THE UNIVERSITY
of EDINBURGH

Multi-Dialect Arabic Broadcast Speech Recognition

Ahmed Ali

Doctor of Philosophy
Institute for Language, Cognition and Computation
School of Informatics
University of Edinburgh
2018

Lay Summary

Multi-dialect speech recognition is an important challenge due to the growing adoption of personal assistant devices and smart phones. In particular, Arabic poses an interesting challenge as the language has many dialects, and dialectal Arabic (DA) does not have standard orthographic rules. Despite the fact that there has been a great deal of speech recognition research in modern standard Arabic (MSA), which constitutes formal speech, there is still no open platform with standard lexicon and training data to benchmark results and advance the state of the art in Arabic automatic speech recognition (ASR). With regards to DA, it is lacking speech resources as well as appropriate methods for evaluating dialectal speech recognition. The standard word error rate (WER) metric assumes a single reference is sufficient for a single speech utterance, which is not true for non-orthographic languages, such as DA. This thesis concerns understanding and evaluating multi-dialect Arabic ASR without prior knowledge about the Arabic dialect that will be given as speech input. Therefore, we address the following three challenges: (1) finding labelled dialectal Arabic speech data, (2) building robust dialectal speech recognition with limited labelled data and (3) evaluating speech recognition for dialects with no orthographic rules. We make the following contributions:

Arabic Dialect Identification: We are concerned with Arabic speech without prior knowledge of the spoken dialect. Arabic dialects are sufficiently diverse to the extent that one can argue to describe them as different languages rather than dialects of the same language. Thus, automatically identifying the input dialect can greatly improve ASR. We look at two main groups of features: acoustic features and linguistic features. For the linguistic features, we look at a wide range of features; addressing words, characters and phonemes. With respect to acoustics, we look at raw features such as mel-frequency cepstral coefficients combined with shifted delta cepstra (MFCC-SDC), bottleneck features and the i-vector as a latent variable. In our work, we classify Arabic into five dialects: (*i*)Egyptian, (*ii*)Levantine, (*iii*)Gulf or Arabic peninsula, (*iv*)North African or Moroccan and finally (*v*)Modern Standard Arabic.

Arabic Speech Recognition: We introduce our effort in building Arabic speech recognition, and we create an open research community platform to advance it. We have two main goals: First, we create a framework for Arabic speech recog-

nition that is publicly available for research. We address our effort in building two multi-genre broadcast (MGB) challenges. MGB-2 focuses on broadcast news using more than 1,200 hours of speech and 130M words for text collected from Al Jazeera broadcast news channel and their website: `Aljazeera.net`. MGB-3, however, focuses on dialectal multi-genre data with limited non-orthographic speech data collected from YouTube, with special attention paid to transfer learning. Second, we build a robust Arabic speech recognition system and reporting a competitive WER results and use it as a benchmark to advance the state of the art in Arabic ASR.

Evaluation: The third part of the thesis addresses our effort in evaluating dialectal speech with no orthographic rules. Our methods learn from multiple transcribers and align the speech hypotheses to overcome the non-orthographic aspect. We have also automated this process by learning from Twitter data's different writing and we propose a new evaluation metric. Finally, we tried to estimate the word error rate with no reference transcription using decoding and language features. We show that our word error rate estimation is robust for many scenarios with and without the decoding features.

ملخص بسيط
(in Arabic)

يعد تحويل الكلام المسموع باللهجات المتعددة إلى كلام مكتوب بصورة آلية تطبيقا مهمًا، وذلك نظرا لزيادة الاعتماد على أجهزة المساعدة الشخصية والهواتف الذكية، ولزيادة الحاجة للتفاعل مع هذه الأجهزة بصورة طبيعية، خاصة من خلال التفاعل الصوتي. تتميز اللغة العربية بعدة سمات تزيد من صعوبة التعامل مع المدخل الصوتي، ولعل من أبرز هذه السمات: تعدد اللهجات بالإضافة إلى اللغة العربية الفصحى، وعدم وجود قواعد هجائية مستقرة للكتابة لأي من اللهجات (مثل اللهجات المصرية والسورية). ركزت أغلب الأبحاث السابقة على التحويل الآلي للعربية الفصحى، ولكن اللهجات العربية لم تحظى بنفس الاهتمام. فعلى سبيل المثال لا يوجد حتى الآن منصة مفتوحة مع تعريف واضح لبيانات التدريب، تمكن من قياس التقدم العلمي في التحويل الآلي للهجات ومعرفة مدي جاهزية التقنية للاستخدام في حياتنا اليومية، كما أنه لا توجد آلية مناسبة لقياس معدل الخطأ في التحويل الآلي للأصوات في اللهجات، حيث أنه قد يكون هناك العديد من الطرق لكتابة الكلمات العاميه بطرق مقبولة.

تختص هذه الرسالة بالأمور التالية:

١- تحضير بيانات صوتية مناسبة للهجات العربية المختلفة.

٢- تدريب نظام للتعرف على اللهجة العربية من الصوت وتحويل النص المنطوق إلى نص مكتوب.

٣- تصميم أسلوب مناسب لتقييم الأداء في التعرف على الكلام للهجات العربية.

هذه الرسالة تقدم ثلاث مساهمات:

- التعرف على لهجة المتحدث من خلال الصوت فقط. نقوم في هذا الجزء من البحث باستخدام الملامح الصوتية والملامح اللغوية ليقوم النظام بالتعرف على اللهجة العربية كواحدة من خمس لهجات، هي اللغة العربية الفصحى، اللهجة الخليجية، واللهجة المصرية، واللهجة الشامية، واللهجة المغاربية.

- تحويل النص المنطوق إلى مكتوب، حيث قمنا بتدريب نظام الشبكات العصبية العميقة على أكثر من ١,٢٠٠ ساعة صوتية تم تجميعها من قناة الجزيرة للتدرب على الملامح الصوتية وأكثر من ١٣٠ مليون كلمة للتدرب على الملامح اللغوية. كما قمنا بإنشاء منصة بحثية مفتوحة لدعم الأبحاث في صوتيات اللغة العربية.

- الجزء الثالث من الرسالة يتناول جهودنا في تقييم مستوى أنظمة تحويل النص المنطوق إلى مكتوب للهجات المختلفة التي لا يوجد قواعد إملائية ثابتة، وذلك من خلال تعلم طرق الكتابة المختلفة من وسائل التواصل الاجتماعية، مثل منصة تويتر، ومن خلال تعدد المراجع للنص المنطوق.

