Printed text and Handwriting recognition

by

Ashish Arora

A thesis submitted to The Johns Hopkins University
in conformity with the requirements for the degree of
Masters of science

Baltimore, Maryland
December, 2018
Abstract

This thesis focuses on modifying the open source speech recognition toolkit, Kaldi, to work for the task of handwriting recognition, also called text recognition. Various methods were explored to improve the performance of the text recognition setup. Text recognition refers to the automatic transcription of handwritten or printed text inputs from sources such as text page images, personal digital assistants, electronic white-boards or other devices. Text recognition can be performed in both online and off-line scenarios. Off-line recognition involves recognition of handwritten images whereas on-line recognition also stores the time trajectory information of each stroke.

Handwriting recognition has long been an active area of research and uses many of the same models used to perform automatic speech recognition (ASR). One such model used in both tasks is the Hidden Markov Model (HMM). In handwriting recognition, the text line images are treated as observations generated by underlying states representing the transcription. In this thesis, a hybrid deep-neural-network-HMM (DNN-HMM) acoustic model used for ASR was adapted for text recognition. To overcome a major challenge of out of vocabulary (OOV) words, a new subword based algorithm was implemented for lexicon and language modeling. Different data augmentation
and language specific modifications such as character decomposition, and bidirectional reordering were studied. To improve the performance of our text recognition setup, shared models, semi-supervised training and a recurrent neural network language modeling were also used. We investigated the performance of the text recognition setup on different languages, as well as when trained on varying amounts of data of different resolution and background. We report competitive results on several commonly used handwritten and printed text datasets.
Acknowledgments

I express my deepest and sincere gratitude to my advisors and mentors Dr. Daniel Povey, Dr. Sanjeev Khudanpur, Hossein Hadian and Dr. Jan Trmal for their invaluable suggestions and guidance during this project. Coming from a computer vision background, I was introduced to the field of Automatic Speech recognition (ASR) during a course taught by Dr. Khudanpur. I would like to thank him for his willingness to let me work in his lab and giving me the opportunity to participate in the SCALE workshop. I am extremely thankful to Dr. Povey and Hossein Hadian for their tremendous support and guidance over these past two years. I would like to thank them for assigning me tasks, helping through the tasks, reviewing the modifications and sharing elegant ways of performing the tasks. I would like to thank Dr. Trmal for his support and help during the mini-SCALE and SCALE workshop.

I am extremely thankful to Yiwen Shao and Hang lyu for sharing knowledge for image segmentation and ASR, performing preprocessing and image augmentation experiments for IAM handwriting recognition setup, building faster decoding setup for text localization and helping while installing different text recognition systems. I would like to thank Jonathan Chang; we both started working on text recognition together and helped each other during the
project. I am thankful to Jonathan Chang for doing experiments and sharing results for IAM, MADCAT Chinese, Yomdle Chinese, Yomdle Farsi, and UW3 dataset. I am extremely thankful to Matthew Wiesner for helping with the writing and David Etter, Paola Garcia for providing feedback for the thesis. I would also like to thank David Etter and Greg Sell for sharing synthetic data and helping with the data augmentations. I would like to thank Babak Rekabdar for his work on Byte pair encoding for MADCAT Arabic handwriting recognition setup, which later became the default approach for all other text recognition setups. I would like to thank Vimal Manohar for his help during semi-supervised experiments. I would like to thank Hainan Xu and Ke Li for their help with the RNNLM setup. I would like to thank Desh Raj for performing experiments with probabilistic sub-word on text recognition setups. I would like to thank Paul Mcnamee, Nolan King, Kevin Duh, John Steinberg, Frederick S. Richardson, Stephens Rawls, Michael Decerbo and Bridget Kennedy for the discussions during SCALE workshop. The work presented in this thesis is the joint work of the above collaboration, and none of it would have been possible without their contribution.

Thanks to all the other CLSP students and SCALE participants, faculties and staff. I would like to thank my parents for their continuous support. I would like to thank everyone else involved for being supportive and helpful.
Dedication

This thesis is dedicated to my Parents.
Table of Contents

Table of Contents vii

List of Tables x

List of Figures xii

1 Introduction 1

1.1 Motivation . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1

1.2 Overview . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2

1.3 Related work . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4

1.3.1 Feature extraction . . . . . . . . . . . . . . . . . . . . . . . 6

1.3.2 Optical model . . . . . . . . . . . . . . . . . . . . . . . . . 7

1.3.3 Lexicon and Language model . . . . . . . . . . . . . . . . 8

1.4 Contributions and Organization . . . . . . . . . . . . . . . . . 9

2 Dataset 11

2.1 Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . . 11

2.2 IAM and UW3 database . . . . . . . . . . . . . . . . . . . . . . 11
5 Dataset specific modifications

5.1 Introduction

5.2 Language

5.2.1 Arabic and Farsi language

5.2.2 Chinese and Korean language

5.2.3 French and English language

5.3 Yomdle and Slam specific modifications

5.3.1 Synthetic data

5.3.2 Semi-supervised training

5.3.3 Shared model

5.4 Summary

6 Conclusion and future work

6.1 Conclusion

6.2 Future work
List of Tables

2.1  Dataset  Number of lines and writers in each dataset  

3.1  GMM-HMM setups results at different stages for IAM dataset

3.2  GMM-HMM Baseline result for different datasets

3.3  WER at different stages for various datasets (CER for Chinese)

3.4  comparison of WER between GMM-HMM and flat-start DMM-HMM (CER for Chinese)

3.5  WER improvement from using Flat-startt alignments

4.1  WER improvement for unknown word decoding setup

4.2  WER improvement from different amount of training text for LM

4.3  WER improvement from using higher order LM

4.4  WER improvement from RNNLM rescoring on different datasets

4.5  Results for different lexicon size for IAM dataset

4.6  WER comparison between a word based setup and a BPE based setup
5.1 WER improvement for Arabic and Farsi datasets 37
5.2 WER improvement from modifications (CER for Chinese) 38
5.3 WER improvements from data augmentation 40
5.4 WER obtained from training acoustic model from real and synthetic data 42
5.5 WER improvement from semi-supervised training 43
5.6 WER improvement from including English data in training 44
# List of Figures

1.1 Decoding setup ........................................ 4

2.1 handwritten text sample line images ..................... 14

2.2 Yomdle/Slam sample line images ........................ 15

3.1 DNN-HMM Optical model ................................. 20

3.2 DNN architecture ......................................... 21

5.1 Text recognition setup ..................................... 36

5.2 Image augmentation ...................................... 41
Chapter 1

Introduction

1.1 Motivation

Since the advent of cuneiform in ancient Sumer, printed or handwritten text has been among the most important methods of transmitting and storing information. Digitizing and extracting information from text could potentially save thousands of human annotation hours. This technology as a number of potential applications which has motivated many companies to actively work on text recognition [1] [48] [10]. One such application is document processing, which consists of two main categories. The first category, deals with applications such as license plate recognition, check reader, credit card reader, postal address recognition, and requires the exact transcription of the document. The second category only requires a majority of the document to be transcribed correctly for later downstream tasks. Applications in this category include keyword spotting in handwritten medieval manuscripts, genealogy fact extraction from newspapers, named entity recognition on biomedical text images, or translation of text images into other languages.
Document processing can be further divided into text localization, script and language identification, and text recognition. Large vocabulary text recognition is a task of transcribing printed text or handwritten line images. Since both text line image recognition and speech recognition are sequence recognition problems, approaches used for transcription of text line images are also similar to speech recognition. Hence, many techniques for text line recognition are borrowed from speech recognition research. Kaldi [37] is an open source toolkit for speech recognition research, containing many algorithms related to speech recognition and provides example recipes for building speech recognition systems in various conditions. In this thesis, we focus on using Kaldi for off-line handwritten or printed text line recognition in different scenarios. We explore different techniques to improve the performance of our models for text line recognition.

