Separating Optical and Language Models through Encoder-Decoder Strategy for Transferable Handwriting Recognition

Adeline Granet, Emmanuel Morin, Solen Quiniou, Christian Viard-Gaudin, Harold Mouchère

Separating Optical and Language Models through Encoder-Decoder Strategy for Transferable Handwriting Recognition

Adeline GRANET, Emmanuel MORIN, Harold MOUCHÈRE, Solen QUINIOU, Christian VIARD-GAUDIN
LS2N, UMR CNRS 6004
Université de Nantes, Nantes, France
firstname.lastname@univ-nantes.fr

***Abstract*—Lack of data can be an issue when beginning a new study on historical handwritten documents. To deal with this, we propose a deep-learning based recognizer which separates the optical and the language models in order to train them separately using different resources. In this work, we present the optical encoder part of a multilingual transductive transfer learning applied to historical handwriting recognition. The optical encoder transforms the input word image into a non-latent space that depends only on the letter-n-grams: it enables it to be independent of the language. This transformation avoids embedding a language model and operating the transfer learning across languages using the same alphabet. The language decoder creates from a vector of letter-n-grams a word as a sequence of characters. Experiments show that separating optical and language model can be a solution for multilingual transfer learning.**

***Index Terms*—Handwriting recognition, knowledge transfer, Optical model, Language model**

I. Introduction

The amount of digitized handwritten historical documents has increased to preserve them and to make heritage accessible to all. However, digitization is not sufficient to make the documents usable: information needs to be extracted in order to index them. Researchers and historians in the humanities and in the social sciences need to be able to query them. Therefore, new projects involve the use of several domains such as language processing, document recognition, and information retrieval for historical studies. In recent years, the number of competitions regarding historical documents has increased [1], [2]. Such systems have to deal with the complexity of the task as well as the document medium, its level of deterioration, or even its written language. All of these can have a strong impact on the system efficiency.

State of the art handwriting systems are built with a Multidimensional Long Short-Term Memory (MDLSTM) network [3] or even with a Convolutional Recurrent Neural Network (CRNN) stacked with a Bidirectional Long Short-Term Memory (BLSTM) [4]. The neural network training includes a Connectionist Temporal Classification (CTC) cost function proposed by [5]. Lately, the trend in handwriting recognition (HWR) is the use of deep neural networks and, more recently, the integration of attention models in the networks [6]. To decode words as sequences of characters, several strategies can be used: a dictionary, a language model, or a Weighted Finite State Transducer (WFST) including several dictionaries. Nevertheless, results are constrained by the size of the vocabulary used. When too many words are out-of-vocabulary, the results are degraded. To improve the results, methods can be employed to increase the size of the vocabulary by using Wikipedia or other available resources. Whether improving the results or training the networks, the HWR systems require a lot of labeled data.

In our case of a study, we deal with a new historical resource without any ground-truth. Within this context, transductive transfer learning [7] is an interesting approach when no ground-truth is provided. Indeed, it uses different sources of data to train a system for a specific task by applying various target data. It enables us to use available existing resources to annotate unknown data. It is eagerly used to supply the lack of data for greedy systems such as word spotting in historical documents [8] or translation models with multi-modal systems [9].

The standard of machine translation is an encoder-decoder system based on a recurrent neural network [10]. The first component encodes the source language into a fixed vector and the last one decodes the sequence into the target language. Similarly, [11] proposed a neural image caption generator that consists of two sub-networks: a pre-trained Convolutional Neural Network (CNN) encoding an image into a fixed size vector and a LSTM model generating the corresponding description. We were inspired by this framework to split up our handwriting recognition system into an encoder and a decoder. First, an image encoder takes a word image as input and, thanks to a fully convolutional network (FCN), converts it into a vector of letter-n-grams as in [12]. Here, the transfer learning is performed during the training step that uses available labeled image data. Then, the decoder is built with a recurrent layer to generate a sequence of characters from the vector of letter-n-grams. Here, a second transfer learning is independently made on the vocabulary. Contrary to the machine translation and description generator works, the sub-networks can be trained independently from each other.