Abstract

Dialectal Arabic speech research suffers from the lack of labelled resources and standardised orthography. There are three main challenges in dialectal Arabic speech recognition: (*i*) finding labelled dialectal Arabic speech data, (*ii*) training robust dialectal speech recognition models from limited labelled data and (*iii*) evaluating speech recognition for dialects with no orthographic rules. This thesis is concerned with the following three contributions:

Arabic Dialect Identification: We are mainly dealing with Arabic speech without prior knowledge of the spoken dialect. Arabic dialects could be sufficiently diverse to the extent that one can argue that they are different languages rather than dialects of the same language. We have two contributions: First, we use crowdsourcing to annotate a multi-dialectal speech corpus collected from Al Jazeera TV channel. We obtained utterance level dialect labels for 57 hours of high-quality consisting of four major varieties of dialectal Arabic (DA), comprised of Egyptian, Levantine, Gulf or Arabic peninsula, North African or Moroccan from almost 1,000 hours. Second, we build an Arabic dialect identification (ADI) system. We explored two main groups of features, namely acoustic features and linguistic features. For the linguistic features, we look at a wide range of features, addressing words, characters and phonemes. With respect to acoustic features, we look at raw features such as mel-frequency cepstral coefficients combined with shifted delta cepstra (MFCC-SDC), bottleneck features and the i-vector as a latent variable. We studied both generative and discriminative classifiers, in addition to deep learning approaches, namely deep neural network (DNN) and convolutional neural network (CNN). In our work, we propose Arabic as a five class dialect challenge comprising of the previously mentioned four dialects as well as modern standard Arabic.

Arabic Speech Recognition: We introduce our effort in building Arabic automatic speech recognition (ASR) and we create an open research community to advance it. This section has two main goals: First, creating a framework for Arabic ASR that is publicly available for research. We address our effort in building two multi-genre broadcast (MGB) challenges. MGB-2 focuses on broadcast news using more than 1,200 hours of speech and 130M words of text collected from the broadcast domain. MGB-3, however, focuses on dialectal multi-genre data with limited non-orthographic speech collected from YouTube, with special

attention paid to transfer learning. Second, building a robust Arabic ASR system and reporting a competitive word error rate (WER) to use it as a potential benchmark to advance the state of the art in Arabic ASR. Our overall system is a combination of five acoustic models (AM): unidirectional long short term memory (LSTM), bidirectional LSTM (BLSTM), time delay neural network (TDNN), TDNN layers along with LSTM layers (TDNN-LSTM) and finally TDNN layers followed by BLSTM layers (TDNN-BLSTM). The AM is trained using purely sequence trained neural networks lattice-free maximum mutual information (LF-MMI). The generated lattices are rescored using a four-gram language model (LM) and a recurrent neural network with maximum entropy (RNNME) LM. Our official WER is 13%, which has the lowest WER reported on this task.

Evaluation: The third part of the thesis addresses our effort in evaluating dialectal speech with no orthographic rules. Our methods learn from multiple transcribers and align the speech hypothesis to overcome the non-orthographic aspects. Our multi-reference WER (MR-WER) approach is similar to the BLEU score used in machine translation (MT). We have also automated this process by learning different spelling variants from Twitter data. We mine automatically from a huge collection of tweets in an unsupervised fashion to build more than 11M n-to-m lexical pairs, and we propose a new evaluation metric: dialectal WER (WERd). Finally, we tried to estimate the word error rate (e-WER) with no reference transcription using decoding and language features. We show that our word error rate estimation is robust for many scenarios with and without the decoding features.

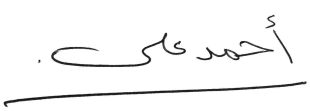
Acknowledgements

I would like to express my gratitude to the following people:

- Steve Renals, thank you for the patience and expert guidance, scientific freedom and the opportunity to pursue my PhD studies in the CSTR. It is a great pleasure to work with Steve.
- QCRI, thanks for giving me the opportunity to pursue my PhD while keeping my full-time job. I am very lucky to be part of the Arabic language Technologies (ALT) group. Special thanks to Stephan Vogel who always helped me to balance work with a study.
- My examiners: Phil Woodland and Hiroshi Shimodaira for peer reviewing this work and for the insightful comments which led to many improvements.
- My family, I owe too much for my family for the lack of time I spent with them. Special thanks for my wife for always being there when I need.
- Preslav Nakov and kareem Darwish, thanks a lot for the proof-reading and your help to wrap-up my thesis, and the discussion.
- ILCC colleagues: Peter Bell, Mirjam Wester, Alexandra Birch. Thank you for helping during my stay and study. You were the first people to ask when I have a problem or need help.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.



(*Ahmed Ali*)

Table of Contents

Part I

Background

This is the first section of the thesis and it has the following four chapters:

Chapter 1 introduces our contributions.

Chapter 2 presents Arabic as a language with an emphasis on the computational linguistic aspects of the language.

Chapter 3 offers an introduction to automatic speech recognition (ASR) and a brief summary of recent work in Arabic ASR.

Chapter 4 introduces language identification and dialect identification (DID). For the DID section, we focus on dialectal Arabic.

Chapter 1

Introduction

This thesis is concerned with the understanding and evaluation of *multi-dialect Arabic Automatic Speech Recognition (ASR)* without prior knowledge of the dialect given as an input. We primarily focus on four major Arabic dialects in addition to modern standard Arabic. This thesis is in line with a great deal of research on Arabic speech recognition that started more than fifteen years ago [Kirchhoff et al., 2003].

Despite recent success in Arabic ASR, the community lacks standard resources to advance the academic research as well as baseline results for understanding the dialect of the Arabic speech, converting speech to text, and finally measuring the quality of the ASR output with a metric that is suitable for dialectal Arabic.

This thesis offers three major contributions:

1. Arabic Dialect Identification: we mainly deal with Arabic speech without prior knowledge of the dialect. Arabic dialects could be sufficiently different to the extent that one can argue that they are different languages rather than dialects of the same language. We look at two main groups of features, namely acoustic and linguistic features. For linguistic features, we look at a wide range of features such as words, characters, and phonemes. With respect to acoustic features, we look at raw features such as mel-frequency cepstral coefficients (MFCC), bottleneck features and i-vectors as latent variables. In our work, we handle five major dialects of Arabic, namely Egyptian, Levantine, Gulf or Arabic peninsula, North African or Moroccan, and finally modern standard Arabic.

2. Arabic Speech Recognition: We introduce our effort in building Arabic speech recognition and in creating an open research community to advance it. We address two main points:

1. Creating a framework for Arabic speech recognition that is available for research. We focus our efforts on building two multi-genre broadcast (MGB) challenges. MGB-2, which focuses on broadcast news using more than 1,200 hours of speech and 130M words of text. MGB-3, which focuses on dialectal multi-genre data with limited non-orthographic speech data collected from YouTube with a special focus on transfer learning.

2. Building a robust Arabic speech recognition system, reporting a competitive word error rate, and using it as benchmark to advance the state of the art in Arabic.