1.2 Overview

Text recognition is a process of converting a sequence of image/pixel vectors into a sequence of words. In an hidden markov model (HMM) based text recognition system, the most likely word sequence is obtained by finding the maximum posterior probability word sequence given the observation sequence. Suppose $x$ represents the observation sequence and $w$ is an arbitrary word sequence. Then, the most likely word sequence $\hat{w}$ is obtained with the help of the following equations:
\[
\hat{w} = \arg \max_w p(w/x) \tag{1.1}
\]
\[
\hat{w} = \arg \max_w p(x/w) * p(w) \tag{1.2}
\]
\[
\hat{w} \approx \arg \max_w p_{\theta_1}(x/w) * p_{\theta_2}(w) \tag{1.3}
\]

Equation 1.1 and 1.2 find the most likely word sequence from the probability of a word sequence given the observation sequence. Since the actual probability distribution \( p(x/w) \), \( p(w) \) in equation 1.2 is unknown, the terms are modeled as parametric distributions \( p_{\theta_1}(x/w) \), \( p_{\theta_2}(w) \). The two terms in the right-hand side of equation 1.3 are modeled separately.

The first term is known as the optical model (OM), \( p_{\theta_1}(x/w) \). It provides the likelihood of the sequence of observations for a given word sequence.

Traditionally, the OM is modelled by an HMM with gaussian mixture model (GMM) emission probabilities. It is trained using a parallel corpus of utterances and transcription. The second term is the language model (LM), \( p_{\theta_2}(w) \). It assigns prior probabilities to word sequences \( w \). Statistical n-gram models are commonly used in the LM. It is usually trained on a large corpus of text.

An HMM-based text recognizer is shown in figure 1. It has three main components: an OM, a LM, and a lexicon. As described above, the OM and the LM provides the likelihood of the observation sequence and the prior probability of word sequence respectively. The feature extraction unit represents a mapping which converts a text line image input into a sequence of feature vectors. And the lexicon provides a mapping between a word level representation and
its sub-word units, typically characters or parts of characters.

![Decoding setup](image)

**Figure 1.1: Decoding setup**

### 1.3 Related work

Text recognition has long been an active area of research. It started gaining momentum after some success was shown on unconstrained offline English handwritten line images using a segmentation free hidden Markov model-based approach [41]. Soon after, to compare the results of different systems, the University of Bern published a large vocabulary English off-line handwriting data set [29].

Many techniques and systems have since emerged in the literature. For example Tesseract [43] is an optical character recognition (OCR) system for printed text maintained by Google. It performs sequence modeling on line images using recurrent neural networks (RNNs) trained with the connectionist temporal classification (CTC) objective function. VistaOCR [40] is a similar pytorch based system based long short-term memory networks (LSTM) and
CTC, but can be used for both printed text and handwritten text. Other common systems available are LAIA [39] and RetuRNN [12]. In addition to the common RNN architectures, RetuRNN also provides a multidimensional LSTM network [52], which has been the state of the art for handwritten recognition data sets.

A common characteristic among these open source toolkits is that they are specialized for OCR/HWR tasks and are not often used for ASR. One other motivation for this thesis is to extend an open source speech recognition toolkit to work for OCR/HWR tasks. As mentioned in section 1.2, HMM-based text recognition systems have three major components. In the subsections below, challenges in the text recognition system and the properties associated with these components will be discussed. The major challenges in text recognition can be roughly categorized as follows:

- Background complexity, page conditions like page degradation, overlay text, and random noise and image capture conditions like blurring, resolution, and illumination can play a major role in degrading transcription results.
- Different writing styles and pen conditions can cause challenges in capturing possible character variations.
- A large vocabulary size for some languages can result in high classification error and out-of-vocabulary (OOV) rate.

In this thesis, different approaches to improving the performance of our system for the challenges above were explored. Data preprocessing, feature
extraction, data augmentation and OM were performed to capture variation in the line images. The use of sub-word lexicons and LM were studied to overcome the challenges of large vocabularies and high OOV rates. The following subsections contain a brief discussion of some of the work most relevant to our proposed method.

1.3.1 Feature extraction

Traditional text recognition systems were based on first performing noise reduction and normalization on the line image and then extracting features from the line images. These systems performed preprocessing on the line image to reduce the data variation. Some common normalizations used were slant correction, skew correction, and contrast normalization [4].

**Handcrafted features**: The feature extraction module aims to find features useful in discriminating between characters. These features were extracted from a sliding window scanned horizontally over the text line image. Traditionally, features were manually designed either for binary [32] or gray-scale line images [30] [51]. They consisted of both statistical and structural features. Some statistical features include center of gravity, movement, concavity, and zero-crossing detection. Some structural features include the detection of loops [25], the direction of strokes, and the interconnection of curves, however, deep neural networks (DNN) based feature extraction outperforms these methods, as resulting features are less sensitive to noise and better able to generalize to new text.

**Feature learning**: In the last decade, feature extraction methods have
focused more on feature learning from raw pixels. Neural networks [7] were used for feature extraction. After training with pixel GMM-HMM models, the training data is realigned for use in a convolutional neural network (CNN) or an HMM model can be used to perform a second pass of training. The combination of a neural network with an HMM usually outperforms GMM-HMM systems using pixel input or handcrafted features.

1.3.2 Optical model

The OM calculates the likelihood of a word sequence for a given feature vector sequence. Hidden Markov models (HMM) have been popularly used as an OM for text recognition. For handwriting recognition, character HMMs are used as a basic OM and, are concatenated to form word or sentence-level OMs. Different neural OMs [33] were proposed exhibiting varying accuracy, latency, and computational complexity. In [7] a CNN was used as feature extractor for a GMM-HMM system. An HMM-LSTM model [22] has also been shown to improve the performance for handwriting recognition. In [47] a CNN followed by a BLSTM neural network under the HMM framework has effectively used to replace the GMM emission probabilities in HMMs.

Choosing an appropriate HMM topology can improve performance. For example, in [17] [13], the number of states for each character HMM was chosen using model length estimation. In [22] the HMM structure consist of segments, and each segment consists of 2 states sharing the same emission distribution.

Optical models can be trained either with frame-level loss criteria such as cross-entropy or with a sequential loss criteria such as connectionist temporal
classification, minimum Bayes risk, or mutual information [36]. Sequence
discriminative loss functions were shown to improve the performance of the
model over frame-level loss functions [53] [38] and have become widespread in
handwritten recognition research. In [53], using maximum mutual information
(MMI) as an objective function has been shown to improve the word error rate
(WER) over cross-entropy objective function. In [21], interpolation of MMI
[36] and CTC [14] was used as a loss function, and improved the performance
over systems trained on only one of the two losses.

