UNIVERSITAT POLITÈCNICA DE VALÈNCIA
DEPARTAMENT DE SISTEMES INFORMÀTICS I COMPUTACIÓ

Arabic Text Recognition and Machine Translation

Thesis
presented by Ihab Al-Khoury
supervised by Dr. Alfons Juan Císcar
and Dr. Jesús Andrés Ferrer

June 2015

Arabic Text Recognition and Machine Translation

Ihab Al-Khoury

Thesis performed under the supervision of doctors
Alfons Juan Císcar and Jesús Andrés Ferrer
and presented at the Universitat Politècnica de València
in partial fulfilment of the requirements for the degree of

Doctor en Informàtica

València, June 2015

The research leading to the results of this thesis has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement no 287755. Also supported by the Spanish Government (Plan E, iTrans2 TIN2009-14511 and AECID 2008-2012 grant).

Acknowledgments

I would never have been able to finish my dissertation without the guidance of my advisers, help from colleagues, and support from my family and friends. I dedicate the last 6 years of effort and hardworking to every person who has encouraged me, advised me, and stood by me during the difficult moments.

To my advisers of UPV...

I would like to express my deepest gratitude to my advisers, Dr. Alfons Juan and Dr. Jesús Andrés-Ferrer, for their excellent guidance, caring, patience, and providing me with an excellent atmosphere for doing research. A Special thanks to Dr. Alfons who patiently corrected my writing and financially supported my research during 2012 under the iTrans2 project.

To my advisers and professors of BU...

I would also like to thank my professors from Bethlehem University, especially Prof. Saleem, Dr. Suhail, Dr. Nader, and Dr. Muna, for guiding my Bachelor studies and my undergraduate project, and helping me to develop my background in computer science and logic.

To evaluators and committee members...

Special thanks go to all professors who were willing to participate in the evaluation process of this dissertation as well as in the defense committee.

To my lab and university colleagues...

I would like to thank my lab colleagues who as good friends, were always willing to help and give their best suggestions. It would have been a lonely lab without them. Many thanks to Adrià, NiCo, Miguel, Joan-Albert, JuanDa, and Isaías. I am equally grateful to my colleagues from other labs: Emilio, JuanFer, David, and others. My research and achievements would not have been possible without their help.

To family and friends...

I would like to thank from all of my heart my parents and my brothers. They were always supporting me and encouraging me with their best wishes. Also, special thanks to my friends: Mohammed(s), Rima, Iman, Allison, Laura, Ivan, Nadim, and all those who gave me moral support and encouragement in every moment.

To the Spanish government...

Also, I would like to thank the Spanish government for giving me the opportunity of my life by offering me a 4-years grant (BECAS MAE 2007-2011).

To you...

Finally, I would like to thank you all for your time in reading this dissertation. It is a pleasure to put this work in your hands; wishing it will help you and the scientific community move one step forward towards supporting humanity and making life easier.

Abstract

Research on Arabic Handwritten Text Recognition (HTR) and Arabic-English Machine Translation (MT) has been usually approached as two independent areas of study. However, the idea of creating one system that combines both areas together, in order to generate English translation out of images containing Arabic text, is still a very challenging task. This process can be interpreted as the translation of Arabic images. In this thesis, we propose a system that recognizes Arabic handwritten text images, and translates the recognized text into English. This system is built from the combination of an HTR system and an MT system.

Regarding the HTR system, our work focuses on the use of *Bernoulli Hidden Markov Models (BHMMs)*. BHMMs had proven to work very well with Latin script. Indeed, empirical results based on it were reported on well-known corpora, such as IAM and RIMES. In this thesis, these results are extended to Arabic script, in particular, to the well-known IfN/ENIT and NIST OpenHaRT databases for Arabic handwritten text.

The need for transcribing Arabic text is not only limited to handwritten text, but also to printed text. Arabic printed text might be considered as a simple form of handwritten text version. Thus, for this kind of text, we also propose Bernoulli HMMs. In addition, we propose to compare BHMMs with state-of-the-art technology based on neural networks.

A key idea that has proven to be very effective in this application of Bernoulli HMMs is the use of a sliding window of adequate width for feature extraction. This idea has allowed us to obtain very competitive results in the recognition of both Arabic handwriting and printed text. Indeed, a system based on it ranked first at the ICDAR 2011 Arabic recognition competition on the Arabic Printed Text Image (APTI) database. Moreover, this idea has been refined by using *repositioning* techniques for extracted windows, leading to further improvements in

Arabic text recognition.

In the case of handwritten text, this refinement improved our system which ranked first at the *ICFHR 2010 Arabic handwriting recognition competition* on IfN/ENIT. In the case of printed text, this refinement led to an improved system which ranked second at the *ICDAR 2013 Competition on Multi-font and Multi-size Digitally Represented Arabic Text* on APTI. Furthermore, this refinement was used with neural networks-based technology, which led to state-of-the-art results.

For machine translation, the system was based on the combination of three state-of-the-art statistical models: the standard phrase-based models, the hierarchical phrase-based models, and the N-gram phrase-based models. This combination was done using the *Recognizer Output Voting Error Reduction (ROVER)* method. Finally, we propose three methods of combining HTR and MT to develop an Arabic image translation system. The system was evaluated on the NIST OpenHaRT database, where competitive results were obtained.

Resumen

El reconocimiento de texto manuscrito (HTR) en árabe y la traducción automática (MT) del árabe al inglés se han tratado habitualmente como dos áreas de estudio independientes. De hecho, la idea de crear un sistema que combine las dos áreas, que directamente genere texto en inglés a partir de imágenes que contienen texto en árabe, sigue siendo una tarea difícil. Este proceso se puede interpretar como la traducción de imágenes de texto en árabe. En esta tesis, se propone un sistema que reconoce las imágenes de texto manuscrito en árabe, y que traduce el texto reconocido al inglés. Este sistema está construido a partir de la combinación de un sistema HTR y un sistema MT.

En cuanto al sistema HTR, nuestro trabajo se enfoca en el uso de los *Bernoulli Hidden Markov Models* (BHMMs). Los modelos BHMMs ya han sido probados anteriormente en tareas con alfabeto latino obteniendo buenos resultados. De hecho, existen resultados empíricos publicados usando corpus conocidos, tales como IAM o RIMES. En esta tesis, estos resultados se han extendido al texto manuscrito en árabe, en particular, a las bases de datos IfN/ENIT y NIST OpenHaRT.

En aplicaciones reales, la transcripción del texto en árabe no se limita únicamente al texto manuscrito, sino también al texto impreso. El texto impreso se puede interpretar como una forma simplificada de texto manuscrito. Por lo tanto, para este tipo de texto, también proponemos el uso de modelos BHMMs. Además, estos modelos se han comparado con tecnología del estado del arte basada en redes neuronales.

Una idea clave que ha demostrado ser muy eficaz en la aplicación de modelos BHMMs es el uso de una ventana deslizante (*sliding window*) de anchura adecuada durante la extracción de características. Esta idea ha permitido obtener resultados muy competitivos tanto en el reconocimiento de texto manuscrito en árabe como en el de texto impreso. De hecho, un sistema basado en este tipo de extracción de características quedó en la primera posición en el concurso *ICDAR 2011 Arabic recognition competition* usando la base de datos *Arabic Printed Text Image (APTI)*. Además, esta idea se ha perfeccionado mediante el uso de técnicas de reposicionamiento aplicadas a las ventanas extraídas, dando lugar a nuevas mejoras en el reconocimiento de texto árabe.

En el caso de texto manuscrito, este refinamiento ha conseguido mejorar el sistema que ocupó el primer lugar en el concurso *ICFHR 2010 Arabic handwriting recognition competition* usando IfN/ENIT. En el caso del texto impreso, este refinamiento condujo a un sistema mejor que ocupó el segundo lugar en el concurso *ICDAR 2013 Competition on Multi-font and Multi-size Digitally Represented Arabic Text* en el que se usaba APTI. Por otro lado, esta técnica se ha evaluado también en tecnología basada en redes neuronales, lo que ha llevado a resultados del estado del arte.

Respecto a la traducción automática, el sistema se ha basado en la combinación de tres tipos de modelos estadísticos del estado del arte: los modelos *standard phrase-based*, los modelos *hierarchical phrase-based* y los modelos *N-gram phrase-based*. Esta combinación se hizo utilizando el método *Recognizer Output Voting Error Reduction (ROVER)*. Por último, se han propuesto tres métodos para combinar los sistemas HTR y MT con el fin de desarrollar un sistema de traducción de imágenes de texto árabe a inglés. El sistema se ha evaluado sobre la base de datos NIST OpenHaRT, donde se han obtenido resultados competitivos.

Resum

El reconeixement de text manuscrit (HTR) en àrab i la traducció automàtica (MT) de l'àrab a l'anglès s'han tractat habitualment com dues àrees d'estudi independents. De fet, la idea de crear un sistema que combine les dues àrees, que directament genere text en anglès a partir d'imatges que contenen text en àrab, continua sent una tasca difícil. Aquest procés es pot interpretar com la traducció d'imatges de text en àrab. En aquesta tesi, es proposa un sistema que reconeix les imatges de text manuscrit en àrab, i que tradueix el text reconegut a l'anglès. Aquest sistema està construït a partir de la combinació d'un sistema HTR i d'un sistema MT.

Pel que fa al sistema HTR, el nostre treball s'enfoca en l'ús dels *Bernoulli Hidden Markov Models (BHMMs)*. Els models BHMMs ja han estat provats anteriorment en tasques amb alfabet llatí obtenint bons resultats. De fet, existeixen resultats empírics publicats emprant corpus coneguts, tals com IAM o RIMES. En aquesta tesi, aquests resultats s'han estès a la escriptura manuscrita en àrab, en particular, a les bases de dades IfN/ENIT i NIST OpenHaRT.

En aplicacions reals, la transcripció de text en àrab no es limita únicament al text manuscrit, sinó també al text imprès. El text imprès es pot interpretar com una forma simplificada de text manuscrit. Per tant, per a aquest tipus de text, també proposem l'ús de models BHMMs. A més a més, aquests models s'han comparat amb tecnologia de l'estat de l'art basada en xarxes neuronals.

Una idea clau que ha demostrat ser molt eficaç en l'aplicació de models BHMMs és l'ús d'una finestra lliscant (*sliding window*) d'amplària adequada durant l'extracció de característiques. Aquesta idea ha permès obtenir resultats molt competitius tant en el reconeixement de text àrab manuscrit com en el de text imprès. De fet, un sistema basat en aquest tipus d'extracció de característiques va quedar en primera posició en el concurs *ICDAR 2011 Arabic recognition competition* emprant la base de dades *Arabic Printed Text Image (APTI)*.

A més a més, aquesta idea s'ha perfeccionat mitjançant l'ús de tècniques de reposicionament aplicades a les finestres extretes, donant lloc a noves millores en el reconeixement de text en àrab. En el cas de text manuscrit, aquest refinament ha aconseguit millorar el sistema que va ocupar el primer lloc en el concurs *ICFHR 2010 Arabic handwriting recognition competition* usant IfN/ENIT. En el cas del text imprès, aquest refinament va conduir a un sistema millor que va ocupar el segon lloc en el concurs *ICDAR 2013 Competition on Multi-font and Multi-size Digitally Represented Arabic Text* en el qual s'usava APTI. D'altra banda, aquesta tècnica s'ha avaluat també en tecnologia basada en xarxes neuronals, el que ha portat a resultats de l'estat de l'art.

Respecte a la traducció automàtica, el sistema s'ha basat en la combinació de tres tipus de models estadístics de l'estat de l'art: els models *standard phrase-based*, els models *hierarchical phrase-based* i els models *N-gram phrase-based*. Aquesta combinació es va fer utilitzant el mètode *Recognizer Output Voting Errada Reduction (ROVER)*. Finalment, s'han proposat tres mètodes per combinar els sistemes HTR i MT amb la finalitat de desenvolupar un sistema de traducció d'imatges de text àrab a anglès. El sistema s'ha avaluat sobre la base de dades NIST OpenHaRT, on s'han obtingut resultats competitius.

الخلاصة

لَقد تمّ في السابق دراسة موضوعَيّ التعرُّف التلقائي على النصوص العربية المكتوبة بخط اليّد (HTR) والترجمة التلقائية (MT) من اللّغة العربية إلى اللّغة الانجليزية كَمجالَيّن مُنفَصِلين عن بعضهما. إنّ فكرة عمل نظام واحد قادر على دمج كلا هذين المجالين معاً من أجل إنتاج الكلمات والنصوص الانجليزية من صورٍ تحتوي على نصوص عربية، مازالت تُعدُّ عملية صعبة المنال. هذه العملية تسمى عملية ترجمة الصور المحتوية على نصوص عربية. في هذه الأطروحة سوف نعرض نظاماً قادراً على فَهم الكلمات والنصوص المكتوبةِ بخط اليّد المدرجة في الصور، وترجمتها إلى اللّغة الانجليزية. إنّ هذا النظام مبنيٌّ من عملية دمج كُلّاً من نظاميّ HTR و MT معاً.

بالنسبة لنظام التعرف التلقائي على النصوص المكتوبة بخط اليّد (HTR) ، فيتركز عملنا على استخدام نماذج ماركوف المخفية باستخدام توزيعات برنولي (BHMMs). لقد أثبتت هذه النماذج قدرتها على العمل بشكل جيد جداً مع النصوص اللاتينية. كدليلٍ لهذا، فقد تم نشر نتائج مهمة لهذه النماذج بعد تجربتها على قواعد بيانات لمجموعة نصوص معروفة مثل IAM و RIMES. في هذه الأطروحة سيتم توسيع هذه النتائج بتجربتها على قواعد بيانات لنصوص عربية مكتوبة بخط اليد مثل IfN/ENIT و OpenHaRT.

إن الحاجة إلى التدوين الأوتوماتيكي للنصوص العربية في يومنا هذا لا يقتصر فقط على النصوص المكتوبة بخط اليّد، وإنما يشمل النصوص المطبوعة أيضاً. يمكننا اعتبار النصوص العربية المطبوعة كنوع مبسّط من النصوص المكتوبة بخط اليد حيث أن مشكلة تنوّع أشكال الكتابة من شخص إلى أخر غير موجودة. لهذا السبب سنقترح في هذه الأطروحة تطبيق نماذج BHMMs مع هذا النوع من النصوص أيضاً. بالإضافة إلى ذلك، فإننا سنُقارن نماذج BHMMs مع إحدى أحدث النماذج المبنية على الشبكات العصبونية الاصطناعية *(neural networks)*.

من أكثر الطرق المستعملة في هذا المجال التي أثبتت فعاليتها مع نماذج BHMMs عند استخراج خصائص الصورة، هي طريقة استعمال النافذة المنزلقة *(sliding window)* ذات العرض المناسب. قد سَمَحَت لنا هذه الطريقة بالحصول على نتائج منافسة جداً في عملية التعرف التلقائي على النصوص المكتوبة بخط اليّد والنصوص المطبوعة. في الواقع، إن نظاماً مبنياً على هذه الطريقة قد تم منحه الدرجة الآولى في مسابقة *ICDAR 2011 Arabic recognition competition* باستخدام قاعدة البيانات *Arabic Printed Text Image (APTI)*. وعلاوة على ذلك، قد تم تطوير هذه الطريقة باستخدام تقنيات إعادة تموضع النوافذ المستخرجة *(repositioning)* ، التي أدّت إلى تحسين عملية التعرف التلقائي على النصوص العربية.

إن طريقة ال *repositioning* قد قادت إلى نظام متطور احتلّ المركز الأول في مسابقة *ICFHR 2010 Arabic handwriting recognition* باستخدام قاعدة البيانات IfN/ENIT في حالة التعرف التلقائي على النصوص المكتوبة، والمركز الثاني في مسابقة *ICDAR 2013 Multi-font and Multi-size Digitally Represented Arabic Text* في حالة التعرف التلقائي على النصوص المطبوعة باستخدام قاعدة البيانات APTI. علاوة على ذلك، فقد تم استخدام هذه الطريقة *(repositioning)* مع النماذج المبنية على الشبكات العصبونية الاصطناعية *(neural networks)* ، الذي حصل على أفضل النتائج ليومنا هذا.

أما بالنسبة للترجمة التلقائية، إن نظامنا قد تم بنائه عن طريق دمج ثلاثةَ من أحدث النماذج المتوفرة، ألا وهي: نماذج *standard phrase-based* ، نماذج *hierarchical phrase-based* ، ونماذج *N-gram phrase-based*. إن عملية الدمج هذه قد تمت باستخدام طريقة *Recognizer Output Voting Error Reduction (ROVER)*.

أخيراً، فإننا نقترح في هذه الأطروحة ثلاثة طرق للجمع بين نظامَيّ HTR و MT لتطوير نظام الترجمة للصور المحتوية على نصوص عربية. قد تم اختبار هذا النظام على قاعدة البيانات *NIST OpenHaRT'13* ، حيث تم الحصول على نتائج منافسة جداً.

Contents

CHAPTER 1

Introduction

Contents

Arabic is spoken by more than 234 million people and important in the cultures of many more. It is one of the six United Nations official languages. Over the past years, the interest in Arabic text recognition and translation has grown. Indeed, many researches had focused on Arabic Handwritten Text Recognition (HTR) and Machine Translation (MT) as two separate areas of study. However, few work has discussed the idea of developing an Arabic image translation system. That is, a system that combines HTR and MT fields together to produce English translations from images containing Arabic text.

In our daily life many applications can be mentioned that involve Arabic image translation systems. For example: the translation of text written on street posters, on walls or on papers. This can be of great help to tourists who wish to communicate with Arabic speakers. Another example could be the transcription and translation of scanned documents. This could be adapted for researchers or businesspeople searching for information originally written in Arabic.

One of the most recent published systems was described in [1] at the OpenHaRT'13 evaluation [2]. In this system, text recognition was based on HMMs and BLSTMs recurrent networks, while the machine translation system was based on Moses toolkit [3]. Another system was described in [4]. It translates Arabic text in signboards into English text by using the camera of mobile phones. For character recognition, the authors suggested a *Template matching algorithm* to find the similarity between the region of interest and the template of a specific letter. For machine translation system, authors used a simple algorithm that searches each word of a sentence in a given dictionary and appends the results to the output string. Also, another system that automatically translates Arabic text embedded in images into English was described in [5]. For Text recognition the authors used a commercial OCR software, *Sakhr Automatic Reader version 8.0 (Platinum Edition)*. For machine translation, authors used a phrase-based statistical MT system called *CMU PanDoRA*.

In this thesis, we propose an Arabic image translation system based on the combination of HTR and MT models. In the case of handwriting recognition of Arabic text, our work will focus on the generative and discriminative Windowed Bernoulli Hidden Markov models (Windowed BHMMs). These models have been successfully used with handwritten text in many languages [6, 7]. In the case of Arabic text translation, our work will focus on the combination of three different state-of-the-art phrase-based translation models: the standard (log-linear) phrase-based models using the Moses [3] toolkit, the hierarchical phrase-based models using the Jane [8] toolkit, and the N-gram phrase based models using the Ncode [9] toolkit. The combination of these models will be performed using the ROVER [10] tool.

1.1 Scientific Goals

The goals of this thesis are pointed out next:

- **Develop an Arabic handwriting recognition system:** Text recognition is the first step of any image translation system. Indeed, it is a main component for developing such system.

  The basic idea is to focus on developing an HTR system based on Windowed Bernoulli Hidden Markov models (Windowed BHMMs). These models have been successfully used with handwritten text in many languages [6].

- **Develop a Printed Arabic recognition system:** If we look back to the examples mentioned in the introduction, most of them are based on printed Arabic text. For an Arabic image translation task, Arabic printed text recognition is considered an important task to be solved towered a robust image translation system.

  Unlike Latin script, Arabic script has a more complicated structure since many characters might overlap. This makes that the conventional OCR techniques for Latin script limited in dealing with Arabic script. The goal here is to develop an Arabic printed system based on the state-of-the-art technology developed for Arabic HTR.

- **Develop an Arabic machine translation system:** The idea here is to develop an Arabic machine translation system using state-of-the-art machine translation models.

- **Propose an Arabic image translation system:** The goal here is to combine the models previously mentioned toward a robust Arabic image translation system.

- **Evaluate these systems on well-known HTR and MT corpora:** As usual, in order to test the performance of any work, it must be evaluated on well-known corpora, and indeed, compared with other results following the same evaluation criteria. For this purpose, we plan to evaluate our work in both fields (text recognition and machine translation) as two separate systems and also in conjunction as one system. These systems are planned to be evaluated on well-known corpora such as IfN/ENIT, NIST OpenHaRT, and APTI.

1.2 Document Structure

To facilitate the reading experience of this thesis, a preliminaries chapter (Chapter 2) was designed to introduce the basic concepts, which are not directly connected to this work.

In chapter 3, our Arabic handwriting recognition system is introduced. It is based on Bernoulli Hidden Markov Models (BHMMs). A description of the databases used to evaluate this system is given, which is followed by a discussion about the experiments. Finally, our participation in ICFHR 2010 and OpenHaRT 2010 competitions is discussed.

In chapter 4, we extend the experiments followed in Arabic handwritten text to Arabic printed text. In this chapter we study the effect of our technology based on BHMMs on this kind of text. A complete series of experiments is carried out on Arabic printed text database. Then, a new technology based on neural networks is introduced and tested. Finally, our participation in ICDAR 2011 and 2013 competitions is discussed.

In chapter 5, the image translation system is introduced. This system is based on the Arabic handwriting recognition system (Chapter 3), and the combination of three of state-of-the-art machine translation systems. Experiments are carried out on the NIST OpenHaRT 2013 database.

Bibliography

[1] O. Morillot, C. Oprean, L. Likforman-Sulem, C. Mokbel, E. Chammas, and E. Grosicki. The UOB-Telecom ParisTech Arabic Handwriting Recognition and Translation Systems for the OpenHart 2013 Competition. *Proc. of the NIST 2013 Open Handwriting and Recognition Workshop*, 2013.

[2] A. Tong and M. Przybocki and V. Märgner and H. El Abed. NIST 2013 Open Handwriting Recognition and Translation (OpenHaRT'13) evaluation. *Proc. of the NIST 2013 Open Handwriting and Recognition Workshop*, 2013.

[3] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, and R. Zens. Moses: Open source toolkit for statistical machine translation. In *Annual meeting-association for computational linguistics*, volume 45, page 2, 2007.

[4] R. A. R. A. Al-Hashemi and S. A. Alsharari. Instant arabic translation system for signboard images based on printed character recognition. *International Journal of Machine Learning and Computing*, pages 384–388, 2013.

[5] Y. Chang, D. Chen, Y. Zhang, and J. Yang. An image-based automatic arabic translation system. *Pattern Recognition*, 42(9):2127 – 2134, 2009.

[6] A. Giménez and A. Juan. Embedded Bernoulli Mixture HMMs for Handwritten Word Recognition. In *Proc. of the 10th Int. Conf. on Document Analysis and Recognition (ICDAR 2009)*, pages 896–900, Barcelona (Spain), July 2009.

[7] A. Giménez, J. Andrés-Ferrer, and A. Juan. Discriminative Bernoulli HMMs for isolated handwritten word recognition. *Pattern Recognition Letters*, 35(0):157 – 168, 2014. Frontiers in Handwriting Processing.

[8] D. Vilar, D. Stein, M. Huck, and H. Ney. Jane: Open source hierarchical translation, extended with reordering and lexicon models. In *Proc. of the Joint 5th Workshop on Statistical Machine Translation and MetricsMATR*, pages 262–270, 2010.

[9] J. B. Marino, R. E. Banchs, J. M. Crego, A. de Gispert, P. Lambert, J. A. R. Fonollosa, and M. R. Costa-jussà. N-gram-based machine translation. *Computational Linguistics*, 32(4):527–549, 2006.

[10] J. G. Fiscus. A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (rover). In *Automatic Speech Recognition and Understanding, 1997. Proc., 1997 IEEE Workshop on*, pages 347–354, 1997.

CHAPTER 2

Preliminaries

Contents

This chapter was designed to outline the general idea of the work presented in this thesis. Also, it will address some important concepts which are not the main focus of this thesis.

2.1 Image Translation System

The idea of an image translation system is to take an image containing some text as an input, and to produce the translation of that text into another language as an output. To discuss this system, we need its mathematical formulation based on probabilities. It can be written in the form of $p(y \mid f)$. In other words, the system should decide in favor of a translation $y^\star$ satisfying,

$$y^\star = \underset{y \in Y}{\operatorname{argmax}}\ p(y|f) \tag{2.1}$$

In practice, giving that Eq. (2.1) is very costly to be computed, it is usually approximated as follows:

$$y^\star \approx \underset{y \in Y}{\operatorname{argmax}}\ [p(y| \max_x \{p(x|f)\})], \tag{2.2}$$

where the probability $p(x|f)$ is usually approximated by a recognition system, while the probability $p(y|x)$ is usually approximated by a statistical machine translation system. Being x a candidate recognized source (Arabic) text and y a candidate translated sentence (in English) corresponding to the input image f.

From here, we can take each posterior probability alone and solve it. The first part of Eq. (2.2) (the transcription system) is the system that maximize the probability of a sentence x giving an image f. It can be expressed using Bayes rule as follows:

$$p(x \mid f) \approx p(x)p(f|x), \tag{2.3}$$

where $p(x)$ is the language model of x, and the $p(f|x)$ is the transcription model, which is usually modeled by Hidden Markov Models. The second part of Eq. (2.2) (the translation system) is based on the noisy channel model. The Bayes rule is used again to calculate the translation probability for translating a sentence x into sentence y as follows:

$$p(y \mid x) \approx p(y)p(x|y), \tag{2.4}$$

with $p(y)$ being the language model of y, and $p(x|y)$ being the translation model.

To clearly illustrate the image translation system including both parts, let's take a look at Figure 2.1. Given a text image of an unknown word, the transcription model $p(f|x)$ computes the maximum probability of the given image with its corresponding word. To compute these probabilities, text images are first preprocessed and then transformed into a sequence of feature vectors. This is also happened with the help of an Arabic Language model $p(x)$ to ensure all output sequence of words or characters belong to a known range of vocabulary (lexicon).

The output of the first system, the text x, is then passed to the second system to be translated. This text is passed through a processing step to clean the noise, which is in some sense transformed into a specific format that the translation model will understand (tokenization a decoding steps). The idea is to find the maximum probability of translating the text x. This can be done using various models such as a phrased-based model, a reordering model, or a language model. The final step is to return the translated text y.

2.2 Text Recognition

A text recognition system including typewritten and handwritten text is the task of transcribing images containing text. In other words, it is the processes of recognizing characters or words given a lexicon or a dictionary [1, 2]. In the past, few research work was carried out on handwritten text recognition due to poor performance achieved by the available recognition systems back in that time [3]. Despite that isolated Optical Character Recognition (OCR) task for some scripts can be considered solved [4], the word recognition task including handwritten or typewritten text is still a challenging task to researchers. Nowadays, text recognition is considered one of the most important tasks. Indeed, it is receiving more attention that ever before.

Some of the current recognition systems handle the whole recognition process in two steps: A pre-processing step where lines and words are extracted from a document, and a recognition step where the extracted text is transcribed. The pre-processing step is introduced in Section 2.2.1.