3. Evaluation: The third part of the thesis addresses our efforts towards evaluating dialectal speech where the corresponding dialectal text does not adhere to orthographic rules. Our methods learn from multiple transcribers and align the speech hypothesis to overcome the lack of standard orthography. We also automate this process by learning from Twitter data's different word spellings, and we propose a new evaluation metric. Finally, we try to estimate the word error rate with no reference transcription using decoding and language features. We show that our word error rate estimation is robust for many scenarios with and without the decoding features.

1.1 Thesis structure

The reminder of the thesis is organised as follows:

- Chapter 2 gives an introduction to Arabic with an emphasis on the computational linguistic aspect of the language. It also covers the required principles for speech and language computing.

- Chapter 3 introduces automatic speech recognition (ASR) and briefly summarises recent work on Arabic ASR.

- Chapter 4 offers an introduction to language identification (LID) and dialect identification (DID). For the DID section, we focus on dialectal Arabic (DA).

- Chapter 5 highlights our efforts in building a dialectal Arabic (DA) corpus using a crowdsourcing approach to label the dialectal data. This corpus is to be used in the dialect identification work.

- Chapter 6 introduces our effort in building dialectal Arabic identification (DID) systems using data from the broadcast domain. We study the linguistic and acoustic features.

- Chapter 7 presents our effort in building Arabic speech recognition and creating an open research community to advance it. The chapter addresses two main areas: (*i*) Arabic broadcast domain with more than 1,200 hours and 130M words (also known as MGB-2), and (*ii*) 16 hours of dialectal data from Youtube for transfer learning and dialectal adaptation (also known as MGB-3).

- Chapter 8 proposes a novel approach in using multiple references to deal with the lack of orthographic rules in dialects to report more appropriate evaluation metric for dialectal ASR. The chapter also introduces *multi-reference* word error rate (MR-WER).

- Chapter 9 builds on chapter 8 to introduce a new method to automate multi-reference generation by learning different spelling variants from Twitter data and proposes a new evaluation metric, namely: dialectal word error rate (WERd).

- Chapter 10 addresses how to estimate the WER with no need for reference transcription and introduces robust quality estimation for large vocabulary speech recognition (LVCSR) system; word error rate estimation (e-WER).

- Chapter 11 concludes our contribution and discusses possible future work.

1.2 Declaration of content

The thesis is almost composed of the work published in the following journal, conference and workshop papers:

- A Ali, S Renals, "Word Error Rate Estimation for Speech Recognition: e-WER", in ACL 2018.
- A Ali, S Vogel, S Renals, "Speech Recognition Challenge in the Wild: Arabic MGB-3", in ASRU 2017.
- A Ali, P Nakov, P Bell, S Renals, "WERd: Using Social Text Spelling Variants for Evaluating Dialectal Speech Recognition", in ASRU 2017.
- S Shon, A Ali, J Glass, "MIT-QCRI Arabic Dialect Identification System for the 2017 Multi-Genre Broadcast Challenge", in ASRU 2017.
- S Khurana, M Najafian, A Ali, T Al Hanai, Y Belinkov, J Glass, "QMDIS: QCRI-MIT Advanced Dialect Identification System", in Interspeech 2017.
- F Dalvi, Y Zhang, S Khurana, N Durrani, H Sajjad, A Abdelali, H Mubarak, A Ali, S Vogel, "QCRI Live Speech Translation System", demo paper in EACL 2017.
- M Zampieri, S Malmasi, N Ljubešić, P Nakov, A Ali, J Tiedemann, "Findings of the VarDial Evaluation Campaign 2017", EACL 2017
- A Ali, P Bell, J Glass, Y Messaoui, H Mubarak, S Renals, Y Zhang, "The MGB-2 Challenge: Arabic Multi-Dialect Broadcast Media Recognition", SLT 2016.
- S Khurana, A Ali, "QCRI Advanced Transcription System (QATS) for the Arabic Multi-Dialect Broadcast Media Recognition: MGB-2 Challenge", SLT 2016.
- A Ali, N Dehak, P Cardinal, S Khurana, SH Yella, J Glass, P Bell, S Renals, "Automatic Dialect Detection in Arabic Broadcast Speech", InterSpeech 2016.
- A Ali, W Magdy, P Bell, S Renals, "Multi-reference WER for evaluating ASR for Languages with no Orthographic Rules", ASRU 2015.

- S Wray, H Mubarak, A Ali, "Best Practices for Crowdsourcing Dialectal Arabic Speech Transcription", ANLP workshop, ACL 2015.
- S Wray, A Ali,"Crowdsource a Little to Label a Lot: Labeling a Speech Corpus of Dialectal Arabic", InterSpeech 2015.
- MH Bahari, N Dehak, L Burget, AM Ali, J Glass, "Non-negative Factor Analysis of Gaussian Mixture Model Weight Adaptation for Language and Dialect Recognition", IEEE/ACM transactions on audio, speech, and language processing, 2014.
- A Ali, Y Zhang, S Vogel, "QCRI Advanced Transcription System (QATS)", demo paper SLT, 2014.
- A Ali, H Mubarak, S Vogel, "Advances in Dialectal Arabic Speech Recognition: A Study Using Twitter to Improve Egyptian ASR", IWSLT 2014.
- A Ali, Y Zhang, P Cardinal, N Dahak, S Vogel, J Glass, "A Complete Kaldi Recipe for Building Arabic Speech Recognition Systems", SLT, 2014.
- P Cardinal, A Ali, Dehak, Najim, Y Zhang, A Hanai, Tuka, Y Zhang, S Vogel, J Glass, "Recent Advances in ASR Applied to an Arabic Transcription System for Al-Jazeera", InterSpeech 2014.

Chapter 2

Arabic Language Background

2.1 Introduction

Arabic is a language spoken by over 350 million speakers (estimated in 2017), primarily known as Arabs. Arabic is the main language in more than 22 countries, which comprises the Arab league. Arabic, the language, has a great dialectal variety, with modern standard Arabic (MSA) being the only standardised dialect [Badawi et al., 2013]. MSA is syntactically, morphologically and phonologically grounded on classical Arabic (CA), the language of the Qur'an (Islam's Holy Book). Lexically, however, it is much more modern [Habash, 2010]. MSA is taught in schools across the Arab region and is the main language in news broadcasts, parliament and formal speech in general. Remarkably, MSA is not a native language of any Arab. Lay people in the Arab world use Dialectal Arabic (DA) as their way of communication; DA is the main language for drama, comedy programs in general in multi-genre broadcast. This is the day-to-day speech. Dialects used to be primarily spoken, not written. However, this has changed since the rise of Web 2.0. DA has become a written, as well as a spoken language. In general, Arabic can be classified into three groups:

- **Modern Standard Arabic (MSA) اللغة العربية الفصحى:** the official language of the Arab World also known as *fus'ha*. MSA is also known as the primary language of all Arabs. MSA is more often written than spoken.
- **Classical Arabic (CA) اللغة العربية التراث:** the language of the Qur'an (Islam's Holy Book). This can be seen as analogous to Shakespearean English. CA also known as *alturath*, used to be the main language in the

pre-Islamic time, more than 1400 years ago. CA is commonly used today in studying Arabic poetry and in Friday prayers' speeches in mosques.