1.3.3 Lexicon and Language model

Language models predict the probability of a word, given the sequence of
previous words. Sometimes OMs produce word sequences which are gram-
matically incorrect or is an unlikely word sequences. Using a LM in decoding
reduces the posterior probability of unlikely word sequences, significantly re-
ducing errors [8]. The LM can also help in scenarios where the OM probability
for two words is very similar (e.g. ‘Coarse’ vs. ‘Course’ or ‘Price’ vs. ‘Prize’).
Different LMs have been used in the literature. The most commonly used
LMs are statistical n-gram LM and recurrent neural network language models
(RNNLMs) [31]. The latter have an advantage of having a larger context than
the former.

As mentioned in Section 1.3.2, OMs of characters can be concatenated to
get word OMs. This concatenation is usually done with the help of a lexicon.
Using a lexicon, we can get an OM for any word present in the lexicon.
However, if a word is not present in the lexicon, then its LM likelihood cannot
be calculated. An approach to solve it is by recognizing the out of vocabulary words as a sequence of characters [23]. This approach therefore requires both a character level LM and a word level LM. Recently, approaches for automatic sub-word decomposition were proposed in machine translation [42] [24]. By using both the lexicon and language model at the sub-word level these approaches are also suitable for ASR.

1.4 Contributions and Organization

In this thesis, we build a text recognition system with the Kaldi toolkit. The thesis makes the following main contributions:

- The DNN-HMM acoustic model from an ASR setup was adapted for text recognition and hyper-parameters were tuned for different amounts of training data and image background conditions.

- To overcome a major challenge of large vocabulary and out of vocabulary words, a subword based algorithm to text recognition was implemented. The algorithm was applied to different datasets, and the datasets have different background conditions and languages. It consistently performed well in different conditions which shows the generalization ability of the approach.

- Different data augmentations were implemented and language specific modifications such as character decomposition, bidirectional reordering were performed.
• The system performs on par with other available systems, and has obtained state of the art results on some of the commonly used datasets in the literature. Also, to the best of our knowledge, this is the first study to be performed on many datasets under different languages and conditions.

The thesis is organized as follows. In Chapter 2 a brief description of datasets is provided, in Chapter 3 a word based text recognition setup and experiments to improve the OM is described, Chapters 4 describes the sub-word based text recognition setup and experiments to enhance the LM. The critical difference among the setups for different datasets is the data augmentation. The other significant difference is the amount of training data, which affects hyper-parameter such as the number of epochs, L2-regularization weight, and the number of jobs. In Chapter 5 we describe the data augmentations and modifications specific to the databases. Finally, Chapter 6 summarizes the thesis contributions and presents thoughts for future work.
Chapter 2

Dataset

2.1 Introduction

This chapter briefly describes the databases used during experiments. Details about the images and the text is provided, which can be useful in creating a better model for the setups. For example, information regarding training size and different characteristics of the line images can help in choosing a suitable number of epochs, layer sizes and the data augmentations used in training. Section 2.2 discusses the details related to an English handwritten and printed text dataset, Section 2.3 contains details regarding a French handwritten dataset, Section 2.4 regarding the MADCAT Arabic and Chinese handwritten datasets and Section 2.5 regarding the Yomdle and Slam printed text dataset for Tamil, as well as Russian, Chinese, Farsi and Korean.

2.2 IAM and UW3 database

IAM [29] is an English off-line handwriting recognition dataset created by University of Bern and its text is taken from the Lancaster-Oslo/Bergen (LOB)
corpus. Since, the IAM dataset provides line images, paragraph images and word images, it can be use for variety of handwritten recognition task. The images were scanned at a resolution of 300 dots per inch (dpi) and saved in gray scale. The dataset contains around 10k unique words and 79 unique characters. This corpus has a few distinct characteristic worth mentioning: The average length of a word is around 120 pixels, compare to 40 frames for a phoneme in speech; the width of punctuation is generally smaller than that of characters; some of the line images are written at significant inclination. Furthermore, the IAM dataset is commonly split into training, development and test sets in two different ways. The first is the original split provided with the dataset and the second is the Aachen split. Both splits are generally used while reporting results in the literature.

The University of Washington database (UW3) [35] is a collection of document images of printed English text. The UW3 database is an older database with clean, printed, English text. There are no established training and testing subsets so some arbitrary partition is used.

2.3 RIMES database

Rimes [3] is a French handwriting recognition database created by A2iA. Similar to IAM it can also be use for page level, line level and isolated word level recognition. It was created for the letter indexing task. The database was created by asking individuals to write letters on a given scenario like change of personal information, payment difficulty, damage declaration. The dataset has been used in several international research including ICFHR 2008,

2.4 MADCAT Arabic and Chinese database

The MADCAT (Multilingual Automatic Document Classification Analysis and Translation) Arabic Corpus [45] is an LDC dataset. It is a high quality database, with images scanned at a resolution of 600dpi. Writers were hired to create the dataset from news related passages and blogs. The XML file for each page provides line segmentation and word segmentation information as well as some writing conditions (writing style, speed, carefulness) of the page. The XML information contains the axis-oriented bounding boxes for the line images. However, a significant portion of the lines in corpus are not straight and the bounding box therefore include part of their surrounding lines. The dataset contains about 95k unique words and 160 unique characters. The dataset has been used in the NIST 2010 [34] and 2013 [49] Openhart Arabic large vocabulary unconstrained handwritten text recognition competition evaluation for line level recognition.

The MADCAT Chinese corpus [44] is also a LDC dataset. The major text is in Chinese, but it also contains English letters and numerals. The dataset contains 3k unique characters.
2.5 Yomdle and Slam database

YOMDLE (Yet One More Deep Learning Enterprise) developed a machine printed dataset that includes complex layouts of document images. It includes mobile camera images of books, newspapers, receipts, Power Point slides, social media and web pages. It contains around 1000 document page images per Language in Arabic, Chinese, English, Farsi, Hindi, Korean, Russian, and Tamil.

CASL (Center for Advanced Studies of Language) at the University of Maryland developed a machine printed dataset which primarily contain scans of documents and images taken from mobile cameras, in which the text is mainly machine printed. Some of the examples images are book scans, form scans, cell phone pictures of documents, images of social media posts, and memes. In contains around 2000 document page images and corresponding Line-Level Transcriptions with the primary language being Chinese, Korean, Farsi, Russian, and Tamil.
Table 2.1: Dataset Number of lines and writers in each dataset

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Number of line images</th>
<th>Number of writers</th>
</tr>
</thead>
<tbody>
<tr>
<td>IAM</td>
<td>9k</td>
<td>283</td>
</tr>
<tr>
<td>UW3</td>
<td>96k</td>
<td>-</td>
</tr>
<tr>
<td>Rimes</td>
<td>13k</td>
<td>1300</td>
</tr>
<tr>
<td>Madcat Arabic</td>
<td>740k</td>
<td>305</td>
</tr>
<tr>
<td>Madcat Chinese</td>
<td>184k</td>
<td>110</td>
</tr>
<tr>
<td>Yomdle</td>
<td>10k - 15k</td>
<td>-</td>
</tr>
<tr>
<td>Slam</td>
<td>10k - 15k</td>
<td>-</td>
</tr>
</tbody>
</table>
Chapter 3

Optical model

3.1 Introduction

As discussed in Chapter 1, an HMM-based text recognition system has three major components, an optical model (OM), language model (LM), and a lexicon. This chapter focuses on HMM-based OM for text recognition. Section 3.2 discusses the details related to the baseline setup HMM-GMM setup. In Section 3.3, we present the modifications for adapting an ASR setup for text recognition. Section 3.4 discusses the popular end-to-end neural network training. We present results from using different types of HMM AM. In this chapter all results are either with a word-based setup or with a character-based setup (for Chinese), a sub-word setup will be presented in chapter 4.