We might think of a recognition step as the task to automatically transform an image f containing text, which is transformed into a sequence of fixed-dimension feature vectors $f = \mathbf{f}_1, \ldots, \mathbf{f}_N$, into its corresponding text $x = \mathbf{x}_1, \ldots, \mathbf{x}_N$. The idea is to find the most probable transcription x giving an observed feature vector f. In other words, the recognizer should decide in favor of a transcription (word) x^* satisfying,

$$x^* = \arg\max_x \ p(x \mid f) \tag{2.5}$$

The well-known Bayes is used to rewrite the right side of the equation as follows:

$$x^* = \arg\max_x \ p(x) \, p(f \mid x) \, , \tag{2.6}$$

where $p(x)$ is the probability that the word x was written, which it is usually approximated by a *language model* (Section 2.4), and $p(f \mid x)$ is the probability of emitting the feature

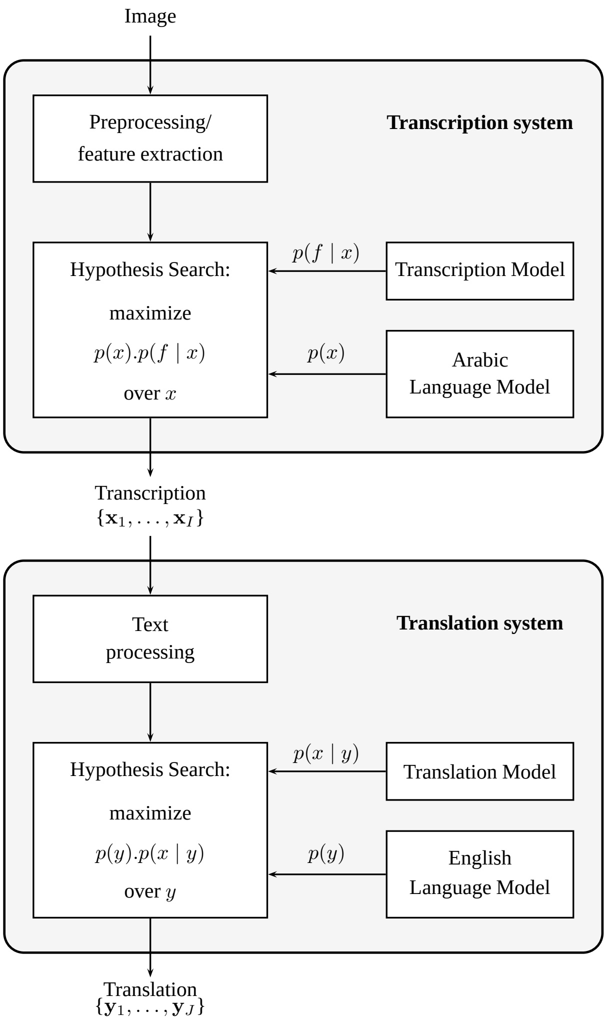


Image
Preprocessing/ feature extraction
Transcription system
Hypothesis Search: maximize $p(x).p(f \mid x)$ over x
$p(f \mid x)$
Transcription Model
$p(x)$
Arabic Language Model
Transcription $\{\mathbf{x}_1, \ldots, \mathbf{x}_I\}$
Text processing
Translation system
Hypothesis Search: maximize $p(y).p(x \mid y)$ over y
$p(x \mid y)$
Translation Model
$p(y)$
English Language Model
Translation $\{\mathbf{y}_1, \ldots, \mathbf{y}_J\}$

Figure 2.1: Basic scheme for an Arabic image translation system, which is shown as a concatenation of two systems: a transcription system (top) and a translation system (bottom)

vector f when observing the text x. In is called the *text image model*. The language model is usually modeled using n-grams approach in continuous text recognition, while in isolated word recognition it is modeled using a table of prior probabilities, one for each word (class), as in traditional statistical classifiers.

As mentioned previously, the OCR task for some scripts can be considered solved, however more complex tasks such as the handwriting recognition task is still challenging. This might be due to the huge amount of variations in handwritten text. To model the writing variations, Hidden Markov Models (HMMs) are used. They have been established as a de-facto standard for speech recognition systems [5], however they are widely used for off-line handwriting recognition in many languages [6–11]. Other models are also used such as those based on artificial neural networks or dynamic time wrapping [12].

In this thesis, we will focus on using HMMs to model the probability $p(f|x)$. A wide-spread criteria used in automatic parameter estimation from a given training set is the Maximum Likely-hood Estimation (MLE). The use of this maximization criteria involves that all observations f are modeled as complete data. However, since this is not the case, the Expectation Maximization (EM) algorithm (Section 2.2.4) is used in order to apply the MLE criteria on HMMs and the mixture models. More details about modeling the $p(f|x)$ using the HMMs will be discussed during this thesis.

2.2.1 Pre-processing and Feature Extraction

Pre-processing involves applying some layout analysis and processing techniques on the image. The idea of processing is to normalize the input image in order to facilitate the recognition. There is no standards concerning the functions of this step, but some techniques are commonly being applied such as: thresholding and background removal, noise removal, skew correction, block or field extraction, or brightness and color normalization. These techniques are usually performed before the lines and words extraction process.

Once the lines and words are extracted, extracted images are processed again using different techniques such as slant correction, size normalization, or binarization to ensure they are ready to be transcribed. For more details about these techniques please refer to [13, 14].

The final stage in this step is the feature extraction process. In this step, text line images are transformed into a sequence of fixed-dimension feature vectors to be fed into the transcription model.

2.2.2 Otsu's Method

The Otsu's method [15] is a binarization technique which belongs to the family of global thresholding binarization techniques. It is a simple ,quick, and robust method for reasonable clean images. The way this method works is by calculating a threshold value in a gray-scale image, which is then used to separate white pixels from black pixels. The threshold value is calculated by maximizing inter-class and minimizing intra-class variances of gray values. That is, for a threshold T, gray scale values are split in two classes: gray values greater than T and gray values smaller than T. Then, the mean and probabilities for each class are calculated as follows:

$$p_1 = \sum_{g=0}^{T} h_g \,, \qquad p_2 = \sum_{g=T+1}^{L-1} h_g = 1 - p_1 \,, \tag{2.7}$$

$$\mu_1 = \frac{1}{p_1} \sum_{g=0}^{T} g h_g \,, \qquad \mu_2 = \frac{1}{p_2} \sum_{g=T+1}^{L-1} g h_g \,, \tag{2.8}$$

where L is the greatest gray level value and h is the normalized histogram of the given input image. Finally, the thresholding value is selected as follows

$$T^* = \underset{T}{\operatorname{argmax}}\; p_1 p_2 (\mu_1 - \mu_2)^2 \tag{2.9}$$

2.2.3 Center of Mass

The center of mass in two-dimensional region is a term referred to the central point of a 2D object in a 2D matrix. In the case of binary or gray images containing text, it is the central point of that text. The center of mas (x_c, y_x) is calculated as follows:

$$x_c = \frac{\sum_x x(\sum_y f(x,y))}{m} \tag{2.10}$$

$$y_c = \frac{\sum_y y(\sum_x f(x,y))}{m}, \tag{2.11}$$

where $f(x,y)$ is the value of (x,y) in the matrix, and m is mass of the image, which is basically the sum of all values in that image. It is calculated as follows:

$$m = \sum_x \sum_y f(x,y) \tag{2.12}$$

2.2.4 EM Algorithm

Given a training set $\mathbf{x}_1^N$, a common criteria used in automatic parameter estimation is the Maximum Likelihood Estimation (MLE), which tries to maximize the log-likelihood $\hat{\boldsymbol{\Theta}}$ of the training set, that is:

$$\hat{\boldsymbol{\Theta}} = \underset{\boldsymbol{\Theta}}{\operatorname{argmax}} \sum_n \log p(\mathbf{x_n} \mid \boldsymbol{\Theta}) \tag{2.13}$$

This equation can be solved in most cases by performing the first derivative, due to the fact that MLE is considered a simple convex optimization problem. An examples on that is the use of MLE in Gaussian distributions, Bernoulli distributions or multinomial distributions. However, when the training set ($\mathbf{x}_1^N$) is modeled as incomplete data, solving Eq. (2.13) is much more complicated. An example on that is the use of hidden Markov models or the mixtures of probabilistic models, which are commonly used in most pattern recognitions fields.

The EM algorithm was proposed in [16] in order to apply the MLE criterion on this kind of models. In EM algorithm, incomplete data is represented as hidden variables denoted as $\mathbf{z}_1^N$. Thus, the probabilistic distribution can be redefined as follows:

$$p(\mathbf{x} \mid \mathbf{\Theta}) = \int p(\mathbf{x}, \mathbf{z} \mid \mathbf{\Theta}) d\mathbf{z}\,, \tag{2.14}$$

so the MLE estimation function in Eq. (2.13) can be rewritten as

$$\hat{\mathbf{\Theta}} = \underset{\mathbf{\Theta}}{\operatorname{argmax}} \sum_n \log \int p(\mathbf{x_n}, \mathbf{z_n} \mid \mathbf{\Theta}) d\mathbf{z_n} \tag{2.15}$$

Equation (2.15) is solved using the EM algorithm following two iterative steps approach: The first step is called the *Expectation (E)* step, since the expected value of the log-likelihood is calculated given a previous estimation of the parameters and the known data. The second step is called the *Maximization (M)* step, where the parameters that maximize the equation of E step are calculated. The EM algorithm is proved to maximize the log-likelihood in each iteration [17]. A schematic description of the EM algorithm is shown below:

Initialization: set $k = 0$ and choose initial $\mathbf{\Theta_{(0)}}$

Loop:

1. **E step:** for all $\mathbf{\Theta}$, compute

$$\mathcal{Q}\left(\mathbf{\Theta} \mid \mathbf{\Theta}^{(k)}\right) = \sum_n E\left(\log p(\mathbf{x_n}, \mathbf{z_n} \mid \mathbf{\Theta}) \mid \mathbf{x_n}, \mathbf{\Theta}^{(k)}\right) \tag{2.16}$$

2. **M step:** compute

$$\mathbf{\Theta}^{(k+1)} = \underset{\mathbf{\Theta}}{\operatorname{argmax}}\, \mathcal{Q}\left(\mathbf{\Theta} \mid \mathbf{\Theta}^{(k)}\right) \tag{2.17}$$

Until: $L\left(\mathbf{\Theta}^{(k+1)}; \mathbf{x}_1^N\right) - L\left(\mathbf{\Theta}^{(k)}; \mathbf{x}_1^N\right) \leq \epsilon$

2.3 Machine Translation

In this section we review the state-of-art applications and approaches that we used to carry out our experiments in the field of *Statistical Machine Translation (SMT)*. We might think of a SMT as a task to automatically translate a source sentence x into a target sentence y. The system is to select the sentence with the higher probability among all possible y.

$$x = x_1 ... \, x_j, x_j \in \mathbf{X}, \; j = 1, ..., J \tag{2.18}$$

$$y = y_1 ... \, y_i, x_i \in \mathbf{Y}, \; i = 1, ..., I \tag{2.19}$$

where x_j and y_i denote source and target words; and $\mathbf{X}$ and $\mathbf{Y}$, the source and target vocabularies respectively.

The state-of-art SMT follow the so called noisy-channel approach (regarding the translation process as a channel which distorts the target sentence and outputs the source sentence) [18–20], where the optimal target sentence y is searched according to

$$y^* = \underset{y}{\operatorname{argmax}} \; p(y \mid x) \tag{2.20}$$

$$= \underset{y}{\operatorname{argmax}} \; p(y) \, p(x \mid y) \tag{2.21}$$

The so-called search problem is to compute a target sentence y for which this probability is maximum. Applying Bayes' theorem we can re-write (eq (2.21)) as shown above, where $p(y \mid x)$ is the *translation model*, and $p(y)$ is the *language model*. The language model describes the correctness of the target language sentence which helps to avoid syntactically incorrect sentences. The translation model is usually decomposed into *lexicon model* and *alignment model*.

Nevertheless, most of the current statistical MT systems present an alternative modeling of the translation process different from that presented in Eq. (2.21). The posterior probability is modeled as a log-linear combination of feature functions [21, 22] under the framework of maximum entropy [23]

$$y^* = \underset{y}{\operatorname{argmax}} \sum_{m=1}^{M} \lambda_m h_m(x, y), \tag{2.22}$$

where λ_m is the log-linear interpolation weight, M is the number of features (models), and $h_m(x, y)$ is a feature function that assigns a score to the sentence pair (x, y).

2.3.1 Pre- and Post-processing

In a machine translation system preprocessing is the process of modifying the raw input data in a way that will increase the efficiency and accuracy of a translation system. It's well known that the preprocessing carried on a parallel corpus has a big impact on translation quality. Usually Latin-based scripts require a similar preprocessing techniques since they all share common characters. However, other scripts like Arabic, require different preprocessing techniques. Indeed a preprocessing techniques used on English for example will definitely not work on Arabic.

For many years, the key features that preprocessing possess for improving a translation system include, but are not limited to lower-casing, sentence shortening, tokenization, sentence splitting, morphological analysis, and part of speech tagging. One of the leading toolkit that perform preprocessing very well is the Freeling toolkit [24]. This toolkit is an open-source multilingual language processing library that provides a wide range of analyzers for several languages, such as English, Spanish, French, Catalan, and Italian. On the other hand, Arabic script can preprocessed using the *MADA+TOKAN* toolkit [25] for example. This toolkit performs some preprocessing techniques on Arabic raw text such as part-of-speech tagging, diacritization, tokenization, and more.

As the input text is encoded (preprocessed), the output must be decoded back to a readable text. This process is called post-processing, because it is applied after the translation process is taken place. For that, some processing techniques, other than the the ones applied for preprocessing, are applied to reverse the effect of the preprocessing techniques once the text is translated. Some of these techniques include detokenizaion and upper-casing.

2.3.2 Word Alignment

The alignment model is one of the main components of a translation model. We might think of an alignment model as a table containing the probability of mapping a source word to a target word. For many years, *GIZA++* toolkit [26] was used to establish word alignments. It is an implementation of several *IBM Models* [27] where, *IBM Model 1* uses only lexical translation probabilities, *IBM Model 2* adds an absolute alignment model, *IBM Model 3* adds a fertility model, *IBM Model 4* replaces the absolute alignment model with a relative alignment model, and *IBM Model 5* fixes a problem with deficiency in the model. In addition to the *IBM models*, *GIZA++* also implements Hidden Markov Models (HMMs).

Training word alignment models on large data, using a single GIZA++ process, is a very time-consuming processes. This process can be accelerated by running it in parallel using a multi-processor system with multi-threading technology, or by using computer clusters. *MGIZA++* [28] was implemented for this specific purpose. It splits the word alignment model calculation into various small processes to be run in parallel. Finally it accumulate them back to generate on big model.

The word alignment process is somehow limited in which the alignment of one target word is only allowed with each source word. To overcome this limitation, a heuristic proposed in [29] is followed. The parallel corpus is aligned bidirectionally: source to target and target to source. This generates two word alignments that have to be integrated. By applying intersection between the two alignments, a high-precision alignment of high-confidence alignment points is obtained. By applying the union between them, a high-recall alignment with additional alignment points is obtained. As an example, let us take a look at the *Arabic-English* alignment integration example in Figure 2.3.2. Please note that the Arabic text is encoded using the *MADA+TOKAN* toolkit explain in Section 2.3.1.

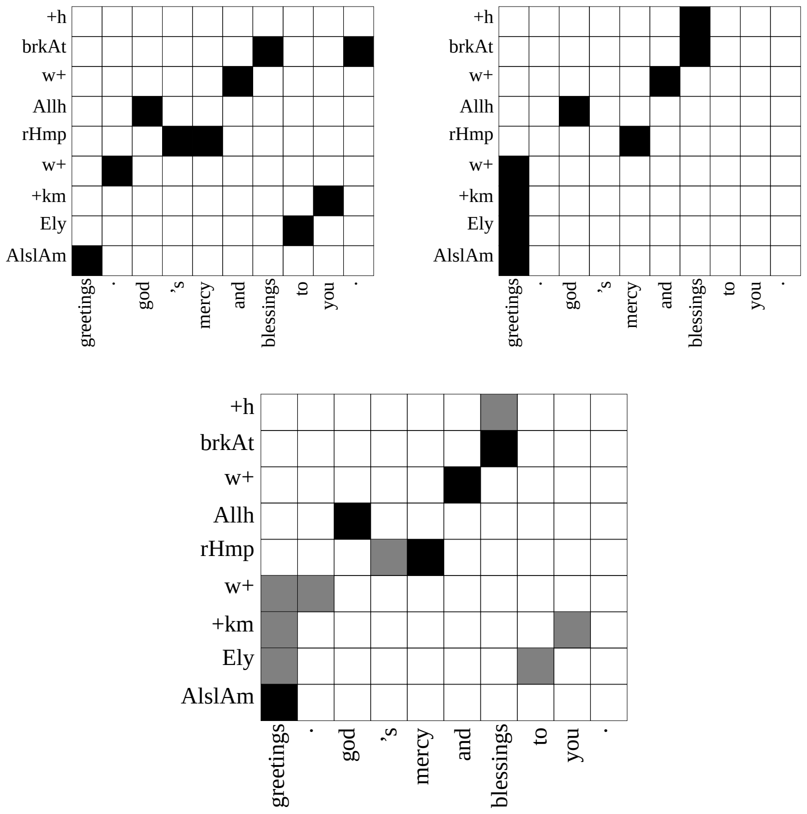


Figure 2.2: Arabic-English and English-Arabic word alignment merging by taking intersection (black) and union (gray)

2.3.3 Phrase-based Models

Unlike Word-based statistical models [27], phrase-based statistical machine translation models are based on the translation of phrases instead of words as atomic units. A phrase can be defined as a continuous multiword sequence. Phrases are mapped one-to-one based on a phrase translation table, and may be reordered. A Phrase translation table can be learned based on a word alignment by finding consistent pairs of phrases. Many methods were presented for learning phrase translation such as: The phrase-based joint probability model that simultaneously generates both the source and target sentences in a parallel corpus [30]. The scoring methods take consistency with the word alignment, lexical translation probabilities, phrase length, etc. [31]. The phrase alignments based on the intersection of the two GIZA++ alignments and points of the union for their expansion [32]. Finally, the standard phrase-based models which was introduced by P. Koehn et al [29]. In addition to these standard models, below we describe two other important models:

Standard Phrase-based Models

Standard Phrase-based models are simple and powerful techniques for machine translation. The use of phrases, which can be any sub-string, allow these models to learn local reordering, translation of short idioms, or insertions and deletions that are sensitive to local context [22, 33] Reordering for these models is handled by either a simple distance-based reordering model or a lexicalized reordering model.

The heuristic estimation of the standard phrase-based models is grounded on the Viterbi alignments computed as a byproduct of word-based alignment models. The Viterbi alignment is defined as the most probable alignment given the source and target sentences and an estimation of the model parameters. A good example of these models is the **Moses** toolkit [34]. Moses is a complete out-of-the-box translation system for academic research. It consists of all the components needed to pre-process the data, and train the language and translation models. It also contains tools for tuning these models using minimum error rate training and evaluating the resulting translations using the BLEU score. Moses uses GIZA++ for word alignments and SRILM for language modeling which are standard external tools.

Hierarchical Phrase-based Models

Hierarchical phrase-based models are one of the current promising approaches to machine translation. They take into account a weighted synchronous context-free grammar is induced from parallel text. In addition to contiguous lexical phrases, hierarchical phrases with usually up to two gaps are extracted. Hierarchical decoding is carried out with a search procedure which is based on CYK+ parsing [35]. A good example on these models is **Jane** toolkit [36].

Jane is an open source translation toolkit which has been developed at RWTH Aachen University and is freely available for non-commercial use. Jane provides efficient C++ implementations for hierarchical phrase extraction, optimization of log-linear feature weights, and parsing-based search algorithms. A modular design and flexible extension mechanisms allow for easy integration of novel features and translation approaches.

n-gram Phrase-based Models

The goal of *n*-gram Phrase-based Models is to define a general inference method for obtaining a finite-state transducer from a corpus of parallel text. The aim is to produce a transducer that is able to generalize the training data and can find the correct translation of new input sentences that have not been seen during the training process [37]. A good example on these models is the **Ncode** toolkit [38]. Ncode is an open source statistical machine translation decoder and its companion tools. Ncode implements the bilingual n-gram approach to machine translation as described in [39, 40]. Ncode main features include the use of multiple n-gram language models estimated over bilingual units, source words and/or target words or any factor decomposition, lexicalized reordering, several tuple (unigram) models, etc.. As for nearly all current statistical approaches to machine translation, these models are embedded in a linear model combination. Ncode splits the reordering and decoding problems of SMT in two separate modules, aiming at better tackling each of the problems. However, hard reordering decisions are avoided by means of using permutation lattices.

2.4 Language Modeling: *n*-gram Models

Language models (LMs) are used to model text properties like syntax and semantic independently from the morphological models [41]. They have many applications in speech recognition, machine translation and text recognition. The aim of these models, is to ensure that the output text is correct and falls within a range of a specific domain of vocabulary, and also to predict the next word in a word sequence.

If we take a look at Eq. (2.3), we need to compute the a priori probability $p(x)$ for every word x. That is:

$$p(x) = p(x_1) \cdot \prod_{t=2}^{T} p\left(x_t \mid x_1^{t-1}\right) \ , \tag{2.23}$$

where $p(x_t \mid x_1^{t-1})$ is the probability of the word x_t when we have already seen the sequence of words $x_1 \ldots x_{t-1}$. The sequence of words prior to x_t is called history. The recognizer must estimate the value of the probability $p(x_t \mid x_1^{t-1})$. In fact estimating this value is difficult and costly since sentences can be very long. For this reason these models are often approximated using smoothed n-gram models, which obtains surprisingly good performance [41], although they only capture short term dependencies.

The *n*-gram models are nowadays the most wide-spread models used for language modeling. It is used practically in all human language technologies [42]. In n-gram models, the probability $p(x_1, \ldots, x_T)$ of observing the sentence $x_1, \ldots, x_T$ is approximated as:

$$p(x) \approx p(x_1) \cdot \prod_{t=2}^{T} p\left(x_t \mid x_{t-(n-1)}^{t-1}\right) \ , \tag{2.24}$$

where the probability of observing the t^{th} word x_t in the context history of the preceding $t-1$ words can be approximated by the probability of observing it in the shortened context history of the preceding $n-1$ words (n^{th} order Markov property).

The parameter estimation can be easily carried out from a training set using the maximum likelihood estimation, since no hidden variables are required in *n*-gram models. The conditional probability can be calculated from n-gram frequency counts as:

$$p(w_t \mid w_{t-(n-1)}^{t-1}) = \frac{\text{count}(w_{t-(n-1)}, \ldots, w_{t-1}, w_t)}{\text{count}(w_{t-(n-1), \ldots, w_{t-1}})} \tag{2.25}$$

To make it simpler, let the history $w_{t-(n-1), \ldots, w_{t-1}}$ be h, the probability of a word x given a history h is calculated as follows:

$$p(w \mid h) = \frac{\text{count}(h, w)}{\text{count}(h)}, \tag{2.26}$$

where count(w, h) and count(h) are the concurrences in the training set of $h \cdot w$ and h respectively.

As you might know, the n-gram probabilities are not derived directly from the frequency counts, this estimation gives a zero probability for all unseen events. This problem is solved by smoothing the model. That is, modifying the original probability distribution in order to obtain similar distribution but without zero probabilities. Various methods are used, from simple “add-one” smoothing (assign a count of 1 to unseen n-grams) to more sophisticated models, such as Good-Turing discounting, interpolation, or back-off models. In the interpolation method, probabilities are smoothed as follows:

$$\tilde{p}(x \mid h) = \lambda_{x,h} p(x \mid h) + B_h \beta(x \mid \bar{h}), \tag{2.27}$$

where $\tilde{p}(x \mid h)$ denotes the smoothed probability, $\lambda_{x,h}$ is a factor used to discount mass probability from the original distribution, B_h is the total amount of discounted probability given history h, that is:

$$B_h = 1 - \sum_{x} \lambda_{x,h} p(x \mid h), \tag{2.28}$$

and finally $\beta(x \mid \bar{h})$ is a more simpler smoothed language model, which is usually a ***(n-1)***-gram model [43].

In the back-off method, discounted probability distributed over non seen events unlike in the interpolation method where it is distributed over all events, that is:

$$\tilde{p}(x \mid h) = \begin{cases} \lambda_{x,h} p(x \mid h) & N(x,h) \neq 0 \\ B_h \hat{\beta}(x \mid \bar{h}) & N(x,h) = 0 \end{cases}, \tag{2.29}$$

where $\hat{\beta}(x \mid \bar{h})$ is the distribution $\beta(x \mid \bar{h})$ normalized over all unseen events, where,

$$\hat{\beta}(x \mid \bar{h}) = \frac{\beta(x \mid \bar{h})}{\sum_{x:N(x,h)=0} \beta(x \mid \bar{h})} \tag{2.30}$$

Regardless of the method used to smooth the *n*-gram models, in all cases, the parameters $\lambda_{x,h}$ are needed. There are several techniques in order to calculate them, which are usually called discounting techniques. The most successful techniques are those based on the *Good-Turing* discounting [44, 45], such as the *Kneser-Ney* discounting [46]. An example of applying those discounting techniques, the equation of the *Good-Turing* discounting can be written as follows:

$$\lambda_{x,h} = \frac{n_{r+1}(r+1)}{rn_r}, \tag{2.31}$$

where $count(x,h) = r$ and n_r are the number of events that have been appeared r times in the training set. For more information about the discount techniques, please refer to [20].

2.5 Evaluation Metrics

In this section, the most used evaluation metrics in classification, recognition, and translation tasks.

2.5.1 Classification Error

Classification error E is a metric used in pattern recognition in classification tasks. It depends on the number of samples incorrectly classified and is calculated by the formula:

$$E = \frac{f}{n} \cdot 100 , \tag{2.32}$$

where f is the number of samples incorrectly classified, and n is the total number of samples. This metric is usually used in the isolated word recognition task, in which each transcribed word is treated as a class, and is classified as true or false. This metric is also used in some continuous text recognition tasks, in which a sequence of words is transcribed. However, it is somewhat a strict metric, since one erroneous word within the sequence implies that the sequence is considered completely wrong.

2.5.2 Word and Character Error Rate

Word Error Rate (WER) is a common metric that is used to measure the performance of a text recognition or speech recognition system. It measures the error in a recognized word sequence compared to a reference word sequence. The way this metric works is by calculating the minimum number of insertions, deletions, and substitutions needed to align the recognized word sequence with the reference word sequence. That is, the WER is calculated as follows:

$$\text{WER} = \frac{I + D + S}{N} \cdot 100\,, \tag{2.33}$$

where I is the number of insertions, D is the number of deletions, S is the number of substitutions, and N is the total number of words in the reference. I, D and S can be easily obtained computing the *Levenshtein distance* between the reference sentence and the recognized sentence.

In some systems, Word accuracy term is used instead of WER, which is basically used to measure the accuracy of a recognized word sequence (opposite of WER). It can be calculated as follows:

$$\text{W}_{accuracy} = 1 - \text{WER} = \frac{N - I - D - S}{N} \cdot 100 \tag{2.34}$$

In some sense, classification error is considered a particular case of WER in the case of having only one word in a the recognized word sequence which is aligned to one word in the reference word sequence. On the other hand, Character Error Rate (CER) is also a common metric, which is very similar to WER with only one difference. CER is performed on a character sequence instead of a word sequence.

It is worth noting that WER can be greater than 100% and thus, the word accuracy can be smaller than 0%. This might happen if the number of recognized word sequence is greater than the reference word sequence, and when the algorithm completely fails to align them with each other. This case is also applied on CER.

2.5.3 Translation Error Rate

Translation Error Rate (TER) is an error metric for machine translation systems that measures the number of edits (insertions, deletions, and substitutions) required to change a machine translation system output into the reference. In another way, It is similar to WER in the way of functionality and it is calculated as follows:

$$\text{TER} = \frac{I + D + S}{N} \cdot 100 \ , \tag{2.35}$$

where I is the number of insertions, D is the number of deletions, S is the number of substitutions, and N is the total number of words in the reference. For more details, please review Section 2.5.2.

2.5.4 Bilingual Evaluation Understudy

The Bilingual Evaluation Understudy (BLEU) [47] is one of the first used metrics in machine translation to measure the correspondence between a machine's output and that of a human. Scores are calculated for individual translated sentences by comparing them with a set of good quality reference translations. Those scores are then averaged over the whole corpus to reach an estimate of the translation's overall quality. Thus, it performs badly if used to evaluate the quality of individual sentences. That is, the geometric mean of the modified precision for different order of n-grams, p_n, (usually from unigram up to 4-grams) is calculated between the target sentence and the reference translation, which is multiplied by an exponential brevity penalty BP factor that penalizes those translations that are shorter than the reference translation. The modified precision score, p_n, is calculated for each n-gram length by summing over the matches for every hypothesis sentence S in the complete corpus C as:

$$p_n = \frac{\sum_{S \in C} \sum_{n\text{-gram} \in S} \text{count}_{\text{matched}}(n\text{-gram})}{\sum_{S \in C} \sum_{n\text{-gram} \in S} \text{count}(n\text{-gram})} \tag{2.36}$$

Each p_n is combined and can be weighted by specifying a weight w_n [48]. On the other hand, BP is computed as follows:

$$\text{BP} = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \leq r \ , \end{cases} \tag{2.37}$$

where c is the length of the corpus of hypothesis translations, and r is the effective reference corpus length. Then, the BLEU score is calculated as follows:

$$\text{BLEU} = \text{BP} \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right) \tag{2.38}$$

The BLEU ranges from 0.0 (worst case) to 1.0 (best case), however, it is a common practice referred as a percentage ranging from 0 (worst score) to 100 (best score) [20].