- **Dialectal Arabic (DA) اللغة العامية:** the daily speech of Arabic native speaker [Maamouri et al., 2006]. DA is also known as *alamia* and is used in everyday speech such as phone calls and family discussions.

Arabic speakers typically do not make an explicit distinction between MSA and Classical Arabic. However, the relationship between MSA and the dialect in a specific region is rather complex. Arabs do not think of these two as separate languages. This particular perception leads to coexistence between the two forms of the language that serve different purposes. This kind of situation is what linguists term diglossia [Holes, 2004], where both MSA and dialectal Arabic exist side by side. Although the two variants have clear domains of prevalence: formal written (MSA) versus informal spoken (dialect), there is a large gray area in between that is often filled with a mix of the two forms.

2.2 Arabic Script

The Arabic script is alphabetically written from right to left. Arabic, the language, is written using Arabic, the script, which is also used to write many languages around the world that are not related to Arabic such as Persian, Kurdish, Urdu and Pashto. Arabic dialects are by default written in Arabic script, although there are no standard dialectal spelling systems. The attempt to call for spelling standardisation of the Arabic dialects is sometimes perceived as a challenge or even as a threat to MSA's hegemony.

Letters: The Arabic letters typically consist of two parts: First, the letter form, which is an essential component in every letter. There are 19 letter forms in total. Second, the letter marks, also called consonantal diacritics, which are mainly dominated by hamzas, and dots. Most commonly, the Arabic alphabet is said to have 28 letters or 29 depending on wheter the hamza is counted or not.

Arabic keyboard: Before the existence of smart devices, where it is easy to localise the keyboard, there were several tools that allowed their users to type in some form of a strict or loose romanisation, e.g., Yamli, Google's ta3reeb and Microsoft's Maren. Some websites, and operating systems also provide a phonetic keyboard for Arabic.

Buckwalter transliteration or UTF-8[1,2]: Buckwalter is a one-to-one mapping allowing non Arabic speakers to understand Arabic scripts, and it is also left-to-right, making it easy to render on most devices. The Buckwalter transliteration can be seen as the binary code for English, which makes it easy for a machine to understand. However, the Buckwalter format is mainly easy to read and debug by non-Arabic-literate researchers. Most Arabic language processing engines use Buckwalter to deal with the text and map one-to-one at the last stage before displaying it to users.

Diacritics also called diacritic, vocalization, vowelization and vowels: In MSA, letters are always written, while diacritics are optional: written Arabic can be fully diacritised, partially diacritised, or entirely undiacritised. Usually, the Arabic text is undiacritised except in religious texts, children educational texts, and some poetry. Some diacritics are indicated in modern written Arabic to help readers disambiguate certain words. In the Penn Arabic Treebank (part 3) [Maamouri et al., 2004], 1.6% of all words have at least one diacritic. There are three types of diacritics: Vowel, Nunation, and Shadda.

Normalisation: Orthographic normalisation is a basic task that researchers working on Arabic language processing always apply with a common goal in mind: reducing the noise and the sparseness in the data. This is true regardless of the task: preparing parallel text for machine translation, documents for information retrieval or text for language modeling. There are four letters in Arabic that are so often misspelled using variants that researchers find it more helpful to completely make these variants ambiguous (normalised). The following are the four letters in order of most commonly normalised to least commonly normalised [Buckwalter, 2007] (the first two are what most researchers do by default, the last two are less commonly applied).

- The Hamza forms of Alef are normalised to Alef
- The Alef-Maqsura is normalised to Ya
- The Ta-Marbuta is normalised to Ha
- The non-Alef forms of Hamza are normalised to the Hamza letter

[1]UTF-8 stands for Unicode Transformation Format. The '8' means it uses 8-bit blocks to represent a character. UTF-8 is a variable width character encoding, it is an efficient representation for many character representation and it has been used for Arabic for some time. Other encodings are also possible.

[2]https://en.wikipedia.org/wiki/UTF-8

2.3 Modern Standard Arabic

Modern Standard Arabic is the official language of the Arab world. MSA is the primary language of the media and education. TV hosts who read prepared scripts, for example on Al Jazeera, are trained to be careful in the pronunciation of certain phonemes (e.g., the realisation of the Classical **jiim** in MSA as **geem** by Egyptians).

Research on MSA in computational linguistics has been extensively investigated over the past two decades: exploring machine translation systems mainly focusing on MSA to English, innovations in speech recognition focusing on broadcast news and natural language processing. Furthermore, there were many treebanks, linguistic and speech corpora created and collected recently addressing MSA in both spoken and written formats. While MSA is the official language of the Arab world, almost no native speakers of Arabic sustain a continuous and spontaneous production of MSA.

2.4 Dialectal Arabic

Dialects are the primary form of Arabic used in unscripted spoken genres; such as conversations, talk shows and interviews. Since the rise of Web 2.0, dialects are increasingly in use in new written media, social networks, news forums, weblogs: e.g., more than 30M Arabic tweets are posted daily. Arabic dialects may be regarded as the true native language forms. How many Arabic dialects are there? There is no single answer for this, linguistics go as far as 27 different dialects of Arabic [Habash, 2010].

Arabic dialects vary across many dimensions: primarily by geography and social class. With respect to the geographical aspect of the language, the Arabic dialects can be divided in many different ways. The following is only one of many (and should not be taken as all members of any particular dialectal group being completely homogeneous linguistically):

- Egyptian Arabic (EGY) covers the dialects of the Nile valley: Egypt and Sudan.
- Levantine (LAV) Arabic includes the dialects of Lebanon, Syria, Jordan, Palestine and Israel.

- Gulf Arabic (GLF) includes the dialects of Kuwait, United Arab Emirates, Bahrain, and Qatar. Saudi Arabia is typically included although there is a wide range of sub-dialects within it. Omani Arabic is sometimes included as well.
- North African (NOR) Arabic (also known as Maghrebi) covers the dialects of Morocco, Algeria, Tunisia and Mauritania. Libyan Arabic is sometimes included too.
- Iraqi Arabic (IRQ) has elements of both Levantine and Gulf.
- Yemeni Arabic (Yem) is often considered its own class.

In this thesis, we will deal with the first four dialects[3] in addition to MSA, which sums up five classes: MSA, EGY, LAV, GLF and NOR. We can summarise the computational aspects in Arabic dialects to the following challenges:

2.4.1 Orthographic Variants

Table 2.1 shows two phrases across the different dialects. It is clear from this example that there are lexical variations across the different dialects including MSA [Ahmed et al., 2016].

