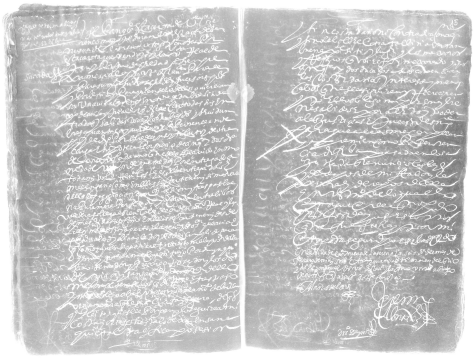
Doctoral Thesis
PhD in Automatic, Electronics and Telecommunication Engineering

Machine Learning for Handwriting Text Recognition in Historical Documents



Author: José Carlos Aradillas Jaramillo
Advisors: Juan José Murillo Fuentes
Pablo Martínez Olmos

Teoría de la Señal y Comunicaciones
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2021

Doctoral Thesis
PhD in Automatic, Electronics and Telecommunication Engineering

Machine Learning for Handwriting Text Recognition in Historical Documents

Author:

José Carlos Aradillas Jaramillo

Advisors:

Juan José Murillo Fuentes

Catedrático de Universidad

Pablo Martínez Olmos

Titular de Universidad

Teoría de la Señal y Comunicaciones
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

2021

Tesis Doctoral: Machine Learning for Handwriting Text Recognition in Historical Documents

Autor: José Carlos Aradillas Jaramillo
Directores: Juan José Murillo Fuentes
Pablo Martínez Olmos

El tribunal nombrado para juzgar la Tesis arriba indicada, compuesto por los siguientes doctores:

Presidente:

Vocales:

Secretario:

acuerdan otorgarle la calificación de:

El Secretario del Tribunal

Fecha:

Agradecimientos

Esta tesis doctoral cierra una etapa muy importante en mi vida. Quiero dedicar las primeras palabras a Juanjo, la persona que me dio la oportunidad de iniciarme en el mundo de la investigación y disfrutar esta experiencia. Mil gracias por tus consejos y ayuda durante todo el camino tanto en la formación investigadora y docente como en todo lo que rodea la realización de una tesis, sin la ayuda del mejor director no hubiera sido posible llegar hasta aquí.

Pablo, mi otro director, tu ayuda también ha sido imprescindible, me has aportado muchas ideas y ofrecido oportunidades por las que siempre te estaré agradecido. Entre ellas el ponerme en contacto con Isabel Valera para realizar una estancia muy importante para mi formación en el Max Planck. Aprovecho para agradecer a Isa y a todo su grupo por acogerme y aportar su granito de arena en esta tesis.

Agradecer la financiación del Gobierno de España mediante la ayuda FPU16 / 04190 que ha posibilitado el desarrollo de la tesis.

He compartido estos años un espacio junto a profesionales en el Departamento de Teoría de la Señal y Comunicaciones, muchos de vosotros fuisteis mis profesores en la etapa previa y ahora sois compañeros y amigos, gracias por todo. Especial agradecimiento a mis compañeros de sala Javi, Irene, Marta, Antonio, Paulina y Juan Antonio por compartir tantos momentos y alegrar las mañanas. También al Prof. Javier Payán por guiarme en la experiencia docente en la asignatura de Comunicaciones Digitales.

Mi lista de amigos ha crecido gracias a los viajes que me ha requerido la tesis. En estos momentos me acuerdo de todos ellos, así como los que han estado ahí desde siempre.

Mi familia ha sido un apoyo importante durante toda mi vida, aunque estos últimos años no he podido estar con vosotros tanto como hubiera querido, sé que

siempre estáis ahí.

A mi hermano Fernando por estar ahí desde que apareciste un día por mi casa, compartir la infancia conmigo y acompañarme de vez en cuando en las escapadas rojiblancas al norte.

La persona que más me ha sufrido y a la vez apoyado frente a las adversidades durante el desarrollo de la tesis ha sido sin duda, Ana, sin ella me habría rendido por el camino. Gracias por compartir tu vida conmigo, disfrutar de tantas experiencias y aguantarme en los momentos difíciles. Mencionar también el cariño de tus padres y tu familia.

Para terminar, agradecer a mis padres José y Gregoria por su apoyo incondicional a pesar de no haberos dedicado el tiempo que os merecéis durante estos años de tesis. Si he llegado hasta aquí ha sido por vosotros gracias a todo lo que me habéis inculcado y ayudado desde pequeño y en las distintas etapas.

Acknoledgements

This doctoral thesis closes a significant period in my life. I want to dedicate the first words to Juanjo, who allowed me to start in the world of research and enjoy this experience. Thank you very much for your advice and support throughout the journey, both in research and teaching training and in everything that surrounds completing a thesis. Without the help of the best director, it would not have been possible to get here.

Pablo, my other director, your help has also been essential. You have given me many ideas and offered opportunities for which I will always be grateful, including getting in touch with Isabel Valera for a necessary stay for my training at Max Planck. I take this opportunity to thank Isa and her entire group for welcoming me and contributing their bit to this thesis. Acknowledge the funding from the Government of Spain through the grant FPU16 / 04190 that has enabled the development of the thesis.

Over the years, I have shared a space with professionals in the Department of Signal Theory and Communications. Many of you were my teachers in the previous stage, and now you are colleagues and friends, thank you for everything. Special thanks to my roommates Javi, Irene, Marta, Antonio, Paulina, and Juan Antonio for sharing so many moments and brightening the mornings. Also, to Prof. Javier Payán for guiding me in the teaching experience in Digital Communications.

My list of friends has grown thanks to the trips that the thesis has required. In these moments, I remember all of them, as well as those who have been there forever.

My family has been substantial support throughout my life, although I have not been able to be with you as much as I would have liked in recent years. I know that you are always there.

To my brother Fernando for being there since you showed up one day at my house, sharing his childhood with me and accompanying me from time to time on the “rojiblancas” getaways to the north.

The person who has suffered me the most and at the same time supported me in the face of adversity during the development of the thesis has undoubtedly been Ana. Without her, I would have given up on the way. Thank you for sharing your life with me, enjoying many experiences, and putting up with me in difficult times. Also, mention the affection of your parents and your family.

To finish, I would like to thank my parents José and Gregoria, for their unconditional support despite not giving you the time you deserve during these years of the thesis. If I have come this far, it has been because of you, thanks to everything you have instilled in me and helped me since I was little and in the different stages.

Abstract

In this thesis, we focus on the handwriting text recognition task over historical documents that are difficult to read for any person that is not an expert in ancient languages and writing style.

We aim to take advantage and improve the neural networks architectures and techniques that other authors are proposing for handwriting text recognition in modern handwritten documents. These models perform this task very precisely when a large amount of data is available. However, the low availability of labeled data is a widespread problem in historical documents. The type of writing is singular, and it is pretty expensive to hire an expert to transcribe a large number of pages.

After investigating and analyzing the state-of-the-art, we propose the efficient application of methods such as transfer learning and data augmentation. We also contribute an algorithm for purging mislabeled samples that affect the learning of models. Finally, we develop a variational auto encoder method for generating synthetic samples of handwritten text images for data augmentation.

Experiments are performed on various historical handwritten text databases to validate the performance of the proposed algorithms. The various included analyses focus on the evolution of the character and word error rate (CER and WER) as we increase the training dataset.

One of the most important results is the participation in a contest for transcription of historical handwritten text. The organizers provided us with a dataset of documents to train the model, then just a few labeled pages of 5 new documents were handled to adjust the solution further. Finally, the transcription of non-labeled images was requested to evaluate the algorithm. Our method raked second in this contest.

Significant contributions of this dissertation have been published in one journal and two conference papers. One more paper will be submitted to a journal soon. These works are listed below:

- José Carlos Aradillas, Juan José Murillo-Fuentes and Pablo M. Olmos, "Boosting Offline Handwritten Text Recognition in Historical Documents With Few Labeled Lines," in IEEE Access, vol. 9, pp. 76674-76688, 2021, doi: 10.1109/ACCESS.2021.3082689.
- José Carlos Aradillas, Juan José Murillo-Fuentes and Pablo M. Olmos, "Improving offline HTR in small datasets by purging unreliable labels," 2020 17th International Conference on Frontiers in Handwriting Recognition (ICFHR), 2020, pp. 25-30, doi: 10.1109/ICFHR2020.2020.00016.
- José Carlos Aradillas, Juan José Murillo-Fuentes and Pablo M. Olmos, "Boosting Handwriting Text Recognition in Small Databases with Transfer Learning," 2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR), 2018, pp. 429-434, doi: 10.1109/ICFHR-2018.2018.00081.
- José Carlos Aradillas, Isabel Valera, Juan José Murillo-Fuentes and Pablo M. Olmos, "Data Augmentation by Variational Autoencoders for Handwritten Text Recognition," 2021, in preparation.

Resumen

En esta tesis, nos centramos en la tarea handwriting text recognition sobre documentos históricos que presentan cierta dificultad de lectura para cualquier persona que no sea un experto en lenguajes antiguos y estilo de escritura.

Nuestro objetivo es aprovechar y mejorar las arquitecturas y técnicas de deep learning que otros autores están proponiendo para handwriting text recognition en documentos manuscritos modernos. Estos modelos realizan esta tarea con mucha precisión cuando se dispone de una gran cantidad de datos. Sin embargo, la baja disponibilidad de datos etiquetados es un problema generalizado en los documentos históricos. El tipo de escritura es singular y resulta bastante caro contratar a un experto para que transcriba una gran cantidad de páginas.

Tras investigar y analizar el estado del arte, proponemos la aplicación eficiente de métodos como el aprendizaje por transferencia y el aumento de datos. También contribuimos con un algoritmo para eliminar muestras mal etiquetadas que afectan el aprendizaje de modelos. Finalmente, desarrollamos un método basado en Variational AutoEncoders para generar muestras sintéticas de imágenes de texto escritas a mano para el aumento de datos.

Se realizan experimentos en varias bases de datos históricas de texto escrito a mano para validar el rendimiento de los algoritmos propuestos. Los diversos análisis incluidos se centran en la evolución de la tasa de error de caracteres (CER) y palabras (WER) a medida que aumentamos el conjunto de datos de entrenamiento.

Uno de los resultados más importantes es la participación en un concurso de transcripción de textos históricos manuscritos. Los organizadores nos proporcionaron un conjunto de datos de documentos para entrenar el modelo, luego se manejaron solo unas pocas páginas etiquetadas de 5 nuevos documentos para ajustar aún más la solución. Finalmente, solicitan la transcripción de imágenes no

etiquetadas para evaluar el algoritmo. Nuestro método obtuvo el segundo lugar en este concurso.

Se han publicado contribuciones significativas de esta tesis en una revista y en dos artículos de conferencias. Pronto se enviará un artículo más a una revista. Estos trabajos se enumeran a continuación:

- José Carlos Aradillas, Juan José Murillo-Fuentes and Pablo M. Olmos, "Boosting Offline Handwritten Text Recognition in Historical Documents With Few Labeled Lines," in IEEE Access, vol. 9, pp. 76674-76688, 2021, doi: 10.1109/ACCESS.2021.3082689.
- José Carlos Aradillas, Juan José Murillo-Fuentes and Pablo M. Olmos, "Improving offline HTR in small datasets by purging unreliable labels," 2020 17th International Conference on Frontiers in Handwriting Recognition (ICFHR), 2020, pp. 25-30, doi: 10.1109/ICFHR2020.2020.00016.
- José Carlos Aradillas, Juan José Murillo-Fuentes and Pablo M. Olmos, "Boosting Handwriting Text Recognition in Small Databases with Transfer Learning," 2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR), 2018, pp. 429-434, doi: 10.1109/ICFHR-2018.2018.00081.
- José Carlos Aradillas, Isabel Valera, Juan José Murillo-Fuentes and Pablo M. Olmos, "Data Augmentation by Variational Autoencoders for Handwritten Text Recognition," 2021, in preparation.

Contents

1 Introduction

1.1 Motivation

Throughout history, humankind has generated a tremendous amount of knowledge and cultural heritage in text, painting, sculpture, or architecture, among others. The conservation of these works in their different formats has been one of the challenges for professionals such as paleontologists, historians, archaeologists, or restorers since they began to worry about the deterioration of the different works. Apart from conservation, a process of dissemination and transfer of knowledge is also essential. It is needed to provide access to any text document, work of art, or architecture archives, whether public or private, for anyone who requires it.

Only in Europe do we have a vast heritage of historical documents. In the Archives Portal Europe (APE)[1] is available information from millions of archival materials stored in hundreds of archival institutions. For example, in the APE we can find almost 100 archives in Spain, 500 archives in Italy, almost 200 archives in Germany, and more than 250 archives in England. Some of the information one can find in the APE is if the documents in some specific archive are digitalized, or they only have the original documents in physical paper.

Spain has a vast network of historical archives, national, regional, and provincial levels. Data published by the Ministry of Education, Science and Culture (MECyC) in 2014 show that, only in the archives managed at the state level, there are more than 427,000 linear meters of occupied shelving [29]. As a reference, the 9000 meters of shelves of the General Archive of the Indies in Seville contain around 80 million pages of documents dated between the 16th and 18th centuries.

[1] https://www.archivesportaleurope.net/directory

This vast amount of information is the object of the painstaking task of organizing and describing each one of the documents by the archivists. The documentary registration process requires a partial transcription of the document to evaluate its historical context and the events or activities described in it for subsequent cataloging within the archive system. The process is entirely manual since it requires the intervention of an archivist in all its stages. According to statistics from 2013, the access system *online* to the digitized funds of the national archives of the MECyC, called Portal de Archives Españoles (PARES) [28], has more than 30 million digitized documents, of which about 6 million have been cataloged and described.

The primary motivation of this thesis is to investigate and advance in the development of tools that facilitate the tasks of conservation and study of ancient manuscript documents found in historical archives. In particular, we aim at improving the transcription stage, one in the whole pipeline, as described next.

1.2 Handling historical documents

Once the good conservation has been achieved, the primary purpose of the archive is to make these documents accessible to the general public and researchers, in particular, informing about the holdings, sections, series, and existing documents [5]. For this, a set of descriptors are included for each document. Since the descriptors of the documents cannot summarize all the content, there is a massive amount of information that, although it may be decisive for the result of an investigation, is not accessible without reading the documents in full. This can be solved partially by digitizing, e.g., by scanning the documents.

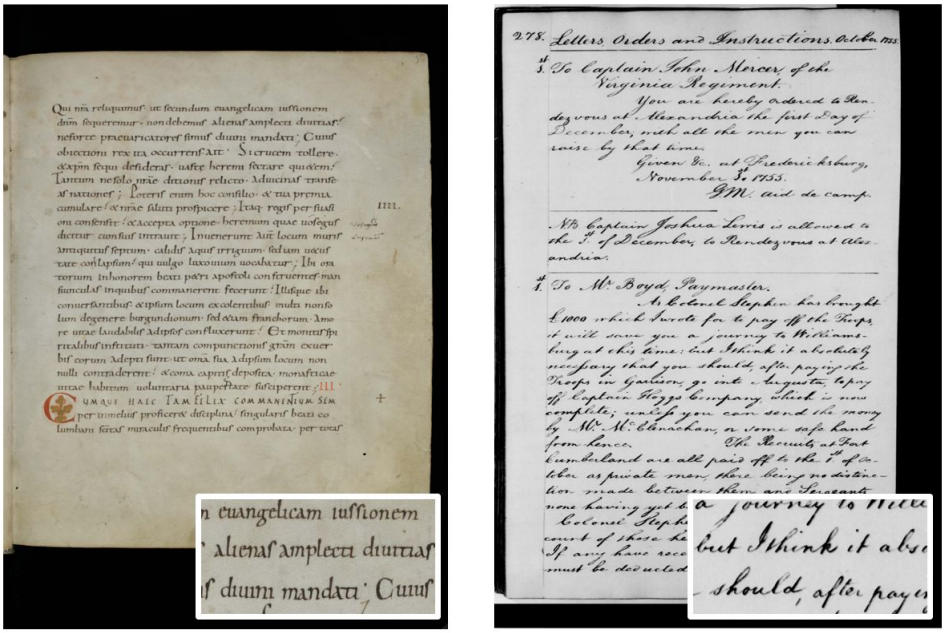
Several aspects must be considered when carrying out a document digitization project, with the main objective of not damaging the original documents. These are:

- **Storage**: how books and historical documents are stored makes a big difference in how well they hold up over time.
- **Handling** books and historical documents must be approached carefully.
- **Scanning**: Choosing a scanning method that does not damage books or historical documents is a top priority.

Once the problem of digitizing documents has been solved by generating scanned images of their pages, the next step would be to facilitate access to the information contained in their texts in the same way that we access the information in digital texts through search engines. This comprises several tasks.

1.3 Handwriting text recognition related tasks

Once we have the image of a page, we need to segment the parts, including text, then proceed with the information retrieval. In Figure 1.1 and Figure 1.2, some examples of historical documents have been included to illustrate the difficulty of this task.



(a) (b)

Figure 1.1 In (a), image from the hagiography *Vita Sancti Galli*. In (b), image from the manuscript notes of George Washington.

1.3.1 Layout analysis and segmentation

An important initial step for information retrieval of historical documents is layout analysis, and segmentation [12, 46, 79, 68]. Historical documents usually have a complex layout, which makes their analysis difficult. [100]. They contain textual elements such as insertions, annotations, and corrections [9]. Moreover, such manuscripts contain decorative elements such as ornaments, illustrations, and comments in the margins and between text lines. See Figure 1.1 for examples of ornaments and side-notes or page numbers. All these elements make the layouts

of the manuscripts very heterogeneous [10]. In some documents, it becomes quite a difficult task. See, for example, in Figure 1.2 some details of pages in the DIVA-HisDB dataset.

(a) CB55, p.25r (b) CB55, p.32v (c) CSG18, p.107 (d) CSG18, p.48 (e) CSG863, p.4 (f) CSG863, p.131

Figure 1.2 Details of pages in the DIVA-HisDB dataset.

Some recent works in this task are [104, 98, 24, 51, 68, 57, 3] where different models are used. In [98, 83] the authors propose a Fully Connected Network (FCN) while in [68] a U-Net is preferred with quite good results. In [3] a Convolutional Neural Network (CNN) is first used to perform further steps later to improve the results, exhibiting good results in the DIVA-HisDB dataset.

It is interesting to remark that we could identify a two-stage process. The first one is where the layout is extracted: we get subregions of the image where we find text in this process. Moreover, a second step is where we segment lines or words of text within these layouts. This segmentation, in turn, can be performed by segmenting the region of the image where a line or word is found or by underlining the text. Images of text lines or words can be binary, gray levels or RGB, and can be saved in different resolutions. In Figure 1.3 we include the result of the text segmentation using the underline approach, while the same page, from the Archivo General de Indias, is segmented by extracting images with lines of text in Figure 1.4.

In some cases, we have images where the text is just part of it. Imagine we want to retrieve the number of a house from a picture of the street. This is known as scene text recognition. In this task, the recognizer tries first to find regions of texts in natural images and afterward recognize the texts in these regions [22, 36].

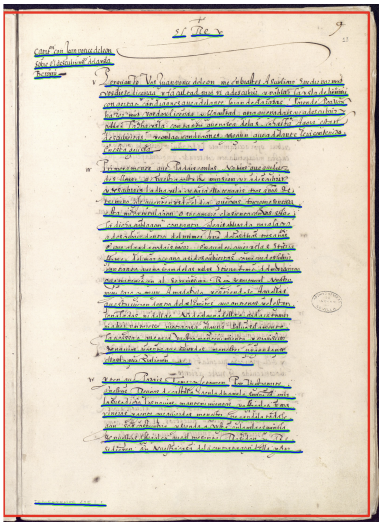


Figure 1.3 Line segmentation using the underline approach, obtained with DhSegment [68] for a document of the General Archive of the Indies. Image from [97].

1.3.2 Information retrieval

Once we have images with pieces of text, we can perform any information retrieval approach. This information can be, e.g.:

- author identification
- search for some information or words
- transcript the full text

Author Identification

Writer and signature identification and verification are other kinds of tasks in the field of handwriting recognition. It consists of identifying the authorship of some text. There are some recent works on this topic [49, 77, 45, 60].

Word spotting

Given a user-defined query, word spotting retrieves a list containing word images relevant to the query. Typically word spotting methods rank all retrieved word images from a given document collection by a specific criterion and sort them by their similarities. [78] Often, this query representation is either an image (Query-by-Example, QbE) or a string defining the sought-after word (Query-by-String, QbS)[93].

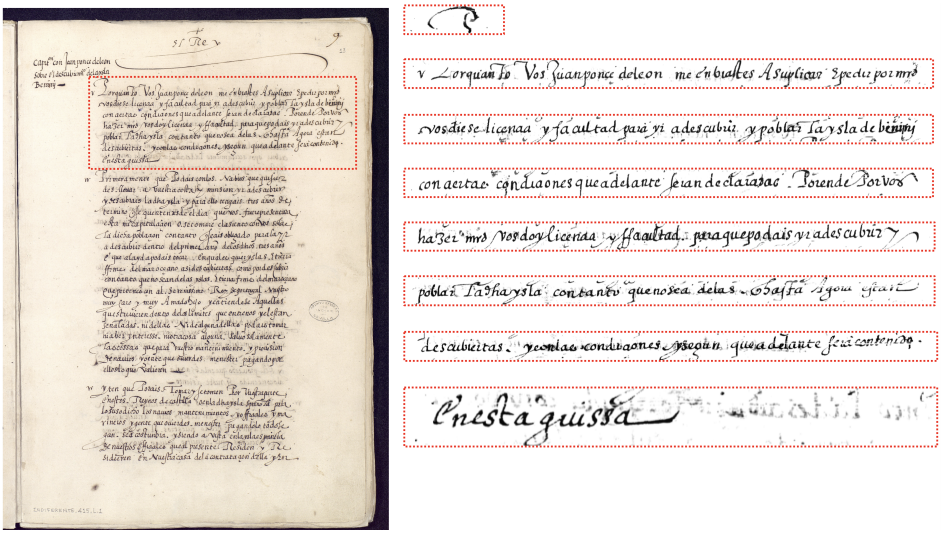


Figure 1.4 Line segmentation providing images of texts, obtained with the Neural Line Segmenter approach [83] for a documente of the General Archive of the Indies. Image from [97].

Some recent works related to this task could be found in [56, 96, 15]. These approaches are helpful when no transcription of the document is available. Note that the problem can be cast as a segmentation one where we are looking for some patterns in the document's images. A demonstrator of one of the proposed tools can be tried at the Carabela Project *http://carabela.prhlt.upv.es/*, where datasets from Archivo General de Indias y al Archivo Histórico Provincial de Cádiz are used.

Text recognition

In this case, we wish to obtain a full-text recognition from an image of a word, line, or paragraph. This problem poses a wide set of difficulties. Here, architectures based on CNN plus Recurrent Neural Network (RNN) and in particular Long Short Term Memory (LSTM) followed by a Connectionist Temporal Classification (CTC) have recently achieved best error rates [99, 16, 74, 94, 7].

The task of text recognition applied to historical documents at line level is the primary motivation of this dissertation. In the next chapter, we give further details about this task.

1.4 Thesis overview

1.4.1 Goal and scope of the thesis

In this thesis, we deal with the Handwriting Text Recognition (HTR) problem applied to ancient documents from the middle ages to the early modern period of history. State-of-the-art techniques have proved to be quite accurate when large homogeneous databases are available. However, the accuracy degrades when tested with new documents that differ from those in the training set. The approaches can be retrained with new labeled (manually transcripted) text from the set of new documents, being the objective to label as few text lines as possible [8, 7]. This is the ultimate goal of this work, to achieve low rates of transcription errors while reducing the number of needed lines to be labeled in the to-be-transcripted dataset. We assume we already have the images with the text and the labels, i.e., the text, and we will work with images of entire lines.

1.4.2 Organization

The organization of this text is as follows. This chapter introduces the importance of the conservation and transference of the knowledge contained in ancient documents and the different analyses and operations we can perform in those documents. The literature about handwriting text recognition applied to current and historical documents is investigated in Chapter 2. The central part of this thesis is focused on the solution of the handwriting text recognition task when we only have a small dataset of a few pages or lines available from some specific author, script, or language to train the Deep Neural Network (DNN) models. In Chapter 3 the datasets used in this thesis are described. In Chapter 4 we analyze how to apply transfer learning from a model previously trained with a considerable amount of data from different writers. In Chapter 5 we take several data augmentation techniques from the literature and investigate how to apply them in combination with the transfer learning method proposed in Chapter 4. In Chapter 6 we propose a method to detect and correct some errors in the labelings of the training set with the aim of improving the performance. In Chapter 7 we propose a method to generate images of handwriting text lines using a Conditional Variational Auto Encoder (C-VAE) model to use the new samples as data augmentation. We compare with the results presented in Chapter 5. The proposed model and the results we achieve in this chapter will be published in a journal paper that is under preparation.

Conclusions and future lines of research are included in Chapter 8

1.4.3 Contributions

The main contributions of this thesis, sketched in Figure 1.5, are[2]:

- We analyze how to perform TL from a massive database to a smaller historical database, determining which model layers need fine-tuning.
- We analyze methods to combine TL and Data Augmentation (DA) efficiently. It is not trivial to apply DA when TL is applied. Authors usually use these techniques by default, and in some cases, it could worsen the performance. We show it in Chapter 5.
- We propose the Corrupted Label Purging (CLP) algorithm to mitigate the effects of incorrect labeling in the training set. The manual transcription of documents by an expert can lead to some errors. We show that these errors in the training set can affect the performance of the models and propose an algorithm to solve it.
- We develop a new Variational Auto Encoder (VAE) approach to generate new images of historical text lines. We use these images as a new data augmentation technique and compare them with the classical ones. We also show how the HTR evaluation method can be used to evaluate other generative methods proposals.

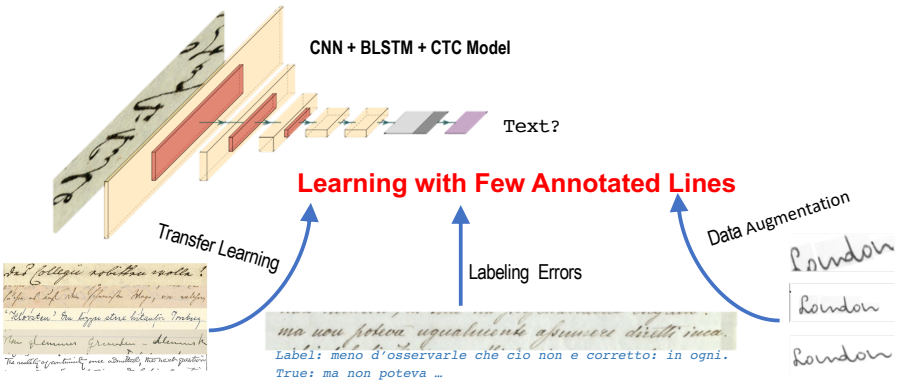


Figure 1.5 Main contributions of the Thesis.

[2] The implementation of the contributions are available in https://github.com/josarajar/

1.4.4 Publications

We have published the contributions of Chapter 4, Chapter 5 and Chapter 6 in one journal paper and two international conference papers listed below:

- José Carlos Aradillas, Juan José Murillo-Fuentes and Pablo M. Olmos, "Boosting Offline Handwritten Text Recognition in Historical Documents With Few Labeled Lines," in IEEE Access, vol. 9, pp. 76674-76688, 2021, doi: 10.1109/ACCESS.2021.3082689.
- José Carlos Aradillas, Juan José Murillo-Fuentes and Pablo M. Olmos, "Improving offline HTR in small datasets by purging unreliable labels," 2020 17th International Conference on Frontiers in Handwriting Recognition (ICFHR), 2020, pp. 25-30, doi: 10.1109/ICFHR2020.2020.00016.
- José Carlos Aradillas, Juan José Murillo-Fuentes and Pablo M. Olmos, "Boosting Handwriting Text Recognition in Small Databases with Transfer Learning," 2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR), 2018, pp. 429-434, doi: 10.1109/ICFHR-2018.2018.00081.

The proposed model and the results we achieve in Chapter 7 will be submitted to a journal, currently under preparation.

