Transfer Learning for a Letter-Ngrams to Word Decoder in the Context of Historical Handwriting Recognition with Scarce Resources

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

Transfer Learning for a Letter-Ngrams to Word Decoder in the Context of Historical Handwriting Recognition with Scarce Resources

Adeline GRANET, Emmanuel MORIN, Harold MOUCHÈRE, Solen QUINIOU, Christian VIARD-GAUDIN

LS2N, UMR CNRS 6004, Université de Nantes, France
firstname.lastname@ls2n.fr

Abstract

Lack of data can be an issue when beginning a new study on historical handwritten documents. In order to deal with this, we present the character-based decoder part of a multilingual approach based on transductive transfer learning for a historical handwriting recognition task on Italian Comedy Registers. The decoder must build a sequence of characters that corresponds to a word from a vector of letter-ngrams. As learning data, we created a new dataset from untapped resources that covers the same domain and period of our Italian Comedy data, as well as resources from common domains, periods, or languages. We obtain a 97.42% Character Recognition Rate and a 86.57% Word Recognition Rate on our Italian Comedy data, despite a lexical coverage of 67% between the Italian Comedy data and the training data. These results show that an efficient system can be obtained by a carefully selecting the datasets used for the transfer learning.

1 Introduction

An increasing amount of handwritten historical documents is becoming digitally available, as a mean for preserving this heritage and making it accessible to all. However, digitization is not sufficient to make the documents usable: it is necessary to extract informations in order to index them. Researchers and historians in the humanities and in the social sciences need to be able to query them and find answers rapidly. Therefore, new projects involve the domains of language processing, document recognition, and information retrieval for historical studies.

In the field of Natural Language Processing (NLP) for historical documents, the main challenge is currently to analyse and normalize the spelling of texts (Garrette and Alpert-Abrams, 2016; Bollmann et al., 2017) whereas challenges of Computer Vision for historical documents are more diverse: they involve segmentation into words, lines, or paragraphs, as well as keyword spotting, or handwriting recognition (HWR). In recent years, the number of competitions regarding historical documents has increased (Cloppet et al., 2016; Pratikakis et al., 2016; Sanchez et al., 2017). 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.

Lately, the trend in HWR is the use of deep neural networks and, more recently, the integration of attention models in the networks (Bluche et al., 2017). The handwriting systems are built with a Multidimensional Long Short-Term Memory (MDLSTM) network or even with a Convolutional Recurrent Neural Network (CRNN) stacked with a Bidirectional Long Short-Term Memory (BLSTM) (Granell et al., 2018). The neural network training includes a Connectionist Temporal Classification (CTC) cost function proposed by (Graves et al., 2006). Those approaches enable the use of all the available contexts. To decode sequences, there are several strategies: 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 data.

We are now working on a new resource (the financial records of the Italian Comedy) with no ground-truth. The changing layout of these documents, the multilingualism (French and Italian), and their special

writing make them more complex to study. Moreover, an annotation step would be time-consuming for experts to create a ground-truth. Therefore, transductive transfer learning 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 (Pan and Yang, 2010). 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 (Lladós et al., 2012) or translation models with multimodal systems such as (Nakayama and Nishida, 2017).

The standard of machine translation is an encoder-decoder system based on a recurrent neural network (Cho et al., 2014). The first component encodes the source language into a fixed vector and the last one decodes the sequence into the target language. Similarly, (Vinyals et al., 2015) 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, using the last hidden layer from GoogleNet, and a LSTM model generating the corresponding description. We were inspired by this idea 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-ngrams as in (Huang et al., 2013; Bengio and Heigold, 2014). Here, the transfer learning is performed during the training step that uses available labelled image data. Then, the decoder is built with a recurrent layer in order to generate a sequence of characters from the vector of letter-ngrams. Here, a second transfer learning is independently made on the vocabulary. Contrary to the machine translation and description generator works, the sub-networks are trained independently from each other.

In this paper, we investigate the robustness of the decoder part to infer rules on the order of characters from the letter-ngrams vector only. Within this context, we present three contributions:

1. we propose a character-based decoder and evaluate it on the historical vocabulary of the Italian Comedy;
2. we show that historical resources are more appropriate than Wikipedia to complement the vocabulary of historical documents;
3. we also show that the transfer learning across languages and periods of time works based on a bag of letter-ngrams.

This paper is organized as follows. We describe our transfer model within the encoder and decoder components in Section 2. Section 3 presents the datasets used and their preprocessing. The experimental setups are presented in Section 4 which is followed by the report of our experiments and results in Section 5.