EGY	GLF	LAV	MSA	NOR	English Gloss
ازايك AzAYk	اشلونك A$lwnk	كيفك / اشلونك kyfk / A$lwnk	كيف حالك kyf HAlk	واش راك wA$ rAk	How are you?
انت فين Ant fyn	وينك wynk	وينك wynk	اين انت Ayn Ant	وين راك wyn rAk	Where are you?

Table 2.1: *Lexical examples in Arabic and Buckwalter format.*

[3]We consider both Iraqi and Yemeni as a subset of the Gulf dialect for two reasons; Iraqi dialects became popular mainly because of the war in Iraq, thus DARPA funded research projects to understand it. However, we see it as a standalone dialect as most of the other dialects in the Gulf, Kuwait for example. While the Yemeni dialect could be different, it was not easy to get access to an abundance of data, and, therefore, we consider it as a subset of the Gulf dialect.

2.4.2 Phonological Variants

Arabic dialects vary phonologically from MSA and from each other. Some of the common variations are shown in table 2.2. In this table, example words are written as they will be pronounced by each dialect. For instance, in the first example, the name can be written as قاسم Qasm as in MSA, but will be pronounced as shown in the table. First example shows the MSA consonant (ق q) is pronounced as (ج g) in Gulf and as (ء ') in Egyptian and Levantine. The second example shows that the MSA alveolar fricative (ج j) is pronounced as (چ j) in Gulf and as (ج g) in Egyptian and Levantine. The last example shows that the MSA consonant (ذ *) is pronounced as (ز z) in Egyptian and North African and (د d) in Levantine. Sometimes the dialectal pronunciation impacts the translation as the name (قذافي q*Afy) which was translated to all the media as the name (جدافي Gaddafy) which is the actual Libyan way of saying the name. More details about phonetic features for Arabic can be found [Biadsy et al., 2009a,b].

EGY	GLF	LAV	MSA	NOR	English Gloss
ء (ءاسم) '('Asm)	ج (جاسم) g (qAsm)	ء (ءاسم) '('Asm)	ق (قاسم) q (qAsm)	ق (قاسم) q (qAsm)	Qasm (person name)
ج (جمل) gml	چ (چمل) jml	ج (جمل) gml	چ (چمل) jml	ج (جمل) gml	Camel
ز (هزا) hzA	ذ (هذا) h*A	د (هدا) hdA	ذ (هذا) h*A	ذ (هذا) h*A	This

Table 2.2: *Phonological examples in Arabic and Buckwalter format.*

2.4.3 Latin Script for Arabic

Before the rise of Web 2.0, and particularity the soft keyboard in smart devices, it was not easy for computer users to write Arabic letters, which is also represented in UTF-8. Therefore, there was a great deal in mapping the shape of some of the Arabic letters to the corresponding available shape on the English keyboard. Table 2.3 shows some of these examples. The importance of these challenges comes when there is a need to process lots of text and harvesting from a social platform, such as Twitter. This kind of writing is called: Arabizi, Arabish, or Franco-Arab. Some companies like Microsoft and Google developed software

capable of dealing with this kind of text and automatically convert it to Arabic script (UTF-8). We had to deal with this problem in this research as we processed a lot of Twitter data, and this was part of the text normalisation pipeline.

Arabic	Latin symbol
أإآءؤئ '<>&}	2
ح h	7
خ x	7’
ع E	3
غ g	3’

Table 2.3: *Latin script examples used for Arabic script.*

2.4.4 No Orthographic Standards

In a standardised language such as English, we know that *enough* is the correct spelling, while *enuf* is not. However, we cannot be sure about the correct spellings of dialectal words; at best, we would know what a preferred or a dominant spelling is. This is because dialects typically do not have an official status, and thus their spelling is not regulated, which widely opens the door to orthographic variation. Table 2.4 shows an example of the same word and various dialectal forms of writing it. All of them are widely used in social network and blogging and all formats are accepted too. Some researchers have looked at this challenge and built guidelines on how to write dialectal forms in a consistent way, such as CODA (a conventional orthography for dialectal Arabic). Nizar Habash has looked at automatically converting dialectal text to a standard convention, which is being termed as codifying the text, converting raw dialectal text to CODA and subsequently correcting various dialectal mistakes.

Spelling Variants	Buckwalter	English Gloss
ماكانش	mAkAn$	He was not
ماكنش	mAkn$	
ما كانش	mA kAn$	
مكنش	mkn$	
قولتله	qwltlh	I told him
قولت له	qwlt lh	
قلتله	qltlh	
قلت له	qlt lh	
على الصبح	ElY AlSbH	By the morning
علي الصبح	Ely AlSbH	
ع الصبح	E AlSbH	
عالصبح	EAlSbH	
عصّبح	ESbH	

Table 2.4: Dialectal phrases with multiple spelling variants: shown in Arabic script and in Buckwalter transliteration.

2.5 Arabic Dialects or Languages

It can be argued that a language is a dialect with an army and navy [Michalowski, 2006]. If we take this perspective into consideration, we can describe the different Arabic dialects as different languages. However, Arabs in general perceive dialects as a deterioration from the classical Arabic, almost using all the same Arabic letters. An objective comparison of the varieties of Arabic dialects could potentially lead to the conclusion that Arabic dialects are historically related, but not synchronically, and are mutually unintelligible languages like English and Dutch. Normal vernacular can be difficult to understand across different Arabic dialects [Holes, 2004]. Arabic dialects are thus sufficiently distinctive. Thus, from a computational prospective, we treat the Arabic dialects as different languages for tasks like dialect identification. However, for ASR, we consider all dialects to build Arabic background models and specific data from each dialect; both speech and text improves transfer learning to have a robust dialectal ASR for each dialect.

2.6 Dialect Identification and Codeswitching

In this thesis, we classify Arabic speech into five dialects: (*i*) EGY, (*ii*) LAV, (*iii*) GLF, (*iv*) NOR and finally (*v*) MSA. Our Arabic Dialect Identification (ADI) classification assumes that each speech segment corresponds to one Arabic native speaker is spoken in a single dialect. We assume that codeswitching happens when an Arabic native speaker switches between MSA and their own dialect.

2.7 Summary

This chapter gave basic background about Arabic, its script, phonetics, and dialects. We highlighted the two major classes of Arabic, dialectal and MSA. Given these highlights, the emphasis of the thesis is to deal with the multi-dialect Arabic speech recognition, and we will, therefore, study the two variants with their domains of prevalence; formal written (MSA) and informal spoken (dialect), in spite of the large grey area that is often filled with a mix of the two forms. This can be described as dialect diarization, where an Arabic native speaker switches between MSA and their own dialect. Also, it can be described as codeswitching. This thesis will illustrate some of these challenges, but will leave dialect diarization open for future research.