- José Carlos Aradillas, Isabel Valera, Juan José Murillo-Fuentes and Pablo M. Olmos, "Data Augmentation by Variational Autoencoders for Handwritten Text Recognition," 2021, in preparation.

2 State of the Art

2.1 Machine learning tools for handwriting recognition

The HTR problem currently falls under the computer vision and pattern recognition areas. In the previous chapter, the different tasks that can be found within the HTR field were introduced. Among these tasks, in this thesis, we focus on the transcription of historical handwritten documents from segmentations of their lines. In the training stage, we also have the image labeled, i.e., the text corresponding to the image.

2.1.1 Handwriting text line recognition

There is a primary classification of the different problems in the machine learning literature depending on the available data. When a set of image-label pairs is available, the task is known as supervised learning which is the case of the HTR problem. When the label, the corresponding text, is not available, the task is referred to as unsupervised learning. Finally, if only scalar reward values are provided, the task is referred to as reinforcement learning. In this thesis, we are usually focused on a supervised task, although in Chapter 7 we take some unsupervised learning tools as an intermediate step to solve the main HTR task.

In the supervised learning task, we have a training set S of input-label pairs $(\mathbf{X},\mathbf{l})$, where $\mathbf{X}$ is an element of the input space $\mathcal{X}$ and $\mathbf{l}$ is an element of the label space $\mathcal{L}$. To measure the performance of the trained model, a test set S' is also provided. It is assumed that both S and S' have been drawn independently from the same distribution $\mathcal{D}_{\mathcal{X}\times\mathcal{L}}$. A portion of the training set is usually extracted to validate the performance of the model during training and prevent overfitting.

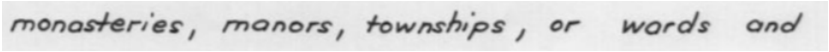


Figure 2.1 Example of image of a text line of the IAM dataset.

In the supervised HTR problem, the format of the input-label data has to be defined. In this text, the input $\mathbf{X}$ consists of an $h \times w \times c$ tensor. The tensor dimensions are referred to as h for the height, w for the width, and c for the number of channels. If we consider an RGB image, $c = 3$ and if we consider a gray image $c = 1$. Each value of the tensor corresponds to the pixel value of an RGB or gray image.

The definition of the label space is a key factor in the way authors face the HTR problem. In the case of HTR over images of full lines of handwriting text, the problem is defined as sequence to sequence problem. As mention in the previous paragraph, the input $\mathbf{X}$ is an image, see Figure 2.1. This image can be interpreted as a sequence of columns vectors, $\mathbf{x}$, with entries the values of the pixels $\mathbf{X} = \left[\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(T)}\right]$, being $T = w$ the length of the input sequence. On the other hand, the label $\mathbf{l}$ is defined as a sequence of characters $\mathbf{l} = \left[l^{(1)}, l^{(2)}, \ldots, l^{(U)}\right]$. The lengths T and U of each input and label sequences are arbitrary so this problem differs from typical supervised task. We can assume that $U \leq T$.

2.1.2 Evaluating the performance

When evaluating the performance of a model in a supervised task, authors usually compare the classification provided by the model with the Ground Truth (GT). When the task consists of identifying each sample with a single label, we can measure the performance by assigning 1 whenever the ground truth label equals the model's output and 0 otherwise. Summing up, by evaluating each sample in the test set and normalizing the outcomes by the number of elements in the set, we estimate the test accuracy. If $h(\mathbf{X}(i))$ is the estimation for the text, given input image $\mathbf{X}(i)$, the accuracy yields,

$$Acc(h,S') = \frac{1}{|S'|} \sum_{i=1}^{|S'|} \begin{cases} 1 & \text{if} \quad h(\mathbf{X}(i)) = \mathbf{l}(i) \\ 0 & \quad\quad \text{otherwise} \end{cases} \tag{2.1}$$

For isolated character or word recognition, the test accuracy could be representative enough. However, in this thesis, we are dealing with sequences of characters or sequences of words. In this case, counting the number of entirely correct sequences is misleading since there would be no difference between a sentence with no correct characters and another with only one error.

If we compare the labeled with the transcribed sequence, we find incorrect characters or words, but sequences might have different lengths due to inserted or deleted characters. Historically in handwriting or speech recognition, the most popular measure of error is based on the Levenshtein edit distance [58], which takes these possible errors into account.

The Levenshtein edit distance counts the number of operations required to transform one string into another. The possible edits are insertions, substitutions, and deletions. The Levenshtein edit distance is computed as follows:

$$ED = n_{ins} + n_{sub} + n_{del} \tag{2.2}$$

where n_{ins}, n_{sub} and n_{del} is the number of insertions, substitutions and deletions measured to transform one text into another, in this case $h(\mathbf{X}(i))$ into $\mathbf{l}(i)$. Normalizing the minimum distance by the number of characters or words in the target sequence give us the Character Error Rate (CER) or Word Error Rate (WER),

$$CER = \frac{n^c_{ins} + n^c_{sub} + n^c_{del}}{n^c} \tag{2.3}$$

$$WER = \frac{n^w_{ins} + n^w_{sub} + n^w_{del}}{n^w} \tag{2.4}$$

The super-index w and c refers to words or characters and the parameter n^x is the overall number of elements in the text in terms of characters ($x = c$) or words ($x = w$).

When measuring the CER, the whitespace character should be taken into account since this symbol is essential to separate words.

2.1.3 State-of-the-art models

In this subsection, we summarize the most relevant training models that other authors have proposed before and during the development of this thesis. Historically, we can classify the models into two main different trends, the Hidden Markov Models (HMM) and the Neural Networks (NN) models.

Since the landmark work by Graves et al. where they proposed the CTC [40], most models include a RNN with the CTC on top.

Hidden markov models

A HMM describes a stochastic process to handle sequential data, involving two random variables: the random variable representing the sequence of observations, denoted by X, and the random variable representing hidden states, denoted by Z.

In particular, HMM are probabilistic graphical models of the joint likelihood of the two variables $p(X,Z)$.

There are techniques to (a) compute the probability of a sequence given a model, (b) find the sequence of hidden states which is most likely to have produced an observed one, and (c) find the parameters of the model to maximize the probability of observing a sequence.

These models have been applied to the HTR task [52, 59, 44, 11, 110]. HMM is a solution for HTR task when constrained to a particular vocabulary, or the set of feasible words is fixed. One of the drawbacks of this model is the fact that the width of the characters has to be similar. Besides, it is necessary to apply some preprocessing steps to reduce the handwriting variability, such as normalizing contrast, normalizing skew, normalizing slant, and normalizing size. The performance of HMM was outperformed by NN models, and they are not being used anymore for HTR of historical documents.

Deep neural networks

NN have become a trend for solving most pattern recognition tasks. They consist of basic processing units linked to each other with weighted connections. These units are called neurons due to the similarity of these models with the biological neurons. The input of each unit is the weighted combination of the outputs of other units.

The basic and quite extended type of unit of a neural network is called perceptron and was proposed in [76], see Figure 2.2 as:

$$y = f(\mathbf{x}) = f(w_1x_1 + w_2x_2 + ... + w_nx_n + b) \tag{2.5}$$

being f a non-linear function known as *activation function*, $\mathbf{x}$ a column vector of n dimensions or features, $\mathbf{w}$ a vector of weights and b a bias term. Some non-linear functions are used in different tasks, for example if we have a binary task in which the model has to choose between two values, we can use the $\text{sign}(x)$ function which takes the values 1 if $x > 0$ and -1 otherwise. In the case the model requires a soft output, for example, a probability of an event occurs, the *sigmoid* function is usually used, see other activations functions in Figure 2.3:

$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{2.6}$$

If we take a set of perceptrons and connect the output of some of them to the input of others, we have a Multi Layer Perceptron (MLP) NN. Every perceptron in this NN is denoted as a neuron or unit. A MLP contains units organized in layers, where several layers can be stacked. The output of the previous layer, $\mathbf{y}(i-1)$, is

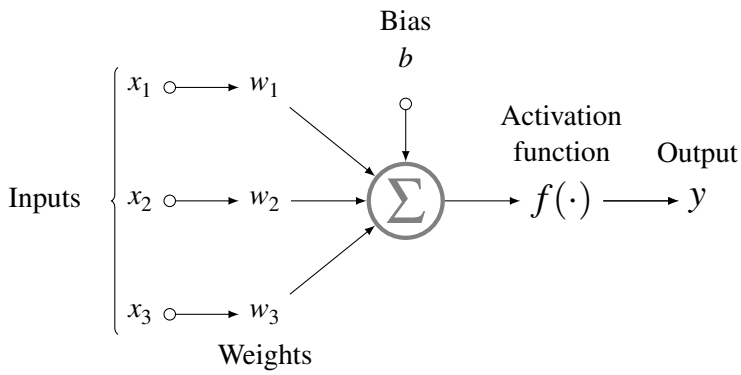


Figure 2.2 Perceptron.

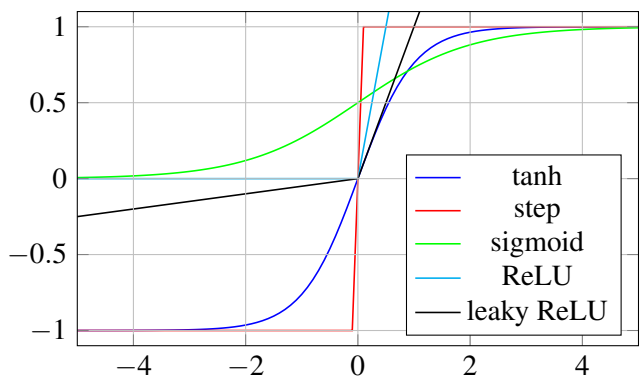


Figure 2.3 Examples of activation functions.

the input to the layer i. The weights of every neuron in a layer can be written as a row of a matrix $\mathbf{W}$ while the biases can be stacked in a vector, $\mathbf{b}$. Since these weights are different, every unit outputs a different output.

We can formulate the output of each layer of a MLP by using matrix notation: in the layer i we has the following expression:

$$\mathbf{y}(i) = f_i(\mathbf{W}\mathbf{y}(i-1) + \mathbf{b}(i)) = f_i(\mathbf{a}(i-1)) \tag{2.7}$$

where if $i = 1$, $\mathbf{y}(i-1) = \mathbf{x}$ and $\mathbf{a}$ are the activations. Also, a different activation function can be used in every layer.

In a multi-class classification problem the last layer might has multiple units, one per class. In this case the *softmax* function [18] is quite extended:

$$\mathbf{y} = \text{softmax}(\mathbf{a}) = \frac{e^{a_i}}{\sum_{k=1}^{n} e^{a_k}} \tag{2.8}$$

where a is a vector with the activations of the last layer,

$$\mathbf{a}(i) = \mathbf{W}\mathbf{y}(i-1) + \mathbf{b}(i) \tag{2.9}$$

and $\mathbf{y}$ is a vector with values in the range 0-1, one per class. A layer providing an activation as in (2.9) is referred to as Fully Connected (FC).

Convolutional neural networks

When dealing with multidimensional input structures such as images, the inputs are usually of high dimensions. In this case, the application of MLP involves a considerable number of weights, as every point, e.g., a pixel in images, is an input to every unit in the first layer. Besides, in these structures, usually, local structures are key features to perform the given task. In this scenario, CNN are preferred, as they exhibit much better performance with a fraction of the number of weights.

The key point in CNN is the application of a convolution with a kernel. The size of the kernel is usually of a few points, but a large number of different filters are used. At every layer i and unit j, the input, organized as a tensor of $h(i-1) \times w(i-1) \times d(i-1)$, is convolved with a kernel $k_j(i)$. Note that we detect relevant local patterns at the lower layers while exploiting large-scale relations in the upper ones by using several layers.

As we go through the layers, we usually reduce the sizes of the inputs, $h(i) \times w(i)$, while increasing the number of kernels, i.e., the depth, $d(i)$. This can be performed either by using the stride or a pooling. See [32] for details. The dimensions of the tensors, the stride, and pooling used are hyperparameters to be pre-defined, while in the training stage, the kernels are learned.

In this thesis, the CNN will be used first in the NN model to extract important features to be later related in a time scale using RNN. As explained later, five convolutional layers will be used in the first process and reduce the dimensions of the input image with the text of a line.

Recurrent neural networks

We already discussed that we face a sequence to sequence task, where a sequence of columns of pixels in an image is translated to a sequence of characters. In this task, RNN is the state-of-the-art. These networks incorporate some mechanism to achieve some memory through recurrent connections.

In a recurrent unit, samples of a sequence input are fed one at a time. In its simplest form, the output is both the corresponding input and the output at the previous timestep. Hence, the previous output plays the role of a state or memory of the unit. The activations, $\mathbf{a}_k^{(t)}$, evolve through time with the following recurrence,

$$a_k^{(t)} = \sum_{i=1}^{I} \omega_{ki}^{in} x_i^{(t)} + \sum_{h=1}^{H} \omega_{kh}^{rec} z_h^{(t-1)} \tag{2.10}$$

where $x_i^{(t)}$ are the inputs at timestep t, ω_{ki}^{in} the weights for the inputs, $\mathbf{z}^{(t-1)}$ the output of the layer at the previous timestep and ω_{kh}^{rec} the corresponding weights.

In general, the output, $\mathbf{y}$, can be any transformation, $g(\cdot)$, of the state or memory, $\mathbf{z}$. See Figure 2.4.

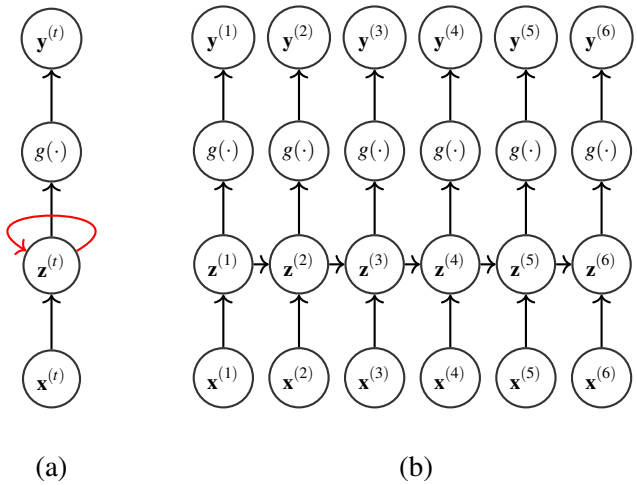


Figure 2.4 Block diagram of a RNN in (a) the folded structure, with the recurrent link in red and (b) unfolded for $T = 6$.

The sequence of outputs is the outputs of the layer. Several layers can be staked. Also, within a layer, the Bidirectional Recurrent Neural Network (BRNN) processes the sequence in both directions. In these networks, the layer has two independent sub-layers: a forward one (see states $\mathbf{z}^{(t)}$ in Figure 2.5), a RNN starting at time 0 and increasing t, and a backward RNN (states $\mathbf{v}^{(t)}$ in Figure 2.5), i.e., starting at the last timestep and decreasing t. Both layers are connected to the same input, and their outputs are combined. Also, the outputs of both sublayers could be staked as input to the next layer.

Other alternative is Multi Dimensional Recurrent Neural Network (MDRNN), which process an input image with four directions in recurrent layers.

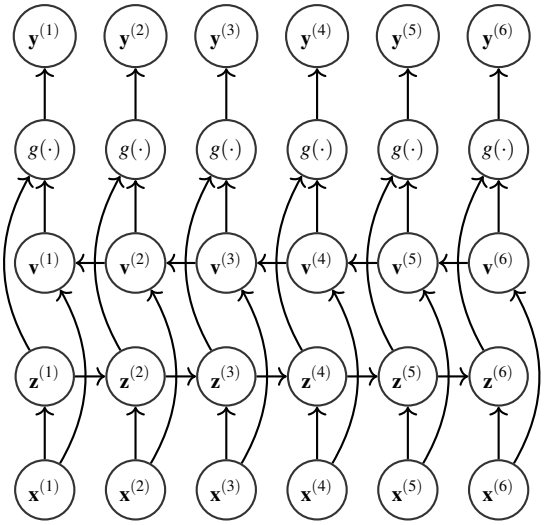


Figure 2.5 Bidirectional RNN, unfolded example for $T = 6$.

Long-short term memory networks

In RNNs, the vanishing gradient issue prevents the network from learning long-time dependencies. In [48] the authors proposed the LSTM as an improved recurrent unit, a.k.a. cell, to solve this problem. In LSTM, we have the input, $\mathbf{x}^{(t)}$, the output, $\mathbf{h}^{(t)}$, and an internal state or memory, $\mathbf{c}^{(t)}$. A gating system controls the flow of information, weighting the input information, the output activation, and the internal state of the unit at the previous timestep to update the current state.The key idea is to let the unit decide how to update the memory given the input. If the memory needs to be entirely updated, we forget it, and it will quite depend on the input. On the contrary, the memory could be preserved and not modified by the input.

The following equations define the behavior of the LSTM unit. There are three so-called gates: input, forget and output. Their outputs are computed, respectively, as:

$$\mathbf{i}^{(t)} = \sigma(\mathbf{W}_x^i x^{(t)} + \mathbf{W}_h^i \mathbf{h}^{(t-1)} + \mathbf{b}^i) \tag{2.11}$$

$$\mathbf{f}^{(t)} = \sigma(\mathbf{W}_x^f x^{(t)} + \mathbf{W}_h^f \mathbf{h}^{(t-1)} + \mathbf{b}^f) \tag{2.12}$$

$$\mathbf{o}^{(t)} = \sigma(\mathbf{W}_x^o x^{(t)} + \mathbf{W}_h^o \mathbf{h}^{(t-1)} + \mathbf{b}^o) \tag{2.13}$$

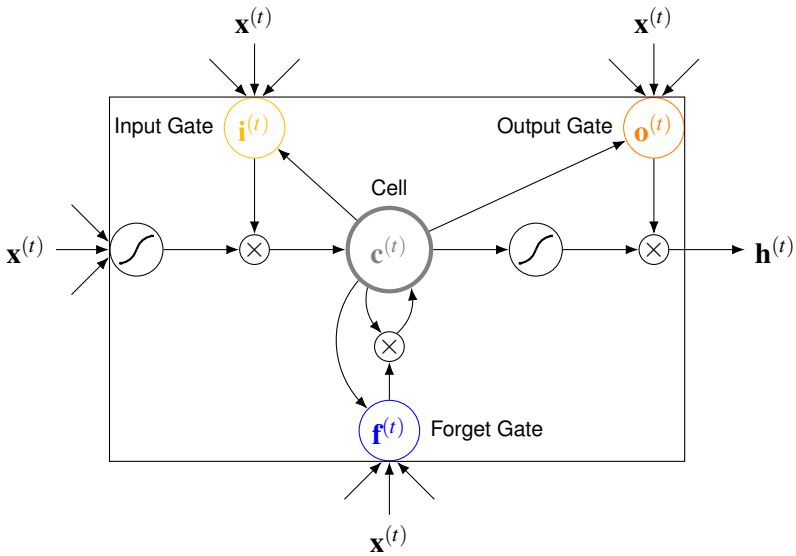


Figure 2.6 LSTM cell.

Then, the new state and output are computed as

$$\mathbf{c}^{(t)} = \mathbf{f}^{(t)} \otimes \mathbf{c}^{(t-1)} + \mathbf{i}^{(t)} \otimes \tilde{\mathbf{c}}^{(t)} \tag{2.14}$$

$$\mathbf{h}^{(t)} = \tanh(\mathbf{c}^{(t)}) \otimes \mathbf{o}^{(t)} \tag{2.15}$$

where

$$\tilde{\mathbf{c}}^{(t)} = \tanh(\mathbf{W}^c_{\mathbf{x}} x^{(t)} + \mathbf{W}^c_h \mathbf{h}^{(t-1)} + \mathbf{b}^j) \tag{2.16}$$

is an auxiliary value and $\otimes$ denotes element-wise multiplication.

This unit can be used as a layer. Also, several layers can be stacked, being the output of the lower input to the upper one. Besides, a bidirectional structure similar to the one explained previously can also be used.

Over the last decade, there has been a trend towards the utilization of LSTMs [41, 72, 99, 94, 14] jointly with CTC [40] in order to have an end-to-end system capable of doing the transcription of raw images containing whole lines of text. RNN-CTC methods for HTR have obtained the lowest error rates in recent HTR contests [41, 72, 99, 17, 14, 42].

Connectionist Temporal Classification

Since the RNN network only outputs local classifications, for that timestep, a post-processing stage is required to give the final label sequence. Suppose we

have the image in Figure 2.1 and we want to recognize the text. Besides, we are not segmenting the image to locate letters. Labeling unsegmented sequence data, denoted as *temporal classification*, is a well-known problem in sequence learning.

Suppose that we feed an RNN with the image in Figure 2.1 to provide $T = 120$ outputs, but we have $U = 44$ letters as labels in *"monasteries, manors, townships, or wards and"*. In training, as we have the labels, i.e., the sequence of letters, we need to *translate* from labels (44) to outputs of the RNN (120). We will need *two temporal indexes*, u for labels, and t for outputs of the RNN. Therefore, several consecutive output indexes of the RNN correspond to the same label index. In Figure 2.1.3 we depict the RNN followed by the CTC, a block translating from timesteps, i.e., columns of pixels in an image of a line of text, to letters. The CTC was proposed in [40] for the labeling of unsegmented data with neural networks. With the CTC data does not need to be pre-segmented, and the output does not have to be post-processed: it is already the sequence of characters.

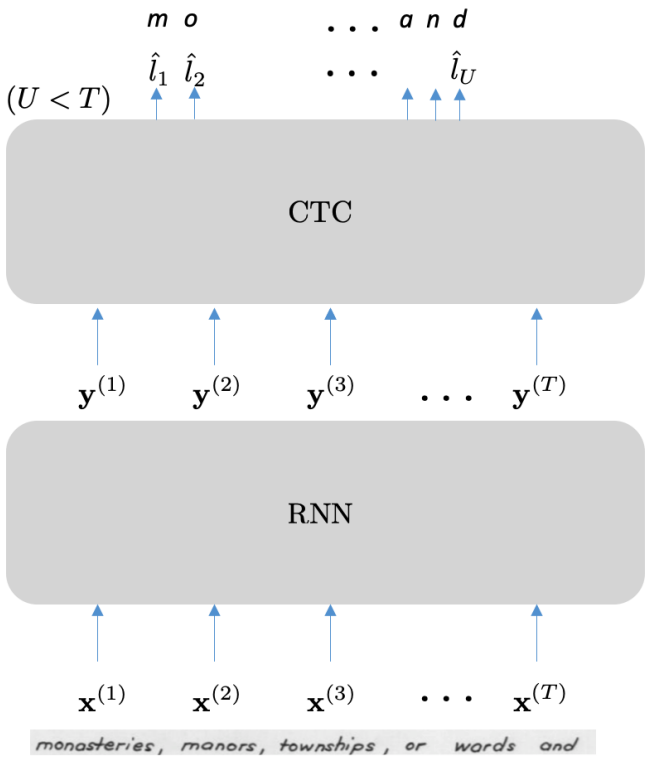


Figure 2.7 RNN and CTC: the output of the RNN is the imput to the CTC, that translates a sequence of features of length T, the size of the input, into a sequence of $U < T$ letters, with U unknown.

Provided that the network outputs for different timesteps are independent, given the input (because there is no connection from the output layer to intermediate layers), the probability of a sequence π for a given $\mathbf{x}$ at the output of the RNN is

$$p(\pi|\mathbf{x}) = \prod_{t=1}^{T} p\left(\mathbf{y}_{\pi_t}^{(t)}(\mathbf{x})\right) \tag{2.17}$$

where $p\left(\mathbf{y}_{\pi_t}^{(t)}(\mathbf{x})\right)$ is the probability of $\mathbf{y}^{(t)}$ being in the sequence π at position t, given $\mathbf{x}$. Since this probability is hard to compute, the activation of the output is interpreted as this probability.

The output of the CTC is an association of every output of the RNN, $\mathbf{y}^{(t)}$, that depends on $\mathbf{x}$, to a letter $l^{(u)}$. In the CTC the mapping $\mathbf{l} = \mathcal{B}(\pi)$ is of major importance. It translates the sequence of outputs of the RNN into a sequence of letters. This mapping, to be calculated within the CTC, decides how many consecutive outputs of the RNN corresponds to a letter or a space between letters. With this mapping, we calculate the posterior probability of a label sequence $\mathbf{l} \in L^{\leqslant T}$ by summing over all possible segmentations:

$$p(\mathbf{l}|\mathbf{x}) = \sum_{\pi \in \mathcal{B}^{-1}(\mathbf{l})} p(\pi|\mathbf{x}) \tag{2.18}$$

This probability is used by the CTC as the objective function to be maximized. This computation can be cumbersome, as we need to check for multiple combinations, and some approximations can be used to reduce the computational complexity, with a quite reduced impact on performance. See [40, 41, 39] for further information.

2.2 State-of-the-art architectures

State-of-the-art architectures for HTR in historical documents combine a CNN [103] with a RNN with LSTM cells [48]. This type of network models the conditioned probability, $p(\mathbf{l}|\mathbf{x})$, of a character sequence of arbitrary length, U, given an image, $\mathbf{X}$, of fixed height and arbitrary width.

These models are configured to minimize the CTC cost function proposed by Graves in [40]. In some works Two Dimensional LSTM (2D-LSTM) [48] networks are used [72, 99, 19, 64]. This RNN has two main drawbacks. On the one hand, it has a vast number of parameters that make learning difficult. On the other hand, it is not parallelizable [74]. For these reasons, after some attempts, it has been discarded in this thesis justified by the analyses we make In Chapter 4.

There are several authors that implement architectures composed of CNN and Bidirectional Long Short Term Memory (BLSTM) networks [84, 79, 108, 20, 92, 8, 90, 81, 107, 6].

Regarding the state-of-the-art DNN models for HTR, some recent works are in the line of avoiding recurrence in the models, developing models based in fully-convolutional networks such as the Gated Convolutional Network (GCN) [23, 30, 65, 26]. This kind of model reduces the number of parameters in the architecture.

2.3 Transfer learning

In the HTR problem with a reduced training set, TL was applied by Soullard *et al.* in [90]. The main idea behind TL is initializing the parameters of a model by those learned from a huge dataset beforehand, denoted as *source*. Then, the available labeled set of samples of the dataset of interest, the *target*, is used to refine the parameters of the model. Usually, just a subset of them [106, 31, 109, 61]. In [61], they analyze how to reduce the dataset shift and enhance the feature transferability in task-specific layers of deep networks. Hence, with TL we start learning a different task to avoid learning the whole set of parameters from scratch, preventing overfitting and favoring convergence. In [90], they proposed a method that applies TL in both the optical and the language model. In this and other similar previous proposals on TL, the authors applied DA in both training and test steps.

2.4 Data augmentation

DA consists in augmenting the training set with synthetically generated samples. Like TL, it reduces the tendency to overfit when training models with many parameters and limited labeled data. In DA for image classification problems, the training set is increased by modifying the original images through transformations such as scaling, rotation, or flipping images, among others [21]. Several authors have proposed specific DA techniques for HTR: in [101] the authors apply methods for augmentation and normalization to improve HTR by allowing the network to be more tolerant of variations in handwriting by profile normalization. In [73] they show some affine transformation methods for data augmentation in HTR. In [55], and [85] they synthesize new lines images by concatenating characters from different datasets. [55] does it from cursive characters, while in [85] they do it from a database of handwritten Chinese characters. Similar to [101], in [87] they also apply some elastic distortions to the original images. In [20] the authors

improve the performance by augmenting the training set with specially crafted multi-scale data. They also propose a model-based normalization scheme that considers the variability in the writing scale at the recognition phase. In these works, they apply DA in relatively large well-known datasets, but here we show that the regularization effect of any DA technique has no impact when making the fine-tuning adaptation to a singular writer in small databases. Accordingly, we will conclude that the combination of TL and DA applied to small datasets has to be done carefully to reduce the final error.

2.5 Mislabeled samples

Mislabeled detection in HTR has been seldom studied. In [67] they face a specific problem in the Institut für Informatik und Angewandte Mathematik (IAM) database: crossed out words that are labeled with the symbol "#". The authors propose a method to avoid how this specific label affects performance. That method is focused on the specific problem of crossed-out text and how it is annotated in the GT. The algorithm we propose in Chapter 6 is more general, addressing this and other possible problems. In related work in [80], the authors apply a method to align the output of a segmentation process with the available GT.