3.1.1 HMM based Optical model

The OM is the heart of text recognition setups. In some end-to-end neural network based systems, it is the only component which is compulsory for text recognition. An OM provides the probability of a character sequence given
the feature sequence. An HMM AM have been the most common in ASR for decades and is adopted for text recognition in this thesis. An HMM models a process in which a hidden signal can only be observed through another process which produces an observable signal. It provides a mapping between the observed signal (feature vector) and the hidden signal (characters). The inherent structure of HMMs allows for producing likelihoods for different word hypotheses given a sequence of feature vectors.

Traditionally, GMM-HMM systems have been used as AMs. The GMM models the probability of emission of a feature vector from a given state and the Markov chain models the probability of transition from one state to another state. DNN-HMM systems have certain significant advantages (discussed in section 3.3) compared to GMM-HMM systems that have translated to improved performance [50]. However, since the training data typically provides only the line level bounding boxes and the corresponding transcript, and not a pixel level alignment of the character in the transcript, our DNN-HMM system still relies on a GMM-HMM system for obtaining initial alignments between feature sequences and text. Recently, many have explored approaches to building DNN-HMM-based AM [19] [18] without relying on GMMs. In the next sections, we will discuss different approaches for HMM-based OM.

### 3.2 Baseline system

This section presents baseline results on the traditional GMM-HMM OM. A GMM-HMM OM provides the likelihood of a given feature vector sequence for a word hypothesis. Feature extraction from line images or raw pixels
extraction is generally performed to obtain a feature sequence. Since usually a second pass DNN-HMM system is trained using the output of GMM-HMM system, it has been shown [5] that raw pixel features can perform as well as the handcrafted features. Generally, a context dependent GMM-HMM OM has 3 main components: the total number of context-dependent HMM states, the total number of Gaussians and the number of space and non-space HMM states. Different techniques such as speaker adaption, combining delta features, applying the linear transform on the spliced features are adopted from ASR for training the GMM-HMM systems. The best parameters for each of these components are usually obtained empirically. The results, the word error rate (WER) and character error rate (CER) values, with the best parameters at different stages of GMM-HMM setup, are presented in Table 3.1.

<table>
<thead>
<tr>
<th>Model</th>
<th>WER</th>
<th>CER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monophone</td>
<td>66.6</td>
<td>46.9</td>
</tr>
<tr>
<td>Tri-delta</td>
<td>61.6</td>
<td>43.0</td>
</tr>
<tr>
<td>Tri-lda-mllt</td>
<td>54.4</td>
<td>40.0</td>
</tr>
<tr>
<td>Tri-sat</td>
<td>52.5</td>
<td>38.9</td>
</tr>
</tbody>
</table>

Table 3.1: GMM-HMM setups results at different stages for IAM dataset

Table 3.2 below shows the results of the best GMM-HMM stage for various datasets. From table 3.2, the baseline system gives good WER for printed text UW3 setup, but for handwritten datasets the results are not as good.
Table 3.2: GMM-HMM Baseline result for different datasets

<table>
<thead>
<tr>
<th>Dataset</th>
<th>WER</th>
<th>CER</th>
</tr>
</thead>
<tbody>
<tr>
<td>IAM</td>
<td>52.5</td>
<td>38.9</td>
</tr>
<tr>
<td>UW3</td>
<td>8.8</td>
<td>1.5</td>
</tr>
<tr>
<td>Madcat Arabic</td>
<td>61.1</td>
<td>37.3</td>
</tr>
<tr>
<td>Madcat Chinese</td>
<td>-</td>
<td>74.6</td>
</tr>
</tbody>
</table>

3.3 Neural Network training

In a DNN-HMM system, a neural network is used to estimate the emission probability from a frame vector instead of a GMM. A DNN-HMM-based AM is shown in figure 3.1 and has significant advantages over a GMM-HMM system. Some of the advantages are as follows:

- DNN-HMMs can model much more context compared to GMM-HMM systems.
- The features extracted by DNN-HMM systems are more discriminative than the raw pixel feature used in GMM-HMM.
- The DNN helps in learning a complex representation from the raw pixel feature vector and hence can work with diverse datasets.

Thus, DNN-HMMs generally outperform GMM-HMM system. However, the DNN-HMM system relies on GMM-HMM systems to obtain alignments between feature sequences and text for building a context dependent models. The second pass of training is often performed by replacing the GMM with a DNN. In the text recognition setup, after training with pixel GMM-HMM model, the training data is force aligned from the GMM-HMM system for
the CNN-TDNN-HMM model. While building a DNN-HMM system for text recognition the following considerations were made:

- Use a sequence-discriminative objective function, instead of frame-wise cross-entropy objective function is used.

- A CNN-TDNN-HMM setup was used because CNN can extract relevant features from the raw pixels and TDNN can effectively model the context on either side of the feature vector.

- L2-regularization, batch normalization, and scheduled dropout [11] are used for faster training and to avoid over-fitting.

- Since the features extracted from line images and features extracted from an acoustic recording files have different properties, different values...
for the tolerance, frame sub-sampling factor and chunk-width hyperparameters were used.

These considerations result in better modeling of the datasets. A DNN architecture with best parameters for IAM dataset is shown in figure 3.2. In the Subsection 3.3.1 below, we will compare this system with an ASR system.

![DNN architecture](image)

**Figure 3.2: DNN architecture**

### 3.3.1 Comparison with ASR

**Second stage LF-MMI training**: Similar to the ASR system, a second pass training with the DNN-HMM setup is performed. However, in the ASR setup, MFCC features are usually extracted when preparing the dataset instead relying on raw pixel features. Since MFCC features are uncorrelated, a GMM
with diagonal covariance can effectively learn the acoustic likelihood for each state. In a pixel GMM-HMM system for text recognition, since decorrelation is not guaranteed, degraded performance of GMM-HMM system is observed. Since the baseline system does not give good results, forced alignments for the baseline GMM-HMM system might not be very reliable. Hence, in comparison to the ASR system, a larger tolerance value was used during training so that the model has more freedom to relearn the alignments. Also, an additional second stage LF-MMI training was performed. In this stage, alignments from first LF-MMI training are used to train a second stage LF-MMI.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>GMM</th>
<th>chain.ali</th>
<th>second chain-ali</th>
</tr>
</thead>
<tbody>
<tr>
<td>IAM</td>
<td>52.5</td>
<td>19.2</td>
<td>15.5</td>
</tr>
<tr>
<td>Madcat Arabic</td>
<td>61.1</td>
<td>14.3</td>
<td>13.3</td>
</tr>
<tr>
<td>Madcat Chinese</td>
<td>74.6</td>
<td>16.6</td>
<td>14.7</td>
</tr>
<tr>
<td>UW3</td>
<td>8.8</td>
<td>5</td>
<td>4.4</td>
</tr>
</tbody>
</table>

Table 3.3: WER at different stages for various datasets (CER for Chinese)

Table 3.3 shows the WER with second stage LF-MMI training for different datasets. As can be observed from the table 3.3, second stage chain alignment system gives more improvement if the results of first stage CNN is comparatively worse. Similar to first stage chain alignment system, a higher tolerance is used.