This paper is organized as follows. Section II describes our transfer model within the encoder and decoder components. Section III presents the architecture of the neural network. Section IV presents the databases used for both components and the evaluation methods. Finally, Section V reports the results obtained with the optical and the language model independently, following by the results on the whole system.

II. Encoder-Decoder for Transfer Learning

Based on the work on image description generation, we define a system that is divided into two complementary components: an optical model and a language model, as shown in Figure 1. The first one encodes a word image into a vector and the second one decodes the vector into a character sequence as a word. The interface vector that links these two parts is a bag of characters or letter-n-grams. A letter-ngram is a sequence of n characters as uni-gram, bi-gram or tri-gram for $n = 1, 2, 3$. We estimated the set of letter-n-grams on all resources used for both the training data and the transfer target data. The originality in using a letter-n-grams vector as a pivot is that we encode the input into a non-latent space which is transferable as long as the training data and transfer target data share the same alphabet. In this paper, we focus on the optical model to encode words into vectors of letter-n-grams. Nevertheless, we will also present the results obtained on the whole system integrating the optical and language models.

A. Letter-n-grams

As suggested by [12], we use letter-n-grams as a pivot in our system. Initially, the authors selected the 50,000 most popular letter-n-grams to represent words. Therefore, we selected around 12,500 letter-n-grams with a maximum length of 3 on all the used datasets adding *[* and *]* to represent the beginning and end of a word. A *joker* class was added to replace the non-selected letter-n-grams (as out-of-vocabulary ngrams). According to Figure 1, the word *Pages* is thus decomposed as {P, a, g, e, s, [P, Pa, ag, ge, es, s], [Pa, Pag, age, ges, es]}. The vector is built by normalizing the frequency of the letter-n-grams.Using frequency enables us to hold information about the word size and to compensate the lost temporality.

B. Optical Model Encoder

We chose to define and to train our own fully convolutional neural network devoted to the grayscale handwriting images. A set of masks is applied to extract local set of features. This is related to use an artificial and pre-defined attention model. We define 3 types of masks: 1) the first mask is the position of the word into the larger image; 2) the second set is composed of two masks for the beginning and end of word into image; 3) the last set is composed of three masks for the beginning, middle and end of word into image. The use of these masks should facilitate the identification of the character order and thus ngrams composing the word. Furthermore it will help to recognized the ngrams using '[' and ']'. The optical encoder ends by two fully-connected layers before the last layer that computes the letter-n-grams vector.

C. Language Model Decoder

Based on machine translation [13] and description generation [11], we turned to neural encoder-decoder architectures in NLP as it enables us to map one sequence (image) to another one (word) without constraining them to have the same length or word order. We define a character-based model as shown on the right part of Figure 1. These components were designed to be the most minimalist: they do not use an embedding layer representing the full words since it might interfere with the transfer learning that has to deal with different languages.

For the sequence generation task, we used a recurrent layer to add a temporal information followed by a softmax layer that gives one character at the time until the end-of-word symbol *]* is given. We choose not to include a bidirectional network contrary to encoder-decoder models that are used in machine translation. Since our character-based decoder uses a vector without any order indication of characters, all the information and context are included in the letter-n-grams vector. Therefore, a bidirectional recurrent layer is useless.

III. Neural networks Modeling

A. Optical Model

The first part of our model is based on a fully-convolutional neural network that extracts and builds features from the word images. We choose not to use an existing pre-trained system. Most of the available networks have been pre-trained on natural images, so not grayscaled handwritten word images. The architecture of the trained FCN is detailed in Table I. The input image size is normalized to 100 pixels of height, so the features extraction varies according to the length called t. The features extraction network is composed of 7 layers of convolution with a kernel size of 3x3 with same-padding. They linked with different max-pooling leading to a representation depending only on the length of the image and the number of filters used to extract the features. Each output convolutional layer used the rectify linear unit (ReLU) [14] that enables to speed up the training step. For each image, we define several masks as shown in the bottom of Figure 1 based on the length of the word image to represent the position of each part. The product results between the six masks and the feature map produces six fix size feature vectors, followed by 2 dense layers with various sizes from 1,024 to 2,048 hidden units and a ReLU activation function. The outputs have to represent the frequency of each letter-n-grams of the word, so their values rarely reach 0.25% (for one character words, the number of letter-ngram is 4), so the difference between the present letters and non-presents is low. In order to have really a difference between the letter-n-grams included into the word and the others, we add an offset of 0.5 during the training, called Off. Thus, we experiment 2 different layers to construct the vector of letter-n-grams. The first one called SIG uses a last fully connected layer with a sigmoid activation function defined as