Chapter 3

Automatic Speech Recognition Overview

Automatic Speech Recognition (ASR) is defined as the process of realising acoustic speech audio into a corresponding word sequence W_h, which is the word sequence that is as close as possible to what a human could transcribe. The input speech data can be represented by a sequence of speech vectors or observations O. Thus, the challenge in predicting the most likely word sequence w_h can be described by solving equation 3.1, which has been described as the fundamental equation of statistical speech recognition [Clark et al., 2013].

$$W_h = \arg\max_w P(w|o) \tag{3.1}$$

Despite the fact that some of the recent work in speech recognition attempted to calculate the posteriors of word sequence directly [Graves and Jaitly, 2014, Hannun et al., 2014, Chorowski et al., 2014, Chan et al., 2015, Chorowski et al., 2015], the main stream in speech recognition still applies Bayes' rule to decompose equation 3.1 into the likelihood $P(o|w)$, the prior $P(w)$, and the denominator $P(o)$ which is independent of the word sequence, and it does not affect the search of the word sequence. Therefore $P(o)$ is removed.

$$\begin{aligned} W_h &= \arg\max_w \frac{p(o|w)P(w)}{P(o)} \\ &= \arg\max_w p(o|w)P(w) \end{aligned} \tag{3.2}$$

$p(o|w)$ is often refereed to as the likelihood of the acoustic model and $P(w)$ is described as the language model. Since the language model probability $P(w)$

does not depend on acoustics, it can be calculated independently, and can use different corpora.

It can be seen from equation 3.2 that the task of building a speech recognition system is based on two main modules: acoustic modelling and language modelling. The aim of acoustic modelling is to train a model that can explain the speech signals given an observation vector o. Due to the sequential nature of speech signals, hidden Markov models (HMMs) are found to be effective for this task [Rabiner, 1989]. The acoustic observation vector $o = [o_1, o_2, ..., o_T]$ is the outcome of the front-end signal processing extracted from the raw waveform, which ideally should be invariant with respect to extraneous factors to speech recognition such as speaker factors, pronunciation variability, and environmental noise. However, in practice, the feature processing step cannot normalise all of the variability and the acoustic models are expected to share the task. Language models, on the other hand, should try to predict the prior distribution of the word sequence w before the observation of speech signals. Conventional language models are based on the frequency of n-grams, which assume that the distribution of each word depends on the previous $n-k$ words, where k is the history of the language model, and is also known as the order of the language model. The rest of this chapter presents an overview of a standard speech recognition system, and at the end will shed light on recent efforts in Arabic speech recognition.

3.1 Front-end feature extraction

The raw waveform is received in continuous time and magnitude. The aim of the signal processing front end is to sample the raw acoustic waveform into feature vectors and to extract the acoustic features that are to be modelled by the acoustic modelling. Ideally, the acoustic feature representation for speech recognition will be compact, without losing much signal information, minimising variability across speakers and environmental acoustic conditions at the same time. The feature extraction step is language-independent, and often the extracted features do not retain information about the glottal source. However, in some case glottal features such as pitch is used Ghahremani et al. [2014]. The initial step relies on sampling the waveform into chunks, also known as windows; typically 25ms are processed with 10ms intervals. Intuitively, the extracted features should contain as much information as possible to distinguish between phones. The first section in figure

3.1 shows vectorising an audio file into overlapping samples and store feature vector representing a frame of the speech signal.

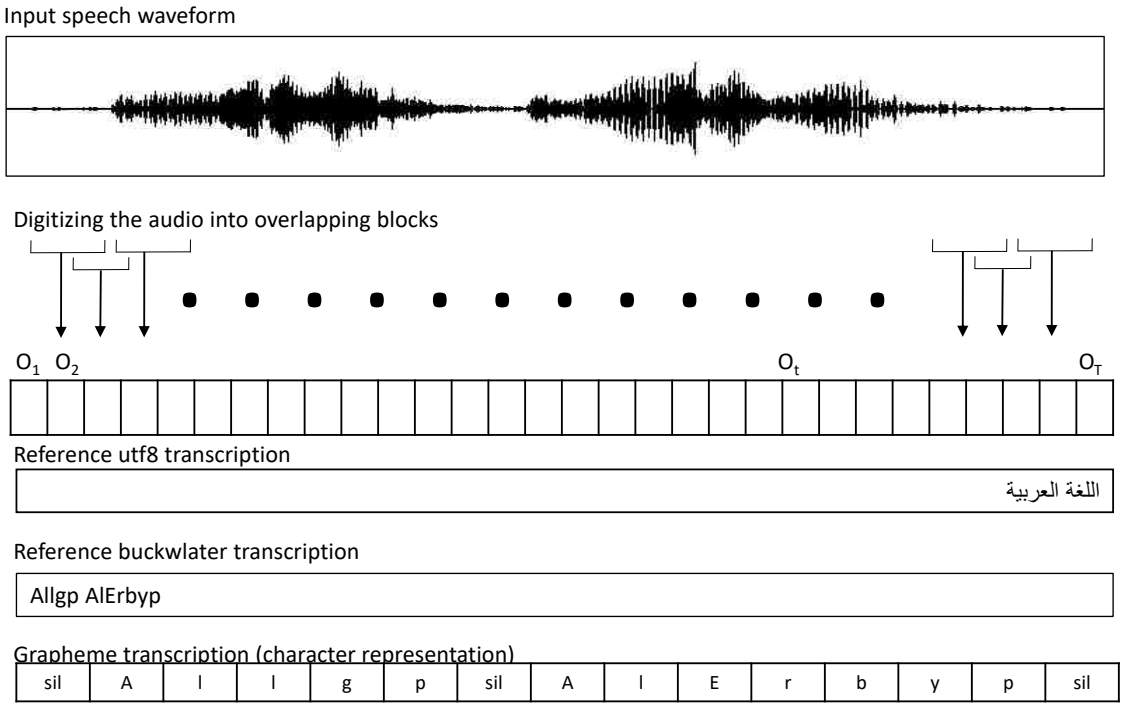


Figure 3.1: Diagram for speech sample digitised into a feature vector, and aligned with word transcription, as well as the corresponding grapheme sequence.

Although a variety of representations are used in speech recognition, one of the common approaches applies the short-time fast Fourier transform on each window to transform it into the spectral domain followed by transformation to power-spectra and smoothed by 20-40 mel filter-bank filters. This is done in order to perform an auditory-based warping of the frequency axis to account for the frequency sensitivity of the human hearing system. Those smoothed power-spectra are further logarithmically compressed and are referred to as mel-filter bank (FBANK) features, and are often used in deep learning and will be used in this thesis to train various neural network acoustic models. The FBANK features can be further processed for diagonal GMMs with a decorrelating discrete cosine transform (DCT) transform resulting in mel-frequency cepstral coefficients features (MFCC) [Davis and Mermelstein, 1980]. Another popular acoustic feature extractor is perceptual linear prediction (PLP) [Hermansky, 1990]. PLP relies on using bark scale to compute the filter-bank filters followed by a linear predictive analysis, from which one then derives a cepstral representation. Both MFCC

and PLP are used quite often in Hidden Markov Models (HMM), which will be discussed in depth in section 3.2.