2 Transfer Models

In this section, we present the architecture of our system. We aim to build a handwriting recognition system for multilingual historical documents using multiple resources with different languages and different creation periods.

2.1 Prior Work

Our first approach was based on a BLSTM-CTC network intergrating the transfer learning (Granet et al., 2018b). Firstly, an image is provided to a fully convolutional neural network to automatically extract features. Then, the extracted features were given to a BLSTM network. The cost was computed by the CTC activation function. The CTC enables the use of an unsegmented input image at a word or a character level and provides a sequence of characters as output. We implemented and evaluated this system with a training and test data from the same resource and, then, used the system on unknown data. The model did not use a dictionary nor a language model during the decoding. The resources were in French, Spanish, and English for the training set and in French, and Italian for the target datatest. The results of this system showed that a language model was learned implicitly to generate the sequence of characters and that the correctly recognized words were words close from one language to the other. Therefore, without a specialized training on a small part of the data to be transcribed, the transfer learning did not work well. The model allowed to reach only 10% of characters which are correctly recognized.

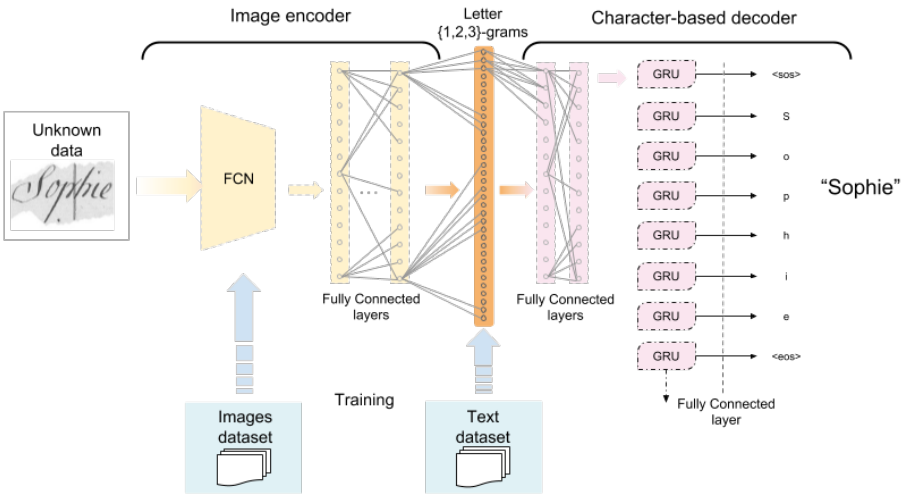

Figure 1: Overview of our encoder-decoder using transfer learning on new historical documents. On the left, the yellow component corresponds to the image encoder from which the features are extracted thanks to a FCN and two connected layers with 1,024 units each and a vector of letter-ngrams is built. On the right, the pink component corresponds to the sequence generation decoder that provides the word, character by character, corresponding to the input vector.

From these results, we concluded that a BLSTM-CTC network was unsuitable for the transfer learning case.

Word classification could have been a solution. Nevertheless, in the field of transfer learning, this strategy quickly reaches its limits when working on target data with a closed vocabulary with many named entities, and a significant number of out-of-vocabulary words. Therefore, classification could not be applied to our data.

2.2 Encoder-Decoder for Transfer Learning

Based on the work on image description generation, we define a new system that is divided into two complementary components: an image encoder and a character decoder, 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 or word. The interface vector that links these two parts is a bag of characters or letter n-grams. We estimated the set of letter-ngrams on all resources used for both the training data and the transfer target data. The originality in using a letter-ngrams 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 character-based decoder to generate corresponding words. Nonetheless, it is interesting to give informations on the whole system to understand the overall approach.

Image Encoder We define our encoder as shown on the left part of Figure 1:

- a fully convolutional network;
- 2 fully connected layers with a ReLU activation function (Nair and Hinton, 2010) and 1,024 hidden units each;
- one last fully connected layer with a sigmoid and L+1 units where L is the number of estimated letter-ngrams and the additional unitias a joker if the letter-ngram is unknown.

The last layer with the sigmoid activation function allows us to have a probability for each letter-ngram independently of the others.

Character-Based Decoder We turned to neural encoder-decoder architectures in NLP as it enables us to map one sequence to another one without having them to have the same length or word order such as in machine translation (Bahdanau et al., 2014) and description generation (Vinyals et al., 2015). We define the architecture for the character-based decoder as shown on the right part of Figure 1:

- one fully connected layer with a ReLU activation function;
- one Gated Recurrent Unit layer (GRU) (Cho et al., 2014);
- one fully connected layer with a softmax activation function.