$$Sigmoid(x) = \frac{1}{1 + e^{-x}}$$

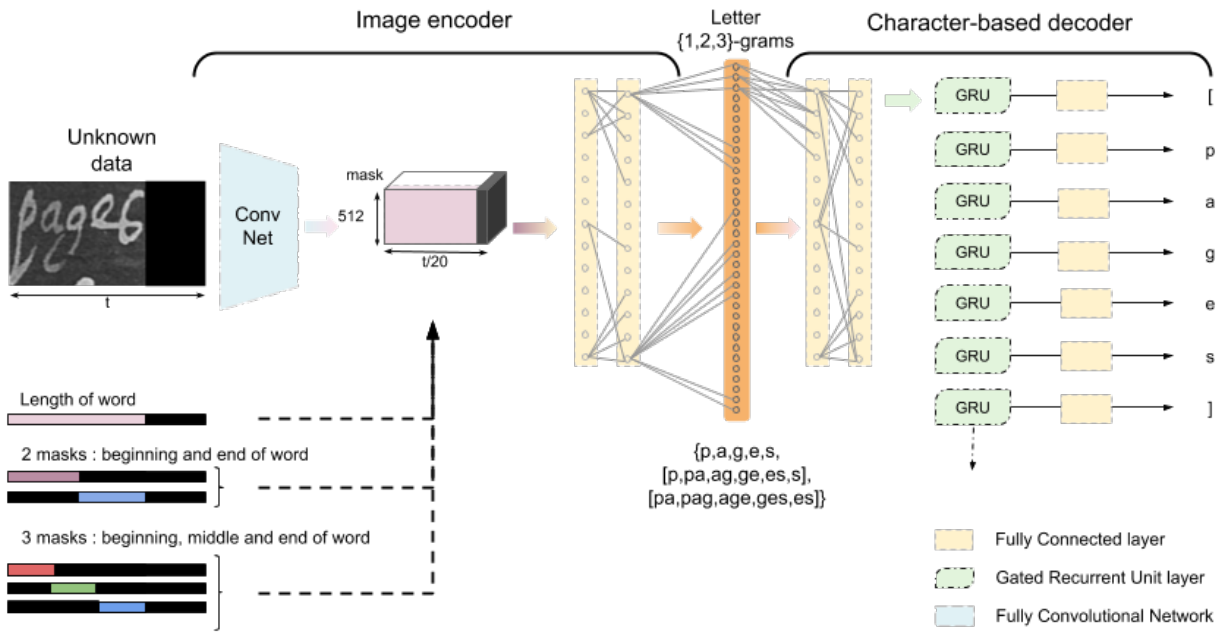

Fig. 1. Overview of our optical encoder and language decoder. On the left, the component corresponds to the image encoder from which the features are extracted thanks to a fully convolutional network, multiply by a set of m masks 1,3,6 and following by two connected layers with a minimum of 1,024 units each and a vector of letter-n-grams is built. On the right, the component corresponds to the sequence generation decoder that provides the word, character by character, corresponding to the input vector.

to have a probability for each letter-ngram independently of the others and mainly used with the offset, The second one called THR uses one fully connected layer with an hyperbolic tangent activation function defined as

$$tanh(x) = 2.Sigmoid(2x) - 1$$

that enables to provide values tending towards -1 for the missing letter-n-grams (instead of 0 with SIG); following by a layer that just applies a ReLU activation function without learning parameters and providing output values between 0 and 1. We trained the model by minimizing different objective functions: the mean squared error, binary cross-entropy and KullbackLeibler. Each time we use the Adam function that controls the learning rate which is initialized at 0.0001 for the stochastic optimization. To prevent computational errors in limits, we fix the (0,1) output values at ($1e^{-6}$, $1\text{-}1e^{-6}$).