Bibliography

[1] Richard buse Zhi Qiang Liu, Jinhai Cai. *Handwriting recognition: soft computing and probabilistic approaches*. Springer, 2003.

[2] L.M. Lorigo and V. Govindaraju. Offline Arabic Handwriting Recognition: A Survey. *PAMI*, 28(5):712–724, May 2006.

[3] Seong-Whan Lee. *Advances in Handwriting Recognition*. World Scientific Publishing Co. Pte. Ltd., 1999.

[4] Dan Claudiu Ciresan, Ueli Meier, Luca Maria Gambardella, and Jürgen Schmidhuber. Deep Big Simple Neural Nets Excel on Handwritten Digit Recognition. *CoRR*, abs/1003.0358, 2010.

[5] Lawrence Rabiner and Biing-Hwang Juang. *Fundamentals of speech recognition*. Prentice-Hall, 1993.

[6] Réjean Plamondon and Sargur N. Srihari. On-Line and Off-Line Handwriting Recognition: A Comprehensive Survey. *IEEE Trans. on PAMI*, 22(1):63–84, 2000.

[7] A. H. Toselli, A. Juan, D. Keysers, J. González, I. Salvador, H. Ney, E. Vidal, and F. Casacuberta. Integrated Handwriting Recognition and Interpretation using Finite-State Models. *International Journal of Pattern Recognition and Artificial Intelligence*, 18(4):519–539, 2004.

[8] 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*, 37:2069–2079, 2004.

[9] Hanhong Xue and Venu Govindaraju. Hidden Markov Models Combining Discrete Symbols and Continuous Attributes in Handwriting Recognition. *IEEE Trans. on PAMI*, 28:458–462, 2006.

[10] V. Märgner and H. E. Abed. ICDAR 2007 - Arabic Handwriting Recognition Competition. In *Proc. of the 9th Int. Conf. on Document Analysis and Recognition (ICDAR 2007)*, volume 2, pages 1274–1278, Curitiba (Brazil), sep 2007.

[11] V. Märgner and H. E. Abed. ICDAR 2009 Arabic Handwriting Recognition Competition. In *Proc. of the 10th Int. Conf. on Document Analysis and Recognition (ICDAR 2009)*, pages 1383–1387, Barcelona (Spain), 7 2009.

[12] Frederick Jelinek. *Statistical methods for speech recognition*. Cambridge, MA: The MIT Press (Language, speech, and communication series), 1997.

[13] M. Pastor. *Aportaciones al reconocimiento automático de texto manuscrito*. PhD thesis, Dep. de Sistemes Informàtics i Computació, València, Spain, Oct 2007.

[14] Adrià Giménez Pastor. *Bernoulli HMMs for Handwritten Text Recognition*. PhD thesis, Universitat Politècnica de Valcència, Valencia, (Spain), May 2014.

[15] Nobuyuki Otsu. A Threshold Selection Method from Gray-Level Histograms. *IEEE Trans. on Systems, Man and Cybernetics*, 9:62–66, 1979.

[16] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.

[17] C. F. Jeff Wu. On the convergence properties of the em algorithm. *The Annals of Statistics*, 11(1):95–103, 1983.

[18] P. F. Brown, J. Cocke, S. A. D. Pietra, V. J. D. Pietra, F. Jelinek, J. D. Lafferty, R. L. Mercer, and P. S. Roossin. A statistical approach to machine translation. *Computational linguistics*, 16(2):79–85, 1990.

[19] Josep M. Crego Clemente. *Architecture and Modeling for N-gram-based Statistical Machine Translation*. PhD thesis, Universitat Politècnica de Catalunya, 2008.

[20] Jesús Andrés-Ferrer. *Statistical approaches for natural language modelling and monotone statistical machine translation*. PhD thesis, Universitat Politècnica de València, Valencia (Spain), Feb 2010. Advisors: A. Juan and F. Casacuberta.

[21] Franz Josef Och and Hermann Ney. Discriminative training and maximum entropy models for statistical machine translation. In *The 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 295–302, Philadelphia, 7 2002.

[22] F. J. Och and H. Ney. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(4):417–449, 2004.

[23] A. L. Berger, V. J. D. Pietra, and S. A. D. Pietra. A maximum entropy approach to natural language processing. *Computational linguistics*, 22(1):39–71, 1996.

[24] Lluís Padró and Evgeny Stanilovsky. Freeling 3.0: Towards wider multilinguality. In *Proc. of the Language Resources and Evaluation Conference (LREC 2012)*, Istanbul, Turkey, May 2012. ELRA.

[25] Owen Rambow Nizar Habash and Ryan Roth. Mada+tokan: A toolkit for arabic tokenization, diacritization, morphological disambiguation, pos tagging, stemming and lemmatization. In Khalid Choukri and Bente Maegaard, editors, *Proc. of the 2nd Int. Conf. on Arabic Language Resources and Tools*, Cairo, Egypt, April 2009. The MEDAR Consortium.

[26] Franz Josef Och and Hermann Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, 2003.

[27] Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. The mathematics of statistical machine translation: Parameter estimation. *Comput. Linguist.*, 19(2):263–311, 6 1993.

[28] Qin Gao and Stephan Vogel. Parallel implementations of word alignment tool. In *In Proc. of the ACL 2008 Software Engineering, Testing, and Quality Assurance Workshop*, 2008.

[29] Philipp Koehn, Franz Josef Och, and Daniel Marcu. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 48–54, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

[30] Daniel Marcu and William Wong. A phrase-based, joint probability model for statistical machine translation. In *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing - Volume 10*, EMNLP '02, pages 133–139, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

[31] Ashish Venugopal, Stephan Vogel, and Alex Waibel. Effective phrase translation extraction from alignment models. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pages 319–326, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

[32] Christoph Tillmann. A projection extension algorithm for statistical machine translation. In Michael Collins and Mark Steedman, editors, *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pages 1–8, 2003.

[33] Richard Zens, Franz Josef Och, and Hermann Ney. Phrase-based statistical machine translation. pages 18–32. Springer Verlag, 2002.

[34] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, and R. Zens. Moses: Open source toolkit for statistical machine translation. In *Annual meeting-association for computational linguistics*, volume 45, page 2, 2007.

[35] J.-C. Chappelier, M. Rajman, and Ch-Lausanne. A generalized cyk algorithm for parsing stochastic cfg. 1998.

[36] D. Vilar, D. Stein, M. Huck, and H. Ney. Jane: Open source hierarchical translation, extended with reordering and lexicon models. In *Proc. of the Joint 5th Workshop on Statistical Machine Translation and MetricsMATR*, pages 262–270, 2010.

[37] Jorge González and Francisco Casacuberta. Great: A finite-state machine translation toolkit implementing a grammatical inference approach for

transducer inference (giati). In *Proc. of the EACL 2009 Workshop on Computational Linguistic Aspects of Grammatical Inference*, CLAGI '09, pages 24–32, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[38] Josep M. Crego, François Yvon, and José B. Mariño. Ncode: an open source bilingual n-gram SMT toolkit. *The Prague Bulletin of Mathematical Linguistics*, 96(-1):49–58, 10 2011.

[39] J. B. Marino, R. E. Banchs, J. M. Crego, A. de Gispert, P. Lambert, J. A. R. Fonollosa, and M. R. Costa-jussà. N-gram-based machine translation. *Computational Linguistics*, 32(4):527–549, 2006.

[40] Josep Maria Crego and José B. Mariño. Improving statistical MT by coupling reordering and decoding. *Machine Translation*, 20(3):199–215, 7 2007.

[41] Verónica Romero Gómez. Multimodal Interactive Transcription of Handwritten Text Images, jun 2010.

[42] Joshua T. Goodman. A bit of progress in language modeling. Technical report, 2001.

[43] Stanley F. Chen and Joshua Goodman. An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, pages 310–318, Morristown, NJ, USA, 1996. Association for Computational Linguistics.

[44] I. J. Good. The population frequencies of species and the estimation of population parameters. *Biometrika*, 40(3-4):237–264, 1953.

[45] A. Nadas. On Turing's formula for word probabilities. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 33(6):1414–1416, dec. 1985.

[46] R. Kneser and H. Ney. Improved backing-off for M-gram language modeling. volume 1, pages 181–184, may. 1995.

[47] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: A method for automatic evaluation of machine translation. In *Proc. of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

[48] Chris Callison-Burch, Miles Osborne, and Philipp Koehn. Re-evaluating the role of bleu in machine translation research. In *In EACL*, pages 249–256, 2006.

CHAPTER 3

Arabic Handwriting Text Recognition

Contents

3.1 Introduction

Hidden Markov Models (HMMs) are widely used in off-line handwritten recognition [1–5]. Given a text (line or word) image, it is firstly transformed into a sequence of fixed-dimension feature vectors, and then fed into an HMM-based decoder to find its most probable transcription.

In principle, each word can be modeled by its own HMM, with no parameters in common with the HMMs associated with other classes. However, this approach becomes impractical for large vocabularies due to lack of training data for infrequent words, which results in poorly estimated HMM parameters and degraded classifier performance. Following the conventional approach in speech recognition [6], from where the HMM methodology was imported, HMMs at global (line or word) level are built from shared, *embedded* HMMs at character (subword) level. In this way, each training text image contributes to the estimation of its constituent character HMMs, all character HMMs are reliably estimated, and infrequent words are better modeled.

HMMs at character level are usually simple in terms of number of states and topology; e.g., 6 states and a linear topology in which each state can only be reached from its preceding state or itself (loop). On the other hand, state-conditional probability (density) functions depend on the type of output that has to be emitted. In the common case of real-valued feature vectors, Gaussian mixtures are generally preferred since, as with finite mixture models in general, their complexity can be adjusted to the available training data by simply varying the number of mixture components. Another good reason for their use is the availability of reliable software from the speech recognition community [7].

After decades of research in speech recognition, the use of certain real-valued speech features and embedded Gaussian mixture HMMs is a de-facto standard [6]. However, in the case of handwritten word recognition, there is no such a standard and, indeed, very different sets of features are in use today. In [8], columns of raw, binary image pixels are directly fed into *embedded Bernoulli mixture HMMs*, that is, embedded HMMs in which the emission probabilities are modeled with Bernoulli mixtures. The basic idea is to ensure that no discriminative information is filtered out during feature extraction, which in some sense is integrated into the recognition model. Empirical results were obtained for Latin script with the well-known IAM database [9] and, despite being much simpler, Bernoulli mixtures achieved error rates similar to those of Gaussian mixtures.

Although embedded Bernoulli mixture HMMs provide good results for Latin script, it is unclear whether these good results also apply to very different scripts such as Arabic. In this chapter, we extend the empirical results reported in [8] to Arabic handwriting. In addition, this basic approach is improved by using a sliding window of adequate width to better capture image context at each horizontal position of the text image. This improvement, is referred to as *windowed BHMMs*. However, the windowed approach, might be limited in dealing with vertical image distortions. In order to circumvent this limitation, we have considered new, adaptive window sampling techniques, as opposed to the conventional, direct strategy in which the sampling window center is applied at a constant height of the text image and moved horizontally one pixel at a time. More precisely, these adaptive techniques can be seen as an application of the direct strategy followed by a *repositioning* step by which the sampling window is repositioned to align its center to the center of gravity of the sampled image. This repositioning step can be done horizontally, vertically or in both directions. Although vertical repositioning is expected to have more influence on recognition results than horizontal repositioning, we have studied both separately and in conjunction, so as to confirm this expectation.

These techniques described above are introduced and extensively tested on two databases for Arabic handwritten text. More precisely, they are tested on the very popular IfN/ENIT database of Arabic handwritten Tunisian town names [10], and the NIST OpenHaRT 2010 databases [11]. Our results are compared with state-of-the-art results on Arabic handwriting recognition, also obtained from the IfN/ENIT and NIST OpenHaRT'10 databases during many international competitions [3, 4, 11–13].

In what follows, we describe Bernoulli mixtures (Section 3.2), Bernoulli HMMs (Section 3.3), BHMM-based handwriting recognition (Section 3.4), maximum likelihood parameter estimation (Section 3.5), windowed BHMMs (Section 3.6), and finally the repositioning approach (Section 3.7). Results are reported in Sections 3.8 and 3.9 for IfN/ENIT and OpenHaRT'10 respectively. Concluding remarks are given in Section 3.10.

3.2 Bernoulli Mixture

Let $\mathbf{o}$ be a D-dimensional feature vector. A finite mixture is a probability (density) function of the form:

$$P(\mathbf{o} \mid \boldsymbol{\Theta}) = \sum_{k=1}^{K} \pi_k \, P(\mathbf{o} \mid k, \boldsymbol{\Theta}') \ , \tag{3.1}$$

where K is the number of mixture components, π_k is the k^{th} component coefficient, and $P(\mathbf{o} \mid k, \boldsymbol{\Theta}')$ is the k^{th} component-conditional probability (density) function. The mixture is controlled by a parameter vector $\boldsymbol{\Theta}$ comprising the mixture coefficients and a parameter vector for the components, $\boldsymbol{\Theta}'$. It can be seen as a generative model that first selects the kth component with probability π_k and then generates $\mathbf{o}$ in accordance with $P(\mathbf{o} \mid k, \boldsymbol{\Theta}')$.

A Bernoulli mixture model is a particular case of (3.1) [14] in which each component k has a D-dimensional Bernoulli probability function governed by its own vector of parameters

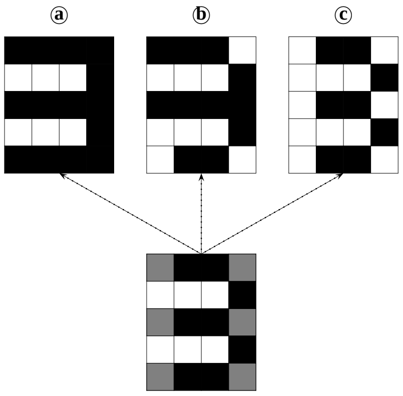


Figure 3.1: Three binary images (**a, b** and **c**) are shown as being generated from a Bernoulli prototype depicted as a gray image (black=1, white=0, gray=0.5).

or *prototype* $\mathbf{p}_k = (p_{k1}, \ldots, p_{kD})^t \in [0,1]^D$,

$$P(\mathbf{o} \mid k, \boldsymbol{\Theta}') = \prod_{d=1}^{D} p_{kd}^{o_d} (1 - p_{kd})^{1-o_d} \ , \tag{3.2}$$

where p_{kd} is the probability for bit d to be 1. Note that this equation is just the product of independent, unidimensional Bernoulli probability functions. Therefore, for a fixed k, it can not capture any kind of dependencies or correlations between individual bits.

Consider the example given in Figure 3.1. Three binary images (**a, b** and **c**) are shown as being generated from a Bernoulli prototype depicted as a gray image (black=1, white=0, gray=0.5). The prototype has been obtained by averaging images **a** and **c,** and it is the best approximate solution to assign a high, equal probability to these images. However, as individual pixel probabilities are not conditioned to other pixel values, there are $2^6 = 64$ different binary images (including **a, b** and **c**) into which the whole probability mass is uniformly distributed. It is then not possible, using a single Bernoulli prototype, to assign a probability of 0.5 to **a** and **c,** and null probability to any other image such as **b.** Nevertheless, this limitation can be easily overcome by using a Bernoulli mixture and allowing a different prototype to each different image shape. That is, in our example, a two-component mixture of equal coefficients, and prototypes **a** and **b,** does the job.

3.3 Bernoulli HMM

Let $O = (\mathbf{o}_1, \ldots, \mathbf{o}_T)$ be a sequence of feature vectors. An HMM is a probability (density) function of the form:

$$P(O \mid \mathbf{\Theta}) = \sum_{q_0,\ldots,q_{T+1}} \prod_{t=0}^{T} a_{q_t q_{t+1}} \prod_{t=1}^{T} b_{q_t}(o_t) \,, \tag{3.3}$$

where the sum is over all possible *paths* (state sequences) $q_0, \ldots, q_{T+1}$, such that $q_0 = I$ (special *initial* or *start* state), $q_{T+1} = F$ (special *final* or *stop* state), and $q_1, \ldots, q_T \in \{1, \ldots, M\}$, being M the number of regular (non-special) states of the HMM. On the other hand, for any regular states i and j, a_{ij} denotes the *transition* probability from i to j, while b_j is the *observation* probability (density) function at j.

A Bernoulli (mixture) HMM (BHMM) is an HMM in which the probability of observing $\mathbf{o}_t$, when $q_t = j$, is given by a Bernoulli mixture probability function for the state j [15]:

$$b_j(\mathbf{o}_t) = \sum_{k=1}^{K} \pi_{jk} \prod_{d=1}^{D} p_{jkd}^{o_{td}} (1 - p_{jkd})^{1-o_{td}} \,, \tag{3.4}$$

where π_{jk} and $\mathbf{p}_{jk}$ are, respectively, the prior and prototype of the kth mixture component in state j.

Consider the upper part of Figure 3.2, where a BHMM example for the number 3 is shown, together with a binary image generated from it. It is a three-state model with single prototypes attached to states 1 and 2, and a two-component mixture assigned to state 3. In contrast to the example in Figure 3.1, prototypes do not account for the whole digit realizations, but only for single columns. This column-by-column emission of feature vectors attempts to better model horizontal distortions at character level and, indeed, it is the usual approach in both speech and handwriting recognition when continuous-density (Gaussian mixture) HMMs are used. The reader can check that, by direct application of Eq. (3.3) and taking into account the existence of two different state sequences, the probability of generating the binary image generated from this BHMM example is 0.063.

As discussed in the introduction, BHMMs at global (line or word) level are built from shared, embedded BHMMs at character level. More precisely, let C be the number of different characters (symbols) from which global BHMMs are built, and assume that each character c is modeled with a different BHMM of parameter vector $\mathbf{\Theta}_c$. Let $\mathbf{\Theta} = \{\mathbf{\Theta}_1, \ldots, \mathbf{\Theta}_C\}$, and let $O = (\mathbf{o}_1, \ldots, \mathbf{o}_T)$ be a sequence of feature vectors generated from a sequence of symbols $S = (s_1, \ldots, s_L)$, with $L \leq T$. The probability of O can be calculated, using embedded

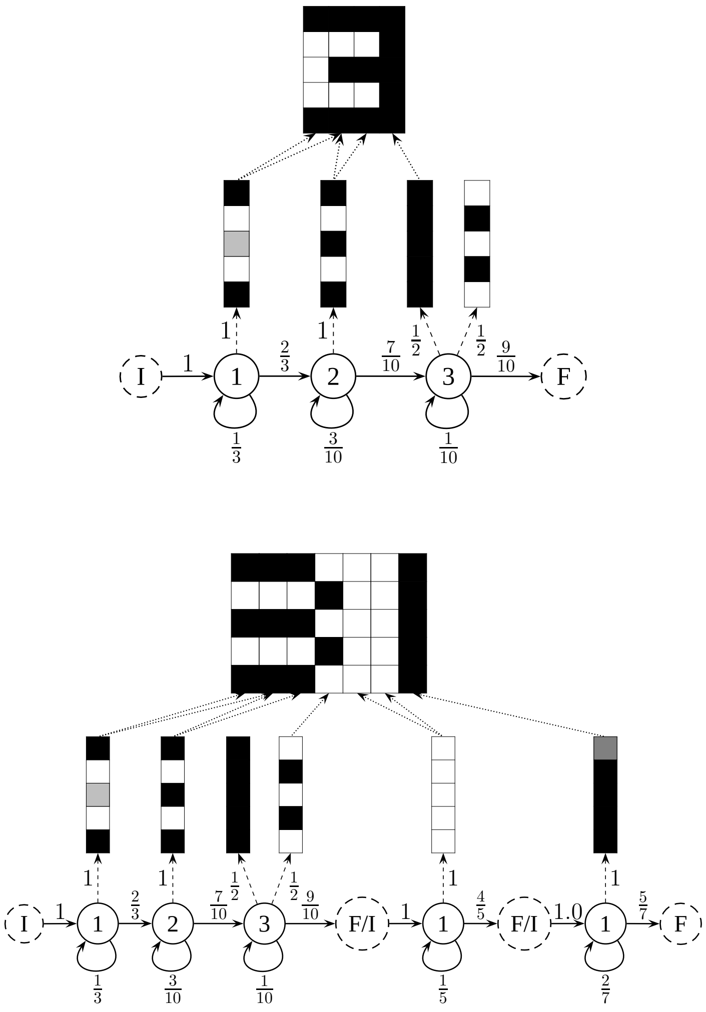


Figure 3.2: BHMM examples for the numbers 3 (top) and 31 (bottom), together with binary images generated from them. Note that the BHMM example for the number 3 is also embedded into that for the number 31. Bernoulli prototype probabilities are represented using the following color scheme: black=1, white=0,gray=0.5 and light gray=0.1.

HMMs for its symbols, as:

$$P(O \mid S, \boldsymbol{\Theta}) = \sum_{i_1,\ldots,i_{L+1}} \prod_{l=1}^{L} P(\mathbf{o}_{i_l}, \ldots, \mathbf{o}_{i_{l+1}-1} \mid \boldsymbol{\Theta}_{s_l}) \,, \tag{3.5}$$

where the sum is carried out over all possible segmentations of O into L segments, that is, all sequences of indices $i_1, \ldots, i_{L+1}$ such that

$$1 = i_1 < \cdots < i_L < i_{L+1} = T + 1;$$

and $P(\mathbf{o}_{i_l}, \ldots, \mathbf{o}_{i_{l+1}-1} \mid \mathbf{\Theta}_{s_l})$ refers to the probability (density) of the lth segment, as given by (3.3) using the HMM associated with symbol s_l.

Consider now the lower part of Figure 3.2. An embedded BHMM for the number 31 is shown, which is the result of concatenating BHMMs for the digit 3, blank space and digit 1, in that order. Note that the BHMMs for blank space and digit 1 are simpler than that for digit 3. Also note that the BHMM for digit 3 is shared between the two embedded BHMMs shown in the Figure. The binary image of the number 31 shown above can only be generated from two paths, as indicated by the arrows connecting prototypes to image columns, which only differ in the state generating the second image column (either state 1 or 2 of the BHMM for the first symbol). It is straightforward to check that, according to (3.5), the probability of generating this image is 0.0004.

3.4 BHMM-based Handwriting Recognition

Given an observation sequence $O = (\mathbf{o}_1, \ldots, \mathbf{o}_T)$, its most probable transcription is obtained by application of the conventional Bayes decision rule:

$$\begin{aligned} w^* &= \operatorname*{argmax}_{w \in W} \; p(w \mid O) && (3.6) \\ &= \operatorname*{argmax}_{w \in W} \; p(w)\, p(O \mid w)\,, && (3.7) \end{aligned}$$

where W is the set of possible transcriptions; $p(w)$ is usually approximated by an *n-gram language model* [16]; and $p(O \mid w)$ is a *text image model* which is modeled as a BHMM (built from shared, embedded BHMMs at character level), as defined in Eq. (3.5). A particularly interesting case arises when the set of possible transcriptions reduces to a (small) finite set of *words (class labels)*. In this case, $p(w)$ is simply the *prior* probability of word w, while $p(O \mid w)$ is the probability of observing O given that it corresponds to a handwritten version of word w.

3.4.1 The forward algorithm

In order to efficiently compute $p(O \mid w)$ as a BHMM probability of the form given in Eq. (3.5), we use a dynamic programming method known as *forward algorithm* [6, 7]. For each time t, symbol s_l and state j from the HMM for symbol s_l, we define the *forward probability* $\alpha_{lt}(j)$ as:

$$\alpha_{lt}(j) = P(O_1^t, q_t = (l, j) \mid S, \mathbf{\Theta}) \,, \tag{3.8}$$

that is, the probability of generating O up to its tth element and ending at state j from the HMM for symbol s_l. This definition includes (3.5) as the particular case in which $t = T$, $l = L$ and $j = F_{s_L}$; that is,

$$P(O \mid S, \mathbf{\Theta}) = \alpha_{LT(F_{s_L})} \tag{3.9}$$

To compute $\alpha_{LT(F_{s_L})}$, we must first take into account that, for each position l in S except for the first, the initial state of the HMM for s_l is joined with final state of its preceding HMM, i.e.

$$\alpha_{lt}(I_{s_l}) = \alpha_{l-1t}(F_{s_{l-1}}) \qquad \begin{matrix} 1 < l \leq L \\ 1 \leq t \leq T \end{matrix} \tag{3.10}$$

Having (3.10) in mind, we can proceed at symbol level as with conventional HMMs. For the final states, we have:

$$\alpha_{lt}(F_{s_l}) = \sum_{j=1}^{M_{s_l}} \alpha_{lt}(j) \, a_{s_l j F_{s_l}} \qquad \begin{matrix} 1 \leq l \leq L \\ 1 \leq t \leq T \end{matrix} \ , \tag{3.11}$$

while, for regular states, $1 \leq j \leq M_{s_l}$, we have:

$$\alpha_{lt}(j) = \left[\sum_{i \in \{I_{s_l}, 1, \ldots, M_{s_l}\}} \alpha_{lt-1}(i) \, a_{s_l i j} \right] b_{s_l j}(\mathbf{o}_t) \,, \tag{3.12}$$

with $1 \leq l \leq L$ and $1 < t \leq T$. The base case is for $t = 1$:

$$\alpha_{l1}(i) = \begin{cases} a_{s_1 I_{s_1} i} \; b_{s_1 i}(\mathbf{o}_1) & l = 1, 1 \leq i \leq M_{s_1} \\ 0 & \text{otherwise} \end{cases} \tag{3.13}$$

The forward algorithm uses a dynamic programming table for $\alpha_{lt}(\cdot)$ which is computed forward in time to avoid repeated computations.

Figure 3.3 shows an application example of the forward algorithm to the BHMM and observation of Figure 3.2 (bottom). Non-null (and a few null) entries of the dynamic programming table are represented by graph nodes aligned with states (vertically) and time (horizontally). Node borders are drawn in black or gray, depending on whether they are in valid paths (i.e. those from which the observation sequence can be generated) or not. Also, those associated with special states are drawn with dotted lines. Numbers at the top of each node refer to $\alpha_{lt}(\cdot)$ and thus, for instance, the probability of generating O up to the third image column and ending at state 2 of the BHMM for the first symbol is $\alpha_{13}(2) = \frac{10}{450}$. Computation dependencies between nodes are represented by arrows, which are labeled above by, first, the transition probability, and then the observation probability at the target state (see Eq. (3.4)). For instance, the numbers above the arrow pointing to node $\alpha_{13}(4)$ are: $a_{s_1 23} \cdot b_{s_1 3}(\mathbf{o}_4) = \frac{7}{10} \cdot (\frac{1}{2} \cdot 0 + \frac{1}{2} \cdot 1^5) = \frac{7}{10} \cdot \frac{1}{2}$.

From Figure 3.3, we can clearly see that, as indicated at the end of Section 3.3, there are only two paths from which the observation can be generated. They share all nodes drawn with black borders except the two nodes aligned with the second observation vector. In accordance with Eq. (3.9), the probability of the observation sequence is $\alpha_{37}(F) = 0.0004$.