The static MFCC and PLP features are extracted for each frame of windowed speech. However, since the speech is not constant frame-to-frame, there is a benefit from adding dynamic features to deal with how the cepstral coefficients change over time. Dynamic features can be calculated by simply using the difference method $\Delta O_{s,t} = O_{s,t+2} - O_{s,t-2}$ or by using linear regression to approximate the temporal derivative as shown in equation 3.3

$$\Delta O_t^r = \frac{\sum_{\delta=1}^{\Delta} \delta(O_{t+\delta}^r - O_{t-\delta}^r)}{2\sum_{\delta=1}^{\Delta} \delta^2} \tag{3.3}$$

One can derive dynamic features for an arbitrary order. However, there is a reduced gain after the second order. Therefore, the final feature vector used for speech recognition is the concatenation of static and dynamic coefficients also known as delta and delta delta, as shown in equation 3.4. Typical MFCC-based ASR features (mainly HMM) are 39 dimensions as shown in equation 3.4.

$$\begin{bmatrix} O_t \\ \Delta O_t^r \\ \Delta^2 O_t^r \end{bmatrix} \tag{3.4}$$

It is worth mentioning that with the rise of convolutional Neural Networks (CNN), there is further research to use less feature engineered from the raw audio data; spectrogram [Tüske et al., 2014, Hannun et al., 2014] and some recent work is modelling the raw audio files with no signal processing [Tüske et al., 2014, Palaz et al., 2015]. However, these will not be used in this thesis.

3.2 Acoustic modelling

The likelihood of the acoustic models $P(o|w)$ is typically estimated from a sizable corpus of speech segments with the corresponding word level transcription; speech segments are often less than 30 seconds each. There are three major challenges to estimate the aforementioned conditional probability directly; first, given that we only know the word sequence in the transcription, the frame-state alignment is unknown. Second, the length of the observation vector is of a variable length. Finally, the observation vector o is of high dimensionality, making direct estimation of the conditional probability difficult. One proposed solution for these

challenges is to model the joint probability $p(o, w)$ using a set of parametric models of word production having parameters μ. Within this generative framework, it is assumed that the sequence of observation vectors for a given word could be generated by a Markov model.

The hidden Markov model (HMM) has proven to be successful in acoustic modelling since it can estimate the time-varying nature of the speech audio quite well. Good reviews of using HMMs for speech recognition can be found in [Rabiner, 1989, Gales and Young, 2008]. HMM can be ergodic, which means that each state can be reached from any state in a finite number of steps. However, given the speech signal is varying in time, a left-to-right HMM topology has been used successfully. An example of an HMM with three emitting states and two non-emitting states is shown in figure 3.2, which are used as building blocks for most HMM-based speech recognition systems.

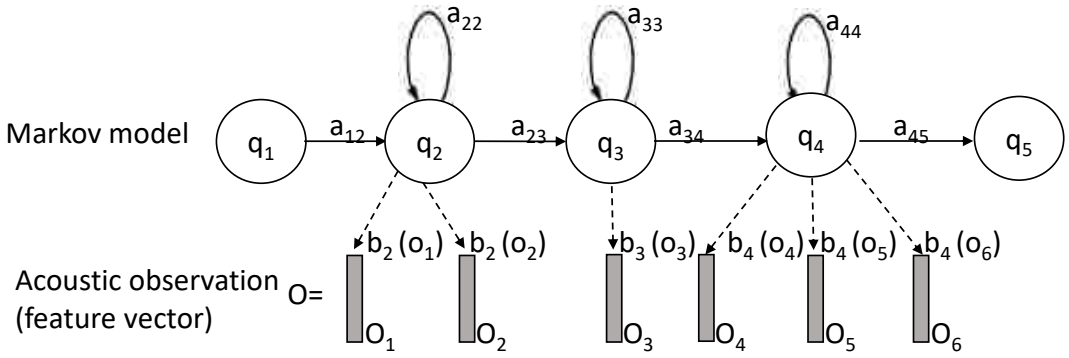


Figure 3.2: Topology of a five states left to right HMM with 3 emitting states and 2 nonemitting states. State 1 is the entrance state and state 5 is the exit state and both of them are non-emitting. a_{ij} denotes the state transition probability from state i to state j. $b_j(o_t)$ is the state output probability distribution for state j at time t.

The HMM used for ASR acoustic modelling is designed following two important assumptions; the HMM is a first-order Markov process, which means that state q_t at time t depends only on the previous state q_{t-1}, as shown in the following equation:

$$p(q_t|q_{t-1}, q_{t-2}, ...q_1) = p(q_t|q_{t-1}) \tag{3.5}$$

The second assumption is that the observation vector O_t at any particular time t is assumed to be conditionally-independent of the previous observations and

states given the state q_t, as shown in the following equation:

$$p(o_t|o_{t-1}, ...o_1, q_t, ..., q_1) = p(o_t|q_t) \tag{3.6}$$

These two assumptions significantly simplify the application of HMM for speech recognition. For instance, assuming $Q = q_1, q_2, ..., q_T$ is the possible state sequence for transcription w, by HMM, the likelihood $P(o|w, \mu)$ can be computed as

$$p(o|w, \mu) = \sum_{Q} p(o|Q, \mu)p(Q|w, \mu) \tag{3.7}$$

Since the state sequence Q is hidden and the space for Q is likely to be large, the likelihood in equation 3.7 is hard to compute. However, using the previous two assumptions, the likelihood can be decomposed as shown in equation 3.8

$$\begin{aligned} p(o|w, \mu) &= \sum_{Q} \prod_{t=1}^{T} p(o_t|q_t, \mu)p(r_t|q_{t-1}, \mu) \\ &= \sum_{Q} a_{q0q1} \prod_{t=1}^{T} b_{qt}(o_t)a_{q_t q_{t+1}} \end{aligned} \tag{3.8}$$

where a_{ij} denotes the state transition probability from state i to state j. $b_j(o_t)$ is the state output probability distribution for state j at time t.

Given that the state sequence $(q_0, q_1, q_2, ..., q_T, q_{T+1})$ is hidden for the observation O, using exhaustive search for evaluating the likelihood can be estimated by summing all the possible state sequences in Q for HMM with N states as $O(N^T)$, which is computationally impractical. However, this challenge can be addressed using the Baum-Welch algorithm [Baum et al., 1970], which is an instance of the expectation maximization (EM) algorithm [Moon, 1996]. This approach reduces the computational cost to $O(NT^2)$.