These components were designed to be the most minimalist. Moreover, they do not use an embedding layer since it might interfer with the transfer learning that has to deal with different languages.

For the sequence generation task, we used a recurrent layer to add a temporality followed by a softmax layer that gives one character at the time until the word end symbol *J* is given. We chose not to include a bidirectional recurrent layer contrary to encoder-decoder models that are used in machine translation. Since our character-based decoder uses a vector without any order indication on characters from words, all the information and context are included in the vector. Therefore, a bidirectional recurrent layer is useless. Overall the system must infer sequences of characters from the vector content. If we analyse the simple short word ”are”, it consists of 3 unigrams, 4 bigrams, and 3 trigrams. This decomposition includes the n-grams with the beginning and end symbols of the word. So the decoder needs to deduce the order of characters from the letter-ngrams.

3 Dataset Collection and Statistics

In this section, we present the resources used in our experiments as well as their representation.

3.1 Target Data: Financial Records of the Italian Comedy

The studied documents consist of more than 28,000 pages of financial records of the Italian Comedy from 1716 to 1791. These pages give the list of daily, monthly and annual receipts, with details on the composition of the troupe of actors for each season. We noticed several evolutions over the years:

- the language switches from Italian (with several dialects) to French;
- the structure changes but keeps the same amount of information;
- the writing style is irregular at the beginning of the century but, after that, a cashier was nominated so the style became stable.

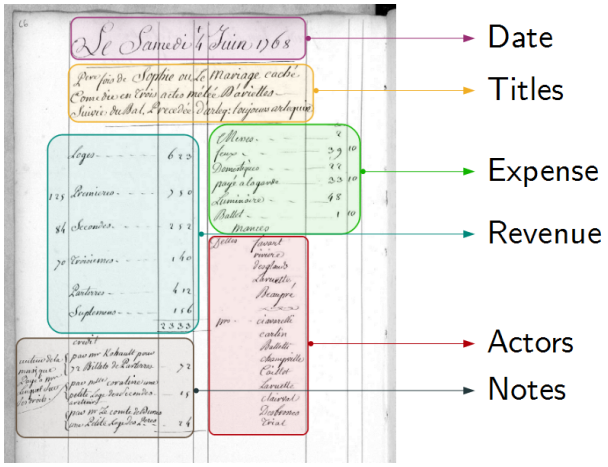

Figure 2: Example of a financial daily record of the Italian Comedy with fields identified.

Figure 2 shows all the information that can be found in a daily record such as the date, the titles of the plays, the revenues and expenses, the actor names, and also some notes. Our study focuses on the title field which contains the list of plays that were performed that day. This list can be extended by some indications as if it was the first performance of the play, if the performance took place at the king’s court, or if a new actor started to play. An example of this additional information is shown in Figure 2: the title field explains that it was a performance of *“Sophie ou le mariage caché”* (lit. *“Sophie or the secret marriage”*) which was a comedy in three acts preceded by *“Arlequin toujours Arlequin”* (lit. *“Arlequin always Arlequin”*). Moreover, a title can be written in many ways with the complete title or a shorter title. Titles were mostly constituted by named entities such as ”Raton and Rosette“ or ”Zemire and Azor“.

The language and writing style are essential for the decoder part. We explained above that the language can be either Italian or French. Compared to the modern writing of these languages, we note some differences such as special characters like the long form of 's', the evolution of the 'y' spelling to 'i' or using 'i' and 'j' indiscriminately, or some weak and strong abbreviations. We define as weak abbreviation, a symbol that replaces few characters of a word, while a strong abbreviation reduces a long title to only two or three words.

Thanks to recent works and a participative annotation website (Granet et al., 2018a), we collected and validated 971 annotated title lines. We used all of these annotations to test our decoder system: this dataset is thereafter called Italian Comedy Registers (ICR).

3.2 Training Datasets

In order to build an efficient decoder, we must carefully select the training datasets. Table 1 presents the vocabulary size for each dataset that are used to train the decoder. We choose to create a new dataset from available digitized historical books, as well as to use three existing image resources (RIMES, George Washington, and Los Esposalles) and the French version of Wikipedia since it is a widely used resource in NLP tasks.