3.4.2 The backward algorithm

The *backward algorithm* is similar to the forward algorithm but, as it name indicates, it uses a dynamic programming table which is instead computed backward in time [6, 7]. The basic definition in this case is the *backward probability:*

$$\beta_{lt}(j) = P\left(O_{t+1}^{T} \mid q_t = (l,j), S, \mathbf{\Theta}\right) \ , \tag{3.14}$$

which measures the probability (density) of generating O_{t+1}^{T} given that the tth vector was generated in state j of the BHMM for the symbol s_l. Using this definition, Eq. (3.5) can be rewritten as:

$$P(O \mid S, \mathbf{\Theta}) = \sum_{j=1}^{M_{s_1}} a_{s_1 I_{s_1} j} \; b_{s_1 j}(\mathbf{o}_1) \; \beta_{11}(j) \tag{3.15}$$

Taking into account that:

$$\beta_{lt}(F_{s_l}) = \beta_{l+1t}(I_{s_{l+1}}) \qquad \begin{array}{l} 1 \leq l < L \\ 1 \leq t < T \end{array} \ , \tag{3.16}$$

the backward probability for the initial and regular states, $i \in \{I_{s_l}, 1, \ldots, M_{s_l}\}$, can be efficiently computed as:

$$\beta_{lt}(i) = a_{s_{nl} i F_{s_l}} \beta_{lt}(F_{s_l}) + \sum_{j=1}^{M_{s_l}} a_{s_l ij} b_{s_l j}(\mathbf{o}_{t+1}) \beta_{lt+1}(j) \qquad \begin{array}{l} 1 \leq l \leq L \\ 1 \leq t < T \end{array} \ , \tag{3.17}$$

where the base case is defined for $t = T$ as

$$\beta_{lT}(i) = \begin{cases} a_{s_L i F_{s_L}} & l = L, 1 \leq i \leq M_{s_L} \\ 0 & \text{otherwise} \end{cases} \tag{3.18}$$

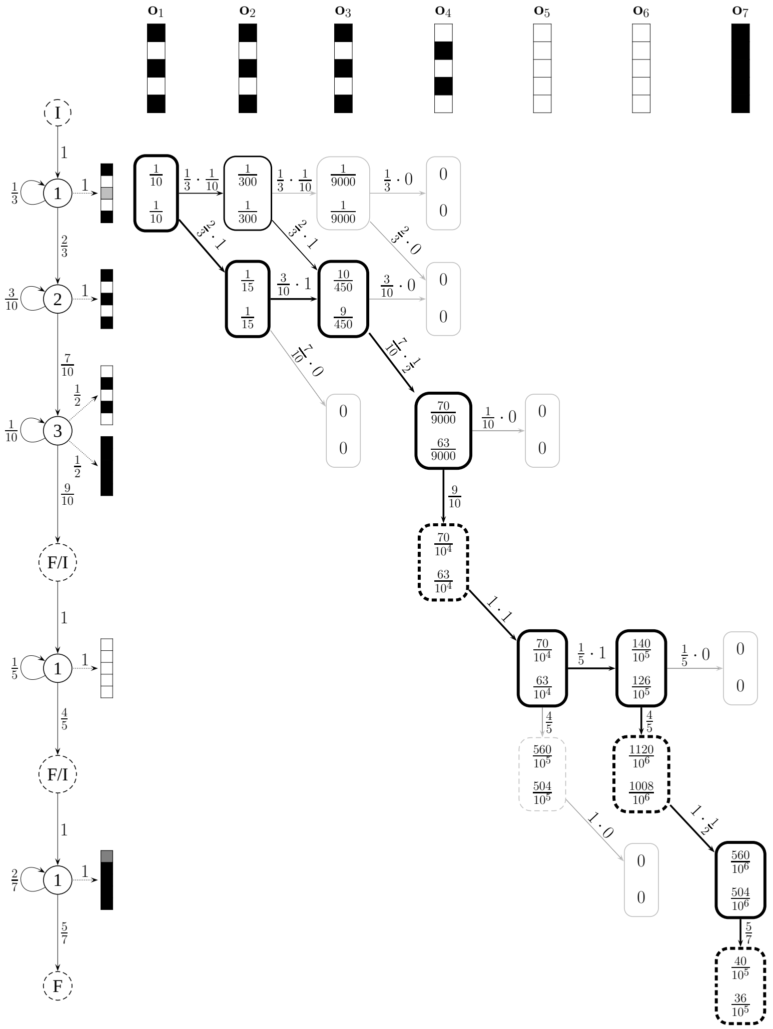


Figure 3.3: Application example of the forward and Viterbi algorithms to the BHMM and observation of Figure 3.2 (bottom). Numbers at the top of the nodes denote forward probabilities, while those at the bottom refer to Viterbi scores.

3.4.3 The Viterbi algorithm

Although the forward and backward algorithms efficiently compute the exact value of $P(O \mid S, \mathbf{\Theta})$, it is common practice to approximate it by the so-called *Viterbi* or *maximum approximation,* in which the sums in Eqs. (3.3) and (3.5) are replaced by the $\max$ operator, i.e.

$$P(O \mid S, \mathbf{\Theta}) \approx \max_{\substack{i_1, \ldots, i_{L+1} \\ q_1, \ldots, q_T}} \prod_{l=1}^{L} \hat{P}(\mathbf{o}_{i_l}^{i_{l+1}-1} \mid \mathbf{\Theta}_{s_l}), \qquad (3.19)$$

where the $\hat{P}$ is defined as:

$$\hat{P}(\mathbf{o}_{i_l}^{i_{l+1}-1} \mid \mathbf{\Theta}_{s_l}) = a_{s_l I_{s_l} q_{i_l}} \cdot \prod_{t=i_l}^{i_{l+1}-2} a_{s_l q_t q_{t+1}} \cdot a_{s_l q_{i_{l+1}-1} F_{s_l}} \cdot \prod_{t=i_l}^{i_{l+1}-1} b_{s_l q_t}(\mathbf{o}_t) \qquad (3.20)$$

In contrast to the exact definition, this approximation allows us to identify a single, best state sequence or *path* associated with the given observation sequence. The well-known *Viterbi algorithm* efficiently computes this approximation, using dynamic programming recurrences similar to those used by the forward algorithm. Formally, we need to compute the probability $Q(l, t, j)$ of the most likely path up to time t that ends with the state j from the BHMM for symbol s_l. For the specials states, it can be computed as:

$$Q(l, t, I_{s_l}) = Q(l-1, t, F_{s_{l-1}}) \qquad \begin{matrix} 1 < l \leq L \\ 1 \leq t \leq T \end{matrix} \qquad (3.21)$$

$$Q(l, t, F_{s_l}) = \max_{1 \leq j \leq M_{s_l}} Q(l, t, j)\, a_{s_l j F_{s_l}} \qquad \begin{matrix} 1 \leq l \leq L \\ 1 \leq t \leq T \end{matrix}, \qquad (3.22)$$

while, for the regular states with $1 \leq l \leq L$ and $1 < t \leq T$, we have:

$$Q(l, t, j) = \left[\max_{i \in \{I_{s_l}, 1, \ldots, M_{s_l}\}} Q(l, t-1, i)\, a_{s_l i j} \right] b_{s_l j}(\mathbf{o}_t), \qquad (3.23)$$

The base case is for $t = 1$:

$$Q(l, 1, i) = \begin{cases} a_{s_1 I_{s_1} i}\; b_{s_1 i}(\mathbf{o}_1) & l = 1, 1 \leq i \leq M_{s_1} \\ 0 & \text{otherwise} \end{cases} \qquad (3.24)$$

Clearly, the Viterbi algorithm can be seen as a minor modification of the forward algorithm in which only the most probable is considered in each node computation. Indeed, the application example shown in Figure 3.3 is used both, for the forward and Viterbi algorithms. Now, however, the relevant numbers are those included at the bottom of each node, which denote $Q(l,t,j)$; i.e., at row 2 and column 3, we have $Q(1,3,2)=\frac{9}{450}$. Consider the generation of the third observation vector at the second state (for the first symbol). It occurs after the generation of the second observation vector, either at the first or the second states, but we only take into account the most likely case. Formally, the corresponding Viterbi score is computed as:

$$Q(1,3,2)=\max\left\{\frac{1}{15}\cdot\frac{3}{10}\cdot 1,\frac{1}{300}\cdot\frac{2}{3}\cdot 1\right\}=\max\left\{\frac{9}{450},\frac{1}{450}\right\}=\frac{9}{450}$$

Note that forward probabilities do not differ from Viterbi scores up to $Q(1,3,2)$, since it corresponds to the first (and only) node with two incoming paths. The Viterbi approximation to the exact probability of generating the observation sequence is obtained at the final node: $Q(3,7,F)=0.00036$. The most likely path, drawn with thick lines, is retrieved by starting at this node and moving backwards in time in accordance with computation of Viterbi scores. As usual in practice, the final Viterbi score in this example (0.00036) is a tight lower bound of the exact probability (0.00040).

3.5 Maximum likelihood parameter estimation

Maximum likelihood estimation of the parameters governing an embedded BHMM does not differ significantly from the conventional Gaussian case, and it can be carried out using the well-known EM (Baum-Welch) re-estimation formulae [6, 7]. Let $(O_1,S_1),\ldots,(O_N,S_N)$, be a collection of N training samples in which the nth observation has length T_n, $O_n=(\mathbf{o}_{n1},\ldots,\mathbf{o}_{nT_n})$, and was generated from a sequence of L_n symbols ($L_n\leq T_n$), $S_n=(s_{n1},\ldots,s_{nL_n})$. At iteration r, the E step requires the computation, for each training sample n, of their corresponding forward and backward probabilities (see (3.8) and (3.14)), as well as the expected value for its tth feature vector to be generated from kth component of the state j in the HMM for symbol s_l,

$$z_{nltk}^{(r)}(j)=\frac{\pi_{s_{nl}jk}^{(r)}\prod_{d=1}^{D}p_{s_{nl}jkd}^{(r)}{}^{o_{ntd}}\left(1-p_{s_{nl}jkd}^{(r)}\right)^{1-o_{ntd}}}{b_{s_{nl}j}^{(r)}(\mathbf{o}_{nt})}\,,$$

for each t, k, j and l.

In the M step, the Bernoulli prototype corresponding to the kth component of the state j in the HMM for character c has to be updated as:

$$\mathbf{p}_{cjk}^{(r+1)}=\frac{1}{\gamma_{ck}(j)}\sum_{n}\frac{\sum_{l:s_{nl}=c}\sum_{t=1}^{T_n}\xi_{nltk}^{(r)}(j)\mathbf{o}_{nt}}{P\left(O_n\mid S_n,\mathbf{\Theta}^{(r)}\right)}\,,\qquad(3.25)$$

where $\gamma_{ck}(j)$ is a normalization factor,

$$\gamma_{ck}(j) = \sum_n \frac{\sum_{l:s_{nl}=c} \sum_{t=1}^{T_n} \xi_{nltk}^{(r)}(j)}{P\left(O_n \mid S_n, \mathbf{\Theta}^{(r)}\right)} , \tag{3.26}$$

and $\xi_{nltk}^{(r)}(j)$ the probability for the tth feature vector of the nth sample, to be generated from the kth component of the state j in the HMM for symbol s_l,

$$\xi_{nltk}^{(r)}(j) = \alpha_{nlt}^{(r)}(j) z_{nltk}^{(r)}(j) \beta_{nlt}^{(r)}(j) \tag{3.27}$$

Similarly, the kth component coefficient of the state j in the HMM for character c has to be updated as:

$$\pi_{cjk}^{(r+1)} = \frac{1}{\gamma_c(j)} \sum_n \frac{\sum_{l:s_{nl}=c} \sum_{t=1}^{T_n} \xi_{nltk}^{(r)}(j)}{P\left(O_n \mid S_n, \mathbf{\Theta}^{(r)}\right)} , \tag{3.28}$$

where $\gamma_c(j)$ is a normalization factor,

$$\gamma_c(j) = \sum_n \frac{\sum_{l:s_{nl}=c} \sum_{t=1}^{T_n} \alpha_{nlt}^{(r)}(j) \beta_{nlt}^{(r)}(j)}{P\left(O_n \mid S_n, \mathbf{\Theta}^{(r)}\right)} \tag{3.29}$$

To avoid null probabilities in Bernoulli prototypes, they can be smoothed by linear interpolation with a flat (uniform) prototype, $\mathbf{0.5}$,

$$\tilde{\mathbf{p}} = (1 - \delta)\, \mathbf{p} + \delta\, \mathbf{0.5} , \tag{3.30}$$

where, for instance, $\delta = 10^{-6}$.

3.6 Windowed BHMMs

Given a binary image normalized in height to H pixels, we may think of a feature vector $\mathbf{o}_t$ as its column at position t or, more generally, as a concatenation of columns in a window of W columns in width, centered at position t. This generalization has no effect neither on the definition of BHMM nor on its maximum likelihood estimation, though it might be very helpful to better capture image context at each horizontal position of the image. As an example, Figure 3.4 shows a binary image of 4 columns and 5 rows, which is transformed into a sequence of 4 15-dimensional feature vectors (first row) by application of a sliding window of width 3. For clarity, feature vectors are depicted as 3×5 subimages instead of 15-dimensional column vectors. Note that feature vectors at positions 2 and 3 would be indistinguishable if, as in our previous approach, they were extracted with no context ($W = 1$).

3.7 Repositioning approach

Although one-dimensional, “horizontal” HMMs for image modeling can properly capture non-linear horizontal image distortions, they are somewhat limited when dealing with vertical image distortions, and this limitation might be particularly strong in the case of feature vectors extracted with significant context. To overcome this limitation, we have considered three methods of window *repositioning* after window extraction: *vertical*, *horizontal*, and *both*. The basic idea is to first compute the compute the center of mass (previously discussed in Section 2.2.3) of the extracted window, which is then repositioned (translated) to align its center to the center of mass. This is done in accordance with the chosen method, that is, horizontally, vertically, or in both directions. Obviously, the feature vector actually extracted is that obtained after repositioning. An example of feature extraction is shown in Figure 3.4 in which the standard method (no repositioning) is compared with the three methods repositioning methods considered.

To illustrate the effect of repositioning with real data, Figure 3.5 shows the sequence of feature vectors extracted from a real sample of the IfN/ENIT database [10], with and without (both) repositioning. As intended, (vertical or both) repositioning has the effect of normalizing vertical image distortions, especially translations.

3.8 Experiments on IfN/ENIT database

Experiments in this section were carried out on the very popular IfN/ENIT database of Arabic handwritten Tunisian town names [10]. More precisely, we used the IfN/ENIT database in version 2.0, patch level 1e (v2.0p1e), which is exactly the version used as training data in the Arabic handwriting recognition competition held at ICDAR (Int. Conf. on Document Analysis and Recognition) in 2007 [3]. It comprises 32492 Arabic word images written by 411 different writers, from a lexicon of 937 Tunisian town/village names. More details about this data set are reported in Section 3.8.1.

3.8.1 IfN/ENIT database

IfN/ENIT database is an Arabic handwritten text database which contains handwritten Tunisian town/villages names [10]. It is a database for isolated word recognition. In last years this database has been used in several Arabic handwritten competitions, see [3, 4, 12, 13, 17], becoming a reference in the Arabic handwritten area. The database consists of 946 Tunisian town/villages. It is written by 411 writers. They were asked to fill 5 forms with 12 names from the possible names with their corresponding postcodes. Forms were made guarantying that each word appears at least 3 times in the database, and each character shape occur at minimum more than 200 times. The only aid to writing was the printing of dark light rectangles in the

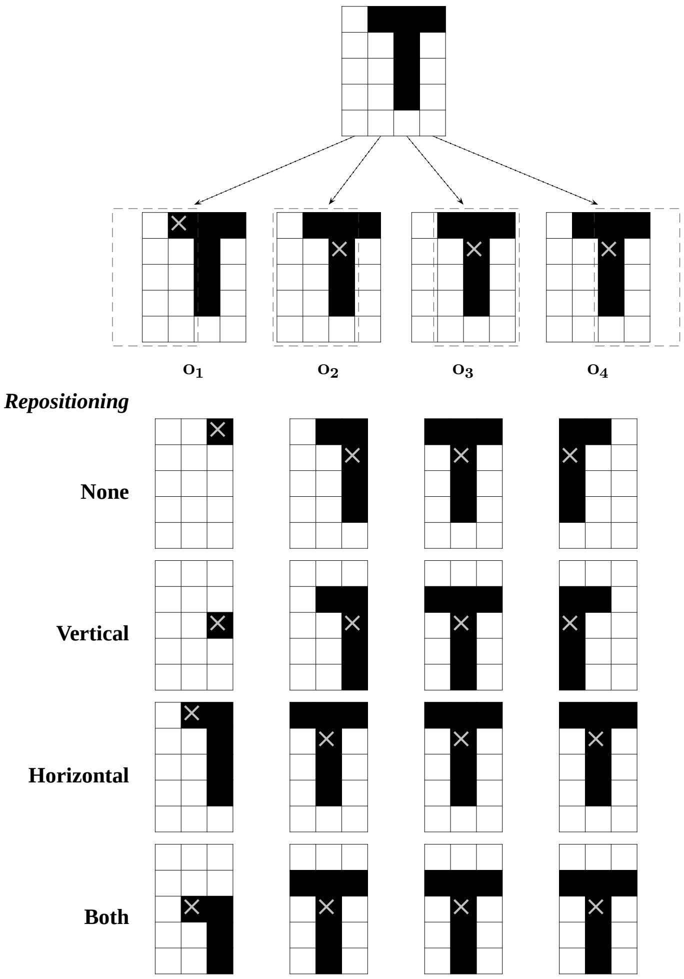


Figure 3.4: Example of transformation of a 4×5 binary image (top) into a sequence of 4 15-dimensional binary feature vectors $O = (\mathbf{o}_1, \mathbf{o}_2, \mathbf{o}_3, \mathbf{o}_4)$ using a window of width 3. After window extraction (illustrated under the original image), the standard method (no repositioning) is compared with the three repositioning methods considered: vertical, horizontal, and both directions. Mass centers of extracted windows are also indicated

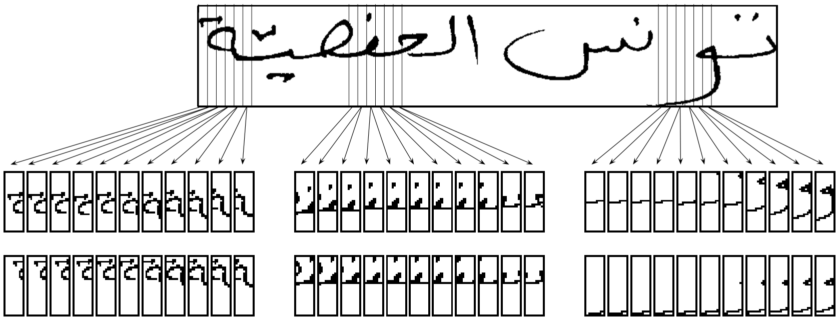


Figure 3.5: Original sample *pf069_011* from IFN/ENIT database (top) and its sequence of feature vectors produced with and without (both) repositioning (center and bottom, respectively)

backside of the form to indicate where to write the words. Figure 3.7 shows two examples of forms.

Forms were scanned with 300 dpi and, binarised and automatically segmented. Using a semi-automatically process segmented images were labeled with the postcode, the Arabic word in codes as “ISO 8859-6” with a sequence of Arabic character shapes from 306 different shapes, since each letter appears in four different forms depending of its position in the word (Begin, Middle, End or Isolated form). It is important to note that “ISO 8859-6” does not encode the shape information.

The resulting database is composed by 32492 different images divided into 5 sets (a, b, c, d and e). The first four sets are the original sets of the database, while the set e was used as test set in the ICDAR 2005 competition see [17], being late released. Thus it is a common practice to public results doing a cross validation experiment with the first four sets, and a final experiment training with sets a, b, c, d and testing the set e. Note that while the number of classes is 946 (postcodes), the size of the lexicon is greater since names are written in different forms. Table 3.1 shows some statistics for the five sets, and Figure 3.6 shows some samples of images.

3.8.2 Preliminary Experiments

In this section we show the effect of the very basic approach on the IfN/ENIT database. That is, our first experiments were carried out using Bernoulli HMMs described in Section 3.3, without applying neither the sliding window nor the repositioning techniques. Each image was rescaled in height to a given dimension D (10, 15, 20, 25, 30, 35), while keeping the original aspect ratio, and then binarized. For binarization, we used the well-known Otsu

Table 3.1: Some statistics of the IfN/ENIT-database sets

	No. of words	Lexicon
a	6537	1588
b	6710	1634
c	6477	1498
d	6735	1564
e	6033	733

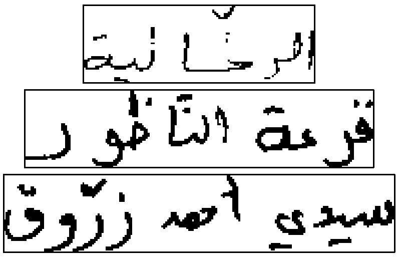


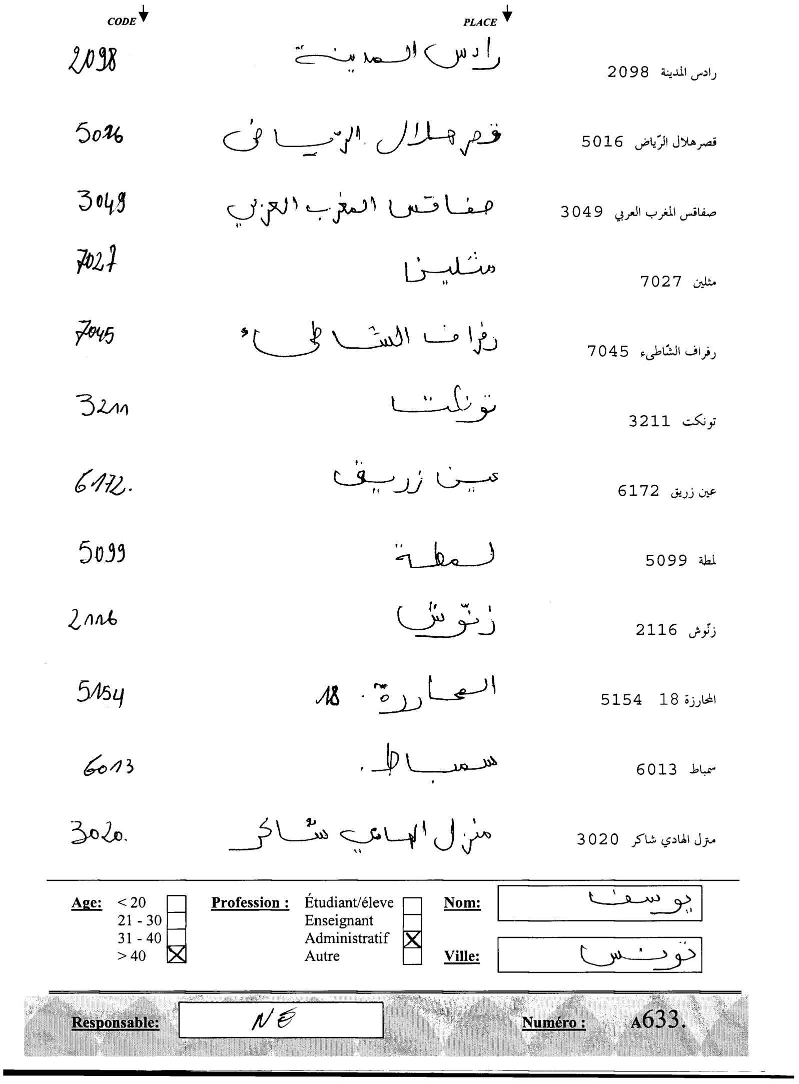
Figure 3.6: Three samples of one, two, and three word images taken from the IfN/ENIT database

algorithm [18], which is a simple and robust method for reasonable clean images. In the results reported below, however, only height 30 is considered since, in a series of preliminary informal tests, it led to better results than other heights.

Experiments were carried out by trying different number of states, $Q \in \{2, 4, 6, 8\}$, and also different number of mixture components per state, $K \in \{1, 4, 16, 64, 128\}$. For $K = 1$, the HMMs were initialized by first segmenting the training set with a “neutral” model, and then using the resulting segments to perform a Viterbi initialization. The initialized HMMs were trained with 4 EM iterations. For $K > 1$, the HMMs were initialized by splitting the mixture components of the models trained with $K/4$ (or $K/2$) mixture components per state. Again, after initialization, HMMs were trained with 4 EM iterations. On the other hand, recognition of test images was performed by using the Viterbi algorithm.

Figure 3.8 shows the Word Error Rate (WER%), as a function of the number of states, for varying number of components. Each WER estimate (plotted point) was obtained by cross-validation with the first 4 standard folds (a, b, c and d).

From the results in Fig. 3.8, it seems that an appropriate value for the number of states is 6, and also an appropriate value for the number of mixture components per state is 64. Using these values, two additional experiments were carried out by using the training-test partitions abcd-e and abcde-e. The resulting WER values are included in Table 3.2 together with



CODE	PLACE	
2098	رادس المدينة	رادس المدينة 2098
5016	قصرهلال الرّياض	قصرهلال الرّياض 5016
3049	صفاقس المغرب العربي	صفاقس المغرب العربي 3049
7027	مثلين	مثلين 7027
7045	رفراف الشّاطىء	رفراف الشّاطىء 7045
3211	تونكت	تونكت 3211
6172.	عين زريق	عين زريق 6172
5099	لمطة	لمطة 5099
2116	زنّوش	زنّوش 2116
5154	المحارزة 18	المحارزة 18 5154
6013	سمباط.	سمباط 6013
3020.	منزل الهادي شاكر	منزل الهادي شاكر 3020

Age: < 20 ☐ 21 - 30 ☐ 31 - 40 ☐ [✓ > 40]

Profession : Étudiant/éleve ☐ Enseignant ☐ [✓ Administratif] Autre ☐

Nom: يوسف

Ville: تونس

Responsable: NE

Numéro : A633.

Figure 3.7: Example of a handwritten form taken from the IfN/ENIT database

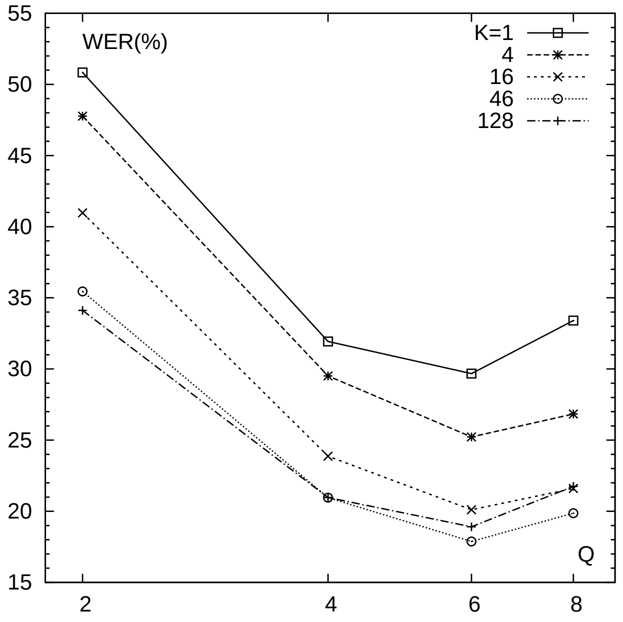


Figure 3.8: Word Error Rate (WER%) as a function of the number of states, for varying number of components (K). Cross-validation using sets (a, b, c, d)

those obtained in the other training-test combinations involved in the 4-fold cross-validation experiment performed previously.

Table 3.2: Word Error Rate (WER%), for $Q = 6$ and $K = 64$, in different training-test combinations of the a, b, c, d and e folds

Training	Test	WER%
abc	d	17.6
abd	c	17.3
acd	b	19.0
bcd	a	17.5
abcd	e	34.3
abcde	e	24.3

From the results in Table 3.2, we can see that the results for the first four folds are very similar, in the range $17.3\% - 19.0\%$, while those for fold e (34.3% and 24.3%) are significantly higher. This might be due to the different age and profession distribution of the writers that contributed to fold e, as compared with those of the first four folds [17]. On the other hand, when compared with the results on fold e (abcde-e) at the ICDAR 2007 competition, our 24.3% would rank in the middle part of the list, far from the best results, but nonetheless above many participating systems.

3.8.3 Effect of the window width

In [15], we found that the sliding window width has a very positive effect on the accuracy of our BHMM-based word recognizer though, as usual, has to combined with an adequate number of components for the state-conditional finite mixture models. This is clearly shown in Figure 3.9, where the Word Error Rate (WER%) of our BHMM-based recognizer is plotted as a function of the number of mixture components (K), for varying sliding window widths (W). Each WER estimate (plotted point) was obtained by cross-validation with the first 4 standard folds (abcd), using BHMMs of 6 states. For $K = 1$, BHMMs were trained by first segmenting the training set with a "neutral" model, and then using the resulting segments to perform a Viterbi initialization followed by 4 EM iterations. For $K > 1$, they were trained by first splitting the components of the models trained with $K/2$ components and then, as before, applying 4 EM iterations. The conventional Viterbi algorithm was used to compute the most probable word for each test word image.

From the results in Figure 3.9 it is clear that the use of a sliding window improves the results to a large extent. In particular, the best result, 7.4%, is obtained for $W = 9$ and $I = 32$, though very similar results are also obtained for $W = 7$ and $W = 11$. It is worth noting that the best result achieved with no sliding windows ($W = 1$) is 17.7%.