2.6 Synthetic handwriting text generation

Previous handwriting generation approaches have focused on the online handwriting generation task where the data is collected in a digital device. The models learn to follow the pen over a digital surface [39, 66, 63, 2].

Recently, since 2019 some authors have proposed a Generative Adversarial Networks (GAN) model in order to generate offline handwritten images [50, 4, 35, 53, 43]. However, all those settings have in common that they only generate images of isolated words.

In [27] Brain Davis et al. propose a GAN for generating images of handwritten lines conditioned on arbitrary text.

In the historical handwritten field, some works consist in the generation of handwritten data, although those tasks and approaches are different of HTR. In [102] they proposed a method to improve the synthesis of word images for the word spotting task. In [89] they perform document enhancement through a GAN.

3 Databases

In this chapter, we introduce the datasets used in this thesis. Although this thesis focuses on historical handwritten documents, we also include two well-known modern databases that researchers use to validate their algorithms. We include these modern databases because we use them as a starting point to compare state-of-the-art models, or they are necessary for some steps of our methods. For example, in Chapter 4 we propose a transfer learning method where these databases are used as source datasets to train the models from scratch.

In each database, we provide information about the language of the documents, the script, some digitalization features, and the number of samples. We also provide some samples from each database.

3.1 The IAM database

The IAM database [62] contains 13353 labeled text lines of modern English handwritten by 657 different writers. The images were scanned at a resolution of 300 dpi and saved as Portable Network Graphics (PNG) images with 256 gray levels. In Figure 3.1 we include an image of this database alongside the GT transcript. The database is partitioned into training, validation, and test sets of 6161, 900, and 2801 lines, respectively[1]. Here, the validation and test sets provided merge in a unique test set. There are 79 different characters in this database, including capital and small letters, numbers, punctuation symbols, and white space.

[1] The names of the images of each set are provided in the *Large Writer Independent Text Line Recognition Task*.

GT: *those in authority to find some simple*

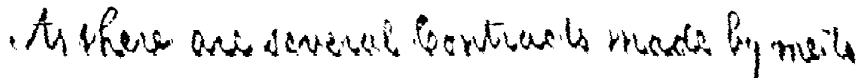
Figure 3.1 IAM handwritten text sample: image of a line and its transcript..

3.2 The RIMES database

The Reconnaissance et Indexation de données Manuscrites et de fac similÉS (RIMES) database is a acquisition of french letters handwritten by 1,300 volunteers who have participated in the RIMES database[2] creation by writing up to 5 emails. The RIMES database thus comprises 12,723 pages corresponding to 5605 mails of two to three pages. In our experiments, we take a set of 12111 lines derived from the International Conference on Document Analysis and Recognition (ICDAR) 2011 line-level competition. There are 100 different characters in this database.

3.3 The Washington database

The Washington database includes 565 text lines of the George Washington letters, handwritten by two authors in the 18th century. Although the language is also English, the text is written in longhand script and the images are binarized as illustrated in Figure 3.2, see [108] for a description of the differences between binarized and binarization-free images when applying HTR tasks. In this database, they provide four possible partitions to train and validate. In this thesis, we have randomly chosen one of them. The train, validation, and test set contain 325, 168 and 163 handwritten lines. There are 83 different characters in the database.



GT: *As there are several Contracts made by me to*

Figure 3.2 Washington handwritten text sample: image of a line and its transcript.

3.4 The Parzival database

The Parzival database [1] contains 4477 text lines handwritten by three writers in the 13th century. The lines are binarized like in the Washington database, but the text is written in gothic script. We include a sample in Figure 3.3. There

[2] see *http://www.a2ialab.com/doku.php?id=rimes_database:icdar_2011*

are 96 different characters in this database. Note that the Parzival database has a considerable number of text lines in contrast to the Washington one. We have randomly picked a training set of approximately the same volume as in the Washington training to emulate learning with a tiny dataset, which is the main goal of this thesis.

GT: *finiv heidenfciv ógen.*

Figure 3.3 Parzival handwritten text sample: image of a line and its transcript.

3.5 The ICFHR 2018 Competition over READ dataset

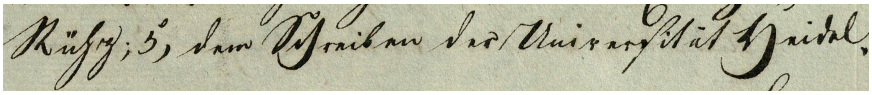
In 2018 they offered the set of documents of the International Conference on Frontiers in Handwriting Recognition (ICFHR)2018 Competition on Automated Text Recognition on a READ Dataset (https://readcoop.eu/) to compare the performance of approaches learning with few labeled pages. The dataset provided for the competition consists of 22 documents segmented at line level [92], written in Italian and modern and medieval German. Each of them was written by only one writer but in different periods and various languages. The training data is divided into a general set (of 17 documents) and a document-specific set (of 5 documents) called Konzilsprotokolle_C, Schiller, Ricordi, Patzig, and Schwerin of an equal script as in the test set.

Hereafter, *general* is used to denote available source labeled databases different from the one of matter, while *document-specific* denotes particular target documents. Also, the Konzilsprotokolle_C dataset, of the University of Greifswald, will be abbreviated as Konzil. The general database comprises roughly 25 pages per document (the precise number of pages varies such that the number of contained characters is almost equal per document). It will be denoted hereafter by ICFHR18-G.

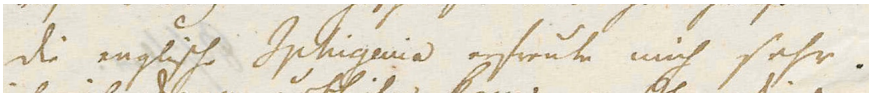
For the 5 document-specific databases the authors provide 16 labeled pages plus 15 unlabeled pages. One can check for the error in the transcription of these databases by sending the authors the 15 transcribed pages. Then, they publish the results of the transcription on the web of the contest. In Figure 3.4, samples from five specific target documents are displayed.

The standard unicode normalization form compatibility decomposition is applied to the GT to provide a common character set over such different documents, with 102 characters. The goal of the competition is to fit a model to transcript

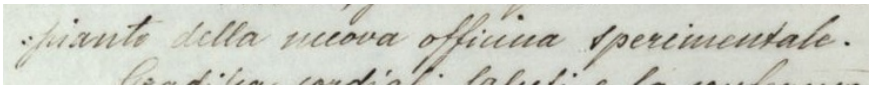
each of the 5 specific target documents with the lowest CER possible, using the 17 source documents available for training. Four experiments are conducted for each document-specific target dataset, simulating that you have 0, 1, 4, or 16 annotated pages available for training.



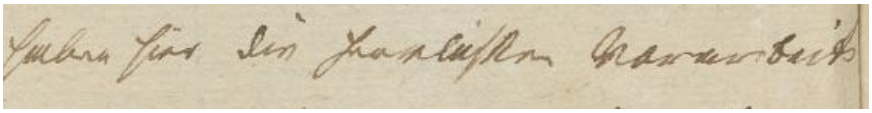
Konzil GT: *Ruhz; 5, dem Schreiben der Universitat Heidel¬*



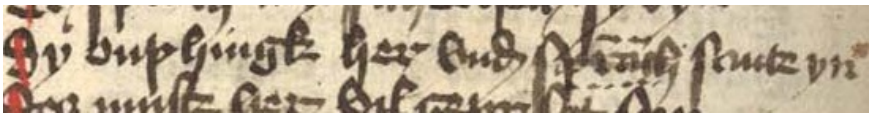
Schiller GT: *Die englische Iphigenia erfreute mich sehr.*



Ricordi GT: *pianto della nuova officina sperimentale.*



Patzig GT: *haben hier die herrlichsten Vorarbeiten*



Schwerin GT: *Dy onphingk her vnd sante yn*

Figure 3.4 From top to bottom: Konzil, Schiller, Ricordi, Patzig and Schwerin handwritten text samples with their transcripts.

In Table 3.1 we include the number of training and test lines available for every database. While we will use all lines in the test sets, the number of lines of the training set used vary through the experiments and we will indicate that number in every case.

Table 3.1 Number of lines available for training and test in each dataset.

	Full train set size	Test set size
RIMES	10163	778
IAM	6152	2912
Washington	325	163
Parzival	350	1328
ICFHR18-G	11424	2878
Konzil	351	118
Schiller	238	90
Ricordi	273	110
Patzig	473	168
Schwerin	782	275

4 HTR in Small Historical Databases: Transfer Learning

4.1 Introduction

The approaches already described in the state-of-the-art and particular NN architectures on HTR exhibit an outstanding performance if applied to modern and massive databases such as IAM and RIMES. However, the performance highly deteriorates when tackling historical databases with the common property of having few labeled lines.

In this chapter, we put forward the first contribution of this thesis to solve this problem by applying TL across different databases. With a smaller number of parameters and good generalization performance, we show that the network in Figure 4.2 can achieve remarkable generalization results for small historical databases once it has been pre-trained over an extensive database. We take the IAM database [62] to pre-train the network, then use this learning to solve HTR in the Parzival database [34] by using only 350 lines of text. We show that a test CER given by the equation (2.3) can achieve as low as 3.3%.

Similar outcomes are also achieved for the Washington database [34]. Furthermore, both databases are binarized, hence making the HTR more challenging. Up to our knowledge, transfer learning is a novel approach to HTR of text lines. We published the results of this chapter in [8].

Previous works where other authors applied TL to a problem related to HTR is presented in [38]. In that work, authors face a different problem in the HTR field: they use TL to face a word-spotting problem in which the objective dataset has no ground truth. After our proposal in [8], other authors started using TL for solving

HTR problems applied to historical documents at line level. The same year there was an international competition with results published in [92]. The organizers provided one of the databases presented in Chapter 3: the database for the ICFHR 2018 Competition over READ dataset. The contestants sent to the organizers the transcriptions of lines for 5 different documents from which they only offered 0, 1, 4, and 16 half-annotated pages in different steps of the competition. As we have shown in Section 5.5, our TL method, along with the architecture presented in Section 4.3 was better than most of the other participants of the competition.

Reported results prove the importance of performing TL as the right way to train HTR solutions based on DNN to get a good generalization over small databases. This is critical in historical documents, typically characterized by small databases and a vast variety of calligraphic styles.

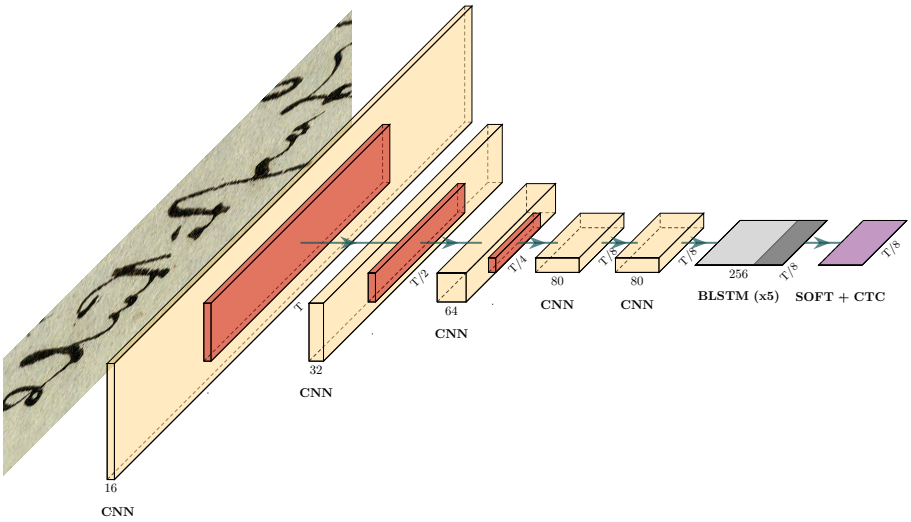


Figure 4.1 The proposed Convolutional Recurrent Neural Network (CRNN) architecture. The number of channels of each CNN layer is shown in this scheme. Pooling layers after the first, second and third CNN layer are also depicted. The number T/k with $k = 1,2,4,8$ is the length of the sequence. Numbers below blocks denote the depth of the layer i, $d(i)$, i.e. the number of filters or kernel used to compute it.

The rest of this chapter is organized as follows: Section 4.2 provides an overview in transfer learning; the neural network used in this chapter, Chapter 5 and Chapter 6 is detailed in Section 4.3; in Section 4.4 we analyze the application of transfer learning to solve HTR tasks over Washington and Parzival databases and, finally,

conclusions are drawn in Section 4.7.

4.2 Transfer learning overview

To cope with a reduced set of labeled inputs, we could first train the DNN model using as source available labeled large datasets. Then, we could apply TL, or domain adaptation strategies [37] to tune the learned model to later transcript a target document. As discussed in Chapter 1, we usually deal with different tasks, where TL has shown useful to share the results of the learning between tasks.

Formally, in HTR, deep learning algorithms have been usually used to solve problems over a domain $\mathcal{D} = \{\mathcal{X}, P(\mathbf{x})\}$, where $P(\mathbf{x})$ is the marginal probability. Typically $\mathbf{x}$ is the image for a segmented line in the text. The task consists of two components: a label space $\mathcal{Y}$ and an objective predictive function $f(\cdot)$ (denoted by $\mathcal{T} = \{\mathcal{Y}, f(\cdot)\}$), which can be learned from the training data. The data consists of pairs $\{\mathbf{x}_i, y_i\}$, where $\mathbf{x}_i \in \mathcal{X}$ and $y_i \in \mathcal{Y}$ [71] and $f(\mathbf{x}) = Q(y|\mathbf{x})$ can be interpreted as the conditional probability distribution.

Given a source domain $\mathcal{D}_S$ and a learning task $\mathcal{T}_S$, transfer learning aims to help improve the learning of another target predictive function $f_T(\cdot)$ in $\mathcal{D}_T$ using the knowledge in $\mathcal{D}_S$ and $\mathcal{T}_S$. In this work, we are interested in *inductive transfer learning* in which the target task is different from the source task, as the domains are different ($\mathcal{D}_S \neq \mathcal{D}_T$). Here we perform TL by retraining a DNN model where 1) all weights are initialized to the ones of the DNN learned for $\mathcal{D}_S$ and $\mathcal{T}_S$ and 2) the parameters of lower layers can be frozen to the values of the ones obtained after training with other available source datasets, used as off-the-shelf feature extractors [75].

To illustrate the performance of the approach, as source domain, we will use the IAM database [62], while the target domain will be the Washington and Parzival databases [34].

4.3 Architecture

In this thesis, we implement a network architecture based on the convolutional recurrent neural network (CRNN) presented in [86]. This approach avoids the use of 2D-LSTM layers, applying convolutional layers as feature extractors and a stack of 1D BLSTM layers to perform classification. Previous DNN architectures for HTR consisted of a combination of 2D-LSTM layers and convolutional layers, with a collapsing stage before the output layer in order to reshape the features tensors from 2D to 1D [99, 72]. The use of 2D-LSTM layers at the first stages has several drawbacks, such as the need for more memory in the allocation of

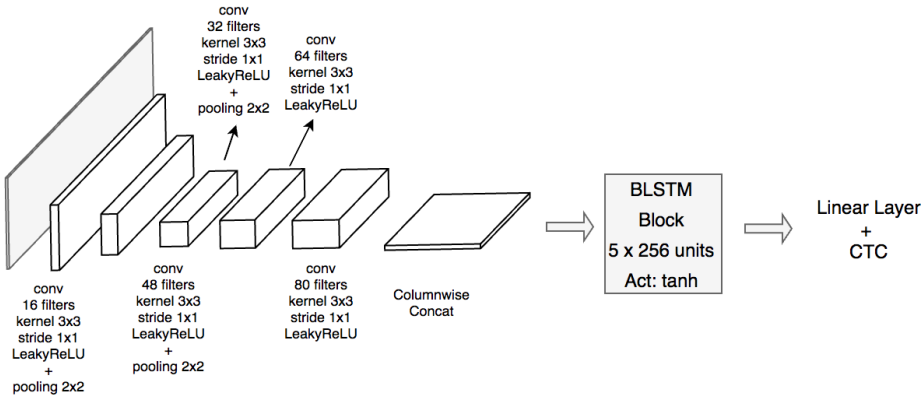


Figure 4.2 The network architecture used in this chapter.

activations and buffers during back-propagation and a longer runtime is required to train the networks since parallel computation cannot be implemented in contrast to a CNN [74]. Recently, it has been shown that CNN in the lower layers of an HTR system obtains similar features than an RNN containing 2D-LSTM units [74].

The CRNN architecture proposed in [86] is comprised of seven CNN with a max-pooling step at the output of four of them, followed by a stack of two BLSTM layers at the top of the network. In [8] we have shown that the CRNN in Figure 4.1, the one used in this work[1], achieves better performance than the original one proposed in [86]. In Figure 4.2 we include the same model with further details of every block.

It uses a CNN with 5 layers at the bottom, with a 3×3 and 1×1 stride kernel, the number of filters are 16, 32, 48, 64 and 80, respectively. We use LeakyReLU, see Figure 2.3, as the activation function. A 2×2 max-pooling is also applied at the output of the first 3 layers to reduce the size of the input sequence. At the output of the CNN, a column-wise concatenation is carried out with the purpose of transforming the 3D tensors of size $w \times h \times d$ (width $\times$ height $\times$ depth) into 2D tensors of size $w \times (h \times d)$ where w and h are the width and height of the input image divided by 8, i.e., after 3 stages of 2×2 max-pooling. The depth, $d = 80$, is the number of features of the last CNN layer. Therefore, at the output of the CNN, we have sequences of length w and depth $h \times 80$ features.

After the CNN stage, 5 1D-BLSTM recurrent layers of 256 units with hyperbolic tangent functions and without peephole connections. Since at the output of each

[1] Implementation is publicly available in https://github.com/josarajar/HTRTF

BLSTM layer we have 256 features in each direction, we perform a depth-wise concatenation to adapt the input of the next layer to the overall size of 512. Dropout regularization [72, 91] is applied at the output of every layer, except for the first convolutional one, with rates 0.2 for the CNN layers and 0.5 for the BLSTM layers.

Finally, each column of features after the 5th BLSTM layer, with depth 512, is mapped into the $L+1$ output labels with a FC layer, where L is the number of characters in the alphabet of each database, e.g., 79, 83, 96 or 102 in the IAM, Washington, Parzival or International Conference on Frontiers in Handwriting Recognition (ICFHR) 2018 Competition databases, respectively. The additional dimension is needed for the blank symbol of the CTC [40], which concludes this architecture. Overall, this CNN-BLSTM-CTC model has, approximately, 9.58×10^6 parameters, depending on the number of characters in each database.

The architecture is implemented in the open-source framework TensorFlow in Python, using the GPU-enabled version. We use the Adam algorithm, a learning rate of 0.003, $\beta_1 = 0.9$ and $\beta_2 = 0.999$. The parameters are updated using the gradients of the CTC loss on each batch of 16 text lines. We apply an early stopping criterion of 10 epochs without average improvement.

The selected model was the one with the best performance out of the $7+3$, $8+0$, $4+4$, $5+5$, and $6+6$ configurations, where $A+B$ corresponds to A convolutional followed by B BLSTM layers. On the other hand, for the CTC we tried best path decoding and beam search decoding, with no significant improvement of the latter, despite the difference in computational complexity. In Table 4.1 we describe the performance achieved by some state-of-the-art architectures and other variations we have proposed. In these experiments, we take the databases that the authors use in their experiments. For a fair comparison, we implement, including training, all the architectures under the same conditions. Hence, the CER and WER values of the experiments may vary from the result indicated in the authors' papers.

The goal of this comparison is to choose a convenient architecture that we will use in all the thesis experiments. The goal of this thesis is not to find the NN architecture that performs best over massive databases such as IAM and RIMES but over small ones.

As we show in Table 4.1, both families of 2D-LSTM based and CNN + BLSTM architectures perform similar. 2D-LSTM architectures achieve the best results in this comparison. However, as mentioned above, these models have some drawbacks. The 2D-LSTM architecture proposed in [99] is depicted in Figure 4.3.

Among those drawbacks, usually related to high computational costs and large numbers of parameters, the most remarkable is presented in Table 4.2. In this table we include the results of TL with the IAM dataset as source and Washington

Table 4.1 Evaluation of CER and WER performance of architectures that achieves state-of-the-art performance over IAM and RIMES datasets. For a fair comparison, we trained all models under the same conditions. The first column indicates the model used, "B" means BLSTM, and "C" means convolutional. The best results are indicated in boldface.

	IAM		RIMES	
	CER	WER	CER	WER
2D-LSTM (a) [99]	6.8	18.3	4.0	10.4
2D-LSTM (b) [19]	**6.7**	**18.1**	**3.8**	**10.0**
7C + 3B	7.5	20.5	4.6	11.4
8C	8.2	25.7	5.7	14.1
4C + 4B [74]	7.3	22.9	4.6	11.3
5C + 5B	7.2	22.2	4.4	10.8
6C + 6B	7.5	22.8	4.4	12.8
5C + 2B	7.8	23.5	4.7	13.2

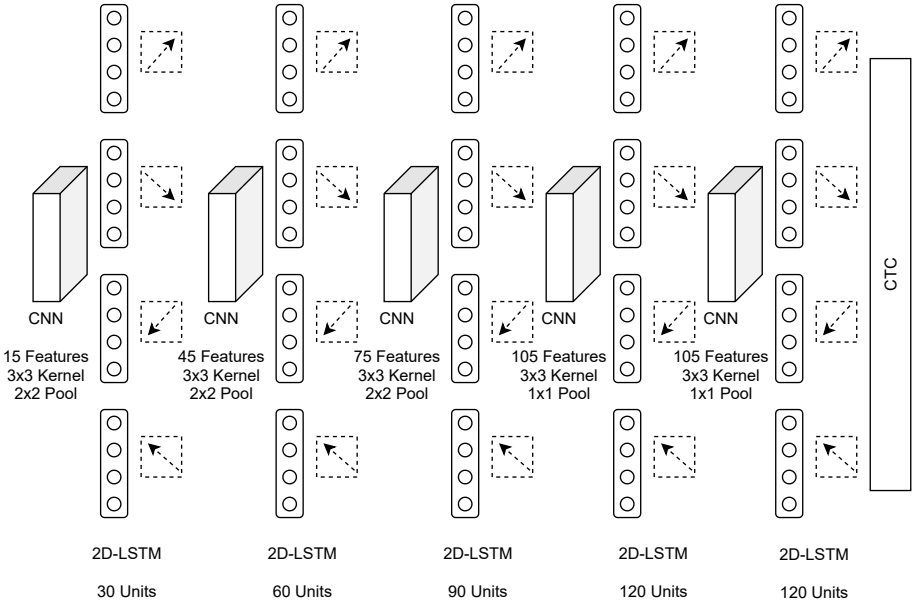


Figure 4.3 The 2D-LSTM based architecture proposed in [99].

and Parzival datasets as targets. Parameters in all layers are retrained using with 350 lines of this datasets, see below in this chapter for further details. It can be observed that the 2D-LSTM based models perform worse than the CNN +

Table 4.2 Evaluation of CER and WER performance of architectures that achieves state-of-the-art performance over Washington and Parzival datasets. For a fair comparison, we trained all models under the same conditions. The first column indicates the model used, "B" means BLSTM, and "C" means convolutional. The models have been pretrained with the IAM database. The best results are indicated in boldface.

	Washington		Parzival	
	CER	WER	CER	WER
2D-LSTM (a) [99]	7.9	27.3	4.8	13.4
2D-LSTM (b) [19]	8.2	26.7	5.0	13.7
7C + 3B	5.6	22.1	3.4	10.8
8C	7.2	23.1	3.5	10.9
4C + 4B [74]	5.5	23.2	3.4	10.8
5C + 5B	**5.3**	**21.9**	**3.3**	**10.5**
6C + 6B	5.9	23.4	3.6	11.0
5C + 2B	5.6	23.1	3.8	10.9

BLSTM families. Furthermore, when used with TL their performance degrades. On the contrary, models based on CNN plus BLSTM, see e.g. the 5C+5B model, experience a remarkable improvement.

4.4 Transfer learning for HTR

In this chapter, we investigate how TL can be applied to reuse the parameters learned during training an extensive database to learn another reduced corpus in the HTR problem. While the extensive database has thousands of lines ($5000-7000$) the training set for the new HTR problem is of a few hundred sizes ($150-350$). The proposed methodology is as follows: We first studied the best strategy to perform TL in detail. We did TL from the IAM database to the Washington one, first with 325 text lines as training data to later face the learning with 250 and 150 text lines. Then, the best strategies found were validated by the Parzival database.

4.4.1 Learning from scratch

When training the CRNN architecture from scratch for the Washington database, the CER tends to 0% if it is evaluated over the training set while a CER of over 40% is reached when evaluating the validation set. It can be concluded that there is not enough number of samples and we have overfitting. The convergence is

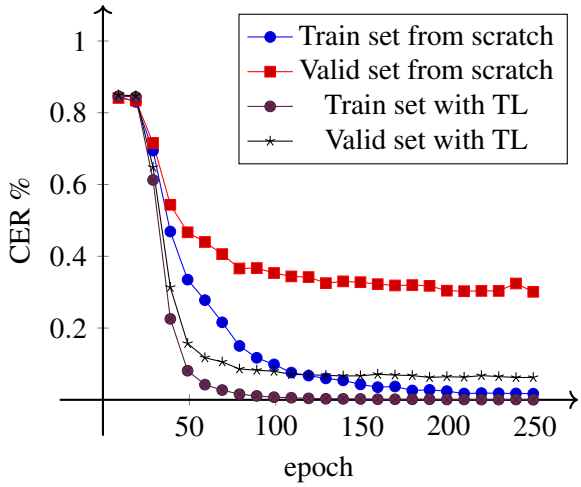


Figure 4.4 Evolution of the error while training the CRNN architecture with random initialization and the Washington database.

depicted in Figure 4.4.

4.4.2 Simple TL by just initialization

In the first instance, we trained the network for an extensive database, IAM, and applied the learned solution to the Washington database. We only transcribed the familiar characters to both databases. We got a CER = 82%. This poor result may be due to the heterogeneity between both databases: 1) their images have different resolutions, 2) the calligraphies of the texts correspond to different centuries, 3) the sets or alphabets of characters are different, and 4) the images used in training are in grayscale while the database where the learning results are applied is binarized. These differences can be observed in Figure 3.1 and Figure 3.2.

At this point, it is interesting to note that due to the stage of column-wise concatenation between CNN and BLSTM layers, the number of parameters in the first BLSTM layer depends on the height of the input images. Therefore, to apply the same CRNN structure to two different databases, it is necessary to resize the images of the target database so that they have the same height as the images of the training database.

4.4.3 Best TL strategy

As a first approach of TL, we added one more BLSTM layer on top of the BLSTM part of the CRNN architecture. We learned this new layer while keeping the rest

Table 4.3 CER evaluated in Washington datasets in a model obtained after retraining a set of layers in the model previously trained over the IAM database. It has been retrained with 325 lines images. Lowest values in boldface.

	CER (%)		
Trainable Layers	Train	Valid	Test
FC	55.1	46.2	47.1
BLSTM5, FC	13.0	22.1	23.6
BLSTM[4,5], FC	4.2	14.4	17.4
BLSTM[3,4,5], FC	0.6	10.6	12.8
BLSTM[2,3,4,5], FC	0.2	8.7	6.7
BLSTM[1,2,3,4,5], FC	0.2	5.3	6.6
Conv[5], BLSTM[1,2,3,4,5], FC	0.2	4.8	6.1
Conv[4,5]. BLSTM[1,2,3,4,5], FC	0.2	5.2	6.3
Conv[3,4,5], BLSTM[1,2,3,4,5], FC	0.3	4.5	5.5
Conv[2,3,4,5], BLSTM[1,2,3,4,5], FC	0.5	**4.3**	5.4
Conv[1,2,3,4,5], BLSTM[1,2,3,4,5], FC	0.2	4.6	**5.3**

of the layers fixed to the values learned for the IAM database. This architecture is prone to overfitting. However, reducing the number of units of this new layer from 256 to 128 or 64 did underfit. These results discouraged us from adding new layers to the already trained architecture.