	Google IC	RIMES	Los Esposalles	Georges Washington	Wikipedia Fr	IC Registers
Train	26,573	4,477	2,565	660	24,456	0
Validation	2,953	1,578	629	521	3,843	0
Test	0	1,627	629	431	1.928	1,431

Table 1: Vocabulary size of the training, validation, and test sets of each dataset.

New Dataset (Frinken et al., 2013) 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 new dataset from digitized books of the same century of our data, with a similar closed vocabulary. To preserve the cultural heritage for future generations, copies of books have been digitized, automatically OCRized, and distributed via *Google Book*, for example. We selected 23 books dealing with the Italian Comedy from (and published in) the 18^{th} century. These books include bilingual scripts of plays (French and Italian), collections of plays, and anecdotal stories of the Italian theater. The extracted texts were cleaned to remove the noise such as the structure of the texts and special characters. This new resource, called Google Italian Comedy (GIC), has the advantage of having the same closed vocabulary with specific writings as in the records of the Italian Comedy.

Existing Datasets For our transfer learning approach, we carried out experiments with four available datasets of images, sharing at least one common feature with our target data, as well as Wikipedia to compare the effect of a modern and general resource on the Italian Comedy data:

- RIMES (RM) stands for Recognition and Indexing of Handwritten Documents and Faxes) and is a French database developed to evaluate automatic systems that recognizes and indexes handwritten letters (Grosicki and El-Abed, 2011);
- George Washington (GW) is an English database created from the George Washington Papers at the Library of Congress (Fischer et al., 2012);
- Los Esposalles (ESP) is a Spanish database compiled from a marriage licence book collection from the 15^th and 17^th centuries (Fischer et al., 2012);
- Wikipedia Fr data (Wiki), as used and distributed by (Bojanowski et al., 2017), provides all words whose frequency is greater than 5 in Wikipedia. From this dataset, we randomly selected 30 000 words.

To our knowledge, no other experiments dedicated to the construction of a decoder use the resources RM and ESP which are initially image resources. However, we found it interesting to observe the impact

of trained models because ESP is mainly composed of named entities, and RM is a contemporary base with an administrative vocabulary. Therefore, they are far from our Italian Comedy data.

3.3 Representation

Preprocessing of the Training Datasets In all the datasets, we replaced the diacritical marks by their simple form, such as {é, è} are *e* characters, and the typical long form of *s* from the 18^{th} as a short form for the decoder input and also the target data. All words are case sensitive. For GIC only, text lines were split on whitespaces and punctuation marks. In contrast, ICR title lines were only split on whitespaces and we kept with the punctuations in the data.

Letter-Ngrams As suggested by (Bengio and Heigold, 2014; Vania and Lopez, 2017), we use letter-ngrams as a pivot in our multimodal encoder-decoder. Initially, the authors selected the 50 000 most popular letter-ngrams to represent words. Adding *[* and *]* to symbolize the beginning and end of a word, respectively, we computed all the possible letter-ngrams with a maximum length of 3 on all the datasets. Therefore, we selected around 12,500 letter-ngrams. The large difference in the number of ngrams comes from the choice of keeping only letter-ngrams appearing in at least 2 different datasets: a *joker* was created to replace the non-selected letter-ngrams. With our example in Figure 1, the name *Sophie* is thus decomposed as {S,o,p,[S,So,op,[So ,. . . ,ie],e]}.

Regarding the choice of having at most 3-grams, this allows to give the system an idea on the order in which the characters are arranged in the word. In the case where there are only unigrams, the learning process is longer and tends towards over-learning to be able to have sequences of characters forming words. Including 4-grams into the vector would dramatically increase its length and would make it more difficult to select the ngrams recognized in a word by the optical model.

Decoder Input and Output The input of the decoder input consists of the normalized frequencies of the n-grams in the input word. The frequencies are normalized by the length of the word to provide a vector with values between 0 and 1, corresponding to probabilities for each n-gram. Using frequencies enables us to hold information about the word length and to compensate the lost temporality. For the sequence generation, we have 79 output units for all the upper and lower characters, the digits, the punctuation symbols including the space, the start and the end of word. The last added symbol is a *blank* label that allows the system to provide an out-of-character answer after the end of a word.

4 Experimental Setup

This section presents the experimental setup of our decoder as well as the evaluation metrics we used.

4.1 Decoder Training

The size of the datasets varies from 660k to 25k words. For this reason, we trained the system with either one dataset or several datasets, except for GW. Its size is too small so we always combined it with other datasets.