B. Language Model

The structure of the character-based decoder part is defined with one fully connected with 1,024 units to extract features, one Gated Recurrent Unit layer (GRU) [10] with 500 hidden units, and one fully connected layer with 79 units and a soft-max activation function. The input of the network has various size from 200 units (unigrams and only bigrams including *[ ]* symbols), 1,883 units (unigrams and all bigrams) and 12 170 units (all 1,2,3-grams). Especially for the GIC dataset, we have computed and kept letter-n-grams with an occurrence greater than 5 (in order to remove the letter-n-grams estimated from the noisy of the documents). Finally, we only kept letter-n-grams appearing in at least 2 different datasets and a *joker* was created to replace the non-selected letter-n-grams. In the scope of sequence generation, we padded the length of the sequence with blank labels up to 50 characters. Therefore, the network is free to define any sequence without any constraints. The training parameters for the network are the same as the optical encoder.

TABLE I
ARCHITECTURE OF OUR FULLY CONVOLUTIONAL NEURAL NETWORK

Layer type	**Filter size**	**Output layer shape**	**Activat. function**
Input image	///	100xt	///
Convolution	$3 \times 3 \times 8$	$8 \times 100 \times t$	ReLU
Convolution	$3 \times 3 \times 16$	$16 \times 100 \times t$	ReLU
Convolution + MaxPooling	$3 \times 3 \times 32$ (2×2)	$32 \times 50 \times \frac{t}{2}$	ReLU
Convolution + MaxPooling	$3 \times 3 \times 64$ (2×2)	$64 \times 25 \times \frac{t}{4}$	ReLU
Convolution + MaxPooling	$3 \times 3 \times 128$ (5×5)	$128 \times 5 \times \frac{t}{20}$	ReLU
Convolution	$3 \times 3 \times 256$	$256 \times 5 \times \frac{t}{20}$	ReLU
Convolution + MaxPooling	$3 \times 3 \times 512$ (5×1)	$512 \times \frac{t}{20}$	ReLU

Note: ReLU corresponds to the Rectified Linear Unit function.

IV. Training and Evaluation

A. Existing Datasets

We carried out experiments with three available datasets of images mixing different languages and time periods to

compare the effect on each others and two linguistic resources:

- RIMES (RM) is a French database developed to evaluate automatic systems that recognizes and indexes handwritten letters [15];
- George Washington (GW) is an English database created from the George Washington Papers at the Library of Congress [16];
- Los Esposalles (ESP) is a Spanish database compiled from a marriage license book collection from the 15^{th} and 17^{th} centuries [16];
- Google Book (GB) composed of 23 available French and Italian digitized historical books dealing with Italian Comedy plays, their transcriptions are used as language model corpus;
- French Wikipedia data (Wiki), as used and distributed by [17], provides all words whose frequency is greater than 5 in Wikipedia. From this dataset, we randomly selected 30 000 words in order get a corpus of comparable size with the size of GB corpus but with different words.

All information is summarized in Table II.

TABLE II
VOCABULARY SIZE AND WORD IMAGE DISTRIBUTIONS OF EACH DATASET.

Distrib.		GB	RM	GW	Wiki	ESP
Images	Train	-	51,739	2,402	-	45,102
	Validation	-	7,464	1,199	-	5,637
	Test	-	7,776	1,292	-	5,637
Vocab.	Train	26,573	4,477	660	24,456	2,565
	Validation	2,953	1,578	521	3,843	629
	Test	0	1,627	431	1.928	629

Authors in [18] built a very large vocabulary gathered from a Google N-grams project and an edition of a manually transcribed book from the 16^{th} century. We had a similar approach to this work as we built a dataset from digitized books of the same century of our data, with a vocabulary containing mainly named entities (GB).