We next kept the architecture and studied how the parameters could be initialized and learned. We initialized the architecture to the values trained with the IAM database and then trained just a subset of layers from top to bottom, using 325 text lines from the Washington database. The result of this analysis can be observed in Table 4.3. This table includes the CER evaluated on the training, evaluation, and test sets. In the first column of the table, we indicate the layers that have been left free during the retraining. The rest of the layers remain fixed and initialized to the result of the learning of the extensive database. For example, BLSTM[3,4,5] FC indicates that the three upper BLSTM and the FC layers have been retrained, keeping the rest of the network fixed. The analysis involves both the layers in the BLSTM networks and the CNN layers.

We first trained just the FC layer of the structure in order to include the number of characters that the Washington database has ($83+1$ for CTC blank character). We retrained this last layer with a training set of 325 lines from the Washington

database, keeping the rest of the layers fixed. In Table 4.3, the first row, it can be observed that just retraining the FC layer, the model tends to underfit, a CER = 52% is obtained, evaluated both in the train set and in the validation set. We next included the BLSTMs in the set of layers to be retrained. See rows 2-6 in Table 4.3. Then we also retrained the CNNs, rows 7 to 11.

The most interesting conclusion of this analysis is that by retraining the first four BLSTM networks, the CER decreases from 47 % to 6.7 %. By retraining the convolutional layers, i.e., the whole network, we get an extra gain from 6.7 % to 5.3 %. The best result in the test set was obtained when the whole model was retrained (CER = 5.3 %), while the validation set was obtained when the first convolutional layer was kept fixed (CER = 4.3 %). Also, by just retraining the top three convolutional layers, we already get 5.5 %. A possible interpretation of this behavior is as follows: While CNNs are extracting features from the images [13], the BLSTM are supporting the classification task. The feature extraction stage is more transferable than the classification step.

From this analysis, we can conclude that a good TL approach would be to initialize and retrain the whole network. However, if the first two CNN layers are kept fixed, we get approximately the same result.

4.4.4 Reducing the training set

We also investigated the performance of this proposed TL approach when the set of training data was reduced from 325 to 250 and 150. We first randomly chose 250 text lines and repeated the same analysis applied over the set of 325 lines to investigate the number of layers we should retrain. The results of this analysis are included in Table 4.4. In this case, the best error rate was achieved when the whole model except the first convolutional layer was retrained for both validation and test sets (CER = 6.0 % and CER = 7.1 % respectively). The results are slightly worse than those of the model trained with 325 text lines. To illustrate the convergence of the proposed TL, we include the convergence along epochs for this case in Figure 4.4, retraining the whole network.

Finally, we reduced the training set to 150 text lines and repeated the experiments. The results of this analysis are included in Table 4.5. In this case, the best result for the validation set was achieved when the whole model was retrained, fixing the two lower convolutional layers (CER = 7.9 %) while in the test set was achieved when keeping just the first layer fixed (CER = 9.4 %). These results could be considered promising if the cost of manually annotating the lines when creating the training dataset is taken into account[2] [84]. It can also be concluded

[2] The total time required for a single expert to annotate 20357 lines was estimated as 500 hours manually.

Table 4.4 CER evaluated in Washington datasets in a model obtained after retraining a set of layers in the model previously trained over the IAM database. It has been retrained with 250 lines images. Lowest values in boldface.

	CER (%)	
Trainable Layers	Validation	Test
BLSTM5, FC	26.0	26.9
BLSTM[4,5], FC	18.9	19.5
BLSTM[3,4,5], FC	12.2	14.4
BLSTM[2,3,4,5], FC	8.4	10.5
BLSTM[1,2,3,4,5], FC	7.1	8.4
Conv[5], BLSTM[1,2,3,4,5], FC	6.6	8.1
Conv[4,5]. BLSTM[1,2,3,4,5], FC	5.8	7.3
Conv[3,4,5], BLSTM[1,2,3,4,5], FC	6.2	7.2
Conv[2,3,4,5], BLSTM[1,2,3,4,5], FC	**6.0**	**7.1**
Conv[1,2,3,4,5], BLSTM[1,2,3,4,5], FC	6.2	7.6

that not retraining the first, or the first and second, CNN layer is a robust strategy with varying training data size.

4.4.5 Validation with the Parzival database

To validate the results obtained on the proposed TL algorithm, we applied it to the Parzival database. The Parzival database contains more than 2000 annotated text lines. We randomly chose a reduced subset of 350, 250, and 150 text lines to perform the TL. The CRNN architecture trained from scratch with this 2000 lines train set achieves a CER = 1.7 % for both validation and test sets. The same model trained for 350 lines got stuck in a value of CER = 18.2 %, similar to the value of this architecture for the Washington database.

Given the previous results, we focused on the CNN layers. The results can be observed in Table 4.6. In this case, the best results were also obtained when the whole model or just the first layer was kept fixed: CER = 3.0 % and 3.3 % in the validation and test set, respectively, in both cases.

When reducing the number of lines, similar results were obtained. We applied TL by training a model with a training set of 250 and 150 images, respectively. The results of these analyses are included in Table 4.7 and Table 4.8. As in the Washington case, the best error rates are obtained when retraining the whole model

Table 4.5 CER evaluated in Washington datasets in a model obtained after retraining a set of layers in the model previously trained over the IAM database. It has been retrained with 150 lines images. Lowest values in boldface.

	CER (%)	
Trainable Layers	Validation	Test
BLSTM5, FC	30.5	31.4
BLSTM[4,5], FC	22.7	24.2
BLSTM[3,4,5], FC	15.7	18.6
BLSTM[2,3,4,5], FC	11.4	14
BLSTM[1,2,3,4,5], FC	10.3	12.6
Conv[5], BLSTM[1,2,3,4,5], FC	9.2	11.2
Conv[4,5]. BLSTM[1,2,3,4,5], FC	8.1	10.1
Conv[3,4,5], BLSTM[1,2,3,4,5], FC	**7.9**	9.5
Conv[2,3,4,5], BLSTM[1,2,3,4,5], FC	8.4	**9.4**
Conv[1,2,3,4,5], BLSTM[1,2,3,4,5], FC	10.4	11.9

Table 4.6 CER evaluated in Parzival datasets in a model obtained after retraining a set of layers in the model previously trained over the IAM database. It has been retrained with a set of 350 lines images. Lowest values in boldface.

	CER (%)	
Trainable Layers	Validation	Test
BLSTM[1,2,3,4,5], FC	4.2	4.1
Conv[5], BLSTM[1,2,3,4,5], FC	3.8	3.8
Conv[4,5]. BLSTM[1,2,3,4,5], FC	3.3	3.6
Conv[3,4,5], BLSTM[1,2,3,4,5], FC	3.4	3.5
Conv[2,3,4,5], BLSTM[1,2,3,4,5], FC	**3.0**	**3.3**
Conv[1,2,3,4,5], BLSTM[1,2,3,4,5], FC	**3.0**	**3.3**

or keeping fixed the lower CNN layers. The best results in this case are CER = 4.0 % for the training with 250 text lines and CER = 5.8 % for the 150 text lines case.

Table 4.7 CER evaluated in Parzival datasets in a model obtained after retraining a set of layers in the model previously trained over the IAM database. It has been retrained with a set of 250 lines images. Lowest values in boldface.

	CER (%)	
Trainable Layers	Validation	Test
BLSTM[12345], FC	5.2	5.4
Conv[5], BLSTM[1,2,3,4,5], FC	5.1	4.8
Conv[4,5]. BLSTM[1,2,3,4,5], FC	4.4	4.5
Conv[3,4,5], BLSTM[1,2,3,4,5], FC	4.0	4.1
Conv[2,3,4,5], BLSTM[1,2,3,4,5], FC	**3.6**	**4.0**
Conv[1,2,3,4,5], BLSTM[1,2,3,4,5], FC	3.9	**4.0**

Table 4.8 CER evaluated in Parzival datasets in a model obtained after retraining a set of layers in the model previously trained over the IAM database. It has been retrained with a set of 150 lines images. Lowest values in boldface.

	CER (%)	
Trainable Layers	Validation	Test
BLSTM[12345], FC	7.4	7.4
Conv[5], BLSTM[1,2,3,4,5], FC	7.2	7.0
Conv[4,5]. BLSTM[1,2,3,4,5], FC	6.8	6.5
Conv[3,4,5], BLSTM[1,2,3,4,5], FC	6.6	6.5
Conv[2,3,4,5], BLSTM[1,2,3,4,5], FC	6.8	6.6
Conv[1,2,3,4,5], BLSTM[1,2,3,4,5], FC	**6.5**	**5.8**

4.5 Application to ICFHR 2018 competition dataset

In previous sections, we analyzed preliminary TL results over the Washington and Parzival databases by using the IAM database as the source, and we investigated which layers should be kept fixed to then apply a fine-tuning process to the others. We concluded that the best choice is to unfreeze all the layers, where the first one can eventually be fixed. In most cases, fixing only the first CNN layer leads to the

best performance.

In Table 4.9 we extend the analysis in [8] to the five specific documents in the ICFHR 2018 Competition dataset, where the 17 documents of the broad set of the database, in the ICFHR18-G, are used as the source. Results are included when fixing layers 1 to 3 of the CNN, as fixing other layers provided more significant errors in all cases. The lowest achieved errors are highlighted in boldface. Training set size is given in the number of lines. It can be observed that, among all databases, the best performance is achieved when unfreezing all layers or, at most, only the first layer is kept frozen. Hereafter the TL is applied by freezing just the first layer of the DNN model. The results shown in all tables hereafter indicate mean values of CER or WER. To get the statistics, the model in Figure 4.1 is trained 10 times, where the parameters to initialize are independently and randomly set. In Table 4.9, a non-parametric bootstrapped confidence interval at 95% [33] is also included. For the remaining tables, the confidence intervals can be found in the appendixes.

Table 4.9 TL perfomance: Mean CER (%) and bootstrapped confidence interval at 95%, in brackets, of the model in Figure 4.1 using TL for the Washington, Parzival, Konzil, Schiller, Ricordi, Patzig and Schwerin datasets (see Section 5.2) as target domains.

	Fixed layers			
	All free	CNN 1	CNN 1,2	CNN 1,2,3
Washing.	**5.32** [5.22-5.41]	5.41 [5.24-5.43]	5.53 [5.41-5.64]	6.30 [5.73-7.1]
Parzival	**3.30** [3.24-3.36]	**3.30** [3.21-3.34]	3.52 [3.36-3.62]	3.63 [3.47-3.69]
Konzil	4.51 [4.33-4.61]	**4.37** [4.24-4.54]	4.42 [4.36-5.49]	4.53 [4.43-4.61]
Schiller	**9.40** [9.31-9.46]	9.42 [9.34-9.46]	9.48 [9.40-9.54]	10.11 [9.21-10.32]
Ricordi	11.21 [11.14-11.25]	**11.20** [10.11-11.23]	11.28 [11.19-11.34]	11.60 [11.41-11.72]
Patzig	10.63 [10.51-10.70]	**10.60** [10.52-10.65]	10.68 [10.57-10.74]	12.4 [12.33-12.46]
Schwerin	**3.50** [3.46-3.53]	**3.50** [3.47 - 3.51]	3.91 [3.81 - 3.94]	4.22 [4.15-4.26]

In Table 4.9 we analyze different strategies of applying TL, with no DA, for the Washington and Parzival target datasets with the IAM database as the source and Konzil, Schiller, Ricordi, Patzig, and Schwerin datasets (see Section 5.2) as target domains with ICFHR18-G as the source. In each of ICFHR 2018 document-specific datasets, 12 pages are used for training. See the corresponding number of lines in Table 3.1. We conclude that the good choice is to freeze the first convolutional layer of the model (column "CNN1"). This solution will be used later in Chapter 5 and Chapter 6 in combination with other techniques.

Table 4.10 CER ICFHR 2018 Competition results for LSTM based models: upper part, other previous approaches and, in the lower part, the results for the approaches in this work. Lowest mean value highlighted in boldface.

	CER (%) per training size				CER (%) per document					Mean
	0	1	4	16	Konzil	Schiller	Ricordi	Patzig	Schwerin	
OSU[101]	31.40	17.74	13.27	9.02	9.39	21.10	23.27	23.17	12.98	**17.86**
ParisTech[20]	32.25	19.80	16.98	14.72	10.49	19.05	35.60	23.83	17.02	20.94
LITIS[95]	35.30	22.51	16.89	11.34	9.14	25.69	30.50	25.18	18.04	21.51
PRHLT	32.79	22.15	17.89	13.33	8.65	18.39	35.07	26.26	18.65	21.54
RPPDI[64]	30.80	28.40	27.25	22.85	11.90	21.88	37.29	32.75	28.55	27.32
Ours[8]	32.77	19.51	15.12	8.26	9.16	21.00	29.39	23.25	13.54	18.93

4.6 Comparison to the state-of-the-art

By using the proposed 5+5 DNN model with CNN and BLSTM layers followed by a CTC, we conclude by analyzing the results of the novel TL approach over the ICFHR 2018 Competition[3]. The results included in Table 6.4 were reported by the organizers of the competition. The contestants provide the transcript of the 15 test pages for every document in the target set: Konzil, Schiller, Ricordi, Patzig, and Schwerin. Then, the organizers evaluate the CER, publicly publishing the results. In this table our results are compared against the 5 original contestants during the competition: OSU [101], ParisTech [20], LITIS [95], Pattern Recognition and Human Language Technology (PRHLT) and RPPDI. These approaches use DNN models based on CNN, LSTM, and CTC, where some modification of the LSTM is used. The results of the proposal in this chapter are included in the lowest row of Table 6.4.

Results are presented in three groups of columns. First, the average CER (%) for the 5 target dataset is included when 0, 1, 4, and 16 pages of the target datasets are used. The second group of 5 columns reports the average CER (%) for the learning with 0, 1, 4, and 16 pages in the dataset for every document. The mean value per row is included in the last column.

4.7 Conclusions

In this chapter, TL applied to a CRNN architecture has been shown to be a promising technique to reduce the number of labeled data when we face an HTR problem over manuscripts belonging to a new domain. Besides the reduced number of labeled data required, this novel procedure also benefits from a speed-up factor

[3] The results are publicly available in the ICFHR competition website: https://scriptnet.iit.demokritos.gr/competitions/10/viewresults/

since the training is much simpler. In the experiments included, where training over thousands of text lines is transferred to an HTR problem with a few hundred, the proposed TL scheme exhibited a good performance when the whole network is initialized and re-trained. Robust results are obtained if the first or the two first layers of the CNN are kept fixed. A good performance, with CER in the range $3-9$ %, has been obtained transferring learning from the solution to the HTR of the IAM database to the HTR of Washington and the Parzival databases, with training data of sizes 150, 250 and 350 and dealing with different resolutions, alphabets and types of images.

In the next chapter, we propose the use of DA that can also be used to reduce the error. Although the combination of DA and TL could be thought of be trivial, in the next chapter, we will present the efficient way to use the combination of these techniques when applied to HTR tasks over small datasets.

5 HTR in Small Historical Databases: Data Augmentation

5.1 Introduction

As mentioned in Chapter 4, once the DNN model to be used has been designed, an enormous number of training samples are required to minimize the number of transcription errors, measured in CER in (2.3) or WER in (2.4), given by the Levenshtein distance [69] between the GT and the output of the model.

However, we might only have a limited number of lines for a given author and document most of the time. Besides, transcription of part of the documents to get labeled samples is expensive either in time or money. Take [84] as an example, where the manual transcription process of a document by an expert in paleography took an average of 35 minutes per page. In this scenario, allowing for a reduction in the transcript needed would significantly improve the viability and cost of the process.

In Chapter 4 we propose the use of TL as an effective method to solve this problem of lack of data. Another additional tool when facing Deep Learning (DL) problems with a small number of labeled data consists in the application of some distortions in the input images in order to augment the database [105]. This technique is also applied in [74] over the IAM database, decreasing the CER from 8.2 % to 6.4 % on the test set and from 5.1 % to 4.4 % on the validation set. The distortions applied are affine transformations such as rotation, shearing, translation, scaling, and some morphological distortions such as erosion and dilation. In this chapter, we analyze the joint performance of TL presented in Chapter 4 and DA

methods when applied to HTR. The proposal within this chapter was published in [7].

5.2 Architecture

In the HTR pipeline, there are several ways to improve the performance of a DNN model: preprocessing steps, the architecture used, regularization techniques, optimization, language model, and dictionary, among others. The methods proposed in this chapter are developed for the same DNN architecture presented en Chapter 4 but can be easily used in the pipeline of any other HTR system to reduce transcript errors. For a fair comparison, in this chapter, we use the same DNN model for all the experiments. Extra correction steps such as adding a Language Model (LM) are not included but could be applied to improve the performance further.

5.3 Data Augmentation without transfer learning

In [101] the authors compare various DA approaches using both RIMES [34] and IAM [62] databases as benchmarks, where transcription is made on the word level. Note that these databases have a considerably large number of labeled lines. In Figure 5.1 and Figure 5.2 samples of -slightly- distorted images are shown.

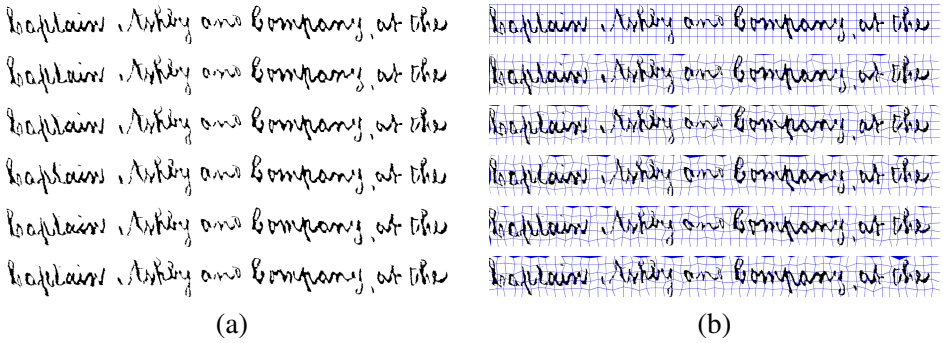


Figure 5.1 In first column, augmented sample from Washington dataset. In (a) the new samples generated using Random Warp Grid Distortion (RWGD). In (b) the grid used to distort them.

When not applying any augmentation technique, they get a CER of 5.35 % (IAM) and 3.69 % (RIMES). The best CER values reported in [101] by using

(a) (b)

Figure 5.2 In first column, augmented sample from Washington dataset. In (a) the new samples generated using RWGD. In (b) the grid used to distort them.

various DA techniques are 3.93 % and 1.36 %, respectively. Which is equivalent to an improvement of approximately 2 percentage points in both databases.

Next, we extend the same analysis to scenarios with small training datasets: Washington, Parzival, Konzil, Schiller, Ricordi, Patzig, and Schwerin databases. As throughout the whole thesis, the transcriptions are made on line level. Results for the IAM, RIMES, and the ICFHR18-G, i.e., the 17 documents of the general dataset in the ICFHR 2018 database, are also analyzed as references. In Table 7.1 we include the CER of our DNN model with no DA and two different DA techniques, affine transformation [73] and random warp grid distortion (RWGD) [101], for all databases in Chapter 3.

We augment the training set by generating ten copies of every line in the training set. One of these copies is the original line without distortions.

In Table 7.1, for the largest databases, the DA improvement is around 2 percentage points (2 percentage points in IAM, 1.9 percentage points in RIMES, and 2.5 percentage points in ICFHR18-G). However, in the small databases, the CER reduction is remarkable, in the range of 5 percentage points to 23.6 percentage

points, see CERs highlighted in boldface. Note that the results in [101] are different to the ones in Table 7.1 because while in [101] transcription is done at the word level here, whole lines are processed. This explains that in IAM without DA we get CER 7.2% while in [101] 5.35% is reported. In any case, It can be concluded that, since the DA acts as a regularization technique to avoid overfitting, the CER reduction is more remarkable as the size of the training set is reduced. At this point, it is most interesting to compare the results of TL and DA techniques when applied independently. It can be observed that TL exhibits, by far, a much larger CER reduction. Next, we face the design and analysis of both techniques combined, where RWGD will be used as the DA approach.

Table 5.1 DA perfomance: Mean CER and WER (%) with affine transformations [73] and RWGD[101] DA approaches evaluated for all datasets in Chapter 3. The DNN is trained from scratch using the number of lines indicated by 'Train size'. Largest DA CER reductions are highlighted in boldface.

	Train size	No method		Affine Transf.		RWGD[101]	
		CER	WER	CER	WER	CER	WER
RIMES	10163	4.4	10.8	2.7	10.7	**2.5**	10.4
IAM	6152	7.2	22.2	5.9	20.3	**5.3**	19.7
Washington	325	41.1	85.3	18.7	69.2	**17.5**	65.2
Parzival	350	18.2	63.0	14.1	56.4	**12.9**	53,6
ICFHR18-G	11424	12.2	43.7	10.6	40.1	**9.7**	38.6
Konzil	351	37.1	95.4	26.2	93.4	**21.5**	90.1
Schiller	238	45.4	88.2	32.4	87.6	**30.1**	85.5
Ricordi	273	37.2	93.1	36.2	91.1	**35.2**	90.3
Patzig	473	24.5	86.3	18.5	81.3	**17.1**	80.5
Schwerin	782	21.1	76.6	17.4	72.9	**16.5**	71.2

5.4 Combining data augmentation and transfer learning

When comparing DA with TL, the large databases are excluded from the comparison. They play the role of source databases in the TL approach. Specifically, the IAM is the source dataset when Washington and Parzival are targets and ICFHR18-G in the Konzil, Schiller, Ricordi, Patzig, and Schwerin case. The RIMES database is only used to enhance the comparisons in this section.

In the combination of TL and DA techniques, there are several possible designs. Here we propose the following two schemes. In a first approach, we perform DA

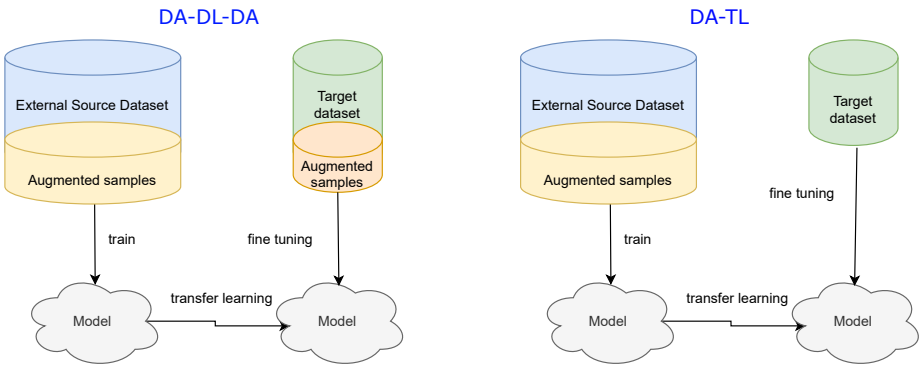


Figure 5.3 Representation of the DA-TL-DA approach (left) and DA-TL approach (right).

at both the learning from the source dataset and the retraining of the model with the target one:

1. Train the model from scratch with a source dataset, applying DA.
2. Retrain the model with the target dataset, applying DA.

We name this proposal DA-TL-DA. In a second proposal, denoted by DA-TL, no DA is applied to the target:

1. Train the model from scratch with a source dataset, applying DA.
2. Retrain the model with the target dataset, **without** applying DA.

The representation of these two different approaches are depicted in Figure 5.3.

With the aim of evaluating the approaches proposed, we perform the same experiments as in Section 4.5, obtaining the results included in Table 5.2. For the sake of completeness and easily comparing the methods, in Table 5.2 are also included the best performance achieved in Section 4.5 (column TL in Table 5.2) and Section 5.3 (column DA in Table 5.2), that is TL with the first layer frozen and DA with RWGD distortions.

In the first step of the DA-TL and DA-TL-DA methods, the model has been trained from scratch with the IAM database. After that, we fine-tune the model using data from the Parzival and Washington databases.

In Table 5.3 we include the results when training the model in Figure 4.1 from scratch with the ICFHR18-G, and being fine-tuned on the 5 specific target data sets provided. For the sake of completeness, we include in Table 5.3 the results for 0 pages in the target dataset, i.e., when no labeled sampled from the target is used. Note that in this case DA-TL-DA cannot be applied.

Table 5.2 TL and DA combined performance: Mean CER (%) evaluated for Washington and Parzival datasets using TL and DA with IAM database as the source. The number of annotated lines used in training is included as 'Train size'.

	Train size #lines	No method	TL	DA	DA-TL-DA	DA-TL
Washington	150	51.6	9.4	22.8	10.0	**9.3**
	250	46.4	7.1	20.4	7.4	**7.0**
	325	41.1	5.4	17.5	5.4	**5.4**
Parzival	150	21.9	5.8	15.7	6.0	**5.6**
	250	20.7	4.0	14.2	4.2	**3.8**
	350	18.2	3.3	12.9	3.4	**3.3**

In the light of Table 5.2 and Table 5.3, it can be concluded that applying DA over the target training set once TL is applied, i.e., DA-TL-DA, either does not reduce the CER or even it slightly increases it, compared to the result of the TL approach alone or the DA-TL method. Except for Schwerin, in which DA-TL-DA slightly improves DA-TL. Put in other words, in general, it is harmful to apply DA to the target dataset if TL has been applied, when just a reduced number of labeled lines are available in the target. On the other hand, DA+TL achieves improvements up to 5 % in the ICFHR 2018 target documents, usually increasing with the reduction of the training set.

From the discussion above, and bearing Table 5.2 and Table 5.3 in mind, it can be concluded that DA-TL is a robust approach. When fine-tuning a DNN that has been previously trained with a similar task (a massive database of HTR samples), the starting point is reasonably good as we can observe in Table 5.3 for the training set sizes of 0 pages in all datasets. We show a good generalization ability of the model for the TL and DA+TL without further training with the target dataset. Afterward, the DNN model is trained with the target database. Only a few samples are available in the target set, representing just a limited part of the support of its marginal distribution, $P_T(\mathbf{x})$. After TL, the parameters of the DNN encode information from both the source and the target training sets. At this point, we conjecture that by using DA in the target dataset and further refining the parameters, the DNN model overfits to the augmented versions of the target samples, forgetting the knowledge learned from the source one, that very much helps to transcript inputs out of the support generated by augmenting the target set. This leads to an increase of the final CER.

Table 5.3 TL and DA combined performance: Mean CER (%) evaluated in ICFHR 2018 Competition Specific datasets as targets using TL and DA with ICFHR18-G as source. The number of annotated pages used in the training is included as 'Train size'.

	Training set size. # pages	None	TL	DA	DA-TL-DA	DA-TL
Konzil	0	–	15.50	–	–	**14.50**
	1 (29 lines)	48.10	10.85	37.30	14.20	**10.80**
	4 (116 lines)	45.30	6.54	28.71	8.00	**6.51**
	12 (351 lines)	37.10	4.37	21.50	5.00	**4.32**
Schiller	0	–	**24.60**	–	–	**24.60**
	1 (21 lines)	53.70	17.36	39.50	21.40	**17.31**
	4 (84 lines)	48.40	12.25	33.20	14.00	**12.22**
	12 (238 lines)	45.40	9.42	30.10	10.00	**9.38**
Ricordi	0	–	39.19	–	–	**34.23**
	1 (19 lines)	56.20	23.66	51.0	24.10	**22.71**
	4 (88 lines)	43.52	21.17	40.81	21.10	**21.02**
	12 (273 lines)	37.21	11.20	35.20	**10.91**	11.14
Patzig	0	–	41.50	–	–	**38.21**
	1 (38 lines)	42.54	27.91	35.31	31.42	**26.7**
	4 (156 lines)	37.63	16.40	30.50	18.32	**16.11**
	12 (473 lines)	24.50	10.60	17.13	11.21	**10.00**
Schwerin	0	–	34.53	–	–	**31.32**
	1 (68 lines)	38.40	12.15	30.20	**10.62**	10.80
	4 (264 lines)	29.31	5.73	24.30	**5.30**	5.52
	12 (782 lines)	21.10	3.51	16.50	**3.30**	3.41

5.5 Comparison to the state-of-the-art

Compared to the state-of-the-art in the ICFHR 2018 Competition Database, we include the results of the proposal in this chapter in the lowest row of Table 5.4. In this table, we show how the efficient combination of TL-DA achieves the best result in the ICFHR 2018 Competition.