To avoid overfitting, we used the classical technique of early stopping. The training process was stopped after five epochs if the validation loss did not decrease anymore. For the sequence generation, we used the loss defined by Equation (1) that computes the multi-classification with a normalization through all the output units.

$$L_i = -\sum_j t_{i,j} \log(p_{i,j}) \tag{1}$$

4.2 Decoder Parameters

We defined the structure of the decoder part with 1,024 units in the fully connected layer to extract features, 500 hidden units in the GRU layer, and 79 units with the Softmax activation function on the last layer. To be sure to stay in the same configuration, we used the Adam function that controls the learning rate which is initialized at 0.0001 for the stochastic optimization. As we have less data than the others experiments of the state-of-the-art, our rate is lower. 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.

4.3 Evaluation Metrics

To evaluate the performance of our system, we used the recognition rate at the character level (*CRR*) and at the word level (*WRR*). The Character Recognition Rate is defined as:

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

with N the number of characters of the reference words, $Subs$ the number of character substitutions, $Dels$ the number of character deletions, and Ins the number of character insertions. The Word Recognition Rate corresponds to the Word Accuracy and is defined as the number of correctly recognized words divided by the total number of words to recognize. The WRR is computed with and without using the Levenshtein edit distance to correct the output sequence with 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.

We also computed the lexical coverage of a training set with respect to the test set. It is defined as the number of words shared between the training set and the test set, normalized by the size of the test vocabulary. This metric defines a high boundary for the word recognition using a dictionary. Since the dictionary is built on the training and the validation sets, out-of-vocabulary words of the test set could not be correctly recognized.

4.4 Baseline Approach

As a baseline approach we used a vector of unigrams to represent each word in the decoder input. We added the symbols for the beginning and the end of words so each word is represented by 2 bigrams and several unigrams.

5 Results and Analysis

In this section, we present the results of our experiments for the sequence generation decoder, in Table 2 which is divided into 3 parts following the test resources. We performed three types of experiments corresponding to the sub-parts of the table, for each validation and test data:

1. the same resource for the training and the test parts (training domain = target domain);
2. adding other resources during the training part (training domain $>$ target domain);
3. different resources for the training and the test parts, *i.e.* transductive transfer learning (training domain $\neq$ target domain).

The experiments are done with the artificial true n-gram vectors so the input is not noisy. Overall, the results in Table 2 prove that using letter-ngrams is a better choice than only using unigrams and the use of a dictionary degrades the results. In all the experiments, the models provide a CRR greater than 75% on the validation sets and greater that 85% on the test sets, even if the lexical coverage is less than 25%. Furthermore, WRR also exceed the lexical coverage. To decode the ICR vocabulary, the model must be trained with at least GIC as a resource. All following analyses are presented on the validation sets to evaluate the best ressources to train the model. The last analysis relates to the test sets.

Baseline Whether experimenting with RM on RM (validation and test set) or with GIC on GIC (validation set) and ICR (test set), we found that the results obtained with letter-ngrams are better than with unigrams. On GIC, letter-ngrams dramatically outperformed unigrams with a 70% relative increase of WRR without the dictionary. It should also be noted that only 5% of the characters were incorrectly recognized with letter-ngrams. These results corroborate other studies using trigrams such as (Vania and Lopez, 2017).