B. Definition Use Case

We are working on title line images about Italian Comedy from 18^{th} century, mainly composed of named entities [19]. We define ESP dataset as the use case to experiment our approach because this is more similar to our historical data with enough images. Let I_{ESP}, a set of ESP images of words considered as a new resource with no ground truth that we want to annotate. The main vocabulary of this historical database is composed of named entities as our data from Italian Comedy. The language used is Spanish with latin alphabet.

Let V_{ESP} the training and validation vocabulary of ESP be another resource from the same domain than images of ESP and used for the training step of the language model.

We train the optical encoder from GW and RIMES composed of the latin characters but in French and English, and coming from various time periods. While, the language model is trained independently with V_{ESP}, GB, Wiki, and also the vocabulary sets of GW and RIMES. Despite the differences of language and domain, GB is an interesting resource sharing the time period as well as the special feature of being composed of named entities (but in French and Italian).

C. Evaluation Metrics

We want to evaluate the performances of our optical model to build a vector of ngrams. We computed the Recall, Precision, Accuracy and F1-score as defined in eq. 1. We measured if all the ngrams composing the word are present (activation greater than the offset), without checking that the frequency of the letter-n-grams is correct. We define the number of ngrams correctly identified in the word vector as the true positives (TP), whereas the number of ngrams outside the word and not detected are the true negatives (TN). The number of ngrams originally included in the vector of word and not added by the model are the false negative (FN). Finally, the number of ngrams originally not included in the decomposition of word but added by the model, are the false positive (FP).

$$\begin{aligned} \text{Recall} = \frac{TP}{TP+FN}, \text{Precision} = \frac{TP}{TP+FP}, \\ \text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}, \\ \text{F1-Score} = 2 \times \frac{Recall \times Precision}{Recall + Precision}. \end{aligned} \quad (1)$$

To evaluate the performance of the language model, we used the recognition rate at the character level (*CRR*) and at the word level (*WRR*). It is defined as:

$$\text{CRR} = \max(\frac{N-(Ins+Subs+Dels)}{N}, 0)$$

with N, Ins, $Subs$, $Dels$ respectively the number of characters of the reference words, the number of character insertions, substitutions and deletions. We set a 0 low bound for the CRR computation to avoid any border effect. The Word Recognition Rate (= Word Accuracy) is defined as the number of correctly recognized words divided by the total number of words to recognize. CRR and WRR are computed using or not the Levenshtein edit distance to correct the output sequence with regards to a multilingual dictionary. The dictionary was built from the training and the validation parts of the vocabulary of all the datasets, except Wikipedia. The dictionary contains 39,051 words.

V. RESULTS & ANALYSIS

A. Optical Model Experiments

Table III presents the results of our experiments for the optical model. The first part of the results enables to define the setup of the system giving the best Recall, Precision and F1-score using ESP to train and test the system. The second part of the results are obtained using the transfer learning from GW and RM to ESP dataset. Among all experiments we did, we selected here the most relevant. We evaluate and select the setup on validation data in this part.

Firstly, we focus on the impact of the setup on the results thanks to the B_i experiments. The experiments B0 and B3

show that target some areas into the features thank to 6 masks is more efficient than to only take globally the full image. The length of the letter-n-grams but also the quantity of the letter-n-grams has a surprising effect on the measures. The recall difference between B1 and B3 is proportional to the number of available output. So, the B1 experiment with the less output using unigrams has a best recall, precision and F1-score than B3. B1 predicts 3 times more unigrams than the ground-truth and B3 predicts 3 times more unigrams too and 7 times more letter-n-grams than the ground-truth. To analyze the effects using the normalized frequency plus the offset values (SIG(Off)) of the output targets we can compare B4 and B6 experiments. The encoding of each output with a binary answer (SIG(1)) is more efficient than the SIG(Off). B6 loses 8.5% of precision and with a similar recall, but the idea of the quantity of letter-n-grams is also lost. To compute the cost, the binary cross-entropy used in B6 provides a better recall of 36.66%, but a dramatically better precision of about 54.17% than using the Kullback-Liebler function (B5).