Although the mean value of CER is better than the other contestants, we see that there are some databases in which our results are worse than the others. In Chapter 6 we propose a new method that improves the results.

Table 5.4 CER ICFHR 2018 Competition results for LSTM based models: upper part, other previous approaches and, in the lower part, the results for the approaches in this Chapter 4 AND Chapter 5. Lowest mean values are highlighted in boldface.

	CER (%) per training size				CER (%) per document					Mean
	0	1	4	16	Konzil	Schiller	Ricordi	Patzig	Schwerin	
OSU[101]	31.40	17.74	13.27	9.02	9.39	21.10	23.27	23.17	12.98	17.86
ParisTech[20]	32.25	19.80	16.98	14.72	10.49	19.05	35.60	23.83	17.02	20.94
LITIS[95]	35.30	22.51	16.89	11.34	9.14	25.69	30.50	25.18	18.04	21.51
PRHLT	32.79	22.15	17.89	13.33	8.65	18.39	35.07	26.26	18.65	21.54
RPPDI[64]	30.80	28.40	27.25	22.85	11.90	21.88	37.29	32.75	28.55	27.32
TL[8]	32.77	19.51	15.12	8.26	9.16	21.00	29.39	23.25	13.54	18.93
DA-TL	31.55[7]	19.21	14.91	8.16	8.58	21.68	27.84	22.35	12.50	**17.83**

5.6 Conclusions

In this chapter, we show that before performing TL, applying DA in the source dataset does reduce the CER. However, applying DA to the target datasets jointly with TL exhibits worse results than using TL alone. Hence, we propose the DA-TL approach where the DA is applied to the source dataset in the TL process. This technique gets the best CER in the ICFHR 2018 Competition.

6 The Corrupted Label Purging (CLP) Algorithm

In previous chapters, we proposed two techniques to improve the performance of HTR models over small datasets, that is, datasets that have few samples of manual-annotated lines. We showed that as TL as DA combined efficiently with TL have a significant impact on the generalization in the learning of the model. However, in a deeper analysis of the results, we detected that when we have a small database, if some of the training lines have some errors, the model's performance quite deteriorates, as we work with few labeled lines.

In this chapter, we focus not only on the impact of the number of lines but also on their quality in the target dataset on the learning process of the DNN model. We first analyze the impact of the performance on the number of *healthy* lines, i.e., lines with no transcription errors in the training dataset. Then we study how this performance degrades with label errors. Finally, we propose an algorithm to detect and remove potential label errors in the dataset. Part of the results of this chapter was presented in [6] and published in [7].

The model used for carrying out the experiments is the same, which was presented in Chapter 4.

6.1 Performance variation with number of labeled samples

When a few lines are available in the target training set, deep learning models are pretty sensitive to minor variations in the number of labeled lines. In this subsection, this sensitivity is evaluated on a specific dataset from the ICFHR

2018 Competition [92]. The chosen training dataset consists of 16 pages from the Konzil, which is segmented at line level. Similar results were obtained for the other datasets.

The ICFHR 2018 target datasets have 16 labeled pages each. Unless otherwise indicated, 4 of them will be used for testing purposes, while up to 12 pages will be used for training. Usually, 10 % out of the used training set is devoted to validation. The ICFHR18-G dataset is used as a source database in the TL-DA approach.

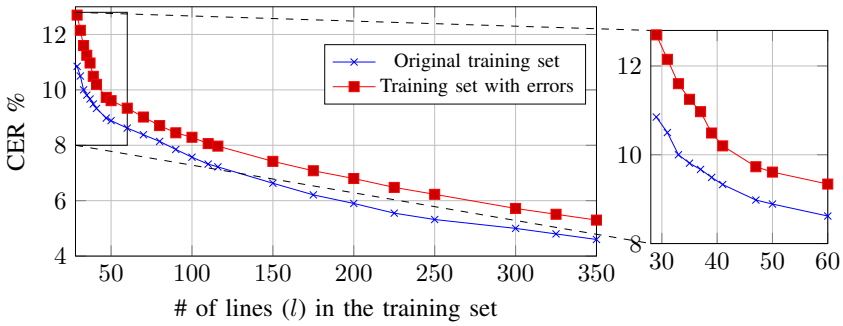
In Figure 6.1.(a) the blue curve in $\times$ represents the TL-DA CER versus the available number of lines, l, of the target training set in the range 29-350 lines, corresponding to 1 and 12 pages, respectively. In the left part of the figure, the CER decreases at a rate of 1 percentage point every 4 new lines added to the training set. After approximately 50 lines, the decreasing rate of the CER changes to approximately 1 percentage point every 100 lines. This is evidenced in Figure 6.1. (b) where the absolute value of the variation of the CER in percentage points is depicted ΔCER, with the increment of the number of annotated lines used in the target to achieve it, Δl. It can be concluded that the sensitivity to the number of samples in the training set is significantly larger for small training sets.

In Figure 6.1 we also include the "Training set with errors" curve (■), which corresponds to the analysis above but where labeling errors have been artificially introduced, as follows. The annotation of a line is modified with probability L. Then, with modified labeling, a character is changed with probability R, in both cases following a Bernoulli distribution. Every changed character is replaced by an independently and randomly selected character, following a discrete uniform probability. In Figure 6.1, where $L = 0.2$ and $R = 0.3$, it is interesting to note that the impact of labeling errors in the CER value is more dramatic for small training sets while the rate at which the CER decreases with the number of lines added remains roughly unaltered.

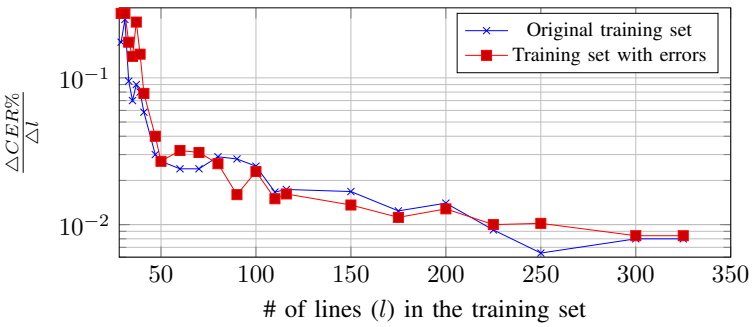
6.2 Types of transcription errors

Before proposing approaches to detect mislabels in the training set, we discuss three typical types and causes of errors in the datasets.

1. *Mislabeled characters.* When labeling a training set, the most common mistake is to confuse a character with another, usually similar. This can be seen in the well-known IAM database [62], where it is indicated that some lines could have some annotation errors in the labels. This type of error is the one simulated in Figure 6.1.



(a)

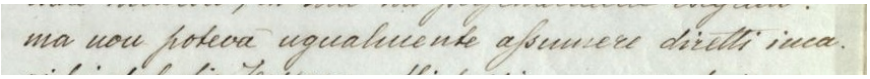


(b)

Figure 6.1 (a) CER (%) divided by the number of annotated lines, l, used and (b) decrement of CER (%) divided by the number of new labeled lines added to obtain it, Δl, in the training of the DNN model with DA-TL approach using the ICFHR18-G dataset as the source and the Konzil dataset as the target with no artificial errors (×) corrupted with artificial errors (■).

2. *Label Misalignment.* The second kind of detected error happens due to a misalignment in the labels. This could be caused by, e.g., a mistake in the name given to some images in the database. This error is encountered several times in the Ricordi dataset from the ICFHR 2018 Competition [92] as illustrated in Figure 6.2. It can be observed in this example that the transcript does not correspond to the handwritten text in the image above.

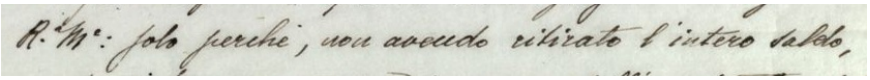
On the contrary, it is pretty close to the model output, after being trained with several lines of the dataset.



GT: *meno d'osservarle che cio non e corretto: in ogni.*

Model output: *ma non poteva nqualuiente assunere direlti inca¬*

Figure 6.2 Sample of a completely mislabeled text at Ricordi dataset.



GT: *R**[icchezz]**a M**[obil]**e solo perche, non avendo ritirato l'intero saldo.*

Model output: *N.° Mi: solo perche, non avendo ritirato l'intero saldo*

Figure 6.3 Sample of special annotations in the GT at the Ricordi dataset.

3. *Special annotations in the ground truth.* Perhaps, the most common source of error is due to special annotations that some transcribers or database managers introduce in some datasets to include some notes inline. In [67] they found this problem in the IAM database: crossed out words that are labeled with the symbol "#" followed by the word behind the blot. Training the model with this labeling might lead to unpredictable behavior since the model could replace the text using "#" at different parts of the text. The model will either be able to recognize the text behind the blot or replace the word with the symbol "#" or both. Another special annotation is included in Figure 6.3, where they write in brackets extra characters that are not in the handwritten text. The output of a model trained with samples of the same dataset is shown below the GT. Despite this line, the CER is about 35%, and it can be observed that the model output is quite similar to the handwritten text.

Manually annotating historical documents remains a challenging task that is prone to errors, even for experts in the field. As discussed in the previous section, when a massive set of annotated samples is available, deep learning models do not suffer from a few mislabeled samples, as they better generalize. However, when a limited set of annotated lines of a specific writer is available to train, mislabeled lines induce overfitting to transcripts with errors, which are pretty hard to tackle via regularization. In the example shown in Figure 6.1, we illustrate this problem when just a few mislabeled lines are introduced.

Algorithm 6.1 Corrupted Labels Purging (CLP)

Given inputs: source set $x_S \in \mathcal{X}_S$ and $y_S \in \mathcal{Y}_S$, target set $x_T \in \mathcal{X}_T$ and $y_T \in \mathcal{Y}_T$ and threshold, ε.

1) Fit the prediction function $f_S(y_S|x_S)$ with the source training set $\{x_S, y_S\}$.

2) Split the target training set into N subsets $\{x_{T_1}, x_{T_2}, \ldots, x_{T_N}\}$, $\{y_{T_1}, y_{T_2}, \ldots, y_{T_N}\}$.

for $n = 1, \ldots, N$ **do**

3) Initialize the prediction function $f_{T_n}(\cdot) = f_S(\cdot)$.

4) Fine tune the prediction function with the whole target set except for the nth, $\{x_{T_i \neq T_n}\}$, $\{y_{T_i \neq T_n}\}$.

5) Include in the new target set, $\{x_{T'}, y_{T'}\}$, all pairs $\{x^{(i)}_{T_n}, y^{(i)}_{T_n}\}$ whose predictions $f_{T_n}(y^{(i)}_{T_n}|x^{(i)}_{T_n})$ have errors below a CER threshold, ε.

end for

6) Initialize the prediction function $f_T(\cdot) = f_S(\cdot)$.

7) Fine tune the prediction function, $f_{T'}(y_{T'}|x_{T'})$, to the modified target set $\{x_{T'}, y_{T'}\}$.

Output:

Function $f_T(y_T|x_T)$ over the target domain $\mathcal{D}_T$.

6.3 Mislabel detection algorithm

As one of our main contributions, we propose an algorithm to detect and remove mislabeled lines from the training set, detailed in Algorithm 6.1. A block diagram of the algorithm is also depicted in Figure 6.4. It divides the target training dataset into N subsets. For every subset, n, the method performs DA-TL using the rest of subsets, $k = 1, \ldots, N$, $k \neq n$, as training sets and it evaluates the CER metric over the subset n. Lines with CER above a threshold, ε, in the nth subset are detected as wrongly transcribed and discarded. Hence, we are implementing some k-fold validation, in which the size of each validation partition is reduced after removing problematic lines. Finally, the DA-TL is applied to the resulting target database. Note that the algorithm performs $N+1$ different training steps. However, the N steps concerning the target subsets could be run in parallel since they are independent of each other. Hence, the run time of applying this algorithm is approximately double the run time of regular training.

In Figure 6.5 we include the histogram of the CER per line for the 5 ICFHR 2018 document-specific datasets using the CLP algorithm with $N = 2$. The ICFHR18-G was used as the source. The histograms were estimated with the CER of the outputs of the $n = 1,2$ stages computed with the lines not used during training, see

the output of "Target subset n" blocks in Figure 6.4. Models have been trained with 4 pages in the left column, while in the right column, they have been trained with 12 pages. Lines are corrupted with artificial errors with probability $L = 0.1$, while every character in the line label is changed with probability $R = 0.3$ to a random value.

Conservatively, we believe that a 10% average number of corrupted lines represent a label error rate similar to the one we encounter in authentic databases. It is interesting to observe that the results for the Schwerin dataset are remarkably better than for the others because it has a significantly more significant number of lines per page. Besides, in the Ricordi dataset, the histogram for 12 pages exhibits large values around 0.8. This dataset is known to have label misalignments.

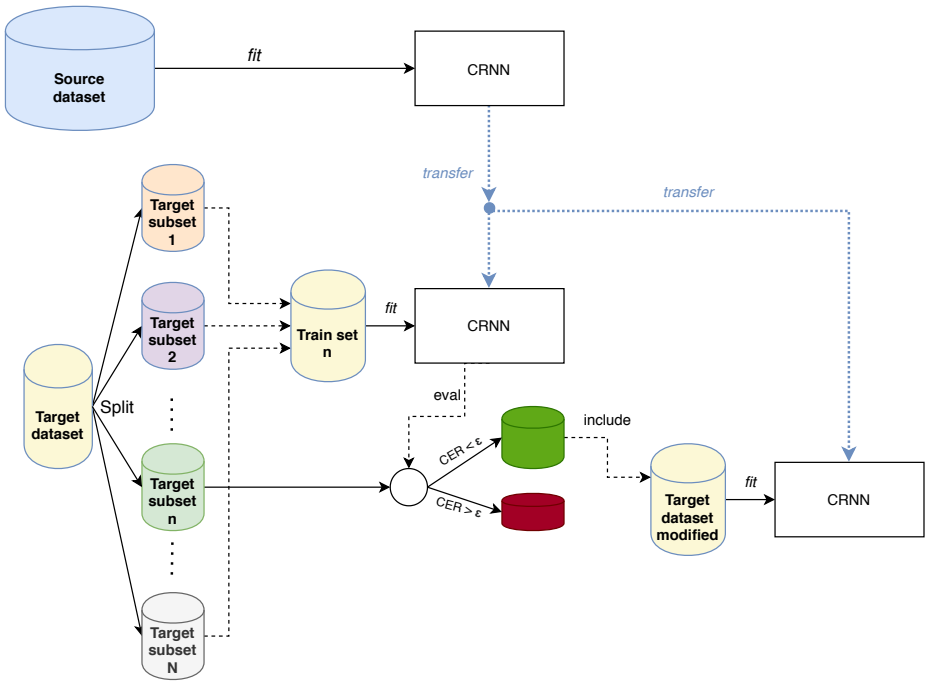


Figure 6.4 Corrupted labels purging algorithm. The algorithm applied over target subset n is depicted. The same procedure should be applied to all the subsets to build the *Target dataset modified*.

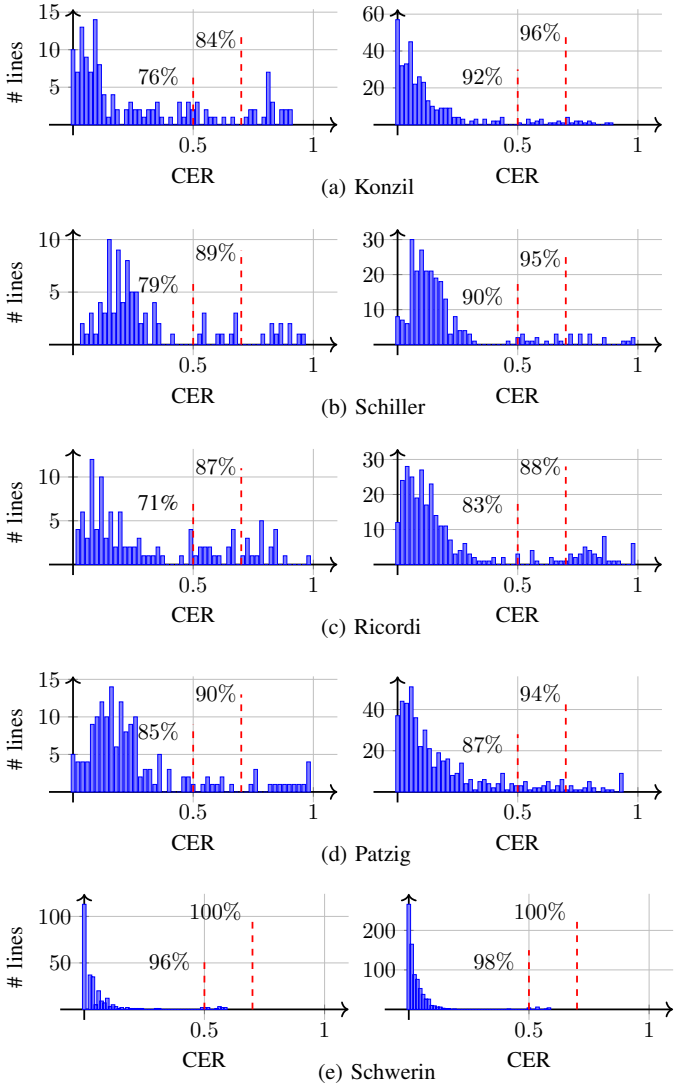


Figure 6.5 Histogram of CER with DA-TL and ICFHR18-G as source dataset for the 5 document-specific datasets using 4 pages (left) and 12 pages (right) of the target dataset. Lines and characters were corrupted with probabilities $L = 0.1$ and $R = 0.3$ respectively. The histograms were evaluated with the outputs of the $N = 2$ target subsets. Red dashed lines indicate the percentage of lines with CER$\leq \varepsilon$ with $\varepsilon = 50\%$ and $\varepsilon = 70\%$, left and right lines, respectively.

6.4 CLP threshold analysis

The selection of the threshold is central to the algorithm performance. In Figure 6.5 the CER of the healthy lines is mainly distributed around a mode value to the left of each histogram, while outliers exhibit larger values. As representative values to be studied, after extensive simulations, we restrict our analysis to the thresholds $\varepsilon = 0.5$ and $\varepsilon = 0.7$, for an average rate $L = 0.1$ of artificially modified lines, and $R = 0.3$. In Figure 6.5 we indicate the percentage of lines with CER equal or lower than 0.5 and 0.7, left and right red dashed lines in the subfigures, respectively. We conclude that almost 10% of lines have a CER above $\varepsilon = 0.7$ when 4 pages for training are available, and the same occurs in the case of 12 pages when $\varepsilon = 0.5$.

The selection for ε should not lead to the deletion of healthy lines. Otherwise, the overall CER would rise. On the other hand, the threshold must ensure a sensitivity when corrupted lines are encountered.

In the following, we study the CLP algorithm in two different scenarios. The first experiment we perform consists of applying the CLP algorithm to the ICFHR 2018 target datasets, with 4 and 12 pages as target training set size. Then we evaluate the CLP for the Washington and Parzival databases, with 150 and 325 lines as target training set sizes. The same procedure is followed through all the scenarios:

1. Fit the model to the source set.
2. Run DA-TL plus CLP with N = 2.

6.4.1 ICFHR 2018 Competition results

We test the CLP algorithm over real databases where we do not have any prior knowledge about the pattern of labeling errors. We do also include artificial errors to evaluate the CLP robustness.

The results of these analyses are reported in Table 6.1 and Table 6.2. Their three last columns include the results for the DA-TL with no CLP as 'Baseline', for the DA-TL+CLP with $\varepsilon = 50\%$, and then for the DA-TL+CLP with $\varepsilon = 70\%$. For every target dataset and training set size three rows are used to report the CER (%) when no artificial errors are introduced, $R = 0$, for $R = 30\%$ and $R = 50\%$.

In this first case, the ICFHR18-G dataset is used as the source. The 17 documents of this corpus have a total number of 11424 lines. The DA-TL plus CLP was applied to the five target documents in the competition: Konzil, Schiller, Ricordi, Patzig, and Schwerin. The results are included in Table 6.1, where it is included the average value for the CER and the number of removed lines by the CLP algorithm.

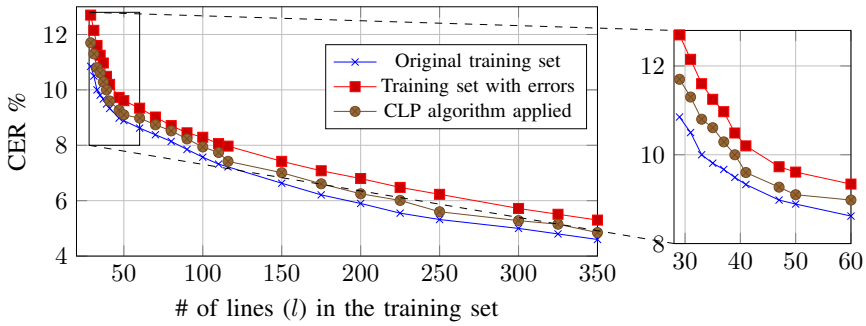


Figure 6.6 CER (%) divided by the number of annotated lines, l, with the DA-TL approach using the ICFHR18-G dataset as source and the Konzil dataset as target with no artificial errors (×), with artificial errors (■) and with artificial errors and CLP used (•).

Given the results, we highlight the following aspects. First, note that, when errors are induced, the threshold $\varepsilon = 70\%$ performs better in most cases when the training set is 4 pages while the threshold $\varepsilon = 50\%$ is the best choice for 12 pages. Exceptions can be observed in Patzig and Schwerin corpora. For the Patzig dataset, we conclude that $\varepsilon = 70\%$ is the best choice for every case. This is due to the distribution of the errors in this dataset, which has a more considerable variance, and therefore more lines are above the $\varepsilon = 50\%$ CER, it can be seen in Figure 6.5. In the Schwerin corpus, the threshold 50% has the best CER in all cases, the opposite that in the Patzig dataset. This is due to the distribution of the errors in this dataset that, due to the more significant number of lines used, has lower variance and most lines are below 10% CER (see Figure 6.5).

It is also interesting to remark that in the Ricordi case, the algorithm improves the CER in the original dataset, i.e., without synthetic errors. This is explained by the fact that in this dataset, as already discussed, there are some mislabeled lines like in the case illustrated in Figure 6.2. Additionally, note that for $R = 0$ and $\varepsilon = 70\%$, a large number of removed lines is quite an indicator of the dataset containing errors in the annotated lines.

For the sake of completeness, we include in Figure 6.6 the evolution of the CER versus the number of lines, l, in Figure 6.1 including the CER for the proposed algorithm (CLP) (○). The introduction of the CLP improves the TL-DA approach when the dataset has corrupted lines. In the range, $l = [40,50]$ the TL-DA with CLP with $l = 40$ achieves the same CER as the DA-TL with $l = 50$ lines in the training set.

6.4.2 Washington and Parzival results

In this second analysis, the model is pre-trained with the IAM database as the source dataset to train the model with DA-TL for the Washington and Parzival targets. There are two main differences to the previous study of the ICFHR 2018 datasets: 1) the number and set of characters are different from the source and targets datasets, and 2) we compare the CER of both targets in terms of the number of lines instead of the number of pages, where we consider two cases, 150 lines and 325 lines, similar to the number of lines used in the previous scenario.

The first rows in Table 6.2 include the results obtained after fine-tuning the model to the Washington dataset. In this study, the threshold $\varepsilon = 70\%$ is the best option when the number of lines is 150 while $\varepsilon = 50\%$ exhibits the lowest CER when the number of lines is 325. This is equivalent to the 4 and 12 pages in the Konzil, Schiller, and Ricordi cases in which the number of lines is similar. We get an improvement of 0.8 and 0.63 in the case of 150 lines and no deterioration over the original dataset for these thresholds. In the case of 325 lines, we get a boost of 0.4 and 0.5 and no deterioration over the original dataset.

Results obtained after fine-tuning the model to the Parzival dataset are also included in Table 6.2, see the lower rows. Similar conclusions can be drawn except for $R = 30\%$ and 150 lines, where the 50% exhibits the best CER. If we choose the threshold as in the previous cases, 70% and 50%, we still get a slight improvement or at least no deterioration.

6.5 Correcting label misalignment

In Section 6.2 we summarized the different types of transcription errors. One of these errors is due to the misalignment of the annotations with the images. When a high number of lines are classified as mislabels, this type of error can be addressed by searching within the outputs of the DNN model for the whole target dataset, the transcript best fitting every annotation in the GT, hence aligning annotations and images in the dataset. This approach is quite similar to the one proposed in [80].

In the case of the Ricordi dataset in the ICFHR 2018 competition, we realized that the CLP detected a high number of mislabeled lines in the dataset. Note the large numbers of removed lines in Table 6.1 for $\varepsilon = 70\%$ and this dataset with $R = 0$. By simply visual inspection, we confirmed that the error was of misalignment of images and annotations. Here, we apply the CLP plus the simple automatic alignment approach described above.

The comparison between simply removing the mislabeled lines and correcting the alignment of the database is shown in Table A.6. In this table, one can observe a significant dropping in the CER when correcting these misalignments of the lines. In training with 4 pages, the overall decrease is 3.7 percentage points. In the 12 pages analysis, the CER drops 0.3 percentage points when removing the lines while it further decreases 0.8 percentage points when correcting them. Note also that the gain is higher when a lower number of annotated lines are used.

6.6 Comparison to the state-of-the-art

Finally, following the comparison made in previous chapters, we sent the transcriptions made using this approach to the ICFHR 2018 Competition. We can see that by deleting the mislabeled samples, we can achieve an improvement of 0.4 percentage points if compared with the results in Chapter 4 and Chapter 5.

The recent work published by Yousef et al. [107] using a DNN model based on a fully GCN, outperformed the LSTM based approaches, with a mean value of 13.02 % providing a 23.35 % CER for a 0-page training size.

The results of the proposal in this work are included in the lowest rows of Table 6.4 where, following the conclusions in Subsection 6.4, we used $\varepsilon = 70\%$ for the 1 and 4 pages training and $\varepsilon = 50\%$ for the 16 pages. Also, the CLP includes an alignment stage. Results are presented in three groups of columns. First, the average CER (%) for the 5 target dataset is included when 0, 1, 4, and 16 pages of the target datasets are used. The second group of 5 columns reports the average CER (%) for the learning with 0, 1, 4, and 16 pages in the dataset for every document. The mean value per row is included in the last column.

6.7 Conclusions

Comparing to the state-of-the-art in the ICFHR 2018 Competition, it can be observed that the DA-TL and CLP outperform all approaches within the CNN + LSTM + CTC class hence underlining the importance of the issues discussed: DA is essential, but in the source dataset, TL is to be considered, and mislabeling detection and correction is essential if the dataset exhibits errors. Besides, the CLP introduces a residual 0.01 percentage points of loss if the datasets have no errors in the labels, while the reduction is important if they have. See the results for the Ricordi corpus, where a reduction of 6.58 percentage points is achieved. The presence of errors in this database was detected by checking the number of removed lines by the CLP.

It is interesting to mention that other variations of the algorithm have been tried to improve the performance further. In this sense, we tried to evaluate the CTC loss [40] to select a threshold ε. We found it complex to deal with because it depends on several factors like the number of epochs in training or if batch normalization has been applied.

Table 6.1 Mean CER (%) evaluated in Konzil, Schiller, Ricordi, Patzig and Schwerin target documents in the ICFHR2018 Competition datasets for DA-TL, DA-TL+CLP with $\varepsilon = 50\%$ and $\varepsilon = 70\%$. DA-TL was applied with both a training set of 4 pages and 12 pages. The annotation for a line is corrupted with probability $L = 0.1$, and a character within it is randomly replaced with probability R. $R = 0$ indicates no error introduced in the labelings. The number of removed lines by the CLP algorithm is included in parentheses in the last two columns. The best-achieved value in every row is in boldface.