To get some insight into the behavior of our BHMMs, the model for character خ, trained from folds abc with $W = 9$ and $K = 32$, is (partially) shown in Figure 3.10 (bottom) together with its Viterbi alignment with a real image of the character خ, extracted from sample *de05_007* (top). As in Figure 3.2 (bottom), Bernoulli prototypes are represented as gray images where the gray level of each pixel measures the probability of its corresponding pixel to be black (white $= 0$ and black $= 1$). From these prototypes, it can be seen that the model works as expected, i.e. each state from right to left accounts for a different local part of خ, as if the sliding window was moving smoothly from right to left. Also, note that the main stroke

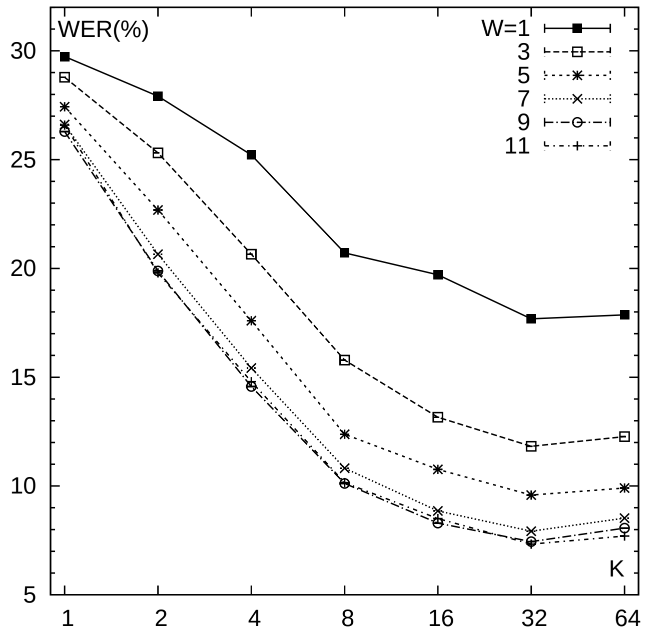


Figure 3.9: WER(%) as a function of the number of mixture components (K) for varying sliding window widths (W)

of the character خ appears almost neatly drawn in most prototypes, whereas its upper dot appears blurred, probably due to a comparatively higher variability in window position.

3.8.4 Effect of the number of states

In accordance with the preliminary results reported in Section 3.8.2, we have only tried BHMMs of 6 states in the experiment described in previous section. However, as discussed in [19], letters in Arabic script differ significantly in length, and thus it might not be appropriate to model all of them using BHMMs of identical number of states. With this idea in mind, a new experiment was carried out, similar to that described above, but with fixed sliding window of $W = 9$ and variable number of states per character. To decide the number of states for each character, we first Viterbi-segmented all training data using BHMMs of 4 states, and then computed the average length of the segments associated with each character. Given an average segment length for character c, $\overline{T}_c$, its number of states was set to $F \cdot \overline{T}_c$, where F is a *factor* measuring the average number of states that are required to emit a feature

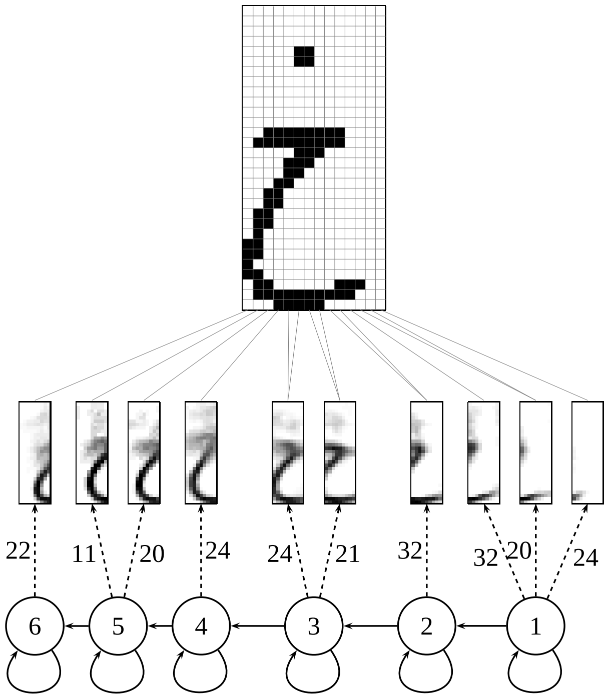


Figure 3.10: BHMM for character خ, trained from folds abc with $W = 9$ and $K = 32$ (bottom), together with its Viterbi alignment with a real image of the character خ, extracted from sample *de05_007* (top).

vector. Thus, its inverse, $\frac{1}{F}$, can be interpreted as a *state load*, that is, the average number of feature vectors that are emitted in each state. For instance, $F = 0.2$ means that only a fraction of 0.2 states is required to emit a feature vector or, alternatively, that $\frac{1}{0.2} = 5$ feature vectors are emitted on average in each state. Figure 3.11 shows the WER obtained as a function of F, $F \in \{0.2, 0.3, 0.4, 0.5\}$, for varying values of the number of mixture components. The best result plotted in Figure 3.11 is a WER of 7.3%, using $F = 0.4$ and $K = 32$. This results is slightly better than the 7.4% obtained with 6 states per character.

In Figure 3.12, the sample *dm33_037* has been recognized using BHMMs with $W = 9$, $K = 32$ and both, 6 states (top) and variable number of states, with $F = 0.4$ (bottom). In both cases, the recognized word has been Viterbi-aligned at character level (background color) and state level (bottom and upper ticks). Although the BHMMs of 6 states produce a

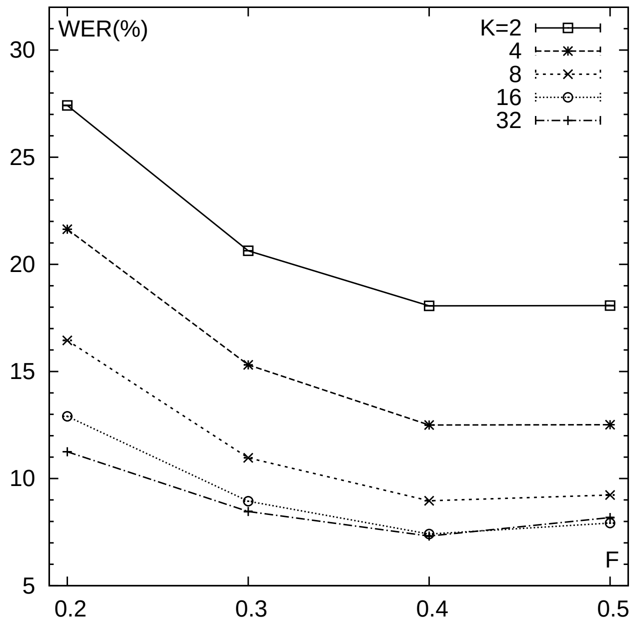


Figure 3.11: WER(%) as a function of the factor F for varying values of the number of mixture components (K)

recognition error, النفاتية (top), the BHMMs of variable number of states are able to recognize the correct word, الدخانية (bottom). Note that there are two letters, 'ﻟ' and 'ﺪ', that are written at the same vertical position or, better to say, at a specific column, and thus it is very difficult for our BHMMs to recognize them as two different letters. On the other hand, the incorrectly recognized word (top) is not very different in shape from the correct one; e.g. the characters 'ﻨ' and 'ﺘ' are very similar (type B [17]).

3.8.5 Effect of repositioning and final results

In the experiments described above, we have not tried window repositioning after window extraction but, as discussed in Sec. 3.6, many recognition errors of our BHMM-based classifier might be due to its limited capability to properly model vertical image distortions. In order to study the effect of repositioning on the classification accuracy, the standard method (no repositioning) was compared with the three repositioning methods described in Sec. 3.6: vertical, horizontal, and both directions. This was done with $W = 9$, $K = 32$, and $F = 0.4$, for the

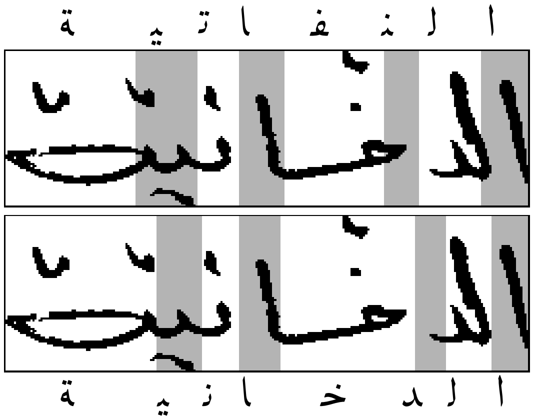


Figure 3.12: Sample *dm33_037* incorrectly recognized with BHMMs of 6 states (top), but correctly recognized with BHMMs of variable number of states (bottom). In both cases, the recognized word has been Viterbi-aligned at character level (background color) and state level (bottom and upper ticks)

four partitions considered in the previous experiments (abc-d, abd-c, acd-b, and bcd-a), and also for the partitions abcd-e and abcde-e, which are commonly used to compare classifiers in the IfN/ENIT task, abcd-e especially. The results are included in Table 3.3.

Table 3.3: Word Error Rate (WER%) of the BHMM-based recognizer (with $W = 9$, $K = 32$, and $F = 0.4$) in different training-test combinations of the abcde folds, for four repositioning methods: none, vertical, horizontal, and both directions

		WER%				
Training	Test	Basic	None	Vertical	Horizontal	Both
abc	d	17.6	7.5	4.7	8.4	4.8
abd	c	17.3	6.9	3.6	7.7	3.8
acd	b	19.0	7.7	4.5	8.1	4.4
bcd	a	17.5	7.6	4.4	8.2	4.6
abcd	**e**	**34.3**	**12.3**	**6.1**	**12.4**	**6.1**
abcde	e	24.3	4.0	2.2	3.9	2.0

As expected, from the results in Table 3.3 it becomes clear that vertical (or both) window repositioning improves very much the results obtained with the standard method or horizontal repositioning alone.

3.8.6 ICFHR 2010 - Arabic Handwriting Recognition Competition

The Arabic handwriting recognition competition held at International Conference on Frontiers in Handwriting Recognition (ICFHR 2010) [12], was the fourth competition on Arabic handwriting recognition. It was very similar to the previous competitions ([3, 4, 17]). It used the IfN/ENIT database containing Arabic handwritten Tunisian town names (Section 3.8.1) for training and testing. In this competition, 6 systems were submitted by 4 participanted groups.

Our submitted systems were based on Bernoulli HMMs (BHMMs), that is, HMMs in which conventional Gaussian mixture density functions are replaced with Bernoulli mixture probability functions (Section 3.2). Also, in contrast to the basic approach described in Section 3.3, in which narrow, one-column slices of binary pixels are fed into BHMMs, the systems were based on a sliding window of adequate width to better capture image context at each horizontal position of the word image (Section 3.6). As an example, Figure 3.13 shows the generation of a 7×5 word image of the number 31 from a sequence of 3 windowed ($W = 3$) BHMMs for the characters 3, "space" and 1.

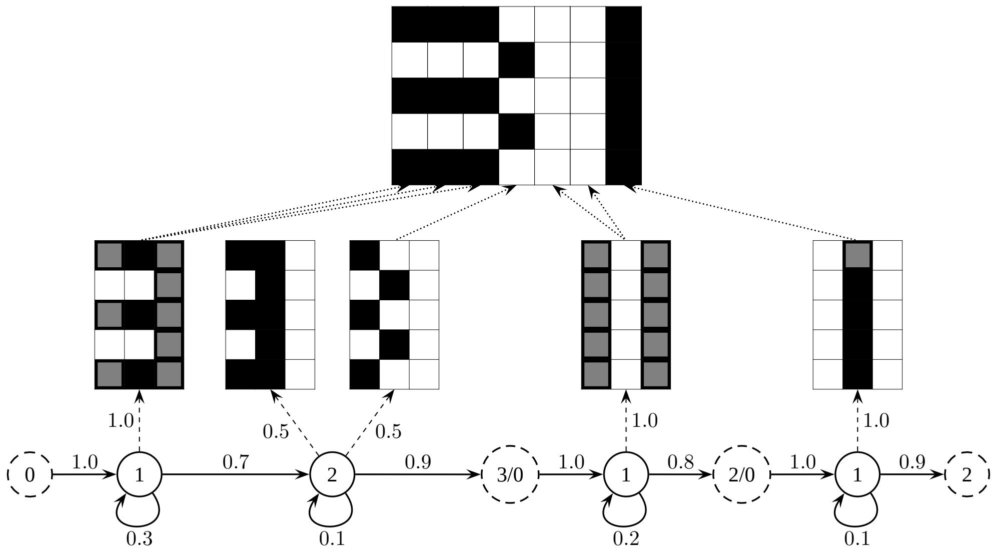


Figure 3.13: Generation of a 7×5 word image of the number 31 from a sequence of 3 windowed ($W = 3$) BHMMs for the characters 3, "space" and 1

These systems were trained from input images scaled in height to 30 pixels (while keeping the aspect ratio), and then binarized by means of the Otsu algorithm. A sliding window of width 9 was applied, and thus the resulting input (binary) feature vectors for the BHMMs had 270 bits. In order to decide the number of states for each character BHMM, we first Viterbi-segmented all training data using BHMMs of 4 states, and then computed the average length of the segments associated with each character. Given an average segment length for character c, $\overline{T}_c$, its number of states was set to $F \cdot \overline{T}_c$, where F is a *factor* measuring the average number of states that are required to emit a feature vector, which was empirically adjusted to 0.4. Similarly, the number of mixture components per state was empirically adjusted to 32. On the other hand, parameter estimation and recognition were carried out using the EM and Viterbi algorithms, respectively.

Two systems were submitted: UPV-BHMM and UPV-BHMM2. They only differ in the way in which the sliding window is applied. In the UPV-BHMM system, the sliding window is applied at each column of the input image, as illustrated above. In the UPV-BHMM2 system, however, the sliding window is repositioned after each application, so as to align its center to the image mass center within the window (Section 3.7). Table 3.4 shows the results of this competition for 6 diferent systems.

Table 3.4: Word Error Rate (WER%) of text image recognition on IfN/ENIT for the 6 different particated systems

System	set *d*	set *e*	set *f*	set *s*
UPV-BHMM	1.1	4.0	12.1	21.6
UPV-BHMM2	0.6	2.0	**7.8**	**15.4**
REGIM	5.9	13.4	21.0	31.6
CUBS-AMA	10.0	19.2	19.7	32.1
RWTH-OCR	**0.0**	**0.2**	9.1	18.9
RWTH-OCR2	0.3	1.2	9.0	19.7

As shown in Table 3.4, the best results for sets *f* and *s* were 7.8 and 15.4 respectively for our system (UPV-BHMM2), following the repositioning approach. Indeed, this system ranked first in this competition due to its great performance on both sets.

3.9 Experiments on OpenHaRT database

Experiments in this section were carried out on the data provided by the Linguistic Data Consortium (LDC) in the 2010 NIST Open Handwriting Recognition and Translation (OpenHaRT-'10) evaluation [11]. In Section 3.9.1 the database is described in details.

3.9.1 OpenHaRT 2010 database

The NITS OpenHaRT 2010 corpus comprises a total of 40053 Arabic image documents written by 148 different scribes, from a lexicon of $100K$ words. Data for training included *MADCAT Phase1* and *MADCAT Phase2* (39050 Arabic image documents). For development, it included *MADCAT Phase1 Pilot Evaluation* (470 Arabic image documents). For testing, it included *MADCAT Phase2 Evaluation* (533 Arabic image documents). Table 3.5 shows more details about this corpus. It is worth noting that the testing set was released after the official publication of the NIST OpenHaRT'10 results. An example of a document from this corpus is shown in Figure 3.14.

Table 3.5: The NIST OpenHaRT database statistics including the number of extracted word images and line images

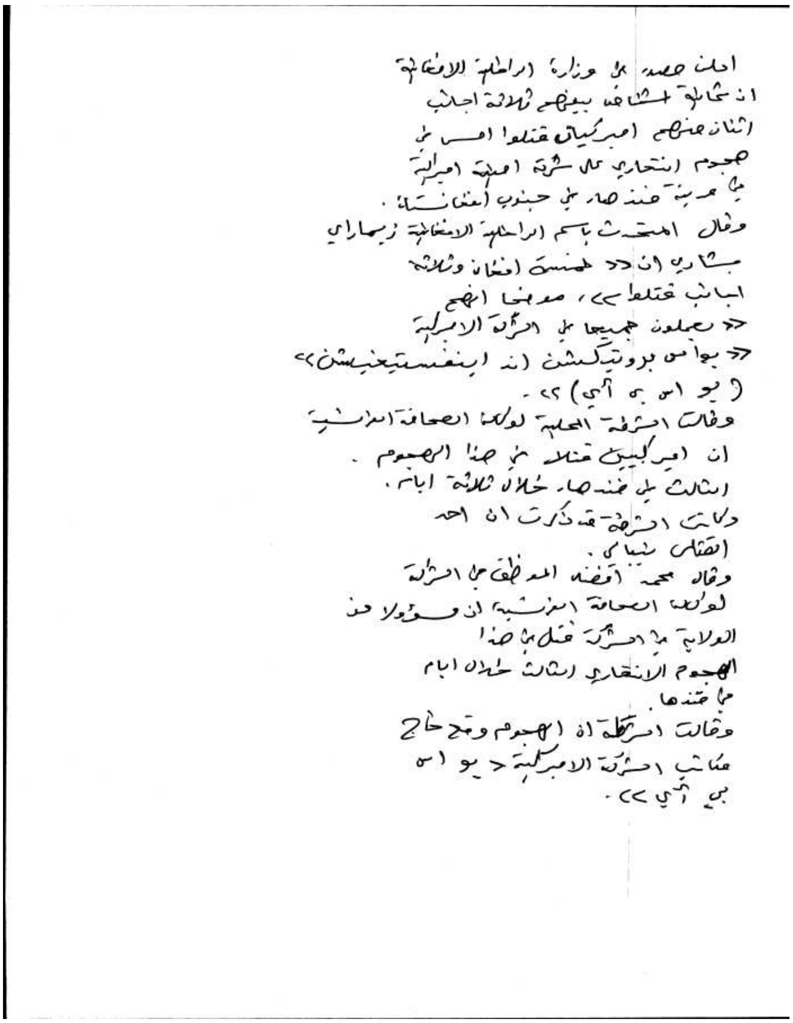
	Num of passages	Num of scribes	Docs	Line images	Word images
Training set	6000	100	39050	$686K$	$3850K$
Development set	100	24	470	$8K$	$48K$
Testing set	100	24	533	$10K$	64
Total	6200	148	40053	$705K$	3963

In addition to the document image given, also a segmentation data file was given for each sample to facilitate the word and line segmentation process. So for example, to take one line out of the document, we crop it giving the segmentation coordinates for that line. Also, to take one word out, we crop the document image giving the segmentation coordinates for that word. An example for a word text image sample followed by a line text image is shown in Figure 3.15.

3.9.2 NIST OpenHaRT 2010 Evaluation

The OpenHaRT 2010 evaluation was created after the DARPA MADCAT program with the idea of encouraging researchers to participate in solving one of the more challenging tasks toward the goal of document understanding.

The OpenHaRT 2010 was the first evaluation of its kind held by NIST. It focused mainly on the document text recognition, text translation, and image translation technologies for documents containing Arabic script [11]. However, the focus of this evaluation was not limited to recognition and translation technologies, but it also included tasks for word and line segmentation to explore the relationship between system performance and the system's ability to segment the data. Segmentations was represented as a series of polygon coordinates indicating the locations of the text segments within the image [11].



اعلن مسؤول في وزارة الداخلية الافغانية
ان ثمانية اشخاص بينهم ثلاثة اجانب
اثنان منهم امريكيان قتلوا امس في
هجوم انتحاري على شركة امنية امريكية
في مدينة قندهار في جنوب افغانستان .
وقال المتحدث باسم الداخلية الافغانية زمراي
بشاري ان «« خمسة افغان وثلاثة
اجانب قتلوا »» ، موضحا انهم
«« يعملون جميعا في الشركة الامريكية
«« يو اس بروتيكشن اند اينفستيغيشن »»
(يو اس بي آي) »» .
وقالت الشرطة المحلية لوكالة الصحافة الفرنسية
ان امريكيين قتلا في هذا الهجوم .
الثالث في قندهار خلال ثلاثة ايام .
وكانت الشرطة قد ذكرت ان احد
القتلى نيبالي .
وقال محمد افضل المتحدث من الشركة
لوكالة الصحافة الفرنسية ان مسؤولا من
الولاية في الشركة قتل في هذا
الهجوم الانتحاري الثالث خلال ايام
في قندهار .
وقالت الشرطة ان الهجوم وقع خارج
مكاتب الشركة الامريكية « يو اس
بي آي »» .

Figure 3.14: An example of Arabic image document extracted from the NIST OpenHaRT 2010 database

More precisely, this evaluation has focused mainly on three tasks: text recognition task which was refereed to as Document Image Recognition (DIR), text translation task, referred to as Document Text Translation (DTT), and image translation task known as Document Im-

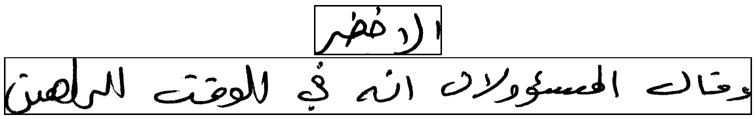


Figure 3.15: Sample of a word text image followed by a sample of line text image taken from the OpenHaRT'10 database

age Translation (DIT). In addition, training included two conditions: a constrained condition that required participants to develop their systems using only the provided LDC data, and an unconstrained condition in which participants are free to use any additional publicly available non-LDC resources for the system development (for more information, please refer to [11]).

In the NIST OpenHaRT 2010 evaluation, we only participated in the DIR task using both word and line segmentation conditions. For image pre-processing, original images were first rescaled to a given height, without changing the original aspect ratio, and then binarized using the Otsu method. A sliding window of fixed width was centered at each column, and then translated to align its center with its mass center (following the repositioning techniques in Section 3.7). The binary image under the translated window was read to construct a local binary feature vector and, in this way, the whole input image was transformed into a sequence of binary feature vectors. Image height was scaled to 30 pixels, and window width was selected as 9 columns. On the other hand, transcriptions were also pre-processed. All Arabic letters were encoded by adding shape information; that is, in Arabic, the shape of letters written at the beginning of the word are different from those written in the middle or at the end.

Each transcription hypothesis was modelled as an HMM in which emission probabilities are modelled as Bernoulli mixture distributions (Bernoulli HMMs) (Section 3.2). To keep the number of independent parameters low, the BHMM at sentence level (transcription hypothesis) was built from BHMMs at character level which depend on their surrounding characters, that is, following a *tri-character* modelling approach. The Viterbi algorithm was used for both training and decoding. For the word condition, we used 64 mixture components per state, and for the line condition, we used 32 mixture components per state. On the other hand, the avarage number of states per character was chosen to be 7.

The likelihood of each transcription hypothesis is weighted by its prior probability. In the word condition case, this prior probability is simply a relative frequency count for each word. In the more general line condition case, it is computed in accordance with a language model in which the probability of occurence of each word is modeled by only considering its preceding word (bigram language model). These word probabilities are estimated as smoothed relative frequency counts.

As a result, our system was ranked first in the line segmentation condition with 47.45%

of WER, and second in the word segmentation condition with 48.93% of WER. For more details please refer to the evaluation report [11].

3.10 Concluding remarks

Embedded Bernoulli HMMs (BHMMs) have been described and tested for Arabic Handwriting Recognition on the well-known IfN/ENIT database of handwritten Tunisian town names and on the OpenHaRT'10 database. We have described our basic approach, in which narrow, one-column slices of binary pixels are fed into BHMMs. Also, we have used a sliding window of adequate width to better capture image context at each horizontal position of the word image. In addition, we have considered three methods of window repositioning after window extraction so as to help our BHMM-based recognizer in dealing with vertical image distortions. The experiments reported have carefully studied the effects of the window width, the number of states, and repositioning. As expected, the best results have been obtained with an adequate adjustment of the window width, number of states, number of mixture components and, what it seems even more important, (vertical) window repositioning after window extraction. A WER of 6.1% has been achieved on the standard abcd-e partition of IfN/ENIT database, and a WER of 47.5% has been achieved on the OpenHaRT'10 database following the line segmentation condition.

Bibliography

[1] M. Dehghan, K. Faez, M. Ahmadi, and M. Shridhar. Handwritten Farsi (Arabic) word recognition: a holistic approach using discrete HMM. *Pattern Recognition*, 34(5):1057–1065, 2001.

[2] S. Günter and H. Bunke. HMM-based handwritten word recognition: on the optimization of the number of states, training iterations and Gaussian components. *Pattern Recognition*, 37:2069–2079, 2004.

[3] V. Märgner and H. E. Abed. ICDAR 2007 - Arabic Handwriting Recognition Competition. In *Proc. of the 9th Int. Conf. on Document Analysis and Recognition (ICDAR 2007)*, volume 2, pages 1274–1278, Curitiba (Brazil), sep 2007.

[4] V. Märgner and H. E. Abed. ICDAR 2009 Arabic Handwriting Recognition Competition. In *Proc. of the 10th Int. Conf. on Document Analysis and Recognition (ICDAR 2009)*, pages 1383–1387, Barcelona (Spain), 7 2009.

[5] E. Grosicki and H. El Abed. ICDAR 2009 Handwriting Recognition Competition. In *ICDAR '09*, pages 1398–1402, Barcelona (Spain), july 2009.

[6] L. Rabiner and B.-H. Juang. *Fundamentals of speech recognition*. Prentice-Hall, 1993.

[7] S. Young et al. *The HTK Book*. Cambridge University Engineering Department, 1995.

[8] A. Giménez and A. Juan. Bernoulli HMMs at Subword Level for Handwritten Word Recognition. In *4th Iberian Conference on Pattern Recognition and Image Analysis*, volume 5524 of *LNCS*, pages 497–504. Springer-Verlag, Póvoa de Varzim (Portugal), June 2009.

[9] U. V. Marti and H. Bunke. The IAM-database: an English sentence database for offline handwriting recognition. *IJDAR*, 5(1):39–46, 2002.

[10] M. Pechwitz, S. S. Maddouri, V. Märgner, N. Ellouze, and H. Amiri. IFN/ENIT - DATABASE OF HANDWRITTEN ARABIC WORDS. In *7th Colloque International Francophone sur l'Ecrit et le Document, CIFED*, pages 21–23, Hammamet (Tunis), oct 2002.

[11] A. Tong. NIST 2010 Open Handwriting Recognition and Translation (OpenHaRT'10) evaluation. *Proc. of the NIST 2010 Open Handwriting and Recognition Workshop*, 2010.

[12] V. Märgner and H. El Abed. ICFHR 2010 Arabic Handwriting Recognition Competition. In *Proc. of the 12th Int. Conf. on Frontiers in Handwriting Recognition (ICFHR 2010)*, pages 709–714, Kolkata (India), 11 2010.

[13] V. Märgner and H. El Abed. ICDAR 2011 - Arabic Handwriting Recognition Competition. In *ICDAR '11*, pages 1444–1448, Beijing (China), sep 2011.

[14] A. Giménez Pastor. *Bernoulli HMMs for Handwritten Text Recognition*. PhD thesis, Universitat Politècnica de Valcència, Valencia, (Spain), May 2014.

[15] A. Giménez, I. Khoury, and A. Juan. Windowed Bernoulli Mixture HMMs for Arabic Handwritten Word Recognition. In *Proc. of the 12th Int. Conf. on Frontiers in Handwriting Recognition (ICFHR)*, Kolkata (India), November 2010.

[16] J. T. Goodman. A bit of progress in language modeling. Technical report, 2001.

[17] V. Märgner, M. Pechwitz, and H. E. Abed. ICDAR 2005 Arabic Handwriting Recognition Competition. In *Proc. of the 8th Int. Conf. on Document Analysis and Recognition (ICDAR 2005)*, volume 1, pages 70–74, Seoul (Korea), 2005.

[18] N. Otsu. A Threshold Selection Method from Gray-Level Histograms. *IEEE Trans. on Systems, Man and Cybernetics*, 9:62–66, 1979.