Test	Train	Expe. Id	Letter ngrams	Lexical Coverage (%)	Validation			Test		
					CRR	WRR	WRR dict.	CRR	WRR	WRR dict.
ICR	GIC	1	1	65.57	69.11	13.05	14.65	69.27	14.54	10.83
	GIC	2	1,2,3		95.85	77.83	66.47	97.10	86.22	39.30
	GIC+RM	3	1,2,3	67.53	95.97	78.78	66.17	**97.27**	**86.57**	**39.23**
	GIC+ESP	4	1,2,3	65.83	**96.49**	**85.87**	**67.22**	96.96	81.46	39.09
	GIC+ESP+GW+RM	5	1,2,3	67.65	95.99	77.86	66.37	95.85	79.65	38.25
	RM	6	1,2,3	14.52	74.69	19.49	23.32	79.70	30.42	17.27
	RM+ESP+GW	7	1,2,3	23.39	78.11	25.36	28.44	83.68	40.21	23.99
	Wiki	8	1,2,3	0.0	**83.67**	**41.18**	**42.75**	**87.32**	**41.40**	**25.24**
RM	RM	9	1	75.09	82.66	37.83	27.58	83.97	43.07	28.49
	RM	10	1,2,3		93.95	79.93	37.45	94.72	79.50	37.78
	GIC+RM	11	1,2,3	83.83	**97.32**	**89.04**	**39.94**	**98.25**	**92.0**	**40.49**
	GIC+ESP+GW+RM	12	1,2,3	83.95	96.8	86.18	39.43	96.22	80.73	39.14
	GIC	13	1,2,3	58.55	**93.93**	**75.48**	**36.56**	**95.51**	**81.53**	**38.58**
	GIC+ESP	14	1,2,3	59.04	93.89	74.33	36.05	95.46	80.61	38.15
	Wiki	15	1,2,3	0.0	87.04	65.99	33.31	90.36	67.57	35.20
ESP	GIC+ESP	16	1,2,3	85.94	**98.25**	**91.61**	**62.92**	**98.57**	**91.11**	56.51
	GIC+ESP+GW+RM	17	1,2,3	86.10	98.14	90.48	61.94	98.40	90.79	**57.62**
	GIC	18	1,2,3	15.96	88.18	54.35	45.48	91.68	**65.87**	44.76
	RM	19	1,2,3	7.27	67.82	14.03	8.87	72.83	18.25	12.70
	GIC+RM	20	1,2,3	17.37	87.4	41.77	**51.61**	**92.05**	64.13	**46.51**
	GIC+RM+GW	21	1,2,3	17.69	**88.77**	**57.1**	44.35	91.68	64.60	44.60
	Wiki	22	1,2,3	0.0	78.89	27.26	24.52	84.52	34.28	32.38

Table 2: Sequence generation results. CRR and WRR with/without the dictionary on the Italian Comedy Records (ICR), Rimes (RM), and Los Esposalles (ESP) validation and test datasets. Several experiments are presented: evaluation of the baseline, combination of the resources, and transductive transfer learning with all the resources. The lexical coverage is computed between the training and the test datasets.

Use of a Dictionary Using the dictionary drops performances except for Expe. Id 6, 7, 8, and 20. We notice that when the lexicon coverage is greater than 20%, WRR always remains lower. We notice that a dictionary misleads the decoder when it generates a correct sequence but there is only another form (e.g., plural) of this word, in the dictionary. However, the dictionary sometimes helps to get closer to the original word even if this word does not exist in the dictionary and if the training and the test data are from different periods or different languages. Nevertheless, these cases are too rare to improve CRR with the dictionary.

Results on ICR (our Target Data) We focus on the central part of the ICR experiments, where GIC is combined with other resources. The CRR and WRR results are very similar to those of the training step with only GIC (Expe. Id 2 vs. Expe. Id 3, 4, and 5). Therefore, the increase in the number of resources does not always have a great impact on the performance. Nonetheless, training on GIC and ESP achieved better than GIC and RM despite the fact that languages are mixed.

For the transfer learning process with an out-of-domain vocabulary (Expe. Id 6, 7, and 8), the lexical coverage is extremely low (around 15%). Nevertheless, the system reaches a maximum of 19.49 % WRR using only RM and 41.18 % WRR using Wikipedia, which are the two modern French resources.

Results on Rimes (RM) These results are impressive because Rimes is the only modern image base we used. For transductive transfer learning, without RM during the training step, CRR and WRR reach the same results obtained with only RM as a training data. Moreover, they exceed the lexical coverage of 15%. When we work with historical training data applied to modern data but with the same language, the lexical coverage is greater than using a modern training data and testing on ICR. It seems that the historical spelling is easier to extend to the modern spelling for the decoder. The results obtained with Wikipedia show a CRR 6.89% lower and a WRR 9.49% lower than with GIC, despite the fact that the

vocabulary used in RM and Wikipedia is modern French.

Results on Los Esposalles (ESP) The vocabulary of Los Esposalles 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 (Expe. Id 19) and WRR outperforms the lexical coverage. We did not experiment the decoding with unigrams on this resource because there are too few words. Expe. Id 21 using GIC, GW, and RM shows that this decoder enables to learn a representation from one language (mainly in French) to another.

Analysis on Test Sets Finally, creating and using a new dataset in the field of the Italian Comedy is an interesting approach to decode ICR. This enables a better word reconstruction since the lexical coverage is increased by more than 20%. Among the unknown but well-recognized words can be found light abbreviations such as *"arleq."* instead of *"Arlequin"*. Contrary to the validation results, the best results are obtained thanks to the combination with RM dataset (Expe. Id 3 versus 4). This can be explain by the difference of the vocabulary, the validation set is composed of GIC while the test set is composed of ICR. For the transductive transfer learning process, the results with Wikipedia are similar ($+1.20\%$) to using mixed resources with less data (Expe. Id 7 versus Expe. Id 8) contrary to the results on the validation set, where there is a significant difference around 15% on WRR with and without dictionary.