With regards to these results, the best setup operates the binary cross-entropy as cost function, using the normalized frequency with the offset of the targets and the features filtered with 6 masks based on the input image. With a recall above 60%, we show that it is possible to build a letter-n-grams vector to represent a word image is possible.

The focus on E_i transfer learning experiments shows that the size of the FC9 layer has no noteworthy effect on the results. Moreover, the THR configuration promotes the precision measure but omits the recall. Overall, all E_i experiments carried out on RIMES and GW reach only the half of the measures estimated on ESP. The best recall measure is obtained with all n-grams and 2,048 units on FC9 (E6), and as on the B_i experiments, following the F1-score, the best results are obtained with the unigrams (E0) and 1,2-grams (E1).

B. Language Model Experiments

Table IV shows the results for the language model component. It clearly appears that the system success to recognize the correct word when the quantity of the letter-n-grams is increased. The difference of the WRR between unigrams (Experiment D1) and bigrams (D2) is more significant with 45.56% than bigrams (D2) and trigrams (D3) with 10.95%. So, if the size of letter-n-grams is the main difficulty for the encoder component, the using of all bigrams can be a good alternative because it represents only 15% of the 1,2,3 letter-n-grams size. The vocabulary of ESP is built primarily with named entities as it is extracted from 18^{th} century Spanish wedding registers. This explains the low lexical coverage for the transductive transfer learning. Nevertheless, CRR is greater than 90% except with RM (D4) and WRR outperforms the lexical coverage. Thus, with more information thanks to the presence of all letter-n-grams, words are more easily generated. We proposed more experiments applied to RIMES and the vocabulary of registers of Italian Comedy in [20].

C. End-to-end Model Experiments

The previous experiments show that the best recall is obtained with unigrams target for the encoder whereas the CRR and WRR are obtained with 1,2,3 letter-n-grams. So, for the last experiments, we combine all best setups for each component. The test is applied to the ESP vocabulary where the frequency of each word is equiprobable. Table V presents the last experiments. The differences between Table III and Table V for the encoder measures come from balancing the vocabulary. We select different experiments with the unigrams as the pivot (E0-D1) because it obtained the best F1-score on the encoder; the 1,2 letter-n-grams with or without transfer (E1-D2, B2-D2) because this is the best compromise between the two components; and the 1,2,3 letter-n-grams combining different encoder architectures (E2-D3, E3-D3, E6-D3) because it obtained the best result on the decoder.

At first sight, we find that the results obtained on the encoder part are similar to these obtained during the training step. To help the system, we realize a training step through the encoder and decoder component from previous weights in order to apply an end to end back-propagation. The fine-tuning applied improves the performance of the encoder when all images are used but degrades the decoder when we realized it with RIMES and GW, so the ESP vocabulary is forgotten. The fine-tuning experiments with E3-D3 increases the F1-score by 8.66%, whereas, with E1-D2, the results remain the same. Detailed analysis shows that the noisy output of the encoder prevents the decoder to work well. For example, the encoder predicts several symbols of the beginning and end of the word or no end symbol. Moreover, except for common words such as “de”, the encoder predicts 7 times more letter-n-grams, which makes it more difficult to recognize words. To improve the performances and restrain the impact of the noisy on the decoder, it would be possible to limit the selection of n more active letter-n-grams and force others at 0.

VI. Conclusion

In this paper, we presented a transfer learning approach dealing with the lack of ground truth, for historical handwriting recognition. We chose an approach that learns independently optical and language models, connected by a vector that did not use an indication of the order of characters in words rather than usual sequence-to-sequence approaches to facilitate the transfer learning. We selected some available resources with the same domain, from the same time period and with the same alphabet to achieve the transfer learning.