[19] P. Dreuw, G. Heigold, and H. Ney. Confidence-Based Discriminative Training for Model Adaptaion in Offline Arabic Handwriting Recognition. *ICDAR*, pages 596–600, 2009.

[20] I. Khoury, A. Giménez, and A. Juan. Arabic Handwritten Word Recognition Using Bernoulli Mixture HMMs. In *Proc. of the 3rd Palestinian Int. Conf. on Computer and Information Technology (PICCIT 2010)*, Hebron (Palestine), 3 2010.

[21] I. Khoury, A. Giménez, and A. Juan. Arabic Handwriting Recognition Using Bernoulli HMMs. In V. Märgner and H. El Abed, editors, *Guide to OCR for Arabic Scripts*, pages 255–272. Springer London, 2012.

CHAPTER 4

Arabic Printed Text Recognition

Contents

4.1 Introduction

In the previous chapter, Bernoulli HMMs were used with off-line handwriting recognition in Arabic script. The main idea was to by-pass feature extraction and directly feed columns of raw, binary pixels into BHMMs. By doing that, we ensure that no discriminative information is filtered out during feature extraction, which in some sense is integrated into the recognition model. This basic approach was tested on the IfN/ENIT database in Section 3.8.2. Then, we improved this approach by using a sliding window of adequate width to better capture image context at each horizontal position of the text image. This approach, which is called *windowed BHMMs* (Sec. 3.6), achieved very competitive results on IfN/ENIT database [1].

Though the *windowed BHMMs* approach achieved competitive results on IfN/ENIT, still, the window repositioning technique (Sec. 3.7) improved the results to around 50% on IfN/-ENIT, and achieved good results on OpenHaRT 2010 which ranked first in OpenHaRT'10 Evaluation [2] following the line segmentation condition.

In this chapter, following a procedure similar to the one describe in the previous chapter, we tried the effect of the previous techniques on Arabic printed text database. More precisely, our experiments will be carried out on the Arabic Printed Text Image (APTI) database [3]. These effects included, window width, fixed and variable number of states, windowed BH-MMs, and repositioning. Indeed, extensive experiments are described from which state-of-the-art results are obtained.

In what follows, we briefly review windowed BHMMs with repositioning (Section 4.2) and its use for printed Arabic recognition by application of the Bayes decision rule (Section 4.3). In Section 4.4, we provide the results of a complete series of experiments on APTI as well as a comparison with results from other authors on this database. In this section we explore a new series of results using vertical repositioning and one of the state-of-the-art techniques based on neural networks. Finally, concluding remarks are given in Section 4.5.

4.2 Repositioning technique on Arabic printed text

As mentioned above, it was very helpful to better capture the image context at each horizontal position of the image by applying the sliding window approach. As an example, please refer to Figure 3.4, where the first row shows a binary image of 4 columns and 5 rows, which is transformed into a sequence of four 15-dimensional feature vectors by application of a sliding window of width 3.

The sliding window technique has shown to be limited when dealing with vertical image distortions particularly in the case of feature vectors extracted with significant context. Therefore, we have applied three methods of window *repositioning* after window extraction: *vertical, horizontal,* and *both*. For more details about this technique please refer to Section 3.7

of the previous chapter. An example of feature extraction is shown in Figure 3.4 in which the standard method (no repositioning) is compared with the three methods repositioning methods considered.

The effect of the repositioning techniques on Arabic handwritten text has similar effect on Arabic printed text. To observe this effect with real data, please refer to Figure 4.1 which shows a sequence of feature vectors extracted from a real sample of the APTI database, with and without (both) repositioning. As expected, (vertical or both) repositioning has the effect of normalizing vertical image distortions, especially translations.

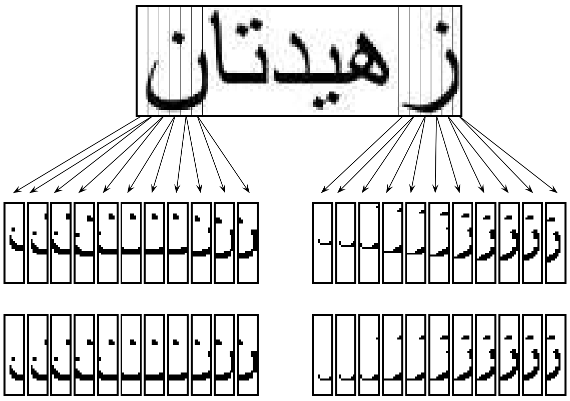


Figure 4.1: Original sample *Image_18_ArabicTransparent_5111* from set1 from APTI database (top) and its sequence of feature vectors produced with and without (both) repositioning (center and bottom, respectively)

4.3 Bernoulli HMMs for printed Arabic recognition

Given an observation O of unknown class, we use the Bayes decision rule to classify O into the class to which it belongs with highest *(posterior)* probability or, equivalently:

$$c^*(O) \;=\; \operatorname*{argmax}_{c=1,\ldots,C} \quad \log P(c) + \log P(O \mid c) \tag{4.1}$$

where C is the total number of classes and, for each class $c = 1, \ldots, C$, $P(c)$ is its *prior* probability and $P(O \mid c)$ is the class-conditional probability (density) for O to come from class c.

Class priors and class-conditional probability (density) functions are usually estimated from a set of *training* observations. The conventional approach to estimate class priors is simply to compute their relative frequencies from the training set. However, the estimation of class-conditional probability (density) functions is more involved and depends on the type of representation space for the observations. Usually, each class-conditional probability (density) function is modeled by an appropriate parametric function whose parameters are estimated by MLE from the training data. As an example, consider the problem of classifying images of *isolated printed Arabic characters*. The number of classes is modest and it is not difficult to collect many training examples for each class. Therefore, class priors can be accurately estimated by the conventional method. Also, if images are represented as sequences of feature vectors, each class-conditional probability function can be modeled by an independent BHMM (Eqs. (3.3) and (3.4)) with parameters estimated by MLE from training observations of its class.

The above approach for the estimation of class priors and class-conditional probability (density) functions is no longer applicable to classification problems with large number of classes due to the lack of training data for each class. Consider, as we do in this work, the problem of classifying images of *printed Arabic words*. Collecting a number of training observations for each word will be really difficult if we are interested in recognizing a large number of different words. Indeed, it will be impossible if we are interested in building an *open-vocabulary* recognizer, that is, one even able to recognize words not "seen" (with no observations) in the training data. As with Arabic handwriting recognition in general, the usual approach in this case consists in using global (word) models defined in terms of local (subword) models. This is the approach followed in this work. Formally, given an observation O of an unknown word, we use Eq. (4.1) to decide to which word corresponds:

$$w^*(O) = \operatorname*{argmax}_w \quad \log P(S_w) + \log P(O \mid S_w, \mathbf{\Theta}) \tag{4.2}$$

where, for each word w, S_w is its sequence of symbols (characters), $P(S_w)$ is its prior probability and $P(O \mid S_w, \mathbf{\Theta})$ is the probability for O to be generated from a BHMM for w (Eq. (3.5)). Word priors are modeled with n-gram language models at character level [4]. Word-conditional probability functions are modeled by BHMMs built from shared, embedded BHMMs at character level (Eq. (3.5)) with parameters trained by MLE.

Clearly, the direct way to measure the error of a word recognizer is to count the (relative) number of misclassified observations in a collection of *test* observations (i.e. samples held out during training). In what follows, this is referred to as the Word Error Rate (WER). Apart from the WER, we also use the Character Error Rate (CER), that is, the (relative) number of misclassified characters. In practice, the CER can be considered equivalent to the WER for comparison purposes.

4.4 Experiments on APTI

As indicated in the introduction, in this Section we provide the results of a complete series of experiments on APTI as well as a comparison with results from other authors on this database. APTI is briefly described in Section 4.4.1 together with its basic preprocessing for the experiments below. Then, two experimental protocols are defined in Section 4.4.2, UPVPC1 and UPVPC2, whose results are reported separately in Sections 4.4.3 and 4.4.4 respectively. Finally, the idea of vertical repositioning is also tried on recent state-of-the-art techniques based on neural networks in Section 4.4.5.

4.4.1 APTI database and preprocessing

The Arabic Printed Text Image (APTI) database is a collection of images of Arabic Printed words. It was recently published by [3] for large-scale benchmarking of open-vocabulary, multi-font, multi-size and multi-style text recognition systems in Arabic. It consists of 113284 different single words, each one available in 10 different fonts, 10 different font sizes, and also 4 different styles. A couple of examples of word images of Arabic transparent font are shown in Figure 4.2.

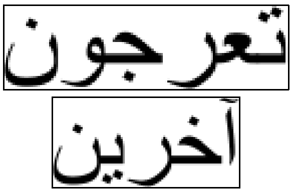


Figure 4.2: Two word image samples of Arabic transparent font, taken from the APTI database

APTI is divided into six equilibrated sets ($set1$, $set2$, $\dots$, $set6$) to allow for flexibility in the design of experimental protocols. Each set has different words, but characters are equally distributed. The five first sets are available for the scientific community. The sixth set is kept by the authors for future evaluation of systems in blind mode.

For the experiments reported below, APTI was preprocessed by scaling all images in the first five sets to a height of D pixels (for 10 different values of D from 30 to 50) while keeping the aspect ratio. Scaled images were then binarized by application of the Otsu's method [5].

4.4.2 Experimental protocols: UPVPC1 and UPVPC2

APTI was used first in the Arabic Recognition Competition of ICDAR 2011 [6]. Two experimental protocols were defined which differ in the number of fonts used: APTIPC1 and APTIPC2. In APTIPC1, only the Arabic Transparent font was used. In APTIPC2, however, five different fonts were used: Arabic Transparent (Trans), Andalus (Anda), Diwani Letter (Diw), Simplified Arabic (Simp), and Traditional Arabic (Trad). In both protocols, only the $Plain$ font style was used, with sizes of 6, 8, 10, 12, 18 and 24 pixels. As indicated above, the first five sets were available to participants for system training, while the sixth set was held-out by the organizers for system comparison in blind mode.

Unfortunately, we could not use the training-test partition used at the ICDAR 2011 competition because the sixth set is not publicly available. Instead, we used the first four sets for training and the fifth set for testing. More precisely, we defined two new protocols: UPVPC1 and UPVPC2. In UPVPC1, 13000 images from the first four sets were randomly drawn (10000 for training and 3000 for testing). In UPVPC2, we used the whole first four sets for training and the whole fifth set for testing. In particular, we used 2266500 images for training, and 566040 for testing. Table 4.1 shows the number of training and test samples from each set used in each experimental protocol.

Table 4.1: Number of training and test samples (in **bold face**) in each set and protocol

		APTI		UPV	
Sets	Words	PC1	PC2	PC1	PC2
Set1	18897	113382	566910	2546	566910
Set2	18892	113352	566760	2463	566760
Set3	18886	113316	566580	2482	566580
Set4	18875	113250	566250	2509	566250
Set5	18868	113208	566040	**3000**	**566040**
Set6	18866	**113196**	**565980**	-	-
Total		679704	3398520	13000	2832540

4.4.3 Results using the UPVPC1 protocol

For (computational) simplicity, the UPVPC1 protocol was used in a first series of experiments to study the effect on the CER of various key parameters. We began with experiments for font size 6, which were then extended to other font sizes. In particular, for each dimension D in $\{30, 32, \ldots, 50\}$, each sliding window width W in $\{1, 3, \ldots, 11\}$, each number of states Q in $\{4, 5, 6, 7, 8\}$ and each number of mixture components K in $\{1, 2, 4, ..., 32\}$, a BHMM-based word recognizer was trained from the training data of font size 6 in the UPVPC1 protocol. For

$K = 1$, BHMMs were initialized by first segmenting training data with a "neutral" model, and then using the resulting segments to perform a Viterbi initialization. Initialized BHMMs were then trained with 4 EM iterations. For $K > 1$, BHMMs were initialized by splitting the mixture components of the models trained with $K/2$ mixture components per state. Again, after initialization, BHMMs were trained with 4 EM iterations. On the other hand, word priors were modeled with 5-gram language models at character level.

The above training procedure led to a different recognizer for each combination of key parameter values (apart from the font size itself). Each of them was of the form given by Eq. (4.2) though, as usual in (Arabic) text recognition, a *Grammar Scale Factor (GSF)* was used to adjust the importance of class priors with respect to word-conditional observation probabilities (i.e. the GSF is a constant multiplier for log-priors). For each combination of parameter values and each value of $GSF \in \{20, 30, 40, 50\}$, the corresponding recognizer was assessed in terms of CER from the test data of font size 6 in the UPVPC1 protocol.

Figure 4.3 shows the CER as a function of D (top left), K (top right), Q (bottom left) and GSF (bottom right); for $W = 1$, 3, 7 and 11 (the curves for $W = 5$ and 9 are similar and have been omitted for clarity). Each plotted point shows the best CER obtained over all values tried for the parameters not given. The best CER obtained is 3.4% for $D = 38$, $W = 7$, $Q = 7$, $K = 32$ and $GSF = 50$. In the plot at the top left, it is shown for $D = 38$ and $W = 7$, as the minimum CER obtained for all values tried for Q, K and GSF.

From the results in Figure 4.3, it is clear that the use of windowed BHMMs is of crucial importance. Indeed, the best CER obtained with no windows ($W = 1$) is 6.6%; i.e. it nearly doubles the best CER with windows. Note also that, as W, the number of mixtures components (K) has a strong effect on the CER. The best error rates were obtained with the maximum value of K tried (32). Therefore, this and larger values of K need to be tried in further experiments with more training data. The dimension (D), number of states (Q) and GSF are also key parameters to be adjusted, though Figure 4.3 does not show wide fluctuations in CER for the ranges of values considered.

As discussed previously in Section 3.8.4, letters in Arabic script differ significantly in length, so it might be more appropriate to model them with variable number of states. To prove the accuracy of this theory on Arabic printed text, an experiment similar to that described above was carried out for $D = 38$, $W = 7$, $K = 32$, $GSF = 50$ and variable number of states. To decide the number of states for each character, we first Viterbi-segmented all training data using BHMMs of 7 states, and then computed the average length of the segments associated with each character. Given an average segment length for character c, $\overline{T}_c$, its number of states was set to $F \cdot \overline{T}_c$, where F is a *factor* measuring the average number of states that are required to emit a feature vector. Thus, its inverse, $\frac{1}{F}$, can be interpreted as a *state load,* that is, the average number of feature vectors that are emitted in each state. For instance, $F = 0.2$ means that only a fraction of 0.2 states is required to emit a feature vector or, alternatively, that $\frac{1}{0.2} = 5$ feature vectors are emitted on average in each state. We tried all

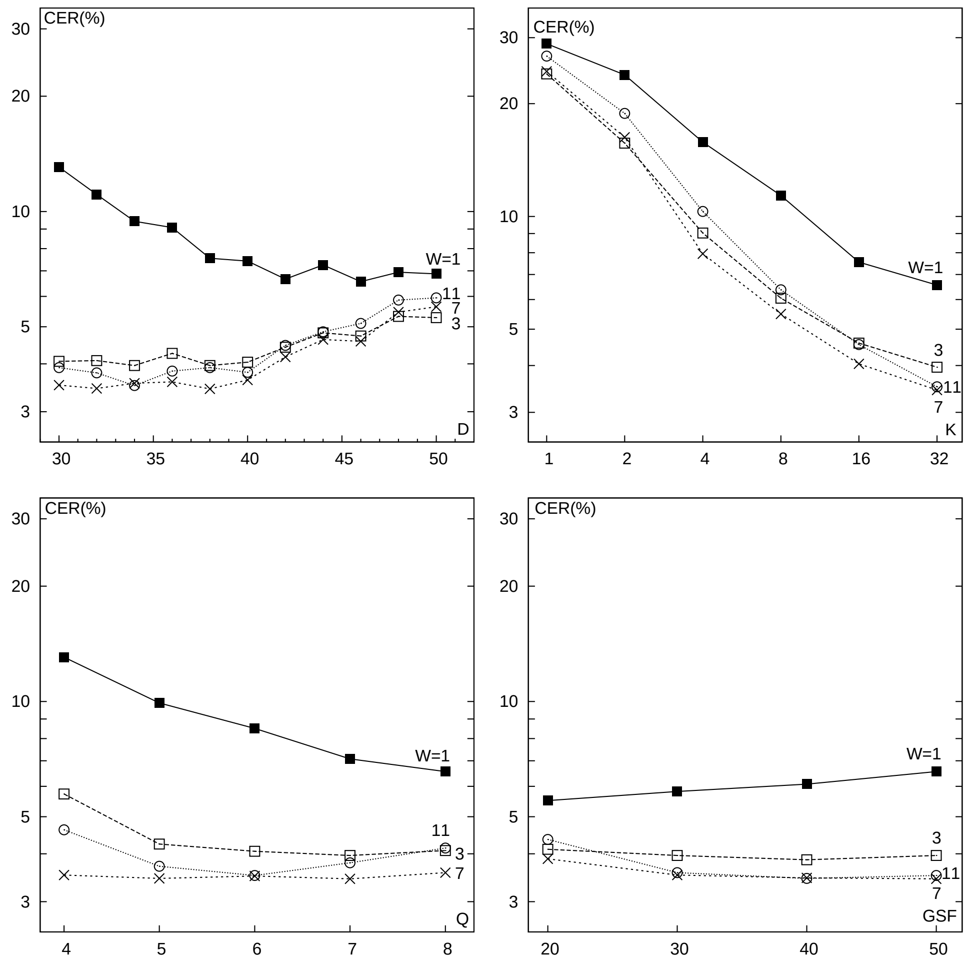


Figure 4.3: CER(%) as a function of the dimension D (top left), number of mixture components K (top right), number of states Q (bottom left) and GSF value (bottom right); for sliding window widths of $W = 1$, 3, 7 and 11

values of F in $\{0.1, 0.2, \ldots, 0.9\}$. The best result achieved is a CER of 3.2%, using $F = 0.5$, which is significantly better than the best result obtained above with fixed number of states (3.4%).

To complete our experiments with font size 6 data in the UPVPC1 protocol, the best recognizer found above was also tested with the four repositioning methods described in Sec. 4.2. As expected, the best CER, 1.1%, was obtained with *vertical* repositioning alone. Also as expected, it was similar to the CER achieved with repositioning in *both* directions (1.2%), and significantly better than those obtained with *horizontal* and *no* repositioning (3.2% for both).

The experiments described above in this Section were extended to all font sizes. More precisely, for each font size $S \in \{8, 10, 12, 18, 24\}$, each $D \in \{30, 32, \ldots, 50\}$, $W \in \{1, 3, \ldots, 11\}$, $Q \in \{5, 6, 7\}$ and $K \in \{1, 2, 4, ..., 32\}$, a BHMM-based word recognizer was trained and tested, for each value of $GSF \in \{30, 40, 50\}$, as described above. Also as above, the best recognizer for each size was then tested with variable number of states ($F \in \{0.3,\ \ldots,\ 0.7\}$) and different repositioning techniques ($R = \{N, V, H, B\}$; where N=None, V=Vertical, H=Horizontal and B=Both vertical and horizontal). The results obtained were similar to those reported in Figure 4.3 for font size 6. More precisely, the best error rates were obtained with windows of width $W \in \{7, 9, 11\}$, $K = 32$ components, $GSF = \{40, 50\}$, variable number of states with $F \in \{0.4, 0.5, 0.6\}$, and vertical repositioning. For brevity, these error rates are not reported here in detail, as those in Figure 4.3 for font size 6. Instead, only a summary of best error rates is reported in Table 4.2 (including font size 6 for completeness). Note that the best recognizer (combination of parameter values) for each font size is trained within the parameter ranges indicated above. Indeed, all recognizers trained within these ranges provide nearly identical error rates.

Table 4.2: Best recognizer (combination of parameter values) and its CER(%) for each size.

Size	D	W	R	F	K	GSF	CER(%)
6	38	7	V	0.5	32	50	1.1
8	40	7	V	0.6	32	40	0.6
10	44	9	V	0.5	32	40	0.6
12	40	9	V	0.5	32	40	0.4
18	40	9	V	0.5	32	40	0.5
24	42	11	V	0.4	32	40	0.8

To get some insight into the behavior of our windowed BHMMs, a real model for the character ﻌ is (partially) shown in Figure 4.4 (bottom) together with its Viterbi alignment with a real image of the character ﻌ, extracted from sample *Image_24_ArabicTransparent_562, set1* (top). Bernoulli prototypes are represented as gray images where the gray level of each pixel measures the probability of its corresponding pixel to be black (white $= 0$ and black $=$ 1). From these prototypes, it can be seen that the model works as expected, i.e. each state from right to left accounts for a different local part of ﻌ, as if the sliding window was moving smoothly from right to left.

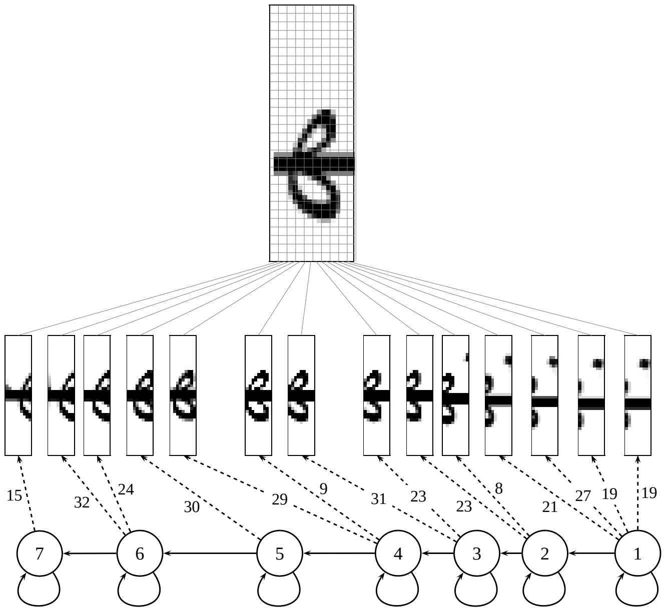


Figure 4.4: Real BHMM example for character ﻌ and its Viterbi alignment on a real image

4.4.4 Results using the UPVPC2 protocol

The UPVPC1 protocol was used to study the effect on the CER of various key parameters, variable number of states, and repositioning. Taking into account the best results obtained with it, the UPVPC2 protocol was used in a new series of experiments to obtain results in conditions similar to those used in the ICDAR 2011 Arabic Recognition Competition (see Sec. 4.4.2). In particular, for each of the five font types considered in UPVPC2, $T \in \{Trans, Anda, Diw, Simp, Trad\}$, and each font size $S \in \{6, 8, 10, 12, 18, 24\}$, a BHMM-based word recognizer was trained and tested from the data in UPVPC2 of font type T and size S. We used $D = 40$, $W = 9$, $R = V$, $F = 0.5$ (on a Viterbi segmentation produced by a recognizer trained with $Q = 7$, $K = 128$ and $GSF = 40$), $K = 128$ and $GSF = 40$. Except for the K, these parameter values are within the parameter ranges leading to the best error rates with the UPVPC1 protocol. However, in the case of K, we used 128 instead of 32. As discussed in Sec. 4.4.3, values of K larger than 32 had to be tried, especially with more training data as with the UPVPC2 protocol. Actually, we tried each $K \in \{1, 2, 4, ..., 128\}$, though $K = 128$ provided the best error rates in all cases.

Table 4.3 shows CER results for each font type and size. The error rates labeled as 2013a in the Year column were obtained as described above. That is, each test sample was accompanied by its font type and size so as to select its appropriate recognizer. However, the error rates labeled as 2013b were obtained in a slightly different way, by only providing the font size of each test sample. In this case, given a test sample of size S, all the five font-dependent recognizers for size S were run in parallel and that producing the highest classification score (see Eq. (4.2)) was chosen to decide the recognized word. The error rates labeled as 2011 are the best results of the ICDAR 2011 competition, which were also obtained by only providing the font size of each test sample.

Table 4.3: CER results for each font type and size (2013a="font type and size given"; 2013b="only font size given"; 2011="best results from the ICDAR 2011 competition")

Font/Size	Year	6	8	10	12	18	24	Mean
Andalus	2013a	0.9	0.2	0.1	0.1	0.0	0.0	0.2
	2013b	0.9	0.2	0.1	0.1	0.0	0.0	0.2
	2011	1.1	5.2	3.9	3.3	3.3	3.0	3.3
Arabic Transparent	2013a	0.6	0.1	0.1	0.0	0.0	0.1	0.2
	2013b	0.6	0.1	0.1	0.0	0.0	0.0	0.1
	2011	1.0	3.5	3.4	3.9	3.8	3.9	3.3
Simplified Arabic	2013a	0.5	0.1	0.1	0.0	0.0	0.0	0.1
	2013b	0.4	0.1	0.1	0.0	0.0	0.0	0.1
	2011	0.8	3.9	3.3	3.1	3.0	2.6	2.8
Traditional Arabic	2013a	6.4	1.3	0.5	0.3	0.2	0.2	1.5
	2013b	6.5	1.3	0.5	0.3	0.2	0.2	1.5
	2011	10.7	18.1	14.1	11.5	12.5	11.7	13.1
Diwani Letter	2013a	10.0	7.2	6.7	6.2	6.1	5.9	7.0
	2013b	10.0	7.2	6.7	6.2	6.1	5.9	7.0
	2011	9.1	24.2	16.6	10.9	5.1	7.4	12.2

A first conclusion that can be drawn from Table 4.3 is that the figures labeled as 2013a and 2013b are virtually identical. Therefore, when font size is known but font type is not, the procedure described above to obtain the 2013b results seems absolutely reliable. Another important conclusion from Table 4.3 is that the results of this work outperform by a large extent those from the competition. Note that, on average, recognition of Andalus, Arabic Transparent and Simplified Arabic is nearly perfect in terms of CER. On the other hand, recognition of Traditional Arabic and Diwani Letter is fairly good and comparatively much better than that of the ICDAR 2011 competition.

Apart from the above multi-font and mono-size recognition results, the ICDAR 2011 competition also included mono-size results on only the Arabic Transparent font. For this particular font, results were published for both, competition participants (IPSAR and UPV) and organizers (DIVA-REGIM). Also, more recent results have been published by [7], and by [8]. The most recent results come from the ICDAR 2013 second competition on APTI, which included three more participants than in its first edition: SID, THOCR and Siemens [9]. All these results are shown in Table 4.4 in terms of CER and WER. UPV-REC1, UPV-BHMM and UPV-2013 refer to our system at, respectively, ICDAR 2011, ICDAR 2013 and this work. Note that the results of UPV-BHMM and UPV-2013 are nearly identical and thus, as expected, the UPVPC2 protocol provides a good approximation to the experimental conditions of the ICDAR competitions on APTI. These results are much better than those of UPV-REC1 and only at a marginal distance from the best system at the ICDAR 2013 second competition on APTI. They are also much better than those reported in [10], where an initial, preliminary part of the experiments and results described here can also be found.

Table 4.4: CER and WER results for the Arabic Transparent font in each size

System	Year		6	8	10	12	18	24	Mean
IPSAR	2011	WER	94.3	26.7	25.0	16.9	22.9	22.5	34.7
		CER	40.6	5.8	4.9	3.1	4.3	3.2	10.3
UPV-REC1	2011	WER	5.5	2.6	3.3	7.5	15.4	15.6	8.3
		CER	1.0	0.4	0.6	1.3	3.1	4.0	1.7
DIVA-REGIM	2011	WER	13.1	4.1	4.3	6.1	2.1	1.1	5.1
		CER	2.0	0.8	0.7	1.2	0.3	0.3	0.9
Awaida et al.	2012	CER	-	-	-	-	-	-	3.4
Dershowitz et al.	2013	WER	72.4	21.1	10.2	6.0	1.0	1.5	18.7
		CER	31.8	5.6	2.5	2.4	0.2	0.4	7.2
UPV-BHMM	2013	WER	2.8	0.3	0.2	0.1	0.1	0.1	0.6
		CER	0.5	0.1	0.1	0.0	0.0	0.0	0.1
SID	2013	WER	5.7	3.8	1.8	1.2	3.4	2.6	3.1
		CER	0.3	0.0	0.0	0.1	0.0	0.0	0.1
THOCR	2013	WER	10.5	4.2	5.2	7.5	5.4	5.0	6.3
		CER	1.7	0.5	0.8	0.9	0.9	0.8	0.9
Siemens	2013	WER	0.1	0.1	0.0	0.1	0.0	0.0	0.1
		CER	0.0	0.0	0.0	0.0	0.0	0.0	0.0
UPV-2013	2013	WER	3.0	0.4	0.3	0.2	0.2	0.2	0.7
		CER	0.6	0.1	0.1	0.0	0.0	0.1	0.2

4.4.5 Results using DNN hybrid HMMs and vertical repositioning

Previous experiments have shown that the results obtained by using BHMMs are improved by applying the vertical repositioning technique. In recent work on handwritten recognition, vertical repositioning has also shown a significant improvement when used with other models than Bernoulli HMMs. In particular, in [11], a notable improvement was reported by using a Long Short Term Memory recurrent neural network (LSTM-RNN) tandem HMM and vertical repositioning on Arabic and French handwriting. This improvement is also observed in [12] where the window repositioning is used as a preprocessing step.