Error Analysis Table 3 shows some mistakes made by the decoder. For words with a repeated character such as *"cavalcade"* and *"clemence"*, it is common for the system to fail to generate the characters inserted between each repetition. Common mistakes are a permutation between two consecutive characters such as *"suitte"* and one doubled character to replace another one. In the last example, *"Soldat"*, the decoder predicts two start symbols: the second one replaces the first letter of the word. This happens when small resources are used to train the system.

Table 4 shows the distribution of the different types of errors presented in Table 3 for 3 selected experimentations (Expe. Id 2, 5, and 8). The most common mistake is a simple switch between two characters: this represents 65.14% to 88.32% of the errors, depending on the considered experiments. For the experiments including GIC in the training step, the redundant character errors and wrong starting character are similar. They represent only 10% and 20% of all the identified errors. Expe. Id 8 has a high rate of wrong starting characters that can be explained by the lack of lexical coverage between Wiki and ICR. Despite the equivalent quantity of words used for the training step with Wiki than with GIC, the system has difficulties finding out the start of words. Indeed, Wiki has been preprocessed and normalized by removing capitalization so the information seems to be important for the system.

To conclude, we can see that, in Expe. Id 2, there is on average 1.14 character error by word while, in Expe. Id 8, there is on average 1.73 character error by word: that can be explained by the 45.8% drop of WRR between these two experiments.

Type Error	Expe. Id	Correct Word	Retrieved Word
Multiple Char.	4	cavalcade	cacaadade
	3	clemence	ccceeene
Swich Char.	6	suitte	usitte
	5	belle	blll
Starting Char.	5	[diverstissemens]	ddevvestissemens]
	6	Soldat	[ollat

Table 3: Examples of mistakes made by our decoder.

Expe. Id	% Error Redundant Character	% Error Switch	% Error Starting Char.
2	9.13	81.22	9.65
5	4.81	88.32	6.87
8	8.44	65.14	26.42

Table 4: Quantitative analysis of the different error types on 3 ICR results.

6 Conclusion and Future Work

In this paper, we presented a transfer learning approach dealing with the lack of ground truth of our Italian Comedy data, for sequence generation. We selected some available resources with the same domain and from the same time period to achieve the transfer learning. We chose an approach 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.

Our results show that a character-based decoder can infer a word from a letter-ngrams vector. Moreover, the lexical coverage between the training and test data is reached by the word recognition rate and mostly exceeded without the dictionary. The dictionary misleads the decoder rather than helping it because only some forms of a word are included in the dictionary. We demonstrated that transfer learning is more efficient from historical spelling to modern vocabulary than the opposite. The difference in languages implies a low coverage of the words but it still enables a correct decoding of the words as long as the resources come from the same time period. Therefore, the letter-ngrams models are efficient across languages sharing the same alphabet. Nevertheless, the decoder encounters some difficulties with specific words. We might be wondering whether the letter-ngram size limit of 3 is large enough, or whether a higher amount of training data could solve this issue. Despite this, our simple decoder built with only 4 layers obtained good results. This strengthens our idea of looking for new untapped resources instead of relying on resources traditionally used such as Wikipedia.

Currently, we are carrying out experiments with this approach at the word-level only and without punctuation marks. The next step for the decoder would be to extend it to the sentence-level, more specifically to deal with the space character. Initially, we aim at implementing a new handwriting recognition system with an encoder-decoder model. We consider that the decoder presented in this article is operational. Our future work is going to look at the encoder components independently. The main difficulty could be to handle the errors by encoding them in the vector of letter-ngrams. Indeed, unlike the experiments carried out in this paper, the input data of the decoder will be noisy.

References

Dzmitry Bahdanau, KyungHyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate.

Samy Bengio and Georg Heigold. 2014. Word embeddings for speech recognition. In *Proceedings of the 15th Annual Conference of the International Speech Communication Association (Interspeech'14)*, Singapore.

Théodore Bluche, Jérôme Louradour, and Ronaldo Messina. 2017. Scan, Attend and Read: End-to-End Handwritten Paragraph Recognition with MDLSTM Attention. In *Proceedings of the 14th International Conference on Document Analysis and Recognition (ICDAR'17)*, Kyoto, Japan.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association of Computational Linguistics*, 5(1):135–146.