Dataset	Train set size	R	Baseline	$\varepsilon = 50\%$	$\varepsilon = 70\%$
Konzil	4 Pages (116 lines)	0%	**7.6**	8.5(-31)	7.9(-7)
		30%	8.7	8.3 (-41)	**7.82** (-14)
		50%	9.1	8.2 (-39)	**7.9** (-16)
	12 Pages (351 lines)	0%	**4.6**	5.3 (-1)	**4.6** (-0)
		30%	5.3	**4.6** (-29)	5.0 (-25)
		50%	5.5	**4.8** (-35)	5.0(-28)
Schiller	4 Pages (84 lines)	0 %	**13.27**	14.72 (-12)	13.61 (-5)
		30 %	15.19	14.81 (-17)	**14.43** (-10)
		50 %	15.64	14.96 (-22)	**13.87** (-12)
	12 Pages (244 lines)	0 %	**9.42**	9.76 (-2)	9.42 (-0)
		30 %	11.31	**10.41** (-22)	10.62 (-22)
		50 %	12.75	10.61 (-24)	**10.51** (-25)
Ricordi	4 Pages (88 lines)	0 %	21.1	18.2 (-16)	**18.2** (-16)
		30 %	23.2	20.8 (-32)	**20.5** (-27)
		50 %	24.31	21.94 (-44)	**20.81** (-27)
	12 Pages (295 lines)	0 %	9.7	**9.4** (-38)	9.4 (-38)
		30 %	10.47	**9.23** (-41)	9.49 (-38)
		50 %	10.8	**9.53** (-52)	9.75 (-44)
Patzig	4 Pages (156 lines)	0 %	**18.32**	18.93 (-7)	**18.32** (-0)
		30 %	21.41	21.6 (-27)	**21.1** (-18)
		50 %	21.84	22.12 (-27)	**21.31** (-18)
	12 Pages (473 lines)	0 %	**11.5**	11.96 (-15)	11.54 (-4)
		30 %	12.28	12.23 (-61)	**11.98 (-52)**
		50 %	12.8	12.67 (-63)	**12.35 (-54)**
Schwerin	4 Pages (264 lines)	0 %	5.3	5.3 (-0)	**5.3** (-0)
		30 %	5.36	**5.31** (-14)	5.36 (-0)
		50 %	5.39	**5.32** (-26)	5.33 (-12)
	12 Pages (782 lines)	0 %	3.3	3.3 (0)	**3.3** (0)
		30 %	3.36	**3.31** (-14)	3.36 (-0)
		50 %	3.53	**3.34** (-75)	3.39 (-22)

Table 6.2 Mean CER (%) evaluated in Washington and Parzival documents for DA-TL, CLP with threshold $\varepsilon = 50\%$ and CLP with threshold $\varepsilon = 70\%$. DA-TL was applied with the IAM dataset as the source and using 150 and 325 lines from the target. The annotation for a line is corrupted with probability $L = 0.1$ and a character within it randomly replaced with probability R. $R = 0$ indicates no error introduced in the labelings. The number of removed lines by the CLP algorithm is included in parentheses in the last two columns.

Dataset	Train set size	R	Baseline	$\varepsilon = \mathbf{50}\%$	$\varepsilon = \mathbf{70}\%$
Washington	150 lines	0 %	9.4	9.5 (-6)	**9.4** (-2)
		30 %	11.3	10.6 (-20)	**10.5**(-14)
		50 %	11.5	11.1 (-31)	**10.87**(-19)
	325 lines	0 %	5.3	**5.3** (-2)	5.3 (-0)
		30 %	6.1	**5.7** (-26)	6.1 (-0)
		50 %	6.3	**5.8** (-34)	6.3 (-0)
Parzival	150 lines	0 %	5.8	5.8 (-0)	**5.8** (-0)
		30 %	6.4	**6.0** (-15)	6.2 (-2)
		50 %	6.6	6.2 (-20)	**6.1**(-14)
	325 lines	0 %	3.3	**3.3** (-0)	3.3 (-0)
		30 %	3.5	**3.5** (-0)	3.5 (-0)
		50 %	3.5	**3.4** (-35)	3.5 (-0)

Table 6.3 Comparison between the CLP algorithm with line removal and the CLP plus alignment of the GT after detection. The mean CER (%) is evaluated for the Ricordi document with a training set of size 4 pages (88 lines) and 12 pages (295 lines).

Train set size	**Method**	Baseline	$\varepsilon = 50\%$	$\varepsilon = 70\%$
4 Pages (88 lines)	CLP	21.1	18.2	18.2
	CLP + alignment	21.1	17.4	17.4
12 Pages (351 lines)	CLP	9.7	9.4	9.4
	CLP + alignment	9.7	8.9	8.9

Table 6.4 CER ICFHR 2018 Competition results for LSTM based models: upper part, other previous approaches and, in the lower part, the results for the approaches in this work. Lowest mean values in both parts are highlighted in boldface.

	CER (%) per training size				CER (%) per document					Mean
	0	1	4	16	Konzil	Schiller	Ricordi	Patzig	Schwerin	
OSU[101]	31.40	17.74	13.27	9.02	9.39	21.10	23.27	23.17	12.98	**17.86**
ParisTech[20]	32.25	19.80	16.98	14.72	10.49	19.05	35.60	23.83	17.02	20.94
LITIS[95]	35.30	22.51	16.89	11.34	9.14	25.69	30.50	25.18	18.04	21.51
PRHLT	32.79	22.15	17.89	13.33	8.65	18.39	35.07	26.26	18.65	21.54
RPPDI[64]	30.80	28.40	27.25	22.85	11.90	21.88	37.29	32.75	28.55	27.32
TL	32.77	19.51	15.12	8.26	9.16	21.00	29.39	23.25	13.54	18.93
DA-TL	31.55	19.21	14.91	8.16	8.58	21.68	27.84	22.35	12.50	17.83
CLP	30.13	19.10	12.40	7.93	8.59	21.69	22.81	22.35	12.51	**17.39**

7 Handwriting Text Generation

7.1 Introduction

In previous chapters, we propose three different methods to improve the performance of HTR models when only a few manually annotated samples are available to train the models. Specifically, in Chapter 4 we propose how to perform TL over state-of-the-art networks which have been previously trained with a huge database of modern handwriting text. In Chapter 5 we perform different HTR data augmentation techniques in combination with the TL method proposed in Chapter 4. Finally, in Chapter 6, we solve a problem with the samples in the training set which have been mislabeled in the process of manual annotation. We propose the CLP algorithm to solve this problem.

In this chapter, our goal is to go further in the data augmentation side we use in Chapter 5 in combination with TL to augment the databases. In Chapter 5 we generate new samples of pairs input (images of handwriting text lines) - labels (sequence of characters corresponding to the text in the input image) by distorting the original images. In that chapter, we differentiated two techniques for distorting the images by classical operations such as rotation, scaling, and shearing and the RWGD distortions proposed in [101].

The techniques mentioned above have one common thing, they only generate new handwritten text images, but the labels are the same. This can lead to a kind of lexicon overfit. In this chapter, we aim to generate new samples of pairs of image labels of handwriting text that have new information in the site of the images and the text that images contain. Our goal is to generate new pairs of images and labels for any given text.

Previous approaches of handwriting generation have been focused on the online handwriting generation task where the data is collected in a digital device, that is,

the input of the models are not images, but the information about the position of the pen over a digital surface when someone is writing text [39, 66, 63, 2].

Recently, in the field of offline HTR, since 2019, some authors proposed generative adversarial networks (GAN) in order to generate offline handwritten images [50, 4, 35, 53, 43]. However, all those settings have in common that they only generate images of isolated words. In [27] Brain Davis et al. propose a GAN for generating images of handwritten lines conditioned on arbitrary text.

In the framework of historical documents, some works focus on the generative approach. In [102] they proposed a method to improve the synthesis of word images for the word spotting task. In [89] they perform document enhancement through a GAN.

All these works have in common that either they generate images of single words, or they use a GAN approach to generate images of lines. As mentioned in previous chapters, state-of-the-art performance by transcribing complete lines of text has lower CER than when transcribing isolated words. For that reason, the approach that matches our pipeline is the GAN method proposed in [27] to generate images of text. In that work, they propose a model to generate images of handwritten text over the IAM and RIMES database, and they evaluate parameters such as Fréchet Inception Distance (FID) [47] and Geometry Score (GS) [54].

In this chapter, we alternatively propose a VAE [70] approach to generate samples of complete lines of handwritten text conditioned to a given text. We compare the quality of the generated images by transcribing them using the methods and architectures presented in previous chapters. It is well known that VAE generative models are, in general, more stable to train. They generalized better (less pruned to the mode-collapse issue), and, in addition, the latest VAE proposals based on hierarchies provide highly realistic samples comparables to GANs. We compare the quality of the images generated with the VAE model with the quality of the images generated by the GAN model in [27] applied to historical databases. Furthermore, we report improvements when using VAEs in many of the scenarios considered.

7.2 VAEs for handwriting text generation

In this chapter, we propose two different architectures to implement a C-VAE that generates images of complete lines of text conditioned to a given transcribed text.

7.2.1 Conditional VAE

The first approach is the use of a simple C-VAE [88] applied to images of full lines of text. Usually, non conditioned VAE model the data probability distribution,

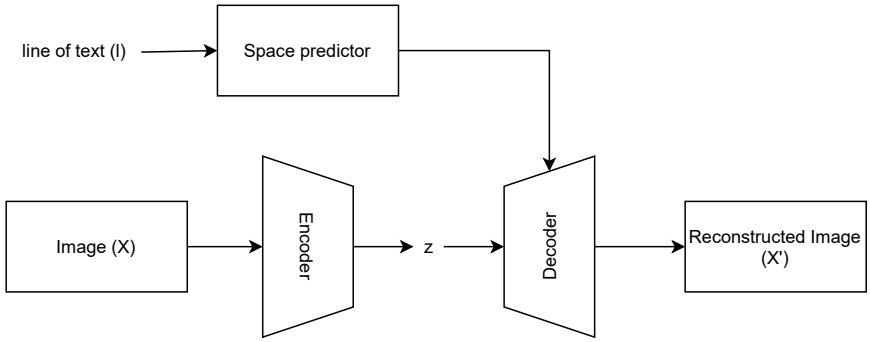


Figure 7.1 Conditional VAE with space predictor use in this thesis.

$p(\mathbf{X})$, which in our case are the images of handwritten text. When generating samples in a VAE, the model uses a latent variable vector, $\mathbf{z}$, from which the data is generated using a likelihood model, $p(\mathbf{x}|\mathbf{z})$. The model has two components, the encoder which infers the probability $p_\theta(\mathbf{z}|\mathbf{X})$ with a variational distribution $q_\phi(\mathbf{z}|\mathbf{X})$ and the decoder which infers the probability $p_\theta(\mathbf{X}|\mathbf{z})$ [70]. Training the model amounts to optimizing the ELBO lower bound given by,

$$\mathcal{L}(\phi,\theta,\mathbf{X}) = -\mathcal{KL}\left[q_\phi(\mathbf{z}|\mathbf{X})\,||\,p_\theta(\mathbf{z})\right] + \mathbb{E}_{z\sim q_\phi(\mathbf{z}|\mathbf{X})}\left[\log p_\theta(\mathbf{X}|\mathbf{z})\right] \tag{7.1}$$

However, this VAE, when trained with handwritten text images, could generate new images without any constraint beyond being similar in style to the images in a database. We are interested in generating new images which represent a given line of text. In that way, we could generate pairs of images-labels that can be used as new data for training the architectures presented in previous chapters.

For that reason, we introduce a new variable $\mathbf{l}$ which represent the line of text to which the generated image is conditioned. If we introduce this new variable in the model, we update equation (7.1) as follows

$$\mathcal{L}(\phi_1,\theta,\mathbf{X},\mathbf{l}) = -\mathcal{KL}\left[q_\phi(\mathbf{z}|\mathbf{X})\,||\,p_\theta(\mathbf{z})\right] + \mathbb{E}_{z\sim q_\phi(\mathbf{z}|\mathbf{X})}\left[\log p_\theta(\mathbf{X}|\mathbf{z},\mathbf{l})\right] \tag{7.2}$$

The model is depicted in Figure 7.1. In the figure, it is shown a block called space predictor. This block is necessary since the length of a line of text is unknown. Each sample would have a line with a different number of characters. For implementation purposes, the width of the input to the decoder has to be related to the width of the image generated. The space predictor takes the line of text and builds an encoding matrix where each character has a specific width. This predictor is trained with the pairs line of text - output of architecture presented in Chapter 4, before the CTC.

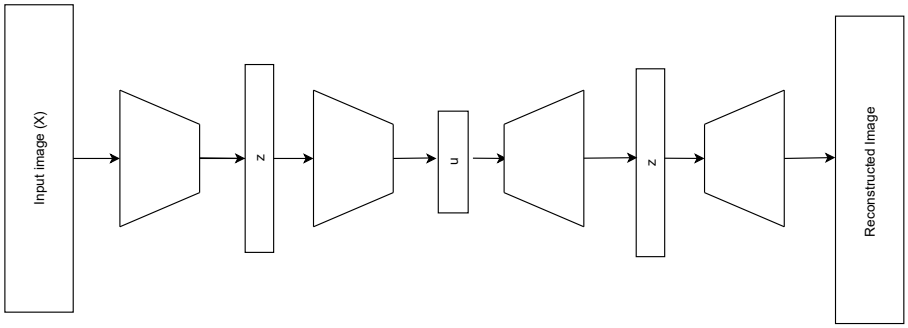


Figure 7.2 Two-stage VAE proposed in [25].

This model is used as a first proposal, and some samples of the image generated can be seen in Figure 7.3. In that figure, it can be observed that the handwritten text generated is too blurred. Intending to reduce this blurriness, in Subsection 7.2.3 we propose a Two-Stage Conditional Variational Auto Encoder (TSC-VAE) based on the two-stage VAE proposed in [25].

7.2.2 TS-VAE model

The fourth contribution of this thesis is presented in this subsection. Based on the Two-Stage Variational Auto Encoder (TS-VAE) in [25], which follows the strategy:

1. Given n observed samples $\{\mathbf{x}^{(i)}\}_{i=1}^{n}$, train a κ-simple VAE, with $\kappa \geq r$, the dimension of the latent space, to estimate the unknown r-dimensional ground-truth manifold $\mathcal{X}$ embedded in $\mathbf{R}^d$ using a minimal number of active latent dimensions. Generate latent samples $\{\mathbf{z}^{(i)}\}_{i=1}^{n}$ via $\mathbf{z}^{(i)} \sim q_\phi(\mathbf{z}|\mathbf{x}^{(i)})$. By design, these samples will be distributed as $q_\phi(\mathbf{z})$, but likely not as $\mathcal{N}(\mathbf{z}|0,I)$.
2. Train a second κ-simple VAE, with independent parameters $\{\theta', \sigma'\}$ and latent representation $\mathbf{u}$, to learn the unknown distribution $q_\phi(z)$, i.e., treat $q_\phi(z)$ as a new ground-truth distribution and use samples $\{\mathbf{z}^{(i)}\}_{i=1}^{n}$ to learn it.
3. Samples approximating the original ground-truth μ_{gt} can then be formed via the extended ancestral process $\mathbf{u} \sim \mathcal{N}(\mathbf{u}|0,I)$, $z \sim p_{\theta'}(z|u)$, and finally $x \sim p_\theta(\mathbf{x}|\mathbf{z})$.

The two-stage VAE proposed in [25] is depicted in Figure 7.2.

In [25] the authors prove that the FID score achieved by this architecture has similar values that state-of-the-art GAN architectures evaluated over MNIST, Fashion MNIST, CIFAR-10, and CelebA datasets.

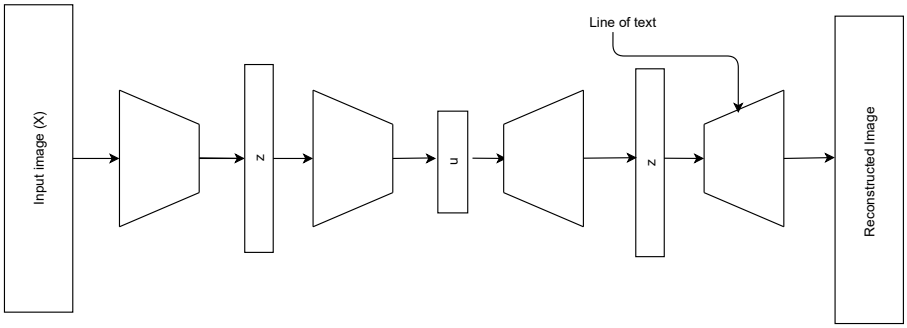


Figure 7.3 Two-stage Conditional VAE in which desired text is the input at the first stage.

We apply this model to HTR, with three remarkable variations: the images are no longer square, images are sequential data, and we want to manage the generated text, so we propose a TSC-VAE.

7.2.3 TSC-VAE model

Two possible choices for providing the desired label to the model are proposed:

1. The desired text sequence, **l**, taking into account at the first stage:

   First stage ELBO:

$$\mathcal{L}\left(\phi_1,\theta_1,\mathbf{X},\mathbf{l}\right) = -\mathcal{KL}\left[q_{\phi_1}\left(\mathbf{z}|\mathbf{X},\mathbf{l}\right)||p_{\theta_1}\left(\mathbf{z}\right)\right] + \mathbb{E}_{z\sim q_{\phi_1}(\mathbf{z}|\mathbf{X},\mathbf{l})}\left[\log p_{\theta_1}\left(\mathbf{X}|\mathbf{z},\mathbf{l}\right)\right] \tag{7.3}$$

   Second stage ELBO:

$$\mathcal{L}\left(\phi_2,\theta_2,z\right) = -\mathcal{KL}\left[q_{\phi_2}\left(\mathbf{u}|z\right)||p_{\theta_2}\left(\mathbf{u}\right)\right] + \mathbb{E}_{u\sim q_{\phi_2}(\mathbf{u}|z)}\left[\log p_{\theta_2}\left(z|\mathbf{u}\right)\right] \tag{7.4}$$

2. The desired text **l** sequence taking into account at the second stage

   First stage ELBO:

$$\mathcal{L}\left(\phi_1,\theta_1,\mathbf{X}\right) = -\mathcal{KL}\left[q_{\phi_1}\left(\mathbf{z}|\mathbf{X}\right)||p_{\theta_1}\left(\mathbf{z}\right)\right] + \mathbb{E}_{z\sim q_{\phi_1}(\mathbf{z}|\mathbf{X})}\left[\log p_{\theta_1}\left(\mathbf{X}|\mathbf{z}\right)\right] \tag{7.5}$$

   Second stage ELBO:

$$\mathcal{L}\left(\phi_2,\theta_2,z,\mathbf{l}\right) = -\mathcal{KL}\left[q_{\phi_2}\left(\mathbf{u}|z\right)||p_{\theta_2}\left(\mathbf{u}\right)\right] + \mathbb{E}_{u\sim q_{\phi_2}(\mathbf{u}|z,\mathbf{l})}\left[\log p_{\theta_2}\left(z|\mathbf{u},\mathbf{l}\right)\right] \tag{7.6}$$

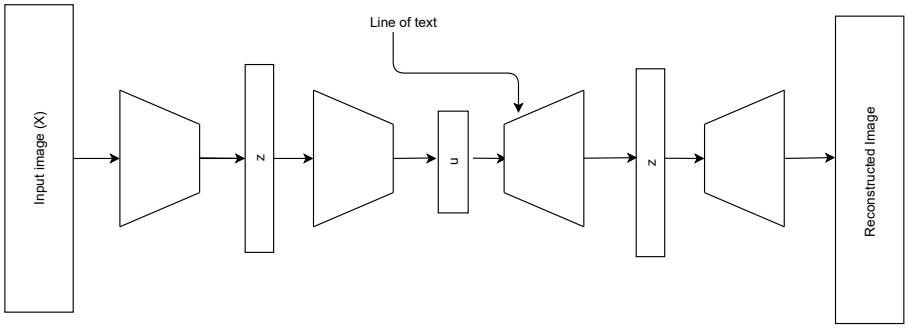


Figure 7.4 Two-stage Conditional VAE in which desired text is the input at the second stage.

7.3 Experiments of generation

In this section, we include some examples of generations of samples with the proposed approaches.

7.3.1 Generation with the TS-VAE

In Figure 7.5 we include some results of data generation with the TS-VAE when the training set used in the experiment is the database ICFHR-2018 Competition. We generated square images of side 128 pixels and appended one after the other to create a full line. This experiment is carried out without taking into account the conditionality of the line of text. The TS-VAE generates random text in images similar to the ones presented in the training set. It can be seen that the strokes of the characters are well defined. In this experiment, we do not have to worry about the width of each character since we are not conditioned the images to any text.

Figure 7.5 ICFHR-2018 Patches generation with a two-stage VAE.

7.3.2 Generation with the C-VAE and TSC-VAE: sequential MNIST

In this series of experiments, we include generated samples when conditioning to a particular text, using the TSC-VAE. As a first simple experiment, we aim at generating a sequence of numbers in one line. The dataset for training is generated by appending images of the MNIST dataset as inputs and concatenating the corresponding numbers to build the label. One significant difference between this database and those used in the subsequent experiments is that the width of each character is the same.

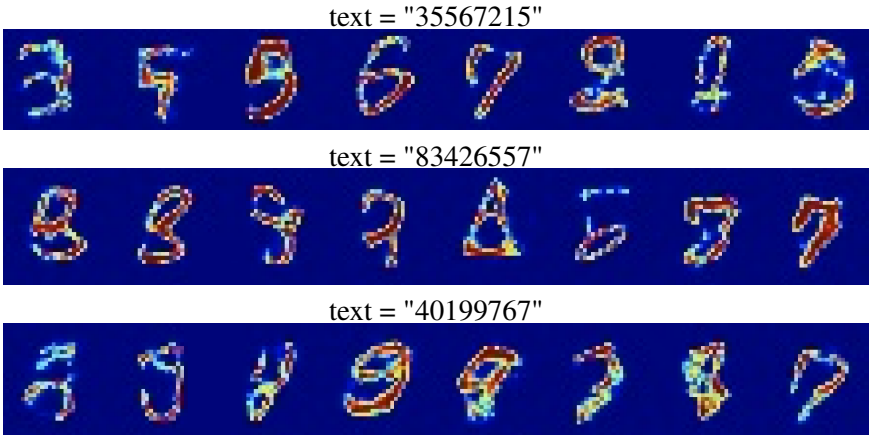


Figure 7.6 Sequential MNIST generated samples in a C-VAE.

If we compare the samples in Figure 7.6 and Figure 7.7, we can observe the differences between the simple C-VAE model and the TSC-VAE model. The images generated by the proposed TSC-VAE model are clearer to human vision than those generated with the single C-VAE.

Once the TSC-VAE model seems to improve the generation of samples from the sequence-MNIST database, in Subsection 7.3.3 we take a set from the ICFHR2018 Competition to show how the model works with historical handwritten text.

7.3.3 Generation with the C-VAE and TSC-VAE: ICFHR 2018 dataset

In this section, we face the challenging problem of generating a new sample of the full line for a historical document dataset. In this case the documents from the ICFHR-2018 Competition database presented en Chapter 3. Some samples of generated images with the C-VAE are included in Figure 7.8 while samples generated with the TSC-VAE are depicted in Figure 7.9. It can be observed that the ones in Figure 7.9 are much near to the ones in the original dataset.

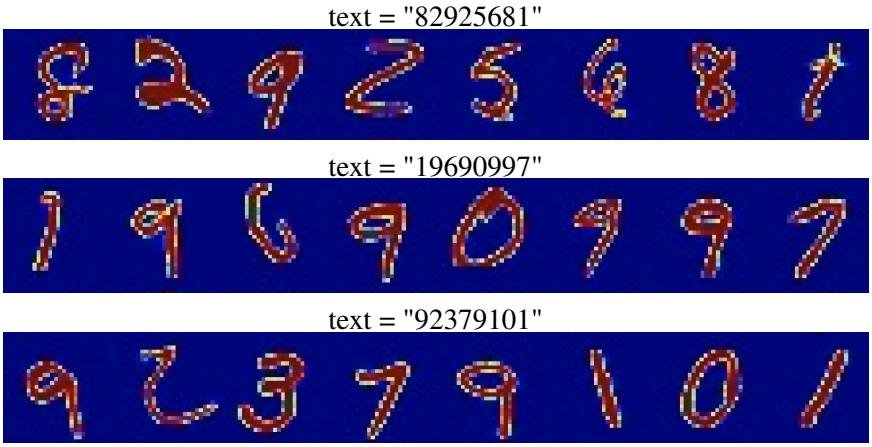


Figure 7.7 Sequential MNIST generated samples in a TSC-VAE.

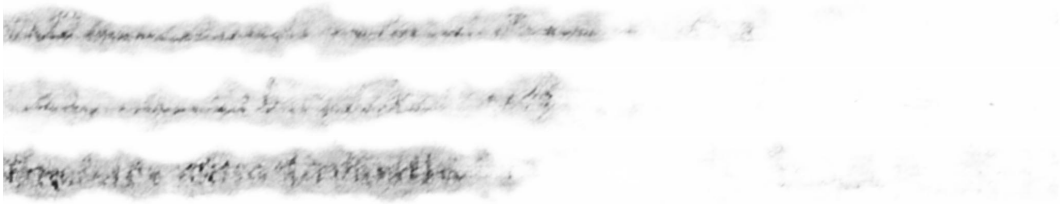


Figure 7.8 ICFHR-2018 Competition full lines generation in a simple C-VAE.

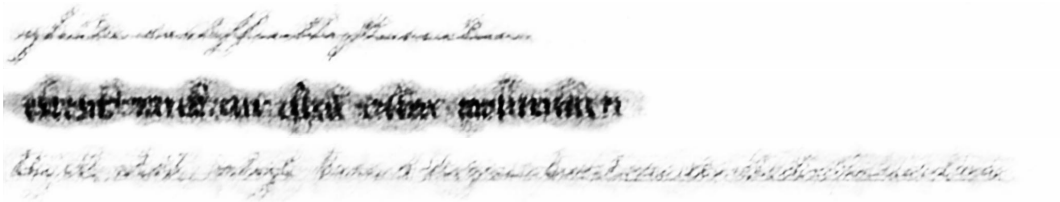


Figure 7.9 ICFHR-2018 Competition full lines generation in TSC-VAE.

7.4 Application to DA

This section of experiments uses the generated samples as DA to reduce overfitting and improve the recognizer performance. We propose data augmentation with images generated by a TSC-VAE model. This new method is compared with those in Chapter 5. In Table 7.1 we include the recognition results measured as CER and WER. As a baseline, we also include a method to generate images with a conditional Generative Adversarial Network (cGAN) proposed in [27], included

as the last column of this table.

In all the experiments shown in Table 7.1 models have been trained from scratch. Neither transfer learning nor another similar technique has been applied. We augment the database with images generated from lines of text taken from some external documents in the same language that the target database but not from the target database.

Table 7.1 Comparison of different DA strategies. Mean CER and WER (%) are evaluated over the test set when models are trained with augmented databases. The augmented images are generated by affine transformations [73], RWGD[101], TSC-VAE and cGAN. The affine transformations and RWGD columns are the same reported in Chapter 5. The DNN is trained from scratch using the number of lines indicated by 'Train size'.

	Train size	None		Affine Transf.		RWGD[101]		TSC-VAE		cGAN	
		CER	WER	CER	WER	CER	WER	CER	WER	CER	WER
RIMES	10163	4.4	10.8	2.7	10.7	**2.5**	10.4	2.6	10.7	2.8	10.7
IAM	6152	7.2	22.2	5.9	20.3	5.3	19.7	**5.2**	20.1	5.4	20.8
Washington	325	41.1	85.3	18.7	69.2	17.5	65.2	19.2	63.1	**17.3**	62.4
Parzival	350	18.2	63.0	14.1	56.4	**12.9**	53,6	13.1	55.2	13.8	54.7
ICFHR18-G	11424	12.2	43.7	10.6	40.1	**9.7**	38.6	10.5	39.4	11.1	38.5
Konzil	351	37.1	95.4	26.2	93.4	**21.5**	90.1	25.4	87.3	23.2	88.6
Schiller	238	45.4	88.2	32.4	87.6	**30.1**	85.5	31.2	88.4	32.4	85.1
Ricordi	273	57.2	93.1	36.2	91.1	35.2	90.3	**34.1**	90.8	36.2	89.5
Patzig	473	24.5	86.3	18.5	81.3	**17.1**	80.5	19.4	82.1	19.7	83.6
Schwerin	782	21.1	76.6	17.4	72.9	**16.5**	71.2	16.9	70.2	17.1	70.9

From Table 7.1, we can see that the four DA methods get similar results. Therefore, the images generated by TSC-VAE and cGAN are confidence samples from each database.