The optical model with a bag of letter-n-grams as target provides a good recall greater than 70% for the case where the source and target datasets were the same. Regarding the learning with other resources, it seems that the amount of data used for the training step is still too low. But, experiments have shown that the architecture of a system decomposed with two independent models could be a solution. Regarding on our results through the two components, optical and language models, in addition to increasing the amount of data for the encoder, it could be interesting to add an attention mechanism

TABLE III

OPTICAL MODEL RESULTS. IN THE TOP PART, EVALUATION OF THE BEST SETUP ON ESP TRAIN DATASET. IN THE BOTTOM PART, RESULTS OBTAIN ON TRANSFER LEARNING WITH GW AND RIMES TRAIN DATASETS, AND ESP VAL DATASET. (%)

Expe. Id	FC9	Config	Funct. cost	Nb masks	Letter n-grams	Rec.	Pre.	F1-Score	Acc.
B0	1024	SIG (off)	Bin.	1	1,2,3	47.28	29.03	35.84	99.89
B1	1024	SIG (off)	Bin.	6	1	66.48	**89.61**	**76.33**	98.33
B2	1024	SIG (off)	Bin.	6	1,2	61.85	79.03	69.40	99.68
B3	1024	SIG (off)	Bin.	6	1,2,3	58.16	72.40	64.50	99.91
B4	2048	SIG (1)	Bin.	6	1,2,3	**93.16**	74.39	82.72	99.96
B5	2048	SIG (off)	Kull	6	1,2,3	55.68	11.72	19.10	99.90
B6	2048	SIG (off)	Bin	6	1,2,3	92.34	65.89	76.91	99.96
E0	1024	SIG (off)	Bin.	6	1	29.09	35.48	**31.97**	95.48
E1	1024	SIG (off)	Bin.	6	1.2	22.51	34.14	27.13	99.16
E2	1024	SIG (off)	Bin.	6	1,2,3	32.43	18.77	23.78	99.88
E3	1024	SIG (off)	MSE	6	1,2,3	34.80	10.73	16.27	99.88
E6	2048	SIG (off)	Bin.	6	1,2,3	**45.16**	10.58	17.15	99.90
E8	2048	THR (off)	Bin.	6	1,2,3	4.77	**59.32**	8.83	98.76

TABLE IV

LANGUAGE MODEL RESULTS. CRR AND WRR WITH THE CASE-SENSITIVITY AND THE DICTIONARY ON ESP TEST DATASET. (%)

Train	Expe. Id	Letter n-grams	Lexical Coverage	CRR	WRR	WRR dict.
GB+ESP	D1	1		85.46	44.43	37.87
GB+ESP	D2	1,2	85.94	92.00	62.31	59.49
GB+ESP	D3	1,2,3		**98.25**	**91.61**	**62.92**
GB+ESP +GW+RM	D4	1,2,3	86.10	98.14	90.48	61.94
GB	D5	1,2,3	15.96	88.18	54.35	45.48
RM	D6	1,2,3	7.27	67.82	14.03	8.87
GB+RM	D7	1,2,3	17.37	87.4	41.77	**51.61**
GB+RM+GW	D8	1,2,3	17.69	**88.77**	**57.1**	44.35
Wiki	D9	1,2,3	0.0	78.89	27.26	24.52

TABLE V

RESULTS OF TRANSFER LEARNING ON ESP TEST DATASET WITH THE TWO COMPONENTS PLACED END TO END: SAVED WEIGHTS FROM PREVIOUS EXPERIMENTS ARE USED TO INITIALIZE THE COMPLETE SYSTEM. (%)

Exp.		Encoder				Decoder
Enc.	**Dec.**	**Rec.**	**Pre.**	**F1-Sc**	**Acc.**	**CRR dict.**
E0	D1	29.78	32.15	30.83	94.73	27.01
E1	D2	29.47	29.75	29.54	99.14	**28.07**
E2	D3	32.64	11.03	16.43	99.84	24.39
E3	D3	40.22	3.42	6.27	99.86	25.70
E6	D3	33.87	9.71	15.02	99.85	17.23
E3	D7	32.73	11.02	16.44	99.85	20.24
B2	D2	45.88	**43.61**	44.72	99.35	**28.02**

on the decoder inputs, or to customize the decoder to sort the useful letter-n-grams. An other solution can be to modify the encoder architecture but keeping the concept of the projection in the common non-latent space. There is still some works to succeed in the recognition of new digitized documents from multilingual and multi-period resources.