**Number of HMM states:** The space between the written words in text recognition is equivalent to the silence between spoken words. However, acoustic silence usually has more variation in speech as compared to background space in text images. Hence fewer states (4) are used to model space between the words in text recognition than acoustic silence (5). Similarly, a
larger number of states (8) for non-space characters improved the performance of the text recognition system. Since the best system had more states for each character, a larger frame sub-sampling factor is used than in standard ASR systems. Similarly, since punctuation marks are small and have less variation, fewer states are used to recognize punctuation as compared to a standard character. Furthermore, since the Chinese language is logographic, a single character requires a greater number of HMM states than the used to model an English character.

3.4 GMM-free training

Recently, end-to-end models with sequential loss function for ASR [19] [18] have become very popular. These models can train DNN-HMM system without relying on GMM-HMM models. Table 3.4 shows the comparison of WER between GMM-HMM system and a flat-start DNN-HMM system. The flat-start DNN-HMM system has shown to give a significantly improved performance over GMM-HMM system for text recognition due to two major reasons.

- The flat-start system uses CNN for feature extraction, it can work well on the diverse background and low-resolution images, where GMM-HMM raw pixel feature system might fail.
- Since the Flat-start system can be built with a mono phone or left bi-phone context dependency, they provide a lot of improvement for languages such as Korean and Chinese which have huge number of unique
characters (around 3000), and it can become hard to build a context dependent GMM-HMM system.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>GMM WER</th>
<th>Flat-start WER</th>
</tr>
</thead>
<tbody>
<tr>
<td>IAM</td>
<td>52.5</td>
<td>15.4</td>
</tr>
<tr>
<td>Madcat Arabic</td>
<td>61.1</td>
<td>12.9</td>
</tr>
<tr>
<td>Madcat Chinese</td>
<td>74.6</td>
<td>13.3</td>
</tr>
</tbody>
</table>

Table 3.4: comparison of WER between GMM-HMM and flat-start DMM-HMM (CER for Chinese)

### 3.4.1 Second pass training using flat-start DNN-HMM

As mentioned in section 3.4, a flat-start DNN-HMM system can provide faster and significantly better results than GMM-HMM training. Hence, an alternative to using GMM-HMM setup for second pass training, the flat-start DNN-HMM setup can be used. It can also avoid the need of second stage DNN-HMM training. Table 3.5 shows the results of second pass training from using the forced alignments from flat-start DNN-HMM setup for different datasets. As seen in the table, the second pass chain alignment result for Madcat Arabic setup is very close to the flat-start result. One possible reason for this can be the large size of the database.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Flat-start WER</th>
<th>Chain-ali WER</th>
</tr>
</thead>
<tbody>
<tr>
<td>IAM</td>
<td>15.4</td>
<td>13.6</td>
</tr>
<tr>
<td>Rimes</td>
<td>13.0</td>
<td>9.0</td>
</tr>
<tr>
<td>Madcat Arabic</td>
<td>12.9</td>
<td>12.5</td>
</tr>
</tbody>
</table>

Table 3.5: WER improvement from using Flat-start alignments
3.5 Summary

In this chapter, different approaches to HMM-based OM are studied. Initially, GMM-HMM based baseline systems were explored, which gave good performance for clean print text line images. To improve the performance, a second pass training and later a second stage training with DNN-HMM was performed. It significantly improved the performance for both printed text and handwritten text datasets. To overcome the challenge of getting good alignments for diverse and challenging backgrounds, a GMM-free, end-to-end neural models were explored. They consistently gave good results in different conditions. Eventually, flatstart DNN-HMM based training and a second pass DNN-HMM training using the alignments from a flat-start training emerged as the best performing approach.
Chapter 4

Sub-word setup

4.1 Introduction

Text recognition and ASR are inherently open vocabulary tasks. But, many text recognition and ASR systems rely on a fixed vocabulary (closed vocabulary setup). As mentioned in Section 1.3.3, in a closed vocabulary word-based text recognition setup, if a word is not present in the lexicon (out of vocabulary word), it cannot be recognized. A dataset where a text image can contain URLs, abbreviations, names of medicines, people, etc. can require a highly specific vocabulary to achieve good performance in the closed vocabulary scenario.

This chapter focuses on the open vocabulary sub-word lexicon and LM setup for text recognition. The system models writing at the sub-word level in both the lexicon and LM, which allows for recognition of out of vocabulary words (OOV) as a sequence of sub-words. Subsections 4.1.1 and 4.1.2 discuss details related to two open vocabulary setups i.e. byte pair encoding (BPE) based setup and unknown word decoding setup. Section 4.2 describes the
sub-word LM. To overcome small context challenge faced in sub-word LM, a higher order LM and a recurrent neural network language model (RNNLM) [31] are studied. In Section 4.3, we present the sub-word lexicon setup and compares it with this unknown word decoding setup. We experimented with different OOV rates to show the usefulness of this approach.

4.1.1 Byte-Pair Encoding

Usually, an HMM-based text recognition system relies on a vocabulary. A closed vocabulary setup faces the following challenges.

- if a word is not present in the lexicon it cannot be recognized by the system.

- Rare words have very low LM probabilities and are often replaced with higher frequency alternatives.

An approach to deal with these challenges is to build an open vocabulary system [9]. This approach is more suitable for text recognition setups because all text recognition setups are graphemics system (primary unit in the lexicon is characters). BPE [42] is one such approach which can create sub-words from the text in a language independent, data-driven way. BPE compression is a greedy algorithm and requires a training text to learn sub-words. For each iteration, it greedily replaces a most frequent pair of bytes with a new byte-symbol. Since at each iteration a new symbol is produced, the number of symbols (vocabulary size) can be controlled by fixing the number of iterations. The minimum sub-word vocabulary size is the number of iterations, and the
maximum sub-word vocabulary is the number of unique words in the text.

### 4.1.2 Unknown word decoding setup

Another approach to building an open vocabulary which provides both a word probability along with the probability of an unknown word [23]. The unknown word can be recognized at the character level. Since the setup is simultaneously hypothesizing an unknown word and identifying it as a sequence of characters, it needs two language models (LM): a word based LM to hypothesize unknown word, and a character based LM for recognizing the word as a sequence of characters. The results for the word-based setup, with and without unknown word decoding support is presented in Table 4.1. As can be seen from the table 4.1, the unknown word decoding setup gives good improvement for the printed text UW3 setup, but for handwriting recognition setup the improvement is less. A possible reason for this is the relative easy recognition of clean and printed English text.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Baseline</th>
<th>Unk-decoding</th>
</tr>
</thead>
<tbody>
<tr>
<td>IAM</td>
<td>14.0</td>
<td>12.5</td>
</tr>
<tr>
<td>UW3</td>
<td>5.0</td>
<td>1.1</td>
</tr>
</tbody>
</table>

Table 4.1: WER improvement for unknown word decoding setup

### 4.2 Language model

Performance of an ASR or text recognition system significantly improves when using an LM in HMM-based recognition systems. The LM provides the probability of a current word given the word history. Different types of LM
have been used in the literature. The most common LMs are statistical n-gram LM and recurrent neural network based LM [31]. Both these LMs require training text to compute the conditional probability of a word given a word history. The probability of a word sequence can therefore be obtained from the Equation 4.1.