In order to asses that the vertical repositioning is useful for printed Arabic recognition with the current state-of-the-art techniques based on neural networks, such as LSTM-RNN, we have carried out a new series of experiments using the UPVPC2 protocol and a Deep Neural Network (DNN) hybrid HMM system [13]. This technique is similar to the Long Short Term Memory (LSTM) technique applied in [14]. It has been implemented in a recently released, open-source toolkit for automatic speech recognition called TLK toolkit [15]. On the basis of our experience on the application of TLK to speech recognition tasks within the transLectures project, we decided to use it also for the additional experiments discussed in this Section. The results of these experiments, with and without vertical repositioning, are shown in Table 4.5.

Table 4.5: CER and WER results for the Arabic Transparent font in each size

System	Year		6	8	10	12	18	24	Mean
Vertical Rep.	2014	WER	0.16	0.13	0.12	0.12	0.13	0.15	0.14
		CER	0.03	0.03	0.02	0.02	0.03	0.03	0.03
Without Rep.	2014	WER	0.22	0.20	0.12	0.13	0.13	0.16	0.16
		CER	0.04	0.04	0.02	0.02	0.03	0.03	0.03

As with the winner of ICDAR 2013 (Table 4.4), the results in Table 4.5 are nearly perfect. Even though the error is nearly zero, vertical repositioning still obtains slight improvements. In particular, for the more challenging font sizes (6 and 8), a modest improvement is achieved when applying repositioning. Specifically, for font size 6 results were 0.16% with repositioning and 0.22% without repositioning. (Note that, as we were using 19000 test samples approximately for each font size, a difference of 0.06% accounts for about 11 classification errors.) In a similar way, for font size 8, results were 0.13 and 0.20 for repositioning and non-repositioning respectively.

4.4.6 ICDAR 2011 and 2013 - Arabic Recognition Competitions

The Arabic recognition competition on Multi-font and Multi-size Digitally Represented Arabic was held in its first edition at the 11th International Conference on Document Analysis and Recognition (ICDAR 2011) [6], and the second edition at ICDAR 2013 [9]. Both competitions used the freely available Arabic Printed Text Image (APTI) database (Section 4.4.1) for training and testing.

As previously mentioned in Section 4.4.2, the ICDAR 2011 followed two protocols to perform the evaluation: APTIPC1 and APTIPC2. The common features between the two protocols is in the use of only the *Plain* font style, and the same font sizes independently. However, the difference between them is the use of different font types. More precisely, APTIPC1 was a mono-font protocol where the evaluation was performed on only the Arabic Transparent font, however, APTIPC2 was a multi-font protocol where the evaluation was performed on 5 different font types [6].

On the other hand, in ICDAR 2013, 4 evaluation protocols were defined: APTI$_2$PC, APTI$_2$PC1, APTI$_2$PC2, and APTI$_2$PC3. APTI$_2$PC is similar to APTPC1 from ICDAR 2011. APTI$_2$PC1 uses Arabic Transparent font and all font sizes $(6, 8, 10, 12, 18, 24)$ independently. APTI$_2$PC2 is similar t APTI$_2$PC1 but it uses DecoType Naskh font instead. The last protocol, APTI$_2$PC3, uses all fonts types and sizes independently [9].

We participated in both competitions using our Bernoulli HMMs-based techniques. More precisely, in the first edition of this competition (2011), we used our *windowed* version of our Bernoulli HMMs approach, which is based on a sliding window of adequate width to better capture image context at each horizontal position of the word image. This approach is described in Section 3.6. As an example, Figure 3.13 shows the generation of a 7×5 word image of the number 31 from a sequence of 3 windowed ($W = 3$) BHMMs for the characters 3, “space” and 1.

The systems presented to this competition were trained from input images scaled in height to 40 pixels (while keeping the aspect ratio) after adding a certain number of white pixel rows to both top and bottom sides of each image, and then binarized with the Otsu algorithm (Sec. 2.2.2). A sliding window of width 9 was applied, and thus the resulting input (binary) feature vectors for the BHMMs had 360 bits.

The number of states per character was adjusted to 5 states for images with font size of 6, and 6 states for other font sizes. Similarly, the number of mixture components per state was empirically adjusted to 64. On the other hand, parameter estimation and recognition were carried out using the EM algorithm.

Two systems were submitted: UPV-PRHLT-REC1 and UPV-PRHLT-REC2. They are used for both protocols. In APTIPC1 there were no differences between systems, where one model for each font size is trained and used later to recognize the test corpus. For the second protocol: In the first system, for each font size, a different model for each font style is trained. The test corpus is recognized on all models, and the recognized text word of the highest probability is selected. In the other system, a different character is considered for each style. A model for all styles together is trained and used to recognize the test corpus.

The systems presented to the 2013 ICDAR competition were build from the Bernoulli HMMs following the repositioning technique described in Section 3.7 to deal with the vertical image distortions.

The presented system (UPV-BHMM) was trained from input images scaled in height to 40 pixels (while keeping the aspect ratio), and then binarized with the Otsu algorithm. A sliding window of width 9 using the vertical repositioning was applied, and thus the resulting input (binary) feature vectors for the BHMMs had 360 bits. The number of states per character was adjusted to 7 states for images with all font sizes $6, 8, 10, 12, 18$, and 24. Similarly, the number of mixture components per state was empirically adjusted to 128. On the other hand, parameter estimation and recognition were carried out using the EM algorithm. Also, we used a 5-grams language model at character level instead of the conventional class priors, due to the huge amount of classes. In addition, a grammar scale factor (a weight on the language model to adjust their importance with respect to word-conditional likelihoods) was adjusted to values between 30 and 50 with respect to the font size.

Three variants of the UPV-BHMM system have been submitted: UPV-PRHLT-REC1 (for protocol APTI$_2$PC), UPV-PRHLT-RECPC2 (protocol APTI$_2$PC1) and UPV-PRHLT-RECPC3 (protocol APTI$_2$PC2). For APTI$_2$PC, where the size selection option is enabled, six different models were trained on the “Arabic Transparent” font images, one model for each font size. For all test images of a specific font size, a specific model was selected to recognize test images. For protocol APTI$_2$PC1, only one model for all font sizes was trained on the “Arabic Transparent” font images. For protocol APTI$_2$PC2, one model for all font sizes was trained on the “DecoType Nash” font images.

Results of both competitions for the Arabic Transparent font in each size is shown in Table 4.4. More details about the results of these competitions please refer to [6, 9].

4.5 Concluding Remarks

Windowed Bernoulli HMMs with repositioning have been described and extensively tested for printed Arabic recognition on the Arabic Printed Text Image (APTI) database. A system based on these models, though with no repositioning, ranked first at the ICDAR 2011 Arabic recognition competition for printed Arabic text, also based on the APTI database. Following evaluation protocols similar to those of the competition, this system has been largely improved by the use of repositioning and an exhaustive experimentation to adjust various key parameters and model topology (variable number of states). Results comparatively much better than those of the competition have been reported on multi-font and mono-size recognition, with nearly perfect performance for most fonts in terms of Character Error Rate. Indeed, a second edition of the competition on APTI was recently held at the ICDAR 2013 and our improved system obtained results nearly identical to those reported here. This second edition was harder than the first and our system ranked second, though only at a marginal distance from the best.

Bibliography

[1] V. Märgner and H. El Abed. ICFHR 2010 Arabic Handwriting Recognition Competition. In *Proc. of the 12th Int. Conf. on Frontiers in Handwriting Recognition (ICFHR 2010)*, pages 709–714, Kolkata (India), 11 2010.

[2] A. Tong. NIST 2010 Open Handwriting Recognition and Translation (OpenHaRT'10) evaluation. *Proc. of the NIST 2010 Open Handwriting and Recognition Workshop*, 2010.

[3] F. Slimane, R. Ingold, S. Kanoun, A. M. Alimi, and J. Hennebert. A New Arabic Printed Text Image Database and Evaluation Protocols. pages 946–950. IEEE, 2009.

[4] F. Jelinek. *Statistical Methods for Speech Recognition*. MIT Press, 1997.

[5] N. Otsu. A Threshold Selection Method from Gray-Level Histograms. *IEEE Trans. on Systems, Man and Cybernetics*, 9:62–66, 1979.

[6] F. Slimane, S. Kanoun, H. E. Abed, A. M. Alimi, R. Ingold, and J. Hennebert. ICDAR 2011 - Arabic Recognition Competition: Multi-font Multi-size Digitally Represented Text. pages 1449–1453. IEEE, 9 2011.

[7] S. Awaida and M. Khorsheed. Developing discrete density Hidden Markov Models for Arabic printed text recognition. In *Computational Intelligence and Cybernetics (CyberneticsCom), 2012 IEEE International Conference on*, pages 35–39, 2012.

[8] N. Dershowitz and A. Rosenberg. *Language, Culture, Computation: Studies in Honor of Yaacov Choueka*, volume 8000 of *Lecture Notes in Computer Science*, chapter Arabic Character Recognition. Springer-Verlag, 2013.

[9] F. Slimane, S. Kanoun, H. El Abed, A. M. Alimi, R. Ingold, and J. Hennebert. ICDAR 2013 Competition on Multi-font and Multi-size Digitally Represented Arabic Text. pages 1465–1469. CPS, 08 2013.

[10] I. Khoury, A. Giménez, A. Juan, and J. Andrés-Ferrer. Arabic Printed Word Recognition Using Windowed Bernoulli HMMs. In A. Petrosino, editor, *Proc. of the 17th Int. Conf. on Image Analysis and Processing – ICIAP 2013*, volume 8156, pages 330–339. Springer Berlin Heidelberg, 2013.

[11] P. Doetsch, M. Hamdani, H. Ney, A. Gimenez, J. Andres-Ferrer, and A. Juan. Comparison of Bernoulli and Gaussian HMMs using a vertical repositioning technique for off-line handwriting recognition. In *International Conference on Frontiers in Handwriting Recognition*, pages 3–7, Bari, Italy, 9 2012.

[12] M. Hamdani, P. Doetsch, M. Kozielski, A. El-Desoky Mousa, and H. Ney. The RWTH Large Vocabulary Arabic Handwriting Recognition System. In *International Workshop on Document Analysis Systems*, France, 4 2014.

[13] G. E. Dahl, S. Member, D. Yu, S. Member, L. Deng, and A. Acero. Context-Dependent Pre-trained Deep Neural Networks for Large Vocabulary Speech Recognition. In *IEEE Transactions on Audio, Speech, and Language Processing*, 2012.

[14] S. F. Rashid, M.-P. Schambach, J. Rottland, and S. von der Nüll. Low Resolution Arabic Recognition with Multidimensional Recurrent Neural Networks. In *Proceedings of the 4th International Workshop on Multilingual OCR*, MOCR '13, pages 6:1–6:5, New York, NY, USA, 2013. ACM.

[15] The transLectures Team - Universitat Politècnica de València. The translectures-upv toolkit (tlk). http://translectures.eu/tlk., 2013.

[16] I. Khoury, A. Giménez, A. Juan, and J. Andrés-Ferrer. Window repositioning for printed Arabic recognition. *Pattern Recognition Letters*, 51(0):86–93, 2015.

CHAPTER 5

Arabic Image Translation

Contents

5.1 Introduction

The Arabic-English image translation system is the process of producing English translations from images containing Arabic text. To our knowledge, there are only few systems that automatically translate Arabic typewritten or handwritten text images into another language. Typically, the available systems are based on the concatenation of two systems: a handwritten text recognition system and a machine translation system.

In this chapter we will describe our Arabic-English handwritten text recognition and translation (image translation) system. In the case of handwriting recognition of Arabic text, our work has focused on the UPV TLK toolkit [1], which implements a generative and discriminative Windowed Bernoulli Hidden Markov models (Windowed BHMMs) [2]. The system is built from character-based windowed BHMMs (Bernoulli HMMs) which are adequately concatenated so as to produce a different word-level windowed BHMM for each word to be recognized (Sec. 3.3). The system follows the newest implemented technique based on window repositioning to deal with the vertical image distortions (Sec. 3.7). This technique has shown very competitive results on both handwritten text [2] and printed text [3].

In the case of Arabic text translation, our work has focused on the combination of three different state-of-the-art phrase-based translation models: the standard (log-linear) phrase-based models using the Moses [4] toolkit, the hierarchical phrase-based models using the Jane [5] toolkit, and the N-gram phrase based models using the Ncode [6] toolkit. The combination of these models was performed using the ROVER [7] toolkit.

This system is tested on the data used in the NIST OpenHaRT 2013 evaluation. The results in this chapter are challenging and significantly outperform our previous results in both OpenHaRT 2010 and 2013 evaluations. To check the results of other participants, please refer to the NIST OpenHaRT 2010 and 2013 evaluation reports [8, 9].

In what follows, we briefly describe the Image Translation System (Sec. 5.2), the Arabic Handwriting Recognition system (Sec. 5.3), and the text translation system (Sec. 5.4). After that, we outline our experiments, results, and a real example in Section 5.5. Concluding remarks are given in Section 5.6.

5.2 Image Translation System

Image translation is an immensely challenging task that requires two well trained systems, a text recognition system and a text translation system. Both systems depend entirely on each other to ensure good translation quality. Despite important research approaches in image recognition for both printed and handwritten data, in most cases their use was limited to constrained tasks such as the Tunisian town names classification task [10] or the check processing task [11], among others. The same thing applies on machine translations systems. Nonetheless, real-word image translation applications require the translation of unconstrained data with serious diversity [12].

An Image Translation System is typically built from the concatenation of two systems: a handwritten text recognition system and a machine translation system. Given a handwritten text image f, by applying the Bayes classification rule, the Image Translation task can be expressed as follows:

$$y^\star = \underset{y \in Y}{\operatorname{argmax}}\ p(y|f) \approx \underset{y \in Y}{\operatorname{argmax}} \sum_x p(x|f)\ p(y|x) \tag{5.1}$$

where x stands for a candidate recognized source (Arabic) text and y for a candidate translated sentence (in English) corresponding to the input image f.

Since the summation over all possible transcriptions in Eq. (5.1) cannot be computed in practice, we have defined three different systems to reduce the search space. In all of them, the probability $p(x \mid f)$ in Eq. (5.1) was approximated by our handwriting recognition system, which is described in Section 5.3. Therefore, the key difference among these systems lay in the translation subsystems. We propose three different approaches:

In the first approach, Eq. (5.1) is approximated as follows:

$$\begin{aligned} y^* &\approx \underset{y \in Y}{\operatorname{argmax}} \left[\max_x \{p\,(x|f)\ \ p(y|x)\}\right] \\ &\approx \underset{y \in Y}{\operatorname{argmax}} \left[p\left(y|\max_x \{p(x|f)\}\right)\right] \end{aligned} \tag{5.2}$$

Letting $x^\star$ be $\max_x\{p(x|f)\}$, $p(y|x^\star)$ is approximated by our statistical machine translation system, which is described in Section 5.4. In other words, the input image was recognized by our handwriting recognition system, and the recognized text was fed into our machine translation system.

In the second approach, we used the same equation described in the first approach. That is, Eq. (5.1) was approximated by Eq. (5.2). The difference here is that the translation system was trained differently. Specifically, the source part of each bilingual training pair was substituted by the transcription obtained by our handwriting recognition system. This approach was expected to better handle the noisy output of the handwriting recognition system.

In the third approach Eq. (5.1) is approximated as follows,

$$y^\star \approx \underset{x \in \text{NBest}(f)}{\operatorname{argmax}} \left\{ \underset{y \in \text{NBest}(f|x)}{\operatorname{argmax}} \left\{ p(x|f)\ \ [p(y|x)]^\theta \right\} \right\} \tag{5.3}$$

where we introduced a scaling factor θ, and the search space was approximated by N-best lists. Specifically, each input image was first recognized using our Handwriting Recognition System into 100-Best transcriptions, and then each transcription was translated using our Machine Translation System into 100-Best translations. The optimal scaling factor θ is found using a grid search in a development set so as to maximize the Bilingual Evaluation Understudy (BLEU) [13]. BLEU is a measure that assesses the correspondence between a machine's output and that of a human.

5.3 Handwriting Recognition System

Handwritten Text Recognition (HTR) is the process of transforming an image containing handwritten text into plain text [14]. Arabic HTR is a very challenging task due to a huge variation in the handwritten text style and size and the immense overlapping between characters. It has caught the interest of many researchers in the last decades. Many models have been tested on this task including Gaussian and Bernoulli HMMs applying generative and discriminative training, as well as the neural networks for both acoustic and language models. Bernoulli HMMs applying discriminative training have shown better performance over Gaussian HMMs [15]. In fact, as mentioned in previous chapters, BHMMs have been ranked first in many competitions for Arabic printed and handwriting recognition [16–18]. In particular, in [17] our system obtained comparatively good results compared to state-of-the-art systems based on recurrent neural networks.

Our handwriting recognition system is based on windowed BHMMs (Sec. 3.6). To keep the number of independent parameters low, the WBHMM at sentence level (transcription hypothesis) is built from BHMMs at character level which depend on their surrounding characters, the so-called tri-character modeling approach [19]. Given a binary image normalized in height to H pixels, each BHMM computes the probability of the given image to be a handwritten version of its corresponding word. To compute these probabilities, text images are first transformed into a sequence of binary feature vectors $\mathbf{o}_t$ as its column at position t or, more generally, as a concatenation of columns in a window of W columns in width, centered at position t. This generalization has no effect neither on the definition of BHMM nor on its MLE, although it might be very helpful to better capture the image context at each horizontal position of the image. In addition, as discussed in previous chapters, this windowed approach are limited when dealing with vertical image distortions. To overcome this limitation, we applied vertical repositioning (only vertical due to its better performance over horizontal and in both directions). This technique is described in Section 3.7.

Parameter estimation is usually carried out using the maximum likelihood estimation (MLE). MLE of BHMM parameters does not differ significantly from the conventional Gaussian case, and it can be efficiently performed using the well-known EM (Baum-Welch) re-estimation formulae [20, 21]. However, discriminative training techniques have been used recently on Arabic HTR to estimate HMM parameters instead of using the conventional Baum-Welch algorithm. That is, discriminative training of Bernoulli HMMs are transformed into equivalent log-linear HMMs (LLHMMs). LLHMMs are then discriminatively trained using the MMI-criterion and the RPROP algorithm. This is usually referred to as discriminative BHMMs. For more details, please refer to [22]. Apart from the conventional BHMMs, which has been extensively explored previously, in this chapter we also carry out experiments using discriminative BHMMs.

5.4 Machine Translation System

The second part of Eq. (5.1), a Statistical Machine Translation (SMT) system follows the Bayes decision rule [23, 24] in which the optimal target sentence y is found by maximizing the posterior probability,

$$y^* = \underset{y}{\operatorname{argmax}}\ p(y \mid x), \tag{5.4}$$

where the posterior probability is modeled as a log-linear model where the normalization constant is neglected (since it is constant in decoding) [25]:

$$y^* = \underset{y}{\operatorname{argmax}} \sum_{m=1}^{M} \lambda_m h_m(x, y), \tag{5.5}$$

with λ_m being the log-linear interpolation weight, M is the number of features (models), and $h_m(x, y)$ is a feature function, such as the logarithm of a language model, or the logarithm of a phrased-based model.

In order to study the effect of the different modeling approaches in machine translation, we compared three state-of-the-art models: the standard phrase-based models [4], the hierarchical phrase-based [5] models, and the N-gram phrase-based models [6]. Obviously, we configured all models as similar as possible for fair comparison. As a results, three different translations have been gathered for each sentence, each one has been translated using a different model. In addition, all these translations were combined together using the Recognizer Output Voting Error Reduction (ROVER) [7] system to yield reduced the translation error rate. The reader might want to review the three state-of-the-art translation models by checking Section 2.3.3.

5.5 Experiments and OpenHaRT'13 evaluation

Our experiments were carried out on the data provided by the Linguistic Data Consortium (LDC) in the 2013 NIST Open Handwriting Recognition and Translation (OpenHaRT'13). In this section we first describe the corpus, the defined tasks, and the corresponding training conditions. Then, we continue by exploring our experiments with details, as well as comparing them to our own published results in the NIST evaluations.

5.5.1 NIST OpenHaRT database

The NIST Open Handwriting Recognition and Translation (OpenHaRT) corpus is a collection of Arabic image documents provided by the Linguistic Data Consortium (LDC). The data for training and development in the 2013 NIST OpenHaRT evaluation included: MADCAT Phase1 Training (LDC2010E15), MADCAT Phase2 Training (LDC2010E17), MADCAT Phase3 Training (LDC2011E97), MADCAT Phase1 Pilot Evaluation (LDC2008E52), MADCAT Phase2 Evaluation (LDC2012E52), and MADCAT Phase3 Evaluation (LDC2012E53). On the other hand, NIST has released the HART13 EVAL set without references for testing purposes. However, references for this set were released after the official publication of the NIST OpenHaRT'13 results. The 2013 OpenHaRT was not the first evaluation of its kind held by NIST. Similar evaluation was also arranged in 2010 (Sec. 3.9.2). Both evaluations focused on core recognition and translation technologies for document images containing primary Arabic handwritten scripts [8, 9].

NIST OpenHaRT has focused on three tasks: text recognition task which was refereed to as Document Image Recognition (DIR), text translation task, referred to as Document Text Translation (DTT), and image translation task known as Document Image Translation (DIT). In addition, training included two conditions: a constrained condition that required participants to develop their systems using only the provided LDC data, and an unconstrained condition in which participants are free to use any additional publicly available non-LDC resources for the system development (For more information, please refer to [9]).

For the constrained training condition our system was trained using the 2013 NIST OpenHaRT corpus. The corpus contains a total of $44K$ of Arabic image documents. A training partition was defined to include the five above mentioned sets containing $42K$ documents, a development partition contained 500 documents, and a testing partition containing 600 documents. Please note that the development set used in this work is the same set that was used as testing set for the 2010 NIST OpenHaRT evaluation. Also, the testing set in this work is the same set used in the 2013 NIST OpenHaRT.

For the unconstrained training condition our system was trained using some of the freely available data for building both translation and language models. The data used in this work has been obtained from *IWSLT 2011* challenge: *MultiUN* [26] and *TED* [27]. Since *MultiUN* corpus is not aligned at sentence level, we used the Champollion [28] tool to align the sentences. Finally, we selected sentences for the training set according to the infrequent N-gram score [29], in order to gather a specific training set to translate our source test sentences. The selection procedure was applied in the text translation system differently than in the image translation system. Particularly, for text translation system models were trained by selecting data using the original source test sentences. However, in the image translation system the source sentences are missing, so models were trained by selecting data using the recognized source sentences. It is worth noting that the number of sentences used for training from MultiUN was $155K$ and $75K$ out of $7M$ sentences for text and image translation respectively. Also, the selected sentences from TED was $48K$ and $47K$ out of $93K$ sentences for text and image translation respectively. Further statistics about each corpus used to train our recognition and translation systems in both training conditions (constrained and unconstrained) are shown in Table 5.1.

Table 5.1: Data (segments) used for training each system and its training conditions

	Constrained	Unconstrained	
System	LDC	MultiUN	TED
Text Recognition	$789,874$	-	-
Text Translation	$41,488$	$155,185$	$48,776$
Image Translation	$41,488$	$75,904$	$47,061$

As you can see from Table 5.1, for the recognition system with constrained condition we used around $790K$ of data lines. For the translation system with constrained condition we used around $41K$ of data segments, and with unconstrained condition we used around $245K$ of data segments for text translation, and around $164K$ for image translation. Note that segments do not necessarily match lines.

5.5.2 System Preparation

The handwriting recognition system was trained from input images scaled in height to 30 pixels while keeping the aspect ratio. Images are then binarized. For binarization, we used the well-known Otsu algorithm [30], which is a simple and robust method for reasonable clean images. A sliding window of width 9 using the vertical repositioning was applied, and thus the resulting input (binary) feature vectors for the BHMMs had 270 bits. In Arabic, the shape of a letter written at the beginning of the word is different from a letter written at the middle or at the end; So all Arabic transcriptions were encoded by adding the position information. Finally, the number of states per character was adjusted to 6 states for all BHMMs. Similarly, the number of mixture components per state was empirically adjusted to 128. Parameter estimation and recognition were carried out using the EM algorithm. Also, we used a 5-gram language model at character level. The language model was smoothed by linear interpolated estimates with absolute modified Kneser-Ney discounting. In addition, the grammar scale factor was adjusted to 30. The handwritten text recognition system was trained using the

TLK toolkit [1] which among other features implements Bernoulli Hidden Markov models (BHMMs). This toolkit was developed by the UPV.

For machine translation system, each source and target sentence was pre-processed. English text was tokenized and pos-tagged using the Freeling tool [31], and Arabic text was tokenized and pos-tagged using the *MADA+TOKAN* tool [32]. Additionally, long sentences (longer than 160 words) were removed. Word Alignment was performed using MGIZA++ [33]. Finally, standard training was performed on the training data, which included: alignment extraction, phrase extraction and MERT. Three systems were trained: The first system is based on the standard phrase-based models. It was trained using Moses toolkit [4] following the standard features: a phrased-based model that includes both direct and inverse phrase translation probabilities and both direct and inverse lexical weights, a language model, a distance-based reordering model, a word penalty, and a lexicalized reordering model. The second system is based on the hierarchical phrase-based models. It was trained using the Jane toolkit [5] following the hierarchical phrase extraction, optimization of log-linear feature weights, and parsing-based search algorithms. The third system is on the N-gram phrase-based models. It was trained using the Ncode toolkit [34] following the multiple N-gram language models estimated over bilingual units, source words and target words, lexicalized reordering, and several tuple models.

All these systems were trained using the standard features as mentioned in the user manual of each machine translation toolkit. In the case of the language model, we used a 5-gram model trained with SRI Language Modeling Toolkit (SRILM) [35], which was smoothed by linear interpolated estimates with absolute modified Kneser-Ney discounting.

5.5.3 Results

As mentioned above, our experiments were carried out on the LDC data provided for the NIST OpenHaRT’13 evaluation. In particular, we used the *HART13 EVAL* set for testing purposes. Below we explore our results for text recognition in Sec. 5.5.3, for text translation in Sec. 5.5.3, and for image translation in Sec. 5.5.3. For the recognition system, results are shown in terms of Word Error Rate (WER%) which estimates the percentage of misrecognized words. For the translation system, results are shown in terms of BLEU score [13], which is an algorithm to evaluate the quality of the translation text. In the case of WER: the lower percentage the evaluation reports, the better text recognition we get. However, in the case of BLEU: The higher score the algorithm reports, the better translation we get.

Text Recognition Results

For the handwritten text recognition task, Table 5.2 shows the WER% for the evaluation of two systems: The first one, which was presented for the NIST 2013 evaluation, is based on generative Bernoulli HMMs [36]. The second one is based on the discriminative Bernoulli HMMs, which is described briefly in Section 5.3 and with more details in [22]. The WER% for the two training conditions (constrained and unconstrained conditions) is shown. The only difference between them is that for the unconstrained training condition, we used a much bigger language model created with the help of the previously mentioned two corpus: *MultiUN* [26] and *TED* [27].

Table 5.2: Text recognition systems together with their Word Error Rate (WER%) for the constrained (Const) and Unconstrained (UnConst) data (2013= ”system due NIST’13”; 2014= ”this work”)

System Disc.	Year	WER [%]	
		Const	UnConst
Generative	2013	29.2	28.3
Discriminative	2014	27.7	**22.8**

As shown in Table 5.2, the results of the discriminative-based (2014) system outperforms those of the generative-based (2010) system for both training conditions. It is worth noting that our best results in similar task in the NIST OpenHaRT’10 evaluation [8] was ranked first with 47.45 of %WER for lines segmentation condition and constrained training condition. The reader can refer to the NIST report for more results about the NIST OpenHaRT’10 [8] and NIST OpenHaRT’13 [9] competitions.