REFERENCES

[1] F. Cloppet, V. Eglin, V. C. Kieu, D. Stutzmann, and N. Vincent, "ICFHR2016 Competition on Classification of Medieval Handwritings in Latin Script," in *Proc. of ICFHR*, 2016, pp. 590–595.

[2] J. A. Sanchez, V. Romero, A. H. Toselli, M. Villegas, and E. Vidal, "ICDAR2017 Competition on Handwritten Text Recognition on the READ Dataset," in *Proc. of ICDAR*, 2017, pp. 1383–1388.

[3] P. Voigtlaender, P. Doetsch, and H. Ney, "Handwriting recognition with large multidimensional long short-term memory recurrent neural networks," in *Proc. of ICFHR*, 2016, pp. 228–233.

[4] E. Granell, E. Chammas, L. Likforman-Sulem, C.-D. Martínez-Hinarejos, C. Mokbel, and B.-I. Cîrstea, "Transcription of spanish historical handwritten documents with deep neural networks," *Journal of Imaging*, vol. 4, no. 1, p. 15, 2018.

[5] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *Proc. of ICML*, 2006, pp. 369–376.

[6] T. Bluche, J. Louradour, and R. Messina, "Scan, attend and read: End-to-end handwritten paragraph recognition with MDLSTM attention," in *Proc. of ICDAR*, 2017.

[7] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. on KDE*, vol. 22, no. 10, pp. 1345–1359, 2010.

[8] J. Lladós, M. Rusiñol, A. Fornés, D. Fernández, and A. Dutta, "On the influence of word representations for handwritten word spotting in historical documents," *International Journal of PRAI*, vol. 26, no. 05, pp. 1 263 002–1–25, 2012.

[9] H. Nakayama and N. Nishida, "Zero-resource machine translation by multimodal encoder–decoder network with multimedia pivot," *Machine Translation*, vol. 31, no. 1-2, pp. 49–64, 2017.

[10] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder–decoder approaches," in *Proc. of SSST*, 2014, pp. 103–111.

[11] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," in *Proc. of CVPR*, 2015, pp. 3156–3164.

[12] S. Bengio and G. Heigold, "Word embeddings for speech recognition," in *Proc. of Interspeech*, 2014.

[13] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *Proc. of ICLR*, 2014.

[14] V. Nair and G. E. Hinton, "Rectified Linear Units Improve Restricted Boltzmann Machines," in *Proc. of ICML*, 2010, pp. 807–814.

[15] E. Grosicki and H. El-Abed, "Icdar 2011 - french handwriting recognition competition," in *Proc. of ICDAR*, 2011, pp. 1459–1463.

[16] A. Fischer, A. Keller, V. Frinken, and H. Bunke, "Lexicon-free handwritten word spotting using character HMMs," *Pattern Recognition Letters*, vol. 33, no. 7, pp. 934–942, 2012.

[17] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Trans. of ACL*, vol. 5, no. 1, pp. 135–146, 2017.

[18] V. Frinken, A. Fischer, and C.-D. Martínez-Hinarejos, "Handwriting recognition in historical documents using very large vocabularies," in *Proc. of HIP*, 2013, pp. 67–72.

[19] A. Granet, B. Hervy, G. Roman-Jimenez, M. Hachicha, E. Morin, H. Mouchre, S. Quiniou, G. Raschia, F. Rubellin, and C. Viard-Gaudin, "Crowdsourcing-based Annotation of the Accounting Registers of the Italian Comedy," in *Proc. of LREC*, 2018.

[20] A. Granet, E. Morin, H. Mouchère, S. Quiniou, and C. Viard-Gaudin, "Transfer Learning for a Letter-Ngrams to Word Decoder in the Context of Historical Handwriting Recognition with Scarce Resources," in *Proc. of COLING*, 2018.