7.4.1 Including TL

By combining DA with TL as in Chapter 5 using the generated samples via generative models, we get the results shown in Table 7.2 for Washington and Parzival databases and in Table 5.3 for the 5 ICFHR2018 Competition datasets. We show that the performance of classical DA and DA via TSC-VAE images (cVDA-TL) and GAN images (cGDA-TL) are similar.

From the results reported in Table 7.2 and Table 7.3, we get the same conclusion as from the previous section, the images generated with the TSC-VAE and cGAN improve the performance in the same way that state-of-the-art DA techniques.

Table 7.2 Mean CER (%) evaluated for Washington and Parzival datasets using TL and DA with IAM database as source. The number of annotated lines used in training is included as 'Train size'.

	Train size #lines	None	TL	DA	DA-TL-DA	DA-TL	cVDA-TL	cGDA-TL
Washington	150	51.6	9.4	22.8	10.0	9.3	**9.2**	9.3
	250	46.4	7.1	20.4	7.4	**7.0**	7.1	**7.0**
	325	41.1	5.4	17.5	5.4	5.4	5.4	**5.3**
Parzival	150	21.9	5.8	15.7	6.0	**5.6**	5.7	**5.6**
	250	20.7	4.0	14.2	4.2	**3.8**	**3.8**	3.9
	350	18.2	3.3	12.9	3.4	3.3	**3.3**	3.3

Table 7.3 Mean CER (%) evaluated in ICFHR 2018 Competition Specific datasets as targets using TL and DA with ICFHR18-G as the source. The number of annotated pages used in training is included as 'Train size'.

	Training set size. # pages	None	TL	DA	DA-TL-DA	DA-TL	cVDA-TL	cGDA-TL
Konzil	0	–	15.5	–	–	14.5	**14.2**	14.3
	1 (29 lines)	48.1	10.85	37.3	14.2	**10.8**	**10.8**	10.9
	4 (116 lines)	45.3	6.54	28.7	8.0	6.51	**6.42**	6.5
	12 (351 lines)	37.1	4.37	21.5	5.0	**4.32**	4.4	4.5
Schiller	0	–	24.6	–	–	24.6	**24.5**	**24.5**
	1 (21 lines)	53.7	17.36	39.5	21.4	**17.31**	17.62	17.9
	4 (84 lines)	48.4	12.25	33.2	14.0	**12.22**	12.4	12.3
	12 (238 lines)	45.4	9.42	30.1	10.0	9.38	9.43	**9.25**
Ricordi	0	–	39.19	–	–	34.2	**33.1**	33.6
	1 (19 lines)	56.2	23.66	51.0	24.1	22.71	22.6	**22.1**
	4 (88 lines)	43.5	21.17	40.8	21.1	**21.02**	21.3	21.2
	12 (273 lines)	37.2	11.2	35.2	**10.9**	11.1	11.0	11.1
Patzig	0	–	41.5	–	–	**38.2**	38.3	37.8
	1 (38 lines)	42.5	27.91	35.3	31.4	26.7	**26.4**	26.5
	4 (156 lines)	37.6	16.4	30.5	18.3	**16.1**	16.8	16.3
	12 (473 lines)	24.5	10.6	17.1	11.2	**10.0**	10.2	10.2
Schwerin	0	–	34.5	–	–	31.3	30.4	**30.1**
	1 (68 lines)	38.4	12.15	30.2	**10.6**	10.8	10.7	10.8
	4 (264 lines)	29.3	5.73	24.3	**5.3**	5.5	5.4	5.4
	12 (782 lines)	21.1	3.5	16.5	**3.3**	3.4	**3.3**	**3.3**

7.5 Evaluation of new generated images

For the last experiment in this chapter, we have to make some consideration about the images we generated for training the models evaluated in the two last columns in Table 7.2 and Table 7.3. As in the previous section, we augment the database with images generated from lines of text taken from some external documents in

Table 7.4 CER evaluated in test set after different train sets. Real train set is composed of only 0,1,4 or 12 pages from the original documents. DA with VAE and DA with GAN train sets are composed of 12 pages, but 1 or 4 are real, the rest are synthetic.

	Original Set Size. # pages	Real Train Set	cVDA-TL	cGDA-TL
Konzil	0	15.50	-	-
	1 (29 lines)	10.85	10.79	**10.65**
	4 (116 lines)	6.54	6.12	**6.04**
	12 (351 lines)	**4.37**	-	-
Schiller	0	24.60	-	-
	1 (21 lines)	17.36	16.93	**16.88**
	4 (84 lines)	12.25	11.86	**11.81**
	12 (238 lines)	**9.42**	-	-
Ricordi	0	39.19	-	-
	1 (19 lines)	23.66	13.41	**13.38**
	4 (88 lines)	21.17	19.53	**19.35**
	12 (273 lines)	**11.20**	-	-
Patzig	0	41.50	-	-
	1 (38 lines)	27.91	25.24	**25.14**
	4 (156 lines)	16.40	14.95	**14.83**
	12 (473 lines)	**10.60**	-	-
Schwerin	0	34.53	-	-
	1 (68 lines)	12.15	10.69	**10.62**
	4 (264 lines)	5.73	5.32	**5.24**
	12 (782 lines)	**3.51**	-	-

the same language that the target database but not from the target database.

In the next experiment carried out in this chapter, we compare how much the performance of the model is affected if, instead of using the original images from the training set, we generate new images but with the exact text that the original ones with the TSC-VAE and cGAN approach. We take the labels from the training dataset and generate the images corresponding to these labels.

The results of these experiments are reported in Table 7.4. In that table, the column “Real Train Set” shows the CER when the real images of handwritten text are used for training, the number of pages in each training is indicated in the “Original Set Size” column. For the experiments in column “DA with VAE” and “DA with GAN”, we always use 12 pages for training. From these 12 pages, the number of pages indicated in the “Original Set Size” column is real. The rest of them are synthetically generated by the VAE or GAN.

From the results shown in that table, we can conclude that although the performance is worse than if we use the original images, these results are better than the results shown in the previous experiments where the training set was augmented with images conditioned to labels of a line of text which not appear in the database.

This implies that using transcribed text that belongs to the actual corpus we are transcribing could lead to further improvements and opens up an exciting approach for DA techniques, in which we generate transcribed text using some generative model trained over the historical corpus, and then we feed the text to our image synthesizer to generate the images.

7.6 Conclusions

This chapter proposes a method to generate new handwritten samples with a conditional VAE model. We have compared the generated images with the images generated for other generative models in the literature, specifically the cGAN model proposed in [27]. We have reported some samples of the images generated with different VAEs models and finally conclude that the best VAE is the proposed TSC-VAE, a combination of the two-stage VAE proposed in [25] with the conditional VAE.

We use the images generated by the TSC-VAE model proposed here to augment the historical databases we have been using in the rest of the chapters in this thesis. The goal is to propose a new DA method based on TSC-VAE generated images. We compare the performance in the recognition step when the models are trained with an augmented dataset with the DA methods presented in Chapter 5. We conclude that the performance of the VAE DA-based method is similar to the other techniques. For the sake of completeness, we also include a method based on images generated by the cGAN model presented in [27]. The results are similar to ours.

Finally, we do an experiment in which we measure the difference in the performance when the recognizer is trained with the original dataset or if we substitute the original images for images generated by the TSC-VAE and cGAN conditioned to the original labels (line of text). We conclude that the performance is better than when the text is taken from another source.

8 Conclusions

This thesis is focused on the handwriting text recognition problem of historical databases. The manual annotation for these databases is usually expensive due to the calligraphy and the language used centuries ago. Therefore, we usually have a few samples annotated for training the state-of-the-art models that solve this task, the neural networks models.

We face this problem of small historical databases motivated by the importance the contents in those archives have for the knowledge of our history. In Chapter 1 we introduce this problem and how authors have solved it in the past. In that chapter, we also overview the different HTR tasks, and we indicate that this thesis is focused on the HTR of complete lines of texts. In Chapter 2, we present the different tools that authors in the field are using for solving the problem when we have many samples in modern datasets. In Chapter 3 we describe the databases these authors usually use for evaluating the tools they propose, and we also introduce the historical databases we use through the different experiments in this thesis.

Our first contribution is in Chapter 4, where we investigate and propose the correct way of applying transfer learning methods to HTR models built of CNN, BLSTM layers, and CTC loss function. In Chapter 5, we introduce our second contribution which is the application of two techniques of data augmentation with the TL approach proposed in Chapter 4.

Our third contribution is presented in Chapter 6 where we focused on the mislabeled samples in the historical databases. This analyzes how the mislabeled samples affect the performance of the model and proposes the CLP algorithm to detect and purge these samples from the training set.

The last contribution is given in Chapter 7 where we go further in the data augmentation techniques by proposing a generative method to generate new pairs

of images-labels of handwritten text. We propose a TSC-VAE for HTR. We also compare with another method in the literature, a cGAN. We compare the data augmentation techniques in Chapter 5 with this novel generative DA.

8.1 Summary of results

Compared to the state-of-the-art in the ICFHR 2018 Competition, it can be observed that the DA-TL proposed in Chapter 5 and CLP proposed in Chapter 6 outperform all approaches within the CNN+LSTM+CTC class hence underlining the importance of the issues discussed: DA is important. However, in the source dataset, TL is to be considered, and mislabeling detection and correction are important if the dataset exhibits errors.

Besides, the CLP introduces a residual 0.01 percentage points of loss if the datasets have no errors in the labels, while the reduction is important if they have. See the results for the Ricordi corpus, where a reduction of 6.58 percentage points is achieved. The presence of errors in this database was detected by checking the number of removed lines by the CLP.

It is interesting to mention that other variations of the algorithm have been tried to improve the performance further. In this sense, we tried to evaluate the CTC loss [40] to select a threshold ε. We found it complex to deal with because it depends on several factors like the number of epochs in training or if batch normalization has been applied.

In Chapter 7 we show that generative models can help to improve the performance of HTR in small historical databases. It is shown that the images generated with TSC-VAE and cGAN augment the training set appropriately, similar to the manual designed DA techniques presented in Chapter 5. We also conclude that the HTR pipeline is an excellent way to evaluate generative models in the literature. Any improvement in some VAE models could be evaluated by training the model with handwritten text images conditioned to a particular text and using CNN + RNN + CTC recognizers to evaluate the performance.

8.2 Future lines

As future lines of research of this thesis we highlight the following:

- We conjecture that new loss approaches applied to the CTC algorithm could improve the performance [82].

- Another promising research line could be introducing TL-DA, and CLP in other DNN models, such as the based on GCN [107], that has a pretty low value for 0 pages, to further improve the CER.
- Besides, introducing LM in the proposed DA-TL and CLP approaches could also be investigated.
- We can also improve the generative method proposed in Chapter 7 by investigating better C-VAE models. Currently, generative models are a main topic of research, and we aim to investigate how the new advances in that topic could help to improve the handwriting text recognition of historical databases performance.
- The most challenging future line of research is to merge the segmentation and recognition task to build a whole document transcriptor system.

Appendix A
Bootstrapped confidence intervals

A.1 Data augmentation analysis table

In Table A.1 we augment the Table 7.1. In this document we report, apart from the mean value from 10 trainings, the bootstrapped confidence interval at 95%.

A.2 Transfer learning and data augmentation combination analysis table

In Table A.2 and Table A.3 we augment the Table 5.2 and Table 5.3 in Chapter 5. In this document we report, apart from the mean value from 10 trainings, the bootstrapped confidence interval at 95%.

A.3 Corrupted label purging algorithm results table

In Table A.4 and Table A.5 we augment the Table 6.1 and Table 6.2 . In this document we report, apart from the mean value from 10 trainings, the bootstrapped confidence interval at 95%.

A.4 Correcting label misalingment results table

In Table A.6 we augment the Table A.6 . In this document we report, apart from the mean value from 10 trainings, the bootstrapped confidence interval at 95%.

Table A.1 Mean CER and WER (%) with affine transformations [73] and RWGD[101] DA approaches evaluated for all datasets. The DNN is trained from scratch using the number of lines indicated by 'Train size'. Largest DA CER reductions are highlighted in boldface.

	Train size	None		Affine Transf.		RWGD[101]	
		CER	WER	CER	WER	CER	WER
RIMES	10163	4.4 [4.38-4.43]	10.8 [10.76-10.84]	2.7 [2.67-2.72]	10.7 [10.58-10.79]	2.5 [2.49 - 2.51]	10.4 [10.38-10.42]
IAM	6152	7.2 [7.16-7.22]	22.2 [21.0 - 23.1]	5.9 [5.86 - 5.94]	20.3 [20.15-20.45]	5.3 [5.28 - 5.32]	19.7 [19.1 - 20.3]
Washington	325	**41.1** [40.3-4.19]	85.3 [81.2 - 89.4]	18.7 [18.2 - 19.2]	69.2 [68.8 - 69.6]	**17.5** [16.9 - 18.2]	65.2 [63.9 - 66.5]
Parzival	350	18.2 [17.9-18.5]	63.0 [62.5-63.5]	14.1 [13.4 - 14.8]	56.4 [55.5-57.3]	12.9 [12.1-13.4]	53.6 [52.1 - 55.1]
ICFHR18-G	11424	12.2 [12.0 - 12.4]	43.7 [42.9 - 44.5]	10.6 [10.5-10.7]	40.1 [38.8 - 41.4]	9.7 [9.64 - 9.76]	38.6 [37.6 - 39.6]
Konzil	351	37.1 [36.5-37.7]	95.4 [92.1 - 98.4]	26.2 [25.7 - 26.7]	93.4 [91.2 - 95.7]	21.5 [20.9 - 22.1]	90.1 [85.4 - 94.8]
Schiller	238	45.4 [43.2-48.2]	88.2 [85.1-92.4]	32.4 [31.0 - 34.8]	87.6 [81.4-93.8]	30.1 [29.1-31.1]	85.5[81.2-89.8]
Ricordi	273	37.2 [34.4-40.0]	93.1 [89.4-97.2]	36.2 [35.1-37.3]	91.1 [85.3-96.4]	35.2 [32.1 - 38.3]	90.3 [84.3 - 96.3]
Patzig	473	24.5 [22.2-26.8]	86.3 [80.2-92.4]	18.5 [17.9-19.1]	81.3 [76.5-86.1]	17.1 [17.0-17.2]	80.5 [77.4-83.6]
Schwerin	782	**21.1** [20.7-21.5]	76.6 [74.3 - 78.9]	17.4 [16.2-18.6]	72.9 [69.1-76.7]	**16.5** [15.9-17.1]	71.2 [68.1-74.3]

Table A.2 Mean CER (%) evaluated for Washington and Parzival datasets using TL and DA with IAM database as source. The number of annotated lines used in training is included as 'Train size'.

	Train size #lines	None	TL	DA	DA-TL-DA	DA-TL
Washington	150	51.6 [50.3-52.9]	9.4 [9.3-9.5]	22.8 [21.9-23.7]	10.0 [9.8-10.2]	**9.3** [9.22 - 9.38]
	250	46.4 [43.2-49.6]	7.1 [7.06 - 7.14]	20.4 [19.6-21.2]	7.4 [7.35-7.45]	**7.0** [6.94 - 7.06]
	325	41.1 [39.1-43.1]	5.4 [5.31-5.49]	17.5 [17.0-18.0]	5.4 [5.34-5.46]	**5.4** [5.38-5.42]
Parzival	150	21.9 [20.7-23.1]	5.8 [5.75-5.85]	15.7 [15.0-16.4]	6.0 [5.92-6.08]	**5.6** [5.53-5.67]
	250	20.7 [19.1-22.3]	4.0 [3.94-4.06]	14.2 [14.1-14.3]	4.2 [4.15-4.25]	**3.8** [3.77-3.83]
	350	18.2 [17.5-18.9]	3.3 [3.22-3.38]	12.9 [12.2-13.6]	3.4 [3.36-3.44]	**3.3** [3.26-3.32]

Table A.3 Mean CER (%) evaluated in ICFHR 2018 Competition Specific datasets as targets using TL and DA with ICFHR18-G as source. The number of annotated pages used in the training is included as 'Train size'.

	Training set size. # pages	None	TL	DA	DA-TL-DA	DA-TL
Konzil	0	–	15.50 [15.3-15.7]	–	–	**14.5** [14.2-14.8]
	1 (29 lines)	48.1 [45.3-50.9]	10.85 [10.8-10.9]	37.3 [35.1-39.5]	14.2 [14.0 - 14.4]	**10.8** [10.78-10.82]
	4 (116 lines)	45.3 [43.2-47.4]	6.54 [6.52-6.56]	28.7 [27.2-30.2]	8.0 [7.95-8.05]	**6.51** [6.48-6.54]
	12 (351 lines)	37.1 [34.6-39.6]	4.37 [4.35-4.39]	21.5 [21.2-21.8]	5.0 [4.92-5.08]	**4.32** [4.30-4.34]
Schiller	0	–	**24.6** [24.6-24.8]	–	–	**24.6** [24.2-24.8]
	1 (21 lines)	53.7 [50.7-56.7]	17.36 [17.2-17.45]	39.5 [37.6-41.9]	21.4 [20.8-22.0]	**17.31** [16.92-17.74]
	4 (84 lines)	48.4 [46.2-51.3]	12.25 [12.12-12.38]	33.2 [30.2-36.2]	14.0 [13.94-14.06]	**12.22** [12.10-12.34]
	12 (238 lines)	45.4 [44.0-46.8]	9.42 [9.40-9.44]	30.1 [28.2-32.0]	10.0 [9.95-10.05]	**9.38** [9.35-9.41]
Ricordi	0	–	39.19 [35.2-42.6]	–	–	**34.2** [32.1-36.3]
	1 (19 lines)	56.2 [51.3-61.1]	23.66 [21.4-25.8]	51.0 [46.8-54.3]	24.1 [21.7-26.8]	**22.71** [20.1-24.82]
	4 (88 lines)	43.5 [40.2-47.0]	21.17 [20.95-21.35]	40.8 [38.5-43 .1]	21.1 [20.96-22.14]	**21.02** [20.76-22.4]
	12 (273 lines)	37.2 [36.5-39.8]	11.2 [11.15-11.25]	35.2 [32.6-38.2]	**10.9** [10.8-11.0]	11.1 [11.03-11.17]
Patzig	0	–	41.5 [38.2-44.6]	–	–	**38.2** [34.6-42.4]
	1 (38 lines)	42.5 [39.2-45.7]	27.91 [25.6-29.96]	35.3 [32.2-38.5]	31.4 [29.3-33.41]	**26.7** [25.12-27.41]
	4 (156 lines)	37.6 [35.4-39.5]	16.4 [16.3-16.6]	30.5 [28.2-32.9]	18.3 [18.1-18.5]	**16.1** [15.9-16.3]
	12 (473 lines)	24.5 [23.9-25.1]	10.6 [10.51-10.69]	17.1 [16.9-17.3]	11.2 [11.11-11.29]	**10.0** [9.95-10.05]
Schwerin	0	–	34.5 [31.5-39.5]	–	–	**31.3** [28.6-34.0]
	1 (68 lines)	38.4 [36.2-40.6]	12.15 [12.08-12.21]	30.2 [28.5-32.7]	**10.6** [10.58-10.62]	10.8 [10.74-10.86]
	4 (264 lines)	29.3 [26.6-32.1]	5.73 [5.61-5.84]	24.3 [23.2-25.4]	**5.3** [5.26-5.34]	5.5 [5.42-5.58]
	12 (782 lines)	21.1 [20.0-22.2]	3.5 [3.44-3.56]	16.5 [15.6-16.9]	**3.3** [3.28-3.32]	3.4 [3.33-3.47]

Table A.4 Mean CER (%) evaluated in Konzilsprotokolle_C, Schiller, Ricordi, Patzig and Schwerin target documents in the ICFHR2018 Competition datasets for DA-TL, DA-TL+CLP with threshold $\varepsilon = 50\%$ and DA-TL+CLP with threshold $\varepsilon = 70\%$. DA-TL was applied with both a training set of 4 pages and 12 pages. 10% of lines were corrupted in $R \neq 0$, where $R(\%)$ of the characters of these lines were randomly replaced by other random ones. The number of removed lines by the CLP algorithm are included in parentheses in the last two columns. Best achieved value in every row is in boldface.

Dataset	Train set size	R	Baseline	$\varepsilon = 50\%$	$\varepsilon = 70\%$
Konzil	4 Pages (116 lines)	0%	**7.6** [7.54-7.66]	8.5 [8.45-8.54](-31)	7.9 [7.8-8.0](-7)
		30%	8.7 [8.66-8.74]	8.3 [8.22-8.38] (-41)	**7.82** [7.79-7.85] (-14)
		50%	9.1 [9.05-9.15]	8.2 [8.13-8.27](-39)	**7.9** [7.86-7.94] (-16)
	12 Pages (351 lines)	0%	**4.6** [4.56-4.64]	5.3 [5.22-5.37](-1)	4.6 [4.56-4.64](-0)
		30%	5.3 [5.26-5.34]	**4.6** [4.58-4.62](-29)	5.0 [4.91-5.09](-25)
		50%	5.5 [5.41-5.59]	**4.8** [4.75-4.85] (-35)	5.0 [4.96-5.04] (-28)
Schiller	4 Pages (84 lines)	0 %	**13.27** [13.01-13.35]	14.72 [14.62-14.82](-12)	13.61 [13.52-13.70](-5)
		30 %	15.19 [14.96-15.23]	14.81 [14.75-14.87](-17)	**14.43** [14.34-14.52] (-10)
		50 %	15.64 [15.49-15.72]	14.96 [14.0-15.02] (-22)	**13.87** [13.82-13.92] (-12)
	12 Pages (244 lines)	0 %	**9.42** [9.38-9.46]	9.76 [9.70-9.82] (-2)	9.42 [9.37-9.47] (-0)
		30 %	11.31 [11.25-11.37]	**10.41** [10.34-10.48] (-22)	10.62 [10.59-10.65] (-22)
		50 %	12.75 [12.70-12.80]	10.61 [10.54-10.81] (-24)	**10.51** [10.46-10.56] (-25)
Ricordi	4 Pages (88 lines)	0 %	21.1 [20.2-22.0]	18.2 [17.6-18.8] (-16)	**18.2** [17.6-18.2] (-16)
		30 %	23.2 [22.8-23.6]	20.8 [19.7-21.9] (-32)	**20.5** [19.9-21.1] (-27)
		50 %	24.31 [22.3-26.29]	21.94 [20.5-23.38] (-44)	**20.81** [19.6-22.0] (-27)
	12 Pages (295 lines)	0 %	9.7 [9.66-9.74]	**9.4** [9.38-9.42] (-38)	9.4 [9.38-9.42] (-38)
		30 %	10.8 [10.77-10.83]	**9.23** [9.20-9.26] (-41)	9.49 [9.42-9.56] (-38)
		50 %	10.47 [10.40-10.54]	**9.53** [9.48-9.58] (-52)	9.75 [9.69-9.81] (-44)
Patzig	4 Pages (156 lines)	0 %	**18.32** [18.28-18.36]	18.93 [18.86-19.0] (-7)	18.32 [18.28-18.36] (-0)
		30 %	21.41 [20.13-22.72]	21.6 [20.6-22.6] (-27)	**21.1** [20.4-21.9] (-18)
		50 %	21.84 [21.12-22.24]	22.12 [21.2-22.7] (-27)	**21.31** [20.84-22.1](-18)
	12 Pages (473 lines)	0 %	**11.5** [11.42-11.58]	11.96 [11.90-12.02] (-15)	11.54 [11.5-11.59] (-4)
		30 %	12.28 [12.21-12.35]	12.23 [12.18-12.28] (-61)	**11.98** [11.95-12.01] (-52)
		50 %	12.8 [12.7-12.9]	12.67 [12.61-12.73] (-63)	**12.35 [12.28-12.42] (-54)**
Schwerin	4 Pages (264 lines)	0 %	5.3 [5.28-5.32]	5.3 [5.28-5.32] (-0)	**5.3** [5.28-5.32] (-0)
		30 %	5.36 [5.32-5.40]	**5.31** [5.25-5.37] (-14)	5.36 [5.32-5.40] (-0)
		50 %	5.39 [5.35-5.44]	**5.32** [5.25-5.39] (-26)	5.33 [5.27-5.39]] (-12)
	12 Pages (782 lines)	0 %	3.3 [3.28-3.32]	3.3 [3.28-3.32] (0)	**3.3** [3.28-3.32] (0)
		30 %	3.36 [3.32-3.4]	**3.31** [3.28-3.34] (-14)	3.36 [3.33-3.39] (-0)
		50 %	3.53 [3.50-3.56]	**3.34** [3.31-3.37] (-75)	3.39 [3.36-3.42] (-22)

Table A.5 Mean CER (%) evaluated in Washington and Parzival documents for DA-TL, CLP with threshold $\varepsilon = 50\%$ and CLP with threshold $\varepsilon = 70\%$. DA-TL was applied with the IAM dataset as source and using 150 and 325 lines from the target. 10% of lines were corrupted for $R \neq 0$, where $R(\%)$ of the characters of these lines were randomly replaced by other random ones. The number of removed lines by the CLP algorithm are included in parentheses in the last two columns.

Dataset	Train set size	R	Baseline	$\varepsilon = 50\%$	$\varepsilon = 70\%$
Washington	150 lines	0 %	9.4 [9.36-9.44]	9.5 [9.45-9.55] (-6)	**9.4** [9.36-9.44] (-2)
		30 %	11.3 [11.22-11.38]	10.6 [10.54-10.66] (-20)	**10.5** [10.41-10.59] (-14)
		50 %	11.5 [11.43-11.57]	11.1 [11.01-11.19] (-31)	**10.87** [10.85-10.89] (-19)
	325 lines	0 %	5.3 [5.21-5.39]	**5.3** [5.21-5.39] (-2)	5.3 [5.21-5.39] (-0)
		30 %	6.1 [6.05-6.15]	**5.7** [5.63-5.77] (-26)	6.1 [6.05-6.15] (-0)
		50 %	6.3 [6.26-6.34]	**5.8** [5.75-5.85] (-34)	6.3 [6.26-6.34] (-0)
Parzival	150 lines	0 %	5.8 [5.78-5.82]	5.8 [5.78-5.82] (-0)	**5.8** [5.78-5.82] (-0)
		30 %	6.4 [6.33-6.47]	**6.0** [5.98 - 6.02] (-15)	6.2 [6.16-6.24] (-2)
		50 %	6.6 [6.52-6.68]	6.2 [6.18-6.22] (-20)	**6.1** [6.06-6.14] (-14)
	325 lines	0 %	3.3 [3.27-3.33]	**3.3** [3.27-3.33] (-0)	3.3 [3.27-3.33] (-0)
		30 %	3.5 [3.47-3.53]	**3.5** [3.47-3.53] (-0)	3.5 [3.47-3.53] (-0)
		50 %	3.5 [3.47-3.53]	**3.4** [3.37-3.43] (-35)	3.5 [3.47-3.53](-0)

Table A.6 Comparison between the CLP algorithm with line removal and the CLP plus alignment of the GT after detection. The CER (%) is evaluated for the Ricordi document with a training set of size 4 pages (88 lines) and 12 pages (295 lines).