\[
p(w_1, w_2, \ldots, w_T) = p(w_1) \times p(w_2 / w_1) \times p(w_3 / w_1, w_2) \times \cdots \times p(w_T / w_1, w_2, \ldots, w_{T-1})
\]

\[
p(w_1, w_2, \ldots, w_T) = \prod_{i=1}^{T} p(w_i / w_{1:i})
\]

(4.1)

where

\[
w_{1:i} = (w_1, w_2, \ldots, w_{i-1})
\]

(4.2)

In the n-gram LM, the word histories that end with same n-1 words are considered identical. Hence, for n-gram LM equation 4.1 can be written as equation 4.3.

\[
p(w_1, w_2, \ldots, w_T) = p(w_1) \times p(w_2 / w_1) \times p(w_3 / w_1, w_2) \times \cdots \times p(w_T / w_{T-n+1}, \ldots, w_{T-1})
\]

\[
p(w_1, w_2, \ldots, w_T) = \prod_{i=1}^{T} p(w_{i-n+1} / w_{n:i})
\]

(4.3)

In practice, in a word based LM, a commonly used n-gram order is 3 or 4. Beyond order 4, the performance generally improves less due to an exponential increase in the training text required to estimate the probabilities. To overcome the data sparsity challenge and to model larger word context, RNNLM were introduced. Theoretically, RNNLM can provide infinite context
for a given word, and the effective word history is learned from the training text. The following subsections discuss the sub-word statistical and RNN LM.

### 4.2.1 Sub-word statistical LM

To overcome the data sparsity challenge mentioned in section 4.2, a large text corpus is usually used. The results with different amount of text for LM training is presented in table 4.2. From Table 4.2, it can be seen that addition of more text helps in better estimation of LM probability and also improves the WER for the Yomdle Russian setup.

<table>
<thead>
<tr>
<th>Text size</th>
<th>perplexity</th>
<th>5-gram WER</th>
</tr>
</thead>
<tbody>
<tr>
<td>265k</td>
<td>31.1</td>
<td>10.3</td>
</tr>
<tr>
<td>990k</td>
<td>24.7</td>
<td>9.9</td>
</tr>
<tr>
<td>3M</td>
<td>22.3</td>
<td>9.4</td>
</tr>
</tbody>
</table>

**Table 4.2: WER improvement from different amount of training text for LM**

In practice, a 3-gram LM is standard in a word based text recognition setup. It looks at the history of the last two words to compute the probability of the current word. In a sub-word setting, a 3-gram LM looks at the history of the previous two sub-words to give the probability of the current sub-word. Depending on the sub-word vocabulary size, a word can have around 1 to 6 sub-words. Hence, a 3-gram language model might not be sufficient to model the probability of current sub-word. Table 4.3 describes the improvement obtained from using higher order LM for both printed text and handwriting recognition setups.
<table>
<thead>
<tr>
<th>Dataset</th>
<th>3-gram</th>
<th>6-gram</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yomdle Tamil</td>
<td>11.6</td>
<td>10.8</td>
</tr>
<tr>
<td>Yomdle Russian</td>
<td>8.9</td>
<td>8.0</td>
</tr>
<tr>
<td>Yomdle Korean</td>
<td>22.1</td>
<td>17.8</td>
</tr>
<tr>
<td>Yomdle Farsi</td>
<td>13.3</td>
<td>12.7</td>
</tr>
<tr>
<td>Madcat Arabic</td>
<td>9.7</td>
<td>7.8</td>
</tr>
</tbody>
</table>

Table 4.3: WER improvement from using higher order LM

### 4.2.2 Sub-word RNNLM

Recently, RNNLMs have become popular in ASR and text recognition [55] [26]. An RNNLMs can provide infinite context for a given word, and the effective word history is learned from the training text. As mention in [26], if a word appears, the words related to that topic are likely to appear more. Hence, a larger history can be helpful for the LM. Since RNNs model the language in continuous space, it can effectively model the long-range dependency even with limited data. Hence it has been shown to improve performance over n-gram LM. Table 4.4 presents the improvement of performance from performing RNNLM rescoring over a n-gram LM.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>6-grm</th>
<th>RNNLM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yomdle Chinese</td>
<td>12.9</td>
<td>12.1</td>
</tr>
<tr>
<td>Gale Arabic</td>
<td>16.4</td>
<td>14.7</td>
</tr>
<tr>
<td>Gale Arabic BPE</td>
<td>16.3</td>
<td>15.2</td>
</tr>
</tbody>
</table>

Table 4.4: WER improvement from RNNLM rescoring on different datasets

### 4.3 Lexicon

A lexicon is an important component in HMM based text recognition systems. In addition to being a major component during decoding, it is also used during
training of OM and getting alignment for training data. It is a dictionary which stores the mapping between a word and a sequence of tokens. These tokens can be the sequence of characters (graphemic lexicon) or can be the sequence of phonemes (phonemic lexicon). As mentioned in Section 1.3.3, an OM of a sequence of characters can be concatenated to get a word sequence using the lexicon. We can get the likelihood from an OM for any word present in the lexicon. Hence, if a word is not present in the lexicon, then the likelihood from a feature sequence for that word cannot be calculated.

### 4.3.1 BPE Vocabulary size

As mentioned in section 4.1, the BPE vocabulary can be controlled by fixing the number of iterations. A change in vocabulary size of a sub-word system can affect the setup in following ways.

- As the vocabulary size increase, the average number of character per sub-word also increases.

- Larger vocabulary size will require more text in an n-gram LM to have robust n-gram estimates.

- Using a larger vocabulary size will increase the system latency due to larger search space during decoding.

Table 4.5 presents the comparison of performance across different vocabulary sizes. These results are obtained with the same OM but different lexicons and LMs.
<table>
<thead>
<tr>
<th>Vocabulary Size</th>
<th>WER</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>12.86</td>
</tr>
<tr>
<td>700</td>
<td>12.79</td>
</tr>
<tr>
<td>1200</td>
<td>13.07</td>
</tr>
<tr>
<td>2000</td>
<td>13.18</td>
</tr>
</tbody>
</table>

Table 4.5: Results for different lexicon size for IAM dataset

### 4.3.2 BPE based ASR setup

Another approach to reducing out of vocabulary (OOV) words is to increase the lexicon size. In a dataset where the utterance can contain URLs, abbreviations or names of medicines, people, etc. increasing lexicon size might not work well but for other increasing the lexicon can be useful. A basic approach to increase the lexicon size is to read unique words from an extra corpus text and add those words in the lexicon.