The evaluation of this task was performed by using exactly the same tool that was published by NIST (OpenHaRT Pipline[a]) in its version number 1.1.2.

Text Translation Results

For the text translation task, Table 5.3 shows the BLEU score for five systems followed the constrained (Const) and unconstrained (UnConst) training conditions: The first system, Moses (2013), which is based on the standard phrase-based models, was ranked first in the NIST OpenHaRT’13 evaluation. The next three systems presented in this work (2014) are based on three different models: the standard phrase-based models (Moses), the hierarchical phrase-based models (Jane), and the N-gram phrase-based models (Ncode). The last system, ROVER, is the combination between the three mentioned models. It is important to mention that the results in this section and next section are not compared somehow to the results of the NIST OpenHaRT evaluation. This is due the calculation of the BLEU score on the tokenized and lowercased version of the testing corpus unlike the NIST OpenHaRT evaluations.

[a] ftp://jaguar.ncsl.nist.gov/openhart/resources/openhart-1.1.2-20130524-1526.tgz

Table 5.3: Text translation systems together with their BLEU score for the constrained (Const) and Unconstrained (UnConst) data (2013= "system due NIST'13"; 2014= "this work")

System	Year	BLEU [%]	
		Const	UnConst
Moses	2013	21.9	24.1
Moses	2014	25.1	28.1
Jane	2014	24.6	26.7
Ncode	2014	25.0	28.1
ROVER	2014	**25.6**	**28.6**

As shown in Table 5.3, the use of standard and N-gram phrase-based models showed better performance over hierarchical phrase-based models. In particular, the results for both Moses (25.1) and Ncode (25.0), which are pretty similar in the constrained training condition, outperform the results of Jane (24.6). In a similar way they showed better results for the unconstrained training condition. On the other hand, the combined system, ROVER, showed even better performance than Jane and Ncode.

Image Translation Results

For the image translation task, Table 5.4 shows the BLEU score for translating the recognized text for seven systems followed the constrained (Const) and unconstrained (UnConst) training conditions: The first three rows show the results of our systems submitted to the NIST OpenHaRT'13 evaluation for the three approaches mentioned in section 5.2. More precisely, the third approach was ranked first in the NIST OpenHaRT'13 evaluation. This set of systems were trained using generative BHMMs to perform the recognition and standard phrase-based models to perform the translation. On the contrary, the next four systems presented in this work (2014), which are based on the first approach as mentioned in Section 5.2 due to its better performance over the second approach, were trained using discriminative BHMMs to perform the recognition. For text translation, each system is based on one of the state-of-the-art models mentioned previously in Section 5.4: The standard phrase-based models (Moses), the hierarchical phrase-based models (Jane), the N-gram phrase-based models (Ncode), and finally a combined system between three mentioned models (ROVER).

As shown in Table 5.4, the use of standard phrase-based models showed better performance over the N-gram and hierarchical phrase-based models. In particular, the results using Moses (17.9) outperform the results using both Jane (17.3) and Ncode (17.4), which are pretty

Table 5.4: Image translation systems together with their BLEU score for the constrained (Const) and Unconstrained(UnConst) data (2013=”system due NIST’13”; 2014=”this work”)

System	Year	BLEU [%]	
		Const	UnConst
1st approach	2013	17.0	–
2nd approach	2013	16.5	–
3rd approach	2013	17.5	–
Moses	2014	17.9	21.0
Jane	2014	17.3	19.7
Ncode	2014	17.4	20.0
ROVER	2014	**18.3**	**21.2**

similar in the constrained training condition. In a similar way Moses showed better results for the unconstrained training condition. In addition, as in text translation task (Sec. 5.4), the combination system, ROVER, showed a slight improvement than each system apart in both constrained and unconstrained conditions. In Particular, results were 18.3 for the constrained condition, and 21.2 for the unconstrained task.

It is worth noting that errors in the recognition process usually remains in the translation, and they might trigger some additional errors. For example, we can notice some improvement of BLEU in the system presented in this work (2014), in particular for Moses, over the system presented in 2013. Indeed, this gain is explained for the improvement of the handwriting recognition system.

From Tables 5.3 and 5.4, it is worth noting that the usage of an additional small set of data (around $20K$) significantly improved the translation accuracy. More precisely, the Unconstrained data-based systems significantly outperforms the Constrained data-based systems. Here, we remind the reader that this additional data was selected according to the infrequent N-gram score [29], in order to gather a specific training set that relates to the source test sentences. The selection technique was applied differently in the text translation system than in the image translation system. In Particular, models for text translation system were trained by selecting data using the original source test sentences. However, for the image translation system, since source sentences are missing, models were trained by selecting data using the recognized source sentences.

5.5.4 Arabic Image Translation Example

In this section we visually inspect the process of recognition and translation of real examples. This process is shown in Table 5.5 using two Arabic image sentences taken from NIST OpenHaRT corpus. The rows "Image" and "Ref" show the original Arabic image sentence and its reference respectively. The "Reco" row demonstrates the recognition result of our discriminative BHMMs based system. On the other hand, each of the next rows show the translation results of the recognized text. The "Moses" row shows the output of the translation process using the standard phrase-based model. The "Ncode" row shows the output of the translation process using the N-gram phrase-based model. The "Jane" row shows the output of the translation process using the hierarchical phrase-based model. The "Comb" row shows the translation output using the combination of the three above mentioned systems. The "Google" row shows the translation using Google engine. The final row "Ref" shows the translation reference.

Table 5.5: Recognition and translation of two real examples taken from OpenHaRT database following the unconstrained conditions

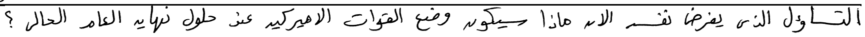
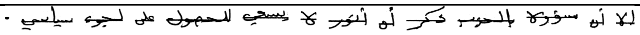
Image:	
Reco:	التساؤل الذي يفرض نفسه الان ماذا سيكون وضع القوات الاميركية عن حلول نهاية العام الحالي ؟
Ref:	التساؤل الذي يفرض نفسه الآن ماذا سيكون وضع القوات الأميركية عند حلول نهاية العام الحالي ؟
Moses:	the question that imposes itself now , what would be the status of the american forces by the end of the current year ?
Ncode:	the question that imposes itself now what will be the status of the american forces by the end of this year ?
Jane:	the same question now , what will be the status of the american forces on by the end of the current year ?
Comb:	the question that imposes itself now , what will be the status of the american forces by the end of the current year ?
Google:	question that arises now is what will be the status of u.s. forces from the end of the year?
Ref:	the compelling question now is what will the status of the american forces be at the end of this year ?

Image:	
Reco:	الا أن مسؤولا بالوس فكر أو أمور لا سعيه للحصول على لجوء سياسي .
Ref:	إلا أن مسؤولا بالحزب ذكر أن أنور لا يسعي للحصول علي لجوء سياسي .
Moses:	however , an official palouš think or other things do not sought to obtain a political asylum .
Ncode:	however , an official thinking or other things are not seeking to obtain political asylum .
Jane:	except that an official palouš a certain view or other things do not his efforts to obtain sought political asylum .
Comb:	however , an official palous think or other things do not sought to obtain a political asylum .
Google:	however, the responsible Palos thought or things do not quest for political asylum.
Ref:	however , a party official mentioned that anwar did not seek to obtain political asylum .

In the upper part of Table 5.5, we show an example in which both recognition and translation processes worked very well. In particular, in the recognized text the diacritics in the words الآن and الأميركية are missing. However, these errors did not change their meaning. In fact, these diacritics are also missing in the original input image. Regarding the translation process, the results make sense in all cases. It is true that neither of the translation outputs are identical to the reference, but the differences are noticed in the use of different word meaning such as using “current” instead of “this” or using “question that imposes itself” instead of “compelling question”. We believe that the closest translation to the reference is the output of the combined system in row “Comb”.

On the other hand, in the bottom part of Table 5.5, we show another example where both processes introduce errors. Moreover, we can notice that the mistakes in the recognition process trigger errors in the translation process. This is shown in words such as: بالحزب, ذكر, أن, أنور, and يسعي. The translation of these wrong recognized words has generated a totally different meaning which lead to reduce the BLEU score in translation. Again, the closest translation to the reference might be the output of the combined system in row “Comb”.

5.6 Conclusion

An image translation system has been described and tested on the LDC data provided for NIST OpenHaRT’13 evaluation (*HART13 EVAL* set). This system is built from the concatenation of two systems: a handwriting recognition system and a machine translation system. For the recognition part, Windowed Bernoulli HMMs with repositioning have been tested using generative and discriminative training. In previous work, generative-based system has proven to work very well with Arabic handwriting recognition. Specifically, this system was ranked first at the NIST OpenHaRT’10 evaluation with line segmentation condition. Following evaluation criteria similar to those of the NIST OpenHaRT evaluation, our results were improved by the use of discriminative training. For the translation part, we studied the effect of three state-of-the-art models: the standard phrase-based models, the hierarchical phrase-based models, and the N-gram phrase-based models. These models were combined using the Recognizer Output Voting Error Reduction (ROVER). For the image translation part, three different approaches were discussed so as to perform the concatenation between a recognition system and a machine translation system.

According to our experiments, a translation system trained using the original source test sentences (first approach) showed better results than a translation system trained using the recognized source sentences (second approach). Furthermore, previous approaches were clearly improved by approximating the search space using N-Best list (third approach). However, this last approach is not a practical solution to the image translation case. Therefore, our results in this work were performed by using the first approach.

The best Word Error Rate (WER%), 22.8, for Arabic handwriting recognition was obtained using discriminative Bernoulli training. For Arabic text translation, best results in BLEU score, 28.6, was obtained using ROVER. In a similar way, the use of ROVER technique has also obtained the best results, BLEU 21.2, for Arabic image translation. Results in this thesis are comparatively much better than those of both NIST OpenHaRT 2010 and 2013 evaluations, where a system based on phrase-based models only was ranked first in both text translation and image translation tasks.

Bibliography

[1] The transLectures Team - Universitat Politècnica de València. The translectures-upv toolkit (tlk). http://translectures.eu/tlk., 2013.

[2] I. Khoury, A. Giménez, and A. Juan. Arabic Handwriting Recognition Using Bernoulli HMMs. In V. Märgner and H. El Abed, editors, *Guide to OCR for Arabic Scripts*, pages 255–272. Springer London, 2012.

[3] I. Khoury, A. Giménez, A. Juan, and J. Andrés-Ferrer. Window repositioning for printed Arabic recognition. *Pattern Recognition Letters*, 51(0):86–93, 2015.

[4] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, and R. Zens. Moses: Open source toolkit for statistical machine translation. In *Annual meeting-association for computational linguistics*, volume 45, page 2, 2007.

[5] D. Vilar, D. Stein, M. Huck, and H. Ney. Jane: Open source hierarchical translation, extended with reordering and lexicon models. In *Proc. of the Joint 5th Workshop on Statistical Machine Translation and MetricsMATR*, pages 262–270, 2010.

[6] J. B. Marino, R. E. Banchs, J. M. Crego, A. de Gispert, P. Lambert, J. A. R. Fonollosa, and M. R. Costa-jussà. N-gram-based machine translation. *Computational Linguistics*, 32(4):527–549, 2006.

[7] J. G. Fiscus. A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (rover). In *Automatic Speech Recognition and Understanding, 1997. Proc., 1997 IEEE Workshop on*, pages 347–354, 1997.

[8] A. Tong. NIST 2010 Open Handwriting Recognition and Translation (OpenHaRT'10) evaluation. *Proc. of the NIST 2010 Open Handwriting and Recognition Workshop*, 2010.

[9] A. Tong and M. Przybocki and V. Märgner and H. El Abed. NIST 2013 Open Handwriting Recognition and Translation (OpenHaRT'13) evaluation. *Proc. of the NIST 2013 Open Handwriting and Recognition Workshop*, 2013.

[10] M. Pechwitz, S. S. Maddouri, V. Märgner, N. Ellouze, and H. Amiri. IFN/ENIT - DATABASE OF HANDWRITTEN ARABIC WORDS. In *7th Colloque International Francophone sur l'Ecrit et le Document, CIFED*, pages 21–23, Hammamet (Tunis), oct 2002.

[11] G. Kim and V. Govindaraju. A lexicon driven approach to handwritten word recognition for real-time applications. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(4):366–379, 1997.

[12] H. Cao, J. Chen, J. Devlin, R. Prasad, and P. Natarajan. Document recognition and translation system for unconstrained arabic documents. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 318–321, 2012.

[13] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: A method for automatic evaluation of machine translation. In *Proc. of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

[14] F. Zamora-Martínez, V. Frinken, S. E. na Boquera, M. Castro-Bleda, A. Fischer, and H. Bunke. Neural network language models for off-line handwriting recognition. *Pattern Recognition*, 47(4):1642 – 1652, 2014.

[15] P. Doetsch, M. Hamdani, H. Ney, A. Gimenez, J. Andres-Ferrer, and A. Juan. Comparison of bernoulli and gaussian hmms using a vertical repositioning technique for off-line handwriting recognition. In *Frontiers in Handwriting Recognition (ICFHR), 2012 International Conference on*, pages 3–7, Sept 2012.

[16] F. Slimane, S. Kanoun, H. E. Abed, A. M. Alimi, R. Ingold, and J. Hennebert. ICDAR 2011 - Arabic Recognition Competition: Multi-font Multi-size Digitally Represented Text. pages 1449–1453. IEEE, 9 2011.

[17] V. Märgner and H. El Abed. ICFHR 2010 Arabic Handwriting Recognition Competition. In *Proc. of the 12th Int. Conf. on Frontiers in Handwriting Recognition (ICFHR 2010)*, pages 709–714, Kolkata (India), 11 2010.

[18] V. Märgner and H. El Abed. ICDAR 2011 - Arabic Handwriting Recognition Competition. In *ICDAR '11*, pages 1444–1448, Beijing (China), sep 2011.

[19] G. Fink and T. Plotz. On the use of context-dependent modeling units for hmm-based offline handwriting recognition. In *Document Analysis and Recognition, 2007. ICDAR 2007. Ninth International Conference on*, volume 2, pages 729–733, Sept 2007.

[20] L. Rabiner and B.-H. Juang. *Fundamentals of speech recognition*. Prentice-Hall, 1993.

[21] S. Young et al. *The HTK Book*. Cambridge University Engineering Department, 1995.

[22] A. Giménez, J. Andrés-Ferrer, and A. Juan. Discriminative Bernoulli HMMs for isolated handwritten word recognition. *Pattern Recognition Letters*, 35(0):157 – 168, 2014. Frontiers in Handwriting Processing.

[23] P. F. Brown, J. Cocke, S. A. D. Pietra, V. J. D. Pietra, F. Jelinek, J. D. Lafferty, R. L. Mercer, and P. S. Roossin. A statistical approach to machine translation. *Computational linguistics*, 16(2):79–85, 1990.

[24] J. Andrés-Ferrer. *Statistical approaches for natural language modelling and monotone statistical machine translation*. PhD thesis, Universitat Politècnica de València, Valencia (Spain), Feb 2010. Advisors: A. Juan and F. Casacuberta.

[25] F. J. Och and H. Ney. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(4):417–449, 2004.

[26] A. Eisele and Y. Chen. Multiun: A multilingual corpus from united nation documents. In D. Tapias, M. Rosner, S. Piperidis, J. Odjik, J. Mariani, B. Maegaard, K. Choukri, and N. C. C. Chair), editors, *Proc. of the 7th conf. on Int. Language Resources and Evaluation*, pages 2868–2872. European Language Resources Association (ELRA), 5 2010.

[27] Ted corpus in the iwslt 2011 evaluation campaign, http://iwslt2011.org/doku.php?id=06_evaluation, 2011.

[28] X. Ma. Champollion: A robust parallel text sentence aligner. In *LREC 2006: Fifth International Conference on Language Resources and Evaluation*, page 489–492, 2006.

[29] G. Gascó, M.-A. Rocha, G. Sanchis-Trilles, J. Andrés-Ferrer, and F. Casacuberta. Does more data always yield better translations? In *Proc. of the 13th Conf. of the European Chapter of the Association for Computational Linguistics*, pages 152–161, Avignon, France, 4 2012. Association for Computational Linguistics.

[30] N. Otsu. A Threshold Selection Method from Gray-Level Histograms. *IEEE Trans. on Systems, Man and Cybernetics*, 9:62–66, 1979.

[31] L. Padró and E. Stanilovsky. Freeling 3.0: Towards wider multilinguality. In *Proc. of the Language Resources and Evaluation Conference (LREC 2012)*, Istanbul, Turkey, May 2012. ELRA.

[32] O. R. Nizar Habash and R. Roth. Mada+tokan: A toolkit for arabic tokenization, diacritization, morphological disambiguation, pos tagging, stemming and lemmatization. In K. Choukri and B. Maegaard, editors, *Proc. of the 2nd Int. Conf. on Arabic Language Resources and Tools*, Cairo, Egypt, April 2009. The MEDAR Consortium.

[33] Q. Gao and S. Vogel. Parallel implementations of word alignment tool. In *In Proc. of the ACL 2008 Software Engineering, Testing, and Quality Assurance Workshop*, 2008.

[34] J. M. Crego, F. Yvon, and J. B. Mariño. Ncode: an open source bilingual n-gram SMT toolkit. *The Prague Bulletin of Mathematical Linguistics*, 96(-1):49–58, 10 2011.

[35] A. Stolcke. SRILM-an extensible language modeling toolkit. In *Proc. of the Inter. Conf. on spoken language processing*, volume 2, page 901–904, 2002.

[36] A. Giménez, I. Khoury, J. Andrés-Ferrer, and A. Juan. Handwriting word recognition using windowed Bernoulli HMMs. *Pattern Recognition Letters*, 35(0):149 – 156, 2014. Frontiers in Handwriting Processing.

[37] I. Khoury, A. Giménez, J. Andrés, A. Juan, and J. A. Sánchez. The UPV Handwriting Recognition and Translation System for OpenHaRT 2013. *Proc. of the NIST 2013 Open Handwriting and Recognition Workshop*, 2013.

CHAPTER 6

Conclusions

Contents

6.1 Summary

The main goal of this thesis was to develop an Arabic image translation system based on the combination of an Arabic Handwritten Text Recognition (HTR) system and an Arabic to English Machine Translation system (MT). More precisely, this system has been proposed in conjunction as one system, that takes an image containing Arabic text and produces the translation of the recognized text into English, and in separation as two different systems. In addition to the HTR system, a system for printed Arabic text was introduced and extensively tested.

In chapter 3 the HTR system was introduced and evaluated on two different Arabic handwritten text corpora: IfN/ENIT database and the OpenHaRT 2010 database. This system is based on Bernoulli Hidden Markov Models (BHMMs). Experiments were carried out by applying a basic approach, in which narrow, one-column slices of binary pixels are fed into BHMMs. This approach was improved by the use of a sliding window of adequate width to better capture image context at each horizontal position of the word image. In addition to the window approach, three methods of window repositioning were considered after window extraction so as to help our BHMM-based recognizer in dealing with vertical image distortions.

Through the experiments, we have carefully studied the effects of the window width, fixed and variable number of states, and repositioning. As expected, the best results have been obtained with an adequate adjustment of the window width, number of states, number of mixture components and the vertical window repositioning. A WER of 6.1% has been achieved on the standard abcd-e partition of IfN/ENIT database. A system based on these techniques ranked first at the ICFHR 2010 Arabic handwriting recognition competition on IfN/ENIT. Also, following the same approach, a WER of 47.5% has been achieved on the OpenHaRT'10 database following the line segmentation condition. The system presented to this evaluation ranked first in the line segmentation condition and second in the word segmentation condition.

In chapter 4, the Arabic printed recognition system was described. Following a procedure similar to the one describe in chapter 3, the effects of the window width, fixed and variable number of states, and repositioning were tested on the Arabic Printed Text Image (APTI) database.

As expected, the use of sliding window and vertical repositioning proved again their ability to deal with text distortions, not only in Arabic handwritten text, but also in Arabic printed text. More precisely, A system based on the sliding window approach, though with no repositioning, ranked first at the ICDAR 2011 Arabic recognition competition for printed Arabic text on the APTI database. Furthermore, a system based on the vertical repositioning approach ranked second at the ICDAR 2013 Arabic recognition competition for printed Arabic text on the APTI database. Additionally, we also carried out experiments using the neural networks technology with vertical repositioning approach, which led us to state-of-the-art results in Arabic printed text

Finally, in chapter 5, an Arabic machine translation and image translation systems were proposed. For machine translation, the system was based on the combination of three state-of-the-art statistical models: the standard phrase-based models, the hierarchical phrase-based models, and the N-gram phrase based models. This combination was performed using the Recognizer Output Voting Error Reduction (ROVER) system. For Arabic image translation, three methods of combining an HTR system and an MT system were proposed. For the HTR system, the best approach based on vertical window repositioning was used (Chapter 3). For the MT system, we used the combination of the three previously mentioned state-of-the-art models.

Experiments in this chapter were carried out on the NIST OpenHaRT 2013 database. The best Word Error Rate (WER%) obtained for Arabic handwriting recognition was 22.8. For Arabic text translation, the best results obtained in BLEU score was 28.6. For Arabic image translation, the best results obtained in BLEU score was 21.2.

In summary, the main contributions of this thesis are the following:

- Bernoulli Hidden Markov Models (BHMMs) have been proposed for Arabic handwritten text recognition system. This system was extensively evaluated on two different corpora for Arabic handwritten text: IfN/ENIT and OpenHaRT 2010.
- BHMMs have been also proposed for Arabic printed text recognition system. Experiments were carried out on the APTI database for Arabic printed text.
- A Machine Translation (MT) system has been developed for Arabic text. This system is based on the combination of three state-of-the-art models.
- An Arabic image translation system was proposed to produce English translations from images containing Arabic text. This system has taken into account the best HTR system and the best MT system. Three methods were proposed for the combining both systems.

6.2 Scientific Publications

A major part of this thesis has been recognized in international competitions and articles in workshops, conferences and journals. In this section, we point out these contributions to the scientific community.

As mentioned previously, chapter 3 was dedicated to the Arabic handwritten text recognition system. Below we list the contributions related to this chapter. These contributions are sorted by year of publication.

We begin with the basic BHMMs and it's first application on the IfN/ENIT database:

- **I. Khoury**, A. Giménez, and A. Juan. Arabic Handwritten Word Recognition Using Bernoulli Mixture HMMs. In Proc. of the 3rd Palestinian Int. Conf. on Computer and Information Technology (PICCIT 2010), Hebron (Palestine), Mar. 2010.

This approach has been improved by using a sliding window of adequate width to better capture the image context.

- A. Giménez, **I. Khoury**, and A. Juan. Windowed Bernoulli Mixture HMMs for Arabic Handwritten Word Recognition. In Proceedings of the 12th International Conference on Frontiers in Handwriting Recognition (ICFHR), pages 533–538, Kolkata (India), November 2010. IEEE Computer Society.

Then, we continued by participating in the first edition of NIST OpenHaRT 2010 evaluation [1] using our windowed BHMMs approach. Our system ranked first in the HTR task following the line segmentation condition, and second following the word segmentation condition. Also, we participated in the ICFHR 2010 competition [2] using the vertical repositioning techniques. In this competition our system ranked first.

To sum up all our work about BHMMs and IfN/ENIT, in 2012 we participated in publishing one chapter in the following book.

- **I. Khoury**, A. Giménez, and A. Juan. Arabic Handwriting Recognition Using Bernoulli HMMs. In Volker Märgner and Haikal El Abed, Guide to OCR for Arabic Scripts, pages 255–272. Springer London, 2012. ISBN 978-1- 4471-4071-9.

In addition to those publications, also some work about Arabic HTR was done in collaboration with Adrià Giménez:

- A. Giménez, **I. Khoury**, J. Andrés-Ferrer, and A. Juan. Handwriting Word Recognition Using Windowed Bernoulli HMMs. Pattern Recognition Letters, 35(0): 149-156, 2012. ISSN 0167-8655. doi: 10.1016/j.patrec.2012.09.002.

Chapter 4 was dedicated to Arabic printed recognition system. Below a list of all publications related to this chapter is provided. As mentioned in this chapter, all experiments of chapter 3 on Arabic Handwritten text was extended to Arabic printed text. In the first article, only preliminary experiments were proposed. In the second article, we introduced the use of repositioning techniques with the state-of-the-art models based on neural networks.

- **I. Khoury**, A. Giménez, A. Juan, and J. Andrés-Ferrer. Arabic Printed Word Recognition Using Windowed Bernoulli HMMs. In 17th International Conference on Image, Analysis and Processing (ICIAP 2013), pages 330–339, Naples (Italy), Sep 2013.

- **I. Khoury**, A. Giménez, A. Juan, and J. Andrés-Ferrer. Window repositioning for printed Arabic recognition. Pattern Recognition Letters, 51(0):86 – 93, 2015. ISSN 0167-8655. doi: 10.1016/j.patrec.2014.08.009.

A system based on repositioning was submitted to two different Arabic printed text competitions at ICDAR 2011 and 2013. In ICDAR 2011 our system ranked first, while in ICDAR 2013 our system ranked second with only a marginal distance from the winner. However, in the PRL article mentioned previously, we managed to achieve state-of-the-arts results by using our repositioning approach and neural networks-based techniques. The results of both competitions can be observed in [3] and [4].

Chapter 5 was dedicated to Arabic-English machine translation and Arabic image translation systems. In this chapter, we proposed to combine our best HTR system based on repositioning techniques and a combination of three state-of-the-art machine translation models. This system obtained competitive results, which was submitted to the PAAA journal on October 2014:

- **I. Khoury**, A. Giménez, J. Andrés, and A. Juan. Image Translation System for Arabic Handwritten Text. Pattern Analysis and Applications (PAAA). (submitted)

In addition, the resulted system was published as a system description through our participation in the second edition of the NIST OpenHaRT 2013 evaluation [5]:

- **I. Khoury**, A. Giménez, J. Andrés, A. Juan, and J. Andreu Sánchez. The UPV Handwriting Recognition and Translation System for OpenHaRT 2013. Proc. of the NIST 2013 Open Handwriting and Recognition Workshop, 2013.
  URL: www.nist.gov/itl/iad/mig/upload/OpenHaRT2013_SysDesc_UPV.pdf.

Finally, during the making of this thesis I have collaborated in several publications, most of them were not directly related to the topic of this thesis. On of these articles is listed below:

- A. H. Toselli, N. Serrano, A. Giménez, **I. Khoury**, A. Juan, and E. Vidal. Language technology for handwritten text recognition. In Doroteo Torre Toledano, Alfonso Ortega Giménez, António Teixeira, Joaquín González Rodríguez, Luis Hernández Gómez, Rubén San Segundo Hernández, and Daniel Ramos Castro, editors, Advances in Speech and Language Technologies for Iberian Languages, volume 328 of Communications in Computer and Information Science, pages 178–186. Springer Berlin Heidelberg, 2012. ISBN 978-3-642-35291-1. doi: 10.1007/978-3-642-35292-8_19.

Bibliography

[1] A. Tong. NIST 2010 Open Handwriting Recognition and Translation (OpenHaRT'10) evaluation. *Proc. of the NIST 2010 Open Handwriting and Recognition Workshop*, 2010.

[2] V. Märgner and H. El Abed. ICFHR 2010 Arabic Handwriting Recognition Competition. In *Proc. of the 12th Int. Conf. on Frontiers in Handwriting Recognition (ICFHR 2010)*, pages 709–714, Kolkata (India), 11 2010.

[3] F. Slimane, S. Kanoun, H. El Abed, A. M. Alimi, R. Ingold, and J. Hennebert. ICDAR 2013 Competition on Multi-font and Multi-size Digitally Represented Arabic Text. pages 1465–1469. CPS, 08 2013.

[4] F. Slimane, S. Kanoun, H. E. Abed, A. M. Alimi, R. Ingold, and J. Hennebert. ICDAR 2011 - Arabic Recognition Competition: Multi-font Multi-size Digitally Represented Text. pages 1449–1453. IEEE, 9 2011.

[5] A. Tong and M. Przybocki and V. Märgner and H. El Abed. NIST 2013 Open Handwriting Recognition and Translation (OpenHaRT'13) evaluation. *Proc. of the NIST 2013 Open Handwriting and Recognition Workshop*, 2013.

List of Figures

List of Tables