(10):2222–2232, 2016.

[13] Abhishek Jha, Vinay P Namboodiri, and C.V. Jawahar. Word spotting in silent lip videos. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 150–159. IEEE, 2018.

[14] Stig Johansson, Eric Atwell, Roger Garside, and Geoffrey N Leech. *The Tagged LOB corpus: users' manual*. Norwegian Computing Centre for the Humanities, 1986.

[15] Lei Kang, Pau Riba, Marçal Rusinol, Alicia Fornés, and Mauricio Villegas. Distilling content from style for handwritten word recognition. In *2020 17th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pages 139–144. IEEE, 2020.

[16] P. Krishnan, K. Dutta, and C. V. Jawahar. Deep feature embedding for accurate recognition and retrieval of handwritten text. In *Proceedings of the* 15^{th} *International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pages 289–294, 2016.

[17] Praveen Krishnan and C.V. Jawahar. Matching handwritten document images. In *European Conference on Computer Vision*, pages 766–782. Springer, 2016.

[18] Praveen Krishnan and C.V. Jawahar. HWNet v2: An efficient word image representation for handwritten documents. *arXiv preprint arXiv:1802.06194*, 2018.

[19] Praveen Krishnan and C.V. Jawahar. Bringing semantics into word image representation. *Pattern Recognition*, 108:107542, 2020.

[20] Praveen Krishnan, Kartik Dutta, and C.V. Jawahar. Word spotting and recognition using deep embedding. In *2018 13th IAPR International Workshop on Document Analysis Systems (DAS)*, pages 1–6. IEEE, 2018.

[21] Gundram Leifert, Tobias Strauß, Tobias Grüning, Welf Wustlich, and Roger Labahn. Cells in multidimensional recurrent neural networks. *The Journal of Machine Learning Research*, 17(1):3313–3349, 2016.

[22] I. Loshchilov and F. Hutter. SGDR: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

[23] U-V Marti and Horst Bunke. The IAM-database: an english sentence database for offline handwriting recognition. *International Journal on Document Analysis and Recognition*, 5(1):39–46, 2002.

[24] Liliane Momeni, Triantafyllos Afouras, Themos Stafylakis, Samuel Albanie, and Andrew Zisserman. Seeing wake words: Audio-visual keyword spotting. In *British Machine Vision Conference*, 2020.

[25] Stavros Petridis, Themos Stafylakis, Pingehuan Ma, Feipeng Cai, Georgios Tzimiropoulos, and Maja Pantic. End-to-end audiovisual speech recognition. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6548–6552. IEEE, 2018.

[26] Arik Poznanski and Lior Wolf. CNN-N-gram for handwriting word recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2305–2314, 2016.

[27] Rohit Prabhavalkar, Tara N Sainath, Bo Li, Kanishka Rao, and Navdeep Jaitly. An analysis of attention in sequence-to-sequence models. In *Interspeech*, pages 3702–3706, 2017.

[28] Ioannis Pratikakis, Konstantinos Zagoris, Basilis Gatos, Joan Puigcerver, Alejandro H. Toselli, and Enrique Vidal. Icfhr2016 handwritten keyword spotting competition (H-KWS 2016). In *15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pages 613–618, 2016.

[29] Joan Puigcerver. Are multidimensional recurrent layers really necessary for handwritten text recognition? In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 1, pages 67–72. IEEE, 2017.

[30] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European Conference on Computer Vision*, pages 525–542. Springer, 2016.

[31] George Retsinas, Georgios Louloudis, Nikolaos Stamatopoulos, and Basilis Gatos. Efficient learning-free keyword spotting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(7):1587–1600, 2018.

[32] George Retsinas, Giorgos Sfikas, Georgios Louloudis, Nikolaos Stamatopoulos, and Basilis Gatos. Compact deep descriptors for keyword spotting. In *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pages 315–320. IEEE, 2018.

[33] George Retsinas, Giorgos Sfikas, Nikolaos Stamatopoulos, Georgios Louloudis, and Basilis Gatos. Exploring critical aspects of cnn-based keyword spotting. a phocnet study. In *13th IAPR International Workshop on Document Analysis Systems (DAS)*, pages 13–18. IEEE, 2018.

[34] George Retsinas, Georgios Louloudis, Nikolaos Stamatopoulos, Giorgos Sfikas, and Basilis Gatos. An alternative deep feature approach to line level keyword spotting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12658–12666, 2019.

[35] Jose A. Rodriguez-Serrano and Florent Perronnin. Label embedding for text recognition. In *British Machine Vision Conference*, 2013.

[36] Giorgos Sfikas, George Retsinas, and Basilis Gatos. A PHOC decoder for lexicon-free handwritten word recognition. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 1, pages 513–518. IEEE, 2017.

[37] Gilbert Strang. *Linear algebra and learning from data*. Wellesley-Cambridge Press, 2019.

[38] S. Sudholt and G. A. Fink. PHOCNet: A deep convolutional neural network for word spotting in handwritten documents. In *Proceedings of the* 15^{th} *International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pages 277–282, 2016.

[39] Sebastian Sudholt and Gernot A. Fink. Evaluating word string embeddings and loss functions for CNN-based word spotting. In *2017 14th iapr international conference on document analysis and recognition (ICDAR)*, volume 1, pages 493–498. IEEE, 2017.

[40] Jorge Sueiras, Victoria Ruiz, Angel Sanchez, and Jose F Velez. Offline continuous handwriting recognition using sequence to sequence neural networks. *Neurocomputing*, 289:119–128, 2018.

[41] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[42] Vasiliki Tassopoulou, George Retsinas, and Petros Maragos. Enhancing handwritten text recognition with n-gram sequence decomposition and multitask learning. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 10555–10560. IEEE, 2021.

[43] A. H. Toselli, E. Vidal, V. Romero, and V. Frinken. HMM word graph based keyword spotting in handwritten document images. *Information Sciences*, 370:497–518, 2016.

[44] Hongxi Wei, Yanke Kang, and Hui Zhang. Word image representation based on sequence to sequence model with attention mechanism for out-of-vocabulary keyword spotting. In *2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, pages 2224–2231. IEEE, 2019.

[45] Curtis Wigington, Seth Stewart, Brian Davis, Bill Barrett, Brian Price, and Scott Cohen. Data augmentation for recognition of handwritten words and lines using a CNN-LSTM network. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 1, pages 639–645. IEEE, 2017.

[46] Tomas Wilkinson and Anders Brun. Semantic and verbatim word spotting using deep neural networks. In *2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pages 307–312. IEEE, 2016.

[47] Yaping Zhang, Shuai Nie, Wenju Liu, Xing Xu, Dongxiang Zhang, and Heng Tao Shen. Sequence-to-sequence domain adaptation network for robust text image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2740–2749, 2019.