Since the Arabic script is phonetic (mostly), the approach to build a BPE based setup here is same as creating a BPE based setup for text recognition. A BPE based sub-word lexicon and LM is implemented for the Gale Arabic dataset. It has around 320 hrs of a training set and 9.3 hrs test set. For a word based setup, it uses an external lexicon which contains approximately 800k words. Almost all words in the test set are present in the dictionary. A part of Arabic gigaword corpus text (10M lines) is used to build a stronger 6-gram LM and RNNLM. Table 4.6 presents the comparison between using a word based setup with almost zero OOV rate and a BPE based setup. From the Table 4.6, in a scenario where all words are present in the lexicon, a word based setup can perform as well as a BPE based setup.
<table>
<thead>
<tr>
<th>LM</th>
<th>Word setup</th>
<th>BPE setup</th>
</tr>
</thead>
<tbody>
<tr>
<td>6-grm LM without external text</td>
<td>16.72</td>
<td>17.23</td>
</tr>
<tr>
<td>6-grm LM with external text</td>
<td>16.47</td>
<td>16.39</td>
</tr>
<tr>
<td>RNNLM with external text</td>
<td>14.71</td>
<td>15.29</td>
</tr>
</tbody>
</table>

Table 4.6: WER comparison between a word based setup and a BPE based setup

4.4 Summary

In this chapter, two different approaches to open vocabulary setup were studied. A hybrid word based LM and character based LM with unknown word decoding was explored which gave good performance for clean print text line images. A sub-word lexicon and LM was implemented to build a complete open vocabulary system. It seemed to perform better as compared to the first approach. In this approach, using a 6-gram LM and RNNLM rescoring improved the performance over the baseline 3-gram sub-word LM. The RNNLM models the text in continuous space and overcomes the data sparsity challenge faced by higher order LMs. Eventually, a sub-word lexicon, 6-gram LM and RNNLM rescoring was found to be the best approach.
Chapter 5

Dataset specific modifications

5.1 Introduction

In chapter 3 and 4, we discussed the experiments leading to improvement in the optical model (OM) and the language model (LM). Using alignments from flat-start DNN-HMM OM for training the second pass DNN-HMM OM and sub-word based lexicon and LM system were found to be best approach. Our best performing setup is shown in figure 5.1. The performance of the setup was investigated on different languages and varying amounts of data. A critical difference among these setups is different data augmentation. The other difference is the modifications due to dataset size and language. This chapter focuses on data augmentations and language-specific modification. Section 5.2 briefly describes some details related different languages and corresponding changes in the dataset. In Section 5.3, we present the modifications particular to the Yomdle and Slam datasets.
5.2 Language

5.2.1 Arabic and Farsi language

Arabic and Farsi are among the most widely used languages in the world. It is cursive both in machine print and handwritten form. A character can have up to 4 shapes, depending on whether it occurred in isolation, beginning, middle or end of the word. Also, it is written right to the left, but the numbers and math expression go left to right. A challenge in text recognition of these languages is due to the existence of unstructured ligatures. Most characters have a corresponding similar shape character that differs in number or position of dots. Apart from the dot, a character can also have small marks called diacritics. This can change the meaning of the words and can create difficulty in recognition. A detailed review of Arabic OCR and language features is provided in [27] [20]. Two databases, MADCAT Arabic and Yomdle Farsi, were used to perform experiments on Arabic and Farsi language. For both Yomdle
Farsi and MADCAT Arabic setup, following dataset specific modifications were made:


- Instead of using axis-aligned line images rotated rectangles were extracted. The rotated rectangle covered the line image with minimum area and helped in making the line straight and removing surrounding lines.

- The training bounding boxes of MADCAT Arabic were expanded to simulate test condition. Similarly, as SLAM dataset have low-resolution images, training images were randomly scaled down and then scaled up for data augmentation.

Table 5.1 shows the improvement in word error rate (WER) from using bidirectional text reordering and image augmentation. From table 5.1, a significant improvement is obtained by using a bidirectional reordering for Yomdle Farsi dataset, but not as much improvement for Madcat Arabic dataset due to significantly more numeric characters in the Slam Farsi.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Baseline WER</th>
<th>Improved WER</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yomdle Farsi</td>
<td>24.3</td>
<td>20.6</td>
<td>bidirectional reordering</td>
</tr>
<tr>
<td>Yomdle Farsi</td>
<td>14.6</td>
<td>14.0</td>
<td>Augmentation</td>
</tr>
<tr>
<td>Madcat Arabic</td>
<td>9.8</td>
<td>9.6</td>
<td>bidirectional reordering</td>
</tr>
<tr>
<td>Madcat Arabic subset</td>
<td>14.24</td>
<td>13.65</td>
<td>Augmentation</td>
</tr>
</tbody>
</table>

Table 5.1: WER improvement for Arabic and Farsi datasets
5.2.2 Chinese and Korean language

Chinese, Korean and Japanese belong to the family of oriental language. It is written at the character level and has no space between the words. There are around 3,000 characters in Chinese and 11,000 in the Korean language. Due to a massive number of characters and many characters being similar to each other, it is difficult to recognize them. A detailed review about 3 Oriental languages (Chinese, Japanese and Korean) from optical character recognition (OCR) point of view is presented in [46]. Three databases, MADCAT Chinese and Yomdle Chinese and Yomdle Korean were used to perform experiments on Chinese and Korean language. For the above databases, following dataset specific modifications were made:

- A characters model is used instead of a word or sub-word model for MADCAT Chinese.


- Line images were randomly scaled down for image augmentation.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Baseline Error rate</th>
<th>Improved Error rate</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yomdle Korean</td>
<td>16.0</td>
<td>12.5</td>
<td>character decomposition</td>
</tr>
<tr>
<td>Yomdle Korean</td>
<td>28.9</td>
<td>27.0</td>
<td>Augmentation</td>
</tr>
<tr>
<td>Yomdle Chinese</td>
<td>16.8</td>
<td>15.2</td>
<td>character decomposition</td>
</tr>
<tr>
<td>Madcat Chinese</td>
<td>13.3</td>
<td>6.8</td>
<td>character based model</td>
</tr>
</tbody>
</table>

Table 5.2: WER improvement from modifications (CER for Chinese)

Table 5.2 shows improvement from data augmentation and character decomposition. Character decomposition results are obtained with the same LM
but changing the lexicon to represent a character/sub-word as a sequence of decomposed character instead of a sequence of character. It effectively reduces the number of symbols to recognize from 3000 to around 250. Madcat Chinese row in the table describes the improvement obtained from using a character based setup instead of a word based setup for MADCAT Chinese. A reason for this improvement is as now the model has one less task of not predicting the space between words. Since in Yomdle Chinese dataset, also, Chinese line images a lot of English line images are also present a BPE based sub-word setup was built instead of a character based setup.

### 5.2.3 French and English language

Two databases, Rimes and IAM, were used to perform experiments on French and English language. For both Rimes and IAM setup, following dataset specific modifications were made:

- For Rimes dataset, training bounding boxes were expanded as data augmentation. Similarly, de-slanting and de-skewing [4] was applied to the IAM line images.

- Paragraph decoding was performed instead of line level decoding for Rimes dataset.

- A smaller HMM topology for punctuations was used as compared to the topology of other characters for IAM dataset.

The three different image augmentations mentioned above i.e. expanding bounding box, de-slanting and de-skewing and, random scaling for respective
datasets is shown in figure 5.2.

Table 5.3 describes the improvement obtained from performing data augmentation and paragraph decoding. From table 5.3, a significant improvement is achieved by paragraph decoding as Rimes dataset is prepared at the paragraph level.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Baseline WER</th>
<th>Improved WER</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rimes</td>
<td>9.0</td>
<td>8.3</td>
<td>augmentation</td>
</tr>
<tr>
<td>Rimes</td>
<td>7.1</td>
<td>6.1</td>
<td>paragraph decoding</td>
</tr>
<tr>
<td>IAM</td>
<td>9.9</td>
<td>9.1</td>
<td>augmentation</td>
</tr>
</tbody>
</table>

Table 5.3: WER improvements from data augmentation

5.3 Yomdle and Slam specific modifications

Yomdle organization and Center for Advanced Studies of Language (CASL) at the University of Maryland developed two printed text datasets Yomdle and Slam respectively. These datasets are created under diverse and challenging conditions. In addition to the training and test set containing a diverse set of images, the training and test set can belong to a different distribution. Hence to check the robustness of the OCR system, Yomdle dataset was used for training, and Slam dataset was used for testing. These datasets have five common languages: Korean, Chinese, Tamil, Farsi, Russian. In the below subsections, modification for improving performance on slam data using yomdle as training data are described.
Figure 5.2: Image augmentation
5.3.1 Synthetic data

Due to the advent of the deep neural network (DNN), the performance of OCR systems has significantly improved in the last decade. However, training a neural network requires a large amount of data. DNN-HMM-based OM for text recognition is trained using a parallel corpus of line images and transcriptions. Obtaining a large amount of parallel corpus data can be costly and time-consuming. Although it requires a large amount of data, for printed text recognition the variation in the text is limited and can be replicated using image augmentations. Hence, a synthetically generated data in different conditions can be used for training OMs. This subsection explores training OM with synthetic data and decoding on real slam data for Korean setup.