Marcel Bollmann, Joachim Bingel, and Anders Søgaard. 2017. Learning attention for historical text normalization by learning to pronounce. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL'17)*, pages 332–344, Vancouver, Canada, July.

Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the Properties of Neural Machine Translation: Encoder–Decoder Approaches. In *Proceedings of the 8th Workshop on Syntax, Semantics and Structure in Statistical Translation (SSST'14)*, pages 103–111, Doha, Qatar.

Florence Cloppet, Véronique Eglin, Van Cuong Kieu, Dominique Stutzmann, and Nicole Vincent. 2016. ICFHR2016 Competition on Classification of Medieval Handwritings in Latin Script. In *Proceedings of the 15th International Conference on Frontiers in Handwriting Recognition (ICFHR'16)*, pages 590–595, Shenzhen, China.

Andreas Fischer, Andreas Keller, Volkmar Frinken, and Horst Bunke. 2012. Lexicon-free handwritten word spotting using character HMMs. *PRL*, 33(7):934–942.

Volkmar Frinken, Andreas Fischer, and Carlos-D Martínez-Hinarejos. 2013. Handwriting recognition in historical documents using very large vocabularies. In *Proceedings of the 2nd International Workshop on Historical Document Imaging and Processing*, pages 67–72. ACM.

Dan Garrette and Hannah Alpert-Abrams. 2016. An unsupervised model of orthographic variation for historical document transcription. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HTL'16)*, pages 467–472, San Diego, CA, USA.

Emilio Granell, Edgard Chammas, Laurence Likforman-Sulem, Carlos-D Martínez-Hinarejos, Chafic Mokbel, and Bogdan-Ionuţ Cîrstea. 2018. Transcription of spanish historical handwritten documents with deep neural networks. *Journal of Imaging*, 4(1):15.

Adeline Granet, Benjamin Hervy, Geoffrey Roman-Jimenez, Marouane Hachicha, Emmanuel Morin, Harold Mouchère, Solen Quiniou, Guillaume Raschia, Francoise Rubellin, and Christian Viard-Gaudin. 2018a. Crowdsourcing-based Annotation of the Accounting Registers of the Italian Comedy. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan.

Adeline Granet, Emmanuel Morin, Harold Mouchre, Solen Quiniou, and Christian Viard-Gaudin. 2018b. Transfer learning for handwriting recognition on historical documents. In *Proceedings of the 7th International Conference on Pattern Recognition Applications and Methods ICPRAM*, pages 432–439.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. 2006. Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks. In *Proceedings of the 23rd International Conference on Machine Learning (ICML'06)*, pages 369–376, Pittsburgh, PA, USA.

Emmanuele Grosicki and Haikal El-Abed. 2011. ICDAR 2011 - French Handwriting Recognition Competition. In *Proceedings of the 11th International Conference on Document Analysis and Recognition (ICDAR'11)*, pages 1459–1463, Beijing, China.

Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pages 2333–2338. ACM.

Josep Lladós, Marçal Rusiñol, Alicia Fornés, David Fernández, and Anjan Dutta. 2012. On the influence of word representations for handwritten word spotting in historical documents. *IJPRAI*, 26(05):1263002–1–25.

Vinod Nair and Geoffrey E Hinton. 2010. Rectified Linear Units Improve Restricted Boltzmann Machines. In *Proceedings of the 27th international conference on machine learning (ICML'10)*, pages 807–814, Haifa, Israel.

Hideki Nakayama and Noriki Nishida. 2017. Zero-resource machine translation by multimodal encoder–decoder network with multimedia pivot. *Machine Translation*, 31(1-2):49–64.

Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359.

Ioannis Pratikakis, Konstantinos Zagoris, George Barlas, and Basilis Gatos. 2016. ICFHR2016 Handwritten Document Image Binarization Contest (H-DIBCO 2016). In *Proceedings of the 15th International Conference on Frontiers in Handwriting Recognition (ICFHR'16)*, pages 619–623, Shenzhen, China.

Joan Andreu Sanchez, Veronica Romero, Alejandro H Toselli, Mauricio Villegas, and Enrique Vidal. 2017. ICDAR2017 Competition on Handwritten Text Recognition on the READ Dataset. In *Proceedings of the 14th International Conference on Document Analysis and Recognition (ICDAR'17)*, pages 1383–1388, Kyoto, Japan.

Clara Vania and Adam Lopez. 2017. From Characters to Words to in Between: Do We Capture Morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL'17)*, pages 2016–2027, Vancouver, Canada.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*, pages 3156–3164. IEEE.