Train set size	Method	Baseline	$\varepsilon = 50\%$	$\varepsilon = 70\%$
4 pages (88 lines)	CLP	21.1 [20.5-21.9]	**18.2** [17.6-18.8]	**18.2** [17.6-18.8]
	CLP + alignment	21.1 [20.5-21.9]	**17.4** [17.25-17.55]	**17.4** [17.25-17.55]
12 pages (295 lines)	CLP	9.7 [9.66-9.74]	**9.4** [9.38-9.42]	9.4 [9.38-9.42]
	CLP + alignment	9.7 [9.66-9.74]	**8.9** [8.82-8.98]	8.9 [8.82-8.98]

Appendix B
Other publications

The author has been involved in the following publications that are not part of the thesis:

- I. Santos, J.J. Murillo Fuentes, J.C. Aradillas, E.M. Arias de Reyna Domínguez, "Channel equalization with expectation propagation at smoothing level". En: IEEE Transactions on Communications. 2020. Vol. 68. Núm. 5. Pag. 2740-2747. 10.1109/Tcomm.2020.2975624
- J. J. Murillo-Fuentes, I. Santos, J. C. Aradillas and M. Sánchez-Fernández, "A Low-Complexity Double EP-Based Detector for Iterative Detection and Decoding in MIMO," in IEEE Transactions on Communications, vol. 69, no. 3, pp. 1538-1547, March 2021, DOI: 10.1109/TCOMM.2020.3043771.
- J. J. Murillo-Fuentes, J.F. Payán Somet and J. C. Aradillas "Laboratorio de Comunicaciones Digitales en Python," Editorial Universidad de Sevilla, 2021.

List of Figures

List of Tables

Bibliography

[1] V. Frinken A. Fischer, A. Keller and H. Bunke, *Lexicon-Free Handwritten Word Spotting Using Character HMMs*, Pattern Recognition Letters **33** (2012), no. 7, 934–942.

[2] Emre Aksan, Fabrizio Pece, and Otmar Hilliges, *DeepWriting: Making Digital Ink Editable via Deep Generative Modeling*, (2018).

[3] Michele Alberti, Lars Vögtlin, Vinaychandran Pondenkandath, Mathias Seuret, Rolf Ingold, and Marcus Liwicki, *Labeling, cutting, grouping: an efficient text line segmentation method for medieval manuscripts*, vol. abs/1906.11894, 2019, pp. 1200–1206.

[4] Eloi Alonso, Bastien Moysset, and Ronaldo Messina, *Adversarial Generation of Handwritten Text Images Conditioned on Sequences*, 2019.

[5] Vicenta Cortés Alonso, *La investigación en el Archivo Histórico Nacional (1977-1990)*, Boletín de la ANABAD (1996), no. 46, 341–358.

[6] J C Aradillas, J J Murillo-Fuentes, and P M. Olmos, *Improving offline HTR in small datasets by purging unreliable labels*, 17th Int. Conf. on Frontiers in Handwriting Recognition (ICFHR) (Dortmund), sep 2020, pp. 25–30.

[7] José Carlos Aradillas, Juan José Murillo-Fuentes, and Pablo M Olmos, *Boosting Offline Handwritten Text Recognition in Historical Documents With Few Labeled Lines*, IEEE Access **9** (2021), 76674–76688.

[8] J C Aradillas Jaramillo, J J Murillo-Fuentes, and P M. Olmos, *Boosting Handwriting Text Recognition in Small Databases with Transfer Learning*,

16th Int. Conf. on Frontiers in Handwriting Recognition (ICFHR), aug 2018, pp. 429–434.

[9] Micheal Baechler, Jean-Luc Bloechle, and Rolf Ingold, *Semi-automatic Annotation Tool for Medieval Manuscripts*, 2010 12th International Conference on Frontiers in Handwriting Recognition, 2010, pp. 182–187.

[10] Micheal Baechler and Rolf Ingold, *Multi Resolution Layout Analysis of Medieval Manuscripts Using Dynamic MLP*, 2011 International Conference on Document Analysis and Recognition, 2011, pp. 1185–1189.

[11] Anne-Laure Bianne-Bernard, Fares Menasri, Laurence Likforman-Sulem, Chafic Mokbel, and Christopher Kermorvant, *Variable length and context-dependent HMM letter form models for Arabic handwritten word recognition*, Document Recognition and Retrieval XIX **8297** (2012), 829708.

[12] Galal M BinMakhashen and Sabri A Mahmoud, *Document Layout Analysis: A Comprehensive Survey*, ACM Computing Surveys (CSUR) **52** (2019), 1–36.

[13] T Bluche, H Ney, and C Kermorvant, *Feature Extraction with Convolutional Neural Networks for Handwritten Word Recognition*, 12th Int. Conf. on Document Analysis and Recognition, 2013, pp. 285–289.

[14] Theodore Bluche, *Joint Line Segmentation and Transcription for End-to-End Handwritten Paragraph Recognition*, Advances in Neural Information Processing Systems 29, Curran Associates, Inc., 2016, pp. 838–846.

[15] Théodore Bluche, Sebastien Hamel, Christopher Kermorvant, Joan Puigcerver, Dominique Stutzmann, Alejandro H Toselli, and Enrique Vidal, *Preparatory KWS Experiments for Large-Scale Indexing of a Vast Medieval Manuscript Collection in the HIMANIS Project*, 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), vol. 01, 2017, pp. 311–316.

[16] Théodore Bluche, Jérôme Louradour, and Ronaldo Messina, *Scan, Attend and Read: End-to-End Handwritten Paragraph Recognition with MDLSTM Attention*, (2016).

[17] Theodore Bluche, Jérôme Louradour, and Ronaldo Messina, *Scan, Attend and Read: End-to-End Handwritten Paragraph Recognition with MDLSTM Attention*, 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), jul 2017, pp. 1050–1055.

[18] John S Bridle, *Probabilistic Interpretation of Feedforward Classification Network Outputs, with Relationships to Statistical Pattern Recognition*, Neurocomputing (Berlin, Heidelberg) (Françoise Fogelman Soulié and Jeanny Hérault, eds.), Springer Berlin Heidelberg, 1990, pp. 227–236.

[19] Dayvid Castro, Byron Bezerra, and Meuser Valenca, *Boosting the Deep Multidimensional Long-Short-Term Memory Network for Handwritten Recognition Systems*, 2018, pp. 127–132.

[20] E Chammas, C Mokbel, and L Likforman-Sulem, *Handwriting Recognition of Historical Documents with Few Labeled Data*, 13th IAPR Int. Workshop on Document Analysis Systems (DAS), apr 2018, pp. 43–48.

[21] Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman, *Return of the Devil in the Details: Delving Deep into Convolutional Nets*, BMVC - Proc. of the British Machine Vision Conf. (2014).

[22] Zhanzhan Cheng, Fan Bai, Yunlu Xu, Gang Zheng, Shiliang Pu, and Shuigeng Zhou, *Focusing Attention: Towards Accurate Text Recognition in Natural Images*, 2017 IEEE International Conference on Computer Vision (ICCV), 2017, pp. 5086–5094.

[23] Denis Coquenet, Clement Chatelain, and Thierry Paquet, *Recurrence-free unconstrained handwritten text recognition using gated fully convolutional network*, 2020.

[24] Andrea Corbelli, Lorenzo Baraldi, Costantino Grana, and Rita Cucchiara, *Historical document digitization through layout analysis and deep content classification*, 2016 23rd International Conference on Pattern Recognition (ICPR), 2016, pp. 4077–4082.

[25] Bin Dai and David Wipf, *Diagnosing and Enhancing VAE Models*, (2019).

[26] Nurseitov Daniyar, Bostanbekov Kairat, Kanatov Maksat, and Alimova Anel, *Classification of handwritten names of cities using various deep learning models*, 15th Int. Conf. on Electronics, Computer and Computation (ICECCO), 2019, pp. 1–4.

[27] Brian Davis, Chris Tensmeyer, Brian Price, Curtis Wigington, Bryan Morse, and Rajiv Jain, *Text and Style Conditioned GAN for Generation of Offline Handwriting Lines*, (2020).

[28] Ministerio de Educación Ciencia y Cultura, *Portal de Archivos Españoles (PARES)*, *http://pares.mcu.es/*.

[29] _____, *Estadística de Archivos Estatales gestionados por el Ministerio de Educación, Cultura y Deporte y por el Ministerio de Defensa de 2014*, http://www.mecd.gob.es/cultura-mecd/areas-cultura/archivos/in/estadisticas.html, 2014.

[30] Arthur Flor de Sousa Neto, Byron Leite Dantas Bezerra, Alejandro Héctor Toselli, and Estanislau Baptista Lima, *HTR-Flor++: A Handwritten Text Recognition System Based on a Pipeline of Optical and Language Models*, Proc. of the ACM Sym. on Document Engineering (New York, USA), 2020.

[31] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell, *DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition*, Proc. of the 31st Int. Conf. on Machine Learning, vol. 32, PMLR, 2014, pp. 647–655.

[32] Vincent Dumoulin and Francesco Visin, *A guide to convolution arithmetic for deep learning*, Tech. report, 2016.

[33] Bradley Efron, *Better Bootstrap Confidence Intervals*, Journal of the American Statistical Association **82** (1987), no. 397, 171–185.

[34] Andreas Fischer, Andreas Keller, Volkmar Frinken, and Horst Bunke, *Lexicon-free handwritten word spotting using character HMMs*, Pattern Recognit. Letters **33** (2012), no. 7, 934–942.

[35] Sharon Fogel, Hadar Averbuch-Elor, Sarel Cohen, Shai Mazor, and Roee Litman, *ScrabbleGAN: Semi-Supervised Varying Length Handwritten Text Generation*, (2020).

[36] Yuting Gao, Zheng Huang, Yuchen Dai, Cheng Xu, Kai Chen, and Jie Guo, *DSAN: Double Supervised Network with Attention Mechanism for Scene Text Recognition*, 2019 IEEE Visual Communications and Image Processing (VCIP), 2019, pp. 1–4.

[37] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, *Deep Learning*, MIT Press, 2016.

[38] Adeline Granet, Emmanuel Morin, Harold Mouchère, Solen Quiniou, and Christian Viard-Gaudin, *Transfer Learning for Handwriting Recognition on Historical Documents*, Int. Conf. on Pattern Recognition Applications and Methods (Madeira, Portugal), 2018.

[39] Alex Graves, *Generating Sequences With Recurrent Neural Networks*, (2013).

[40] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber, *Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks*, 2006, pp. 369–376.

[41] Alex Graves and Jürgen Schmidhuber, *Offline Handwriting Recognition with Multidimensional Recurrent Neural Networks*, Advances in Neural Information Processing Systems 21, 2009, pp. 545–552.

[42] E Grosicki and H E Abed, *ICDAR 2009 Handwriting Recognition Competition*, 10th Int. Conf. on Document Analysis and Recognition, 2009, pp. 1398–1402.

[43] Mingyang Guan, H Ding, Kai Chen, and Qiang Huo, *Improving Handwritten OCR with Augmented Text Line Images Synthesized from Online Handwriting Samples by Style-Conditioned GAN*, 17th Int. Conf. on Frontiers in Handwriting Recognition (ICFHR) (2020), 151–156.

[44] Simon Günter and Horst Bunke, *HMM-based handwritten word recognition: on the optimization of the number of states, training iterations and Gaussian components*, Pattern Recognition **37** (2004), no. 10, 2069–2079.

[45] Luiz G Hafemann, Robert Sabourin, and Luiz S Oliveira, *Learning features for offline handwritten signature verification using deep convolutional neural networks*, Pattern Recognition **70** (2017), 163–176.

[46] Sheng He and Lambert Schomaker, *DeepOtsu: Document Enhancement and Binarization using Iterative Deep Learning*, Pattern Recognit. **91** (2019), 379–390.

[47] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter, *GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium*, 2017.

[48] Sepp Hochreiter and Jürgen Schmidhuber, *Long Short-Term Memory*, Neural Comput. **9** (1997), no. 8, 1735–1780.

[49] Mariko Hosoe, Tomoki Yamada, Kunihito Kato, and Kazuhiko Yamamoto, *Offline Text-Independent Writer Identification Based on Writer-Independent Model using Conditional AutoEncoder*, 2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR), 2018, pp. 441–446.

[50] Bo Ji and Tianyi Chen, *Generative Adversarial Network for Handwritten Text*, (2019).

[51] Panagiotis Kaddas and Basilis Gatos, *A deep convolutional encoder-decoder network for page segmentation of historical handwritten documents into text zones*, 2018.

[52] A Kaltenmeier, T Caesar, J M Gloger, and E Mandler, *Sophisticated topology of hidden Markov models for cursive script recognition*, Proceedings of 2nd International Conference on Document Analysis and Recognition (ICDAR '93), 1993, pp. 139–142.

[53] Lei Kang, Pau Riba, Yaxing Wang, Marçal Rusiñol, Alicia Fornés, and Mauricio Villegas, *GANwriting: Content-Conditioned Generation of Styled Handwritten Word Images*, (2020).

[54] Valentin Khrulkov and Ivan Oseledets, *Geometry Score: A Method For Comparing Generative Adversarial Networks*, (2018).

[55] Praveen Krishnan and C V Jawahar, *Matching Handwritten Document Images*, Computer Vision – ECCV (Cham) (Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, eds.), Springer International Publishing, 2016, pp. 766–782.

[56] Eva Lang, Joan Puigcerver, Alejandro Héctor Toselli, and Enrique Vidal, *Probabilistic Indexing and Search for Information Extraction on Handwritten German Parish Records*, 2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR), 2018, pp. 44–49.

[57] Joonho Lee, Hideaki Hayashi, Wataru Ohyama, and Seiichi Uchida, *Page Segmentation using a Convolutional Neural Network with Trainable Co-Occurrence Features*, 2019 International Conference on Document Analysis and Recognition (ICDAR), 2019, pp. 1023–1028.

[58] Vladimir I Levenshtein, *Binary codes capable of correcting deletions, insertions, and reversals*, Soviet physics. Doklady **10** (1965), 707–710.

[59] E Levin and R Pieraccini, *Dynamic planar warping for optical character recognition*, [Proceedings] ICASSP-92: 1992 IEEE International Conference on Acoustics, Speech, and Signal Processing, vol. 3, 1992, pp. 149–152 vol.3.

[60] Li Liu, Linlin Huang, Fei Yin, and Youbin Chen, *Offline signature verification using a region based deep metric learning network*, Pattern Recognition **118** (2021), 108009.

[61] M Long, Y Cao, Z Cao, J Wang, and M I Jordan, *Transferable Representation Learning with Deep Adaptation Networks*, IEEE Trans. on Pattern Analysis and Machine Intelligence **41** (2019), no. 12, 3071–3085.

[62] U.-V. Marti and H Bunke, *The IAM-database: an English sentence database for offline handwriting recognition*, Int. Journal on Document Analysis and Recognition **5** (2002), no. 1, 39–46.

[63] M Mayr, M Stumpf, A Nicolaou, Mathias Seuret, Andreas Maier, and Vincent Christlein, *Spatio-Temporal Handwriting Imitation*, ArXiv **abs/2003.1** (2020).

[64] Bastien Moysset and Ronaldo Messina, *Are 2D-LSTM really dead for offline text recognition?*, Int. Journal on Document Analysis and Recognition, vol. 22, nov 2019, pp. 193–208.

[65] Fatemeh Naiemi, Vahid Ghods, and Hassan Khalesi, *A novel pipeline framework for multi oriented scene text image detection and recognition*, Expert Systems with Applications **170** (2021), 114549.

[66] Reiichiro Nakano, *Neural Painters: A learned differentiable constraint for generating brushstroke paintings*, (2019).

[67] H Nisa, J A Thom, V Ciesielski, and R Tennakoon, *A deep learning approach to handwritten text recognition in the presence of struck-out text*, 2019 Int. Conf. on Image and Vision Computing New Zealand (IVCNZ), dec 2019, pp. 1–6.

[68] Sofia Ares Oliveira, Benoit Seguin, and Frederic Kaplan, *Dhsegment: A generic deep-learning approach for document segmentation*, 2018.

[69] Jose Oncina and Marc Sebban, *Learning stochastic edit distance: Application in handwritten character recognition*, Pattern Recognit. **39** (2006), no. 9, 1575–1587.

[70] Diederik P Kingma and Max Welling, *Auto-Encoding Variational Bayes*, 2014.

[71] S J Pan and Q Yang, *A Survey on Transfer Learning*, IEEE Trans. on Knowledge and Data Engineering **22** (2010), no. 10, 1345–1359.

[72] V Pham, T Bluche, C Kermorvant, and J Louradour, *Dropout Improves Recurrent Neural Networks for Handwriting Recognition*, 14th Int. Conf. on Frontiers in Handwriting Recognition, sep 2014, pp. 285–290.

[73] A Poznanski and L Wolf, *CNN-N-Gram for HandwritingWord Recognition*, IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), jun 2016, pp. 2305–2314.

[74] Joan Puigcerver, *Are Multidimensional Recurrent Layers Really Necessary for Handwritten Text Recognition?*, 14th IAPR Int. Conf. on Document Analysis and Recognition (ICDAR), IEEE, nov 2017, pp. 67–72.

[75] A S Razavian, H Azizpour, J Sullivan, and S Carlsson, *CNN Features Off-the-Shelf: An Astounding Baseline for Recognition*, IEEE Conf. on Computer Vision and Pattern Recognition Workshops, 2014, pp. 512–519.

[76] Frank Rosenblatt, *The perceptron: a probabilistic model for information storage and organization in the brain.*, Psychological review **65 6** (1958), 386–408.

[77] Victoria Ruiz, Ismael Linares, Angel Sanchez, and Jose F Velez, *Off-line handwritten signature verification using compositional synthetic generation of signatures and Siamese Neural Networks*, Neurocomputing **374** (2020), 30–41.

[78] Eugen Rusakov, Leonard Rothacker, Hyunho Mo, and Gernot A Fink, *A Probabilistic Retrieval Model for Word Spotting Based on Direct Attribute Prediction*, 2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR), 2018, pp. 38–43.

[79] Raid Saabni, Abedelkadir Asi, and Jihad El-Sana, *Text line extraction for historical document images*, Pattern Recognition Letters **35** (2014), no. 1, 23–33.

[80] A B Salah, J p. Moreux, N Ragot, and T Paquet, *OCR performance prediction using cross-OCR alignment*, 13th Int. Conf. on Document Analysis and Recognition (ICDAR), aug 2015, pp. 556–560.

[81] Joan Andreu Sánchez, Verónica Romero, Alejandro H Toselli, Mauricio Villegas, and Enrique Vidal, *A set of benchmarks for Handwritten Text Recognition on historical documents*, Pattern Recognit. **94** (2019), 122–134.

[82] Harald Scheidl, Stefan Fiel, and Robert Sablatnig, *Word Beam Search: A Connectionist Temporal Classification Decoding Algorithm*, 2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR), 2018, pp. 253–258.

[83] Patrick Schone, Christian Hargraves, Jon Morrey, Rachael Day, and Mindy Jacox, *Neural text line segmentation of multilingual print and handwriting with recognition-based evaluation*, 2018.

[84] Nicolás Serrano, Francisco Castro, and Alfons Juan-Císcar, *The RODRIGO Database*, LREC, 2010.

[85] X Shen and R Messina, *A Method of Synthesizing Handwritten Chinese Images for Data Augmentation*, 15th Int. Conf. on Frontiers in Handwriting Recognition (ICFHR), oct 2016, pp. 114–119.

[86] B Shi, X Bai, and C Yao, *An End-to-End Trainable Neural Network for Image-Based Sequence Recognition and Its Application to Scene Text Recognition*, IEEE Trans. on Pattern Analysis and Machine Intelligence **39** (2017), no. 11, 2298–2304.

[87] P Y Simard, D Steinkraus, and J C Platt, *Best practices for convolutional neural networks applied to visual document analysis*, 7th Int. Conf. on Document Analysis and Recognition. Proc., aug 2003, pp. 958–963.

[88] Kihyuk Sohn, Honglak Lee, and Xinchen Yan, *Learning Structured Output Representation using Deep Conditional Generative Models*, Advances in Neural Information Processing Systems (C Cortes, N Lawrence, D Lee, M Sugiyama, and R Garnett, eds.), vol. 28, Curran Associates, Inc., 2015.

[89] M A Souibgui and Y Kessentini, *DE-GAN: A Conditional Generative Adversarial Network for Document Enhancement*, IEEE Transactions on Pattern Analysis and Machine Intelligence (2020), 1.

[90] Y Soullard, W Swaileh, P Tranouez, T Paquet, and C Chatelain, *Improving Text Recognition using Optical and Language Model Writer Adaptation*, Int. Conf. on Document Analysis and Recognition (ICDAR), 2019, pp. 1175–1180.

[91] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, *Dropout: A Simple Way to Prevent Neural Networks from Overfitting*, Journal of Machine Learning Research **15** (2014), 1929–1958.

[92] T Strauß, G Leifert, R Labahn, T Hodel, and G Mühlberger, *ICFHR2018 Competition on Automated Text Recognition on a READ Dataset*, 16th Int. Conf. on Frontiers in Handwriting Recognition (ICFHR), aug 2018, pp. 477–482.

[93] Sebastian Sudholt and Gernot A Fink, *Evaluating Word String Embeddings and Loss Functions for CNN-Based Word Spotting*, 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), vol. 01, 2017, pp. 493–498.

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

[95] W Swaileh, T Paquet, Y Soullard, and P Tranouez, *Handwriting Recognition with Multigrams*, 14th IAPR Int. Conf. on Document Analysis and Recognition (ICDAR), vol. 01, 2017, pp. 137–142.

[96] Alejandro Héctor Toselli, Verónica Romero, Joan Andreu Sánchez, and Enrique Vidal, *Making Two Vast Historical Manuscript Collections Searchable and Extracting Meaningful Textual Features Through Large-Scale Probabilistic Indexing*, 2019 International Conference on Document Analysis and Recognition (ICDAR), 2019, pp. 108–113.

[97] Jorge Ugarte, *Deep learning: segmentation of documents from the archivo general de indias with dhsegment and neurallinesegmenter*, 2019.

[98] Quang Nhat Vo, Soo Hyung Kim, Hyung Jeong Yang, and Guee Sang Lee, *Text line segmentation using a fully convolutional network in handwritten document images*, IET Image Processing (2017).

[99] Paul Voigtlaender, Patrick Doetsch, and Hermann Ney, *Handwriting Recognition with Large Multidimensional Long Short-Term Memory Recurrent Neural Networks*, 15th Int. Conf. on Frontiers in Handwriting Recognition (ICFHR), IEEE, oct 2016, pp. 228–233.

[100] Hao Wei, Micheal Baechler, Fouad Slimane, and Rolf Ingold, *Evaluation of SVM, MLP and GMM Classifiers for Layout Analysis of Historical Documents*, 2013 12th International Conference on Document Analysis and Recognition, 2013, pp. 1220–1224.

[101] C Wigington, S Stewart, B Davis, B Barrett, B Price, and S Cohen, *Data Augmentation for Recognition of Handwritten Words and Lines Using a CNN-LSTM Network*, 14th IAPR Int. Conf. on Document Analysis and Recognition (ICDAR), vol. 01, nov 2017, pp. 639–645.

[102] F Wolf, K Brandenbusch, and G A Fink, *Improving Handwritten Word Synthesis for Annotation-free Word Spotting*, 17th Int. Conf. on Frontiers in Handwriting Recognition (ICFHR), 2020, pp. 61–66.

[103] Yi-Chao Wu, Fei Yin, and Cheng-Lin Liu, *Improving handwritten Chinese text recognition using neural network language models and convolutional neural network shape models*, Pattern Recognit. **65** (2017), 251–264.

[104] Yue Xu, Wenhao He, Fei Yin, and Cheng-Lin Liu, *Page Segmentation for Historical Handwritten Documents Using Fully Convolutional Networks*, 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), vol. 01, 2017, pp. 541–546.

[105] Larry Yaeger, Richard Lyon, and Brandyn Webb, *Effective Training of a Neural Network Character Classifier for Word Recognition*, Proc. of the 9th Int. Conf. on Neural Information Processing Systems (Cambridge, MA, USA), NIPS'96, MIT Press, 1996, pp. 807–813.

[106] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson, *How transferable are features in deep neural networks?*, Advances in Neural Information Processing Systems, vol. 27, 2014.

[107] Mohamed Yousef, Khaled F Hussain, and Usama S Mohammed, *Accurate, data-efficient, unconstrained text recognition with convolutional neural networks*, Pattern Recognit. **108** (2020), 107482.

[108] M R Yousefi, M R Soheili, T M Breuel, E Kabir, and D Stricker, *Binarization-free OCR for historical documents using LSTM networks*, 13th Int. Conf. on Document Analysis and Recognition (ICDAR), 2015, pp. 1121–1125.

[109] L Zhu, Z Huang, Z Li, L Xie, and H T Shen, *Exploring Auxiliary Context: Discrete Semantic Transfer Hashing for Scalable Image Retrieval*, IEEE Trans. on Neural Networks and Learning Systems **29** (2018), no. 11, 5264–5276.

[110] M Zimmermann and H Bunke, *Hidden Markov model length optimization for handwriting recognition systems*, Proceedings Eighth International Workshop on Frontiers in Handwriting Recognition, 2002, pp. 369–374.

Acronyms

2D-LSTM Two Dimensional LSTM. 21, 33, 35

APE Archives Portal Europe. 1

BLSTM Bidirectional Long Short Term Memory. 22, 33–37, 39, 45, 85, 101

BRNN Bidirectional Recurrent Neural Network. 17

C-VAE Conditional Variational Auto Encoder. XI, 7, 72, 76–78, 87

CER Character Error Rate. 13, 27, 31, 35–53, 56–59, 61–65, 67–69, 72, 82, 87, 98, 99, 101–103

cGAN conditional Generative Adversarial Network. 79–82, 86, 103

CLP Corrupted Label Purging. 8, 59, 62–65, 67–69, 71, 85–87, 99, 103

CNN Convolutional Neural Network. 4, 6, 16, 21, 22, 32, 34–46, 65, 85, 86, 98

CRNN Convolutional Recurrent Neural Network. 32, 38, 41, 45, 98

CTC Connectionist Temporal Classification. 6, 13, 19–21, 35, 39, 45, 66, 73, 85, 86, 97

DA Data Augmentation. XI, 8, 22, 23, 44, 47–57, 59, 61–65, 67, 68, 78–82, 86, 87, 98, 99, 102–104

DL Deep Learning. 47

DNN Deep Neural Network. 7, 22, 32, 33, 44, 45, 47–50, 52, 53, 55, 57, 64, 65, 87, 98, 102

FC Fully Connected. 16, 35, 39

FCN Fully Connected Network. 4, 22

FID Fréchet Inception Distance. 72, 75

GAN Generative Adversarial Networks. 23, 72, 75, 79, 81, 82, 104

GCN Gated Convolutional Network. 65, 87

GS Geometry Score. 72

GT Ground Truth. 12, 23, 25, 47, 58

HMM Hidden Markov Models. 13, 14

HTR Handwriting Text Recognition. V, VII, 7–9, 11, 12, 14, 19, 21–23, 26, 31–34, 37, 45, 48, 52, 55, 71, 75, 85, 86

IAM Institut für Informatik und Angewandte Mathematik. 23, 25, 29, 35–37, 42–44, 48–51, 56, 58, 64, 68, 72, 101–103

ICDAR International Conference on Document Analysis and Recognition. 26

ICFHR International Conference on Frontiers in Handwriting Recognition. 27, 29, 32, 35, 44, 45, 49–53, 55–57, 59, 61–65, 67, 69, 86, 98, 99, 102, 103

LM Language Model. 48, 87

LSTM Long Short Term Memory. 6, 18, 19, 21, 33, 34, 45, 53, 65, 69, 86, 102, 103

MDRNN Multi Dimensional Recurrent Neural Network. 17

MLP Multi Layer Perceptron. 14–16

NN Neural Networks. V, VII, 13, 14, 16, 31, 35

PNG Portable Network Graphics. 25

PRHLT Pattern Recognition and Human Language Technology. 45

RIMES Reconnaissance et Indexation de données Manuscrites et de fac similÉS. 26, 29, 36, 48–50, 72, 101

RNN Recurrent Neural Network. 6, 13, 16–21, 33, 34, 86, 97

RWGD Random Warp Grid Distortion. 48–51, 71, 98, 102

TL Transfer Learning. X, XI, 8, 22, 23, 31–33, 35, 37, 38, 40, 41, 43–47, 50–55, 65, 71, 79, 80, 85, 86, 102, 103

TS-VAE Two-Stage Variational Auto Encoder. XI, 74, 76

TSC-VAE Two-Stage Conditional Variational Auto Encoder. XI, 74–82, 86, 99, 103

VAE Variational Auto Encoder. 8, 72, 73, 82, 86

WER Word Error Rate. 13, 35–37, 44, 47, 50, 101, 102