An advantage of using synthetic setup is an unlimited amount of training data. A Korean synthetic line image dataset with 430k line images was generated from a Korean newswire corpus. It had a random rotation, cropping, and padding, gaussian blur, and shearing as data augmentation. Table 5.5 describes the results obtained from using synthetic data as training set and slam data as the test set. As can be observed from table 5.4, comparable results can be obtained from both real and synthetic data.

<table>
<thead>
<tr>
<th>Stage</th>
<th>Synthetic setup WER</th>
<th>Real setup WER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flat-start</td>
<td>28.6</td>
<td>26.5</td>
</tr>
<tr>
<td>Chain-ali</td>
<td>22.6</td>
<td>19.9</td>
</tr>
<tr>
<td>Semi-supervised</td>
<td>20.90</td>
<td>17.5</td>
</tr>
</tbody>
</table>

Table 5.4: WER obtained from training acoustic model from real and synthetic data
5.3.2 Semi-supervised training

Since the Yomdle and Slam datasets have a diverse and challenging set of line images such as text images from books, newspaper, social media, poster, etc. The training data might not be sufficient to model this diversity. However, it is easy to obtain a large number of untranscribed line images. Semi-supervised training is one such approach which can utilize the untranscribed line images in training. A common approach to semi-supervised training is self-learning. Here an already trained model is used to generate a transcript for untranscribed line images, which are then used for second stage training of the model. Recently, lattice supervision based semi-supervised approach [28] has shown to perform better than self-learning approach for ASR datasets. In this approach, a pruned set of possible hypothesis along with their probability stored in the form of the lattice is used during training. Instead of using only best transcription it includes the lattice of word sequences obtained from the decoding of an utterance. Table 5.5 demonstrates the results from training the OM with the additional untranscribed line images.

<table>
<thead>
<tr>
<th>Language</th>
<th>Chain-ali WER</th>
<th>semi-supervised WER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yomdle Korean-decomposed</td>
<td>13.9</td>
<td>12.5</td>
</tr>
<tr>
<td>Yomdle Korean</td>
<td>19.9</td>
<td>17.5</td>
</tr>
<tr>
<td>Yomdle Tamil</td>
<td>10.6</td>
<td>9.5</td>
</tr>
</tbody>
</table>

Table 5.5: WER improvement from semi-supervised training

5.3.3 Shared model

As mentioned in section 5.3, line images of the Yomdle training set and slam test set have a different distribution. Both Yomdle and Slam datasets contain
English language line images. Since the given Yomdle language dataset might not provide sufficient English text for training, the OM was trained with an additional English dataset. For example, to decode Slam Tamil setup, the acoustic model was trained with both Yomdle English and Yomdle Tamil dataset. Table 5.6 demonstrates the results from training the OM with both English and given language.

<table>
<thead>
<tr>
<th>Language</th>
<th>Not including English WER</th>
<th>Including English WER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tamil</td>
<td>18.4</td>
<td>12.9</td>
</tr>
<tr>
<td>Russian</td>
<td>13.1</td>
<td>11.0</td>
</tr>
<tr>
<td>Korean</td>
<td>25.4</td>
<td>24.4</td>
</tr>
<tr>
<td>Farsi</td>
<td>13.8</td>
<td>13.7</td>
</tr>
</tbody>
</table>

Table 5.6: WER improvement from including English data in training

5.4 Summary

In this chapter, different dataset specific modifications such as data augmentation, paragraph decoding, hmm-topology, etc were explored. Language-specific modifications such as character decomposition, bi-directional reordering, character-based model, etc also seemed to help a lot. To improve the performance of text recognition setup, shared models, semi-supervised training and synthetic AM were studied.
Chapter 6

Conclusion and future work

6.1 Conclusion

In this thesis, we build text recognition setups with Kaldi toolkit. The problem of out of vocabulary (OOV) words was effectively removed by using byte-pair encoding (BPE) based sub-word lexicon and language model (LM). The setup was tested on different languages and varying amounts of data of varying resolution and background. We found that convolution neural network (CNN) and time delay neural network (TDNN) based optical model (OM), sub-word 6-gram LM and recurrent neural network language model (RNNLM) rescoring performed well in almost all cases. This thesis presents the following contributions text recognition:

- It successfully adapts the Kaldi software toolkit to perform handwriting recognition and optical character recognition (OCR).
- It applies a novel idea of sub-word decomposition from the automatic text translation literature, named BPE, to LM for OCR and handwriting recognition, and demonstrates that not only does the decomposition
not weaken the model much, but that the reduced vocabulary size and elimination of OOV words significantly improve recognition.

- It demonstrates the effectiveness of the two advances noted above via a comprehensive empirical evaluation in Arabic, Chinese, English, Farsi, French, Korean, Russian and Tamil on seven different datasets commonly used/cited in the OCR and handwriting recognition research literature. The Kaldi-based OCR/handwriting recognition systems match or outperform state of the art in almost all cases.

6.2 Future work

In this thesis, we restricted our attention to text line image recognition where line images were obtained from the manual annotation. Although, simultaneous work towards automatic extraction of line image from page image was also performed. More recently, end-to-end full-page text recognition [54] have become popular and have shown to perform well on some popular datasets in the literature. As a future work, we will explore different techniques to combine text localization and text recognition and improve the performance of our models for full-page text recognition.
Bibliography


[34] Kay Peterson. “OpenHaRT”. In: (2010).


[40] Stephen Rawls et al. “Combining Convolutional Neural Networks and LSTMs for Segmentation Free OCR”. In: *Proc. ICDAR*. 2017. DOI: 10.1109/ICDAR.2017.34. URL: https://doi.org/10.1109/ICDAR.2017.34.


[54] Curtis Wigington et al. “Start, Follow, Read: End-to-End Full-Page Handwriting Recognition”. In:

vita

Ashish Arora received the B.Tech. degree in Electrical and Electronic Engineering from Indian Institute of Technology, Guwahati in July, 2013. He worked as a Senior software developer engineer at Optym India private limited. He joined Masters program in the Electrical and Computer Engineering program at Johns Hopkins University in August, 2016. He became member of the Center for Language and Speech Processing in May, 2017. His research focus is on document processing. His previous related work was automatic scoring filled-in bubbles (OMR), signature and name extraction on printed forms, word segmentation based OCR for Hindi text. He is an active developer for the open source speech recognition toolkit Kaldi and image segmentation and text localization toolkit Waldo. He was a participant of the SCALE workshop for text recognition.