The proposed HMM architecture so far has assumed that each model in the HMM represents a word level in the speech recognition system. However, practically, in a large vocabulary speech recognition system (LVCSR) there is not enough data to model each word in the vocabulary. English systems typically use about 60K words, and in this thesis, Arabic uses a vocabulary of more than one million words. This will be explained in more detail in chapter 7. It is therefore more common to have sub-word representation; most systems use phoneme

representation. One approach to map words to phoneme automatically using a grapheme to phoneme approach (G2P); more detail can be found here [Bisani and Ney, 2008]. This thesis will focus on using grapheme representation for Arabic as shown in the last block in figure 3.1. In the grapheme-based system, each model in the HMM represents a character level in the speech recognition system. This simplifies the process of building the word to sub-word units map considerably. Using the sub-word for modelling the acoustics ensures that there is enough training data to estimate the model parameters robustly. Context-dependent models are often used as well, which leads to increasing the set of sub-word representations. The mapping between words and sub-word representation is known as the speech recognition lexicon.

3.2.1 Neural network acoustic model

The objective of the acoustic modelling is to have the right label for certain frame, e.g., each 10 msec. The Gaussian mixture model takes generative probability role in modelling the acoustics, whereas the neural network looks at the probability of each phone given the input feature data, which can be described as a probability estimation problem. The idea of using multilayer perceptrons in speech recognition has been studied in hybrid approach more than 20 years ago, where researchers achieved good results by using single hidden layer of a neural-network to predict the HMM states from windows of acoustic coefficients [Bourlard and Morgan, 1993]. However, neither the hardware nor the learning algorithm were adequate to train neural network with many hidden layers on large amount of data. An excellent overview can be found here [Morgan and Bourlard, 1995]. The recent advances in both machine learning algorithms and computer hardware have led to more efficient methods for training DNNs that contain many hidden layers with a very large output layer to accommodate large vocabulary speech recognition systems. This is primarily to deal with the large number of HMM states that arise when each phone is modeled by a number of different triphone HMMs that take into account the phones on either side. Even when many of the states of these triphone HMMs are tied together, there can be thousands of tied states (context-dependent states) [Hinton et al., 2012].

We can interpret the output of the neural network as an estimate of the probability of the phone given the input data. While predicting the phone can be

done using a single frame, this is usually done with a window of λ frames context, typically λ being between 3 and 4. Figure 3.3 shows a context of 3 frames to right and 3 frames to the left to O_t as the central frame input to the network. It also shows a feed-forward neural network with one-input layer accepting sampled waveform data; such as MFCC, PLP or filter bank. For example, an MFCC feature vector is of 39 dimensions per frame, times 7 frames (3-left context, central phone and 3-right context) will be $7 * 39 = 273$ MFCC input as a feature vector. This is followed by two hidden layers and one output layer. The hidden layers help to build rich representations to deal with many variations of the feature vector coefficient; such as different accents and diversity in speech rate.

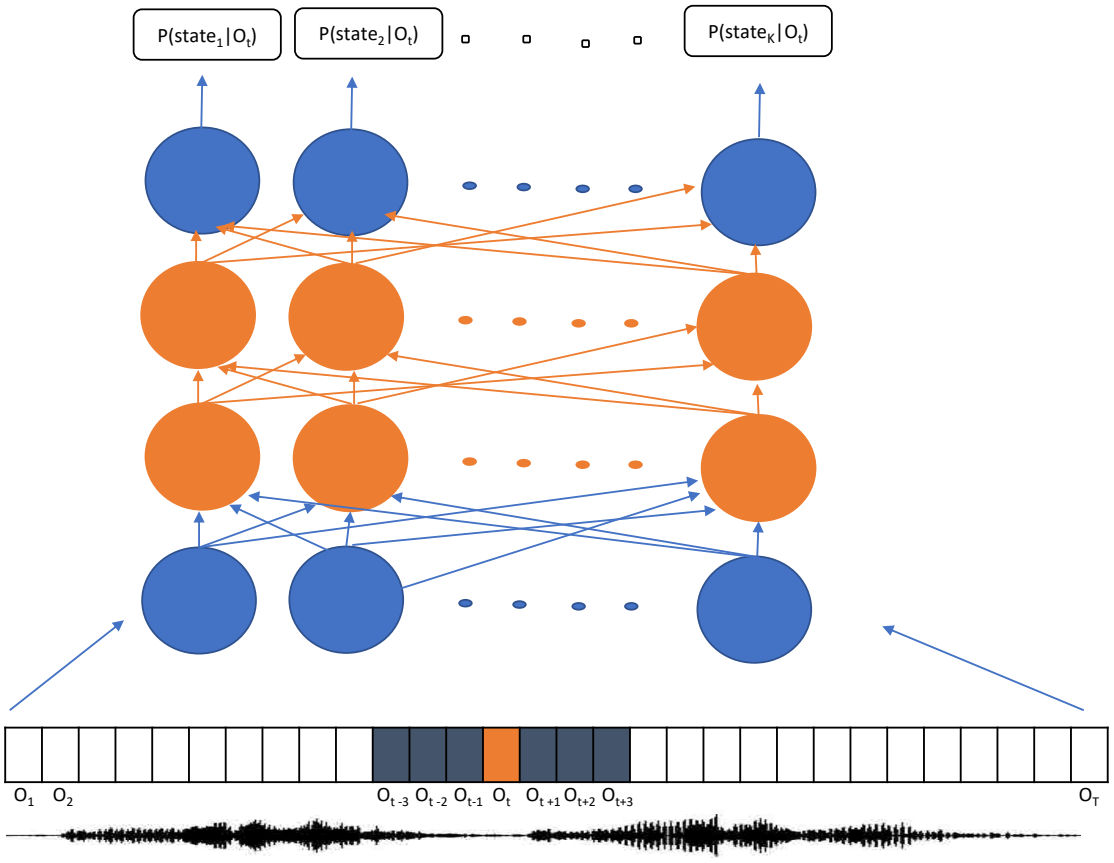


Figure 3.3: Diagram for feed-forward neural network for acoustic modelling.

Gradient descent is used to estimate the weights and biases at each layer, using back-propagation to estimate the required gradients.

There are many error functions to use in a neural network classification problem, such as mean square error (MSE) and cross entropy (CE). The CE objective function is matched with the softmax activation function, which makes CE widely

used in acoustic modelling. In a hybrid HMM/DNN ASR system, there are K context-dependent CD states. Ideally, the output layer of the neural network calculates the score for each state corresponding to the central frame of speech. Typical English LVCSR will have a few thousand classes, e.g., there are 9,304 CD state outputs in a DNN acoustic model for switchboard [Hinton et al., 2012].

The neural network output is typically followed by a soft-max function, which converts scores at the output-layer to a probability for all output states. Soft-max will ensure two things: (i) probability for each CD state is between zero and one, and (ii) the summation of the probabilities will add to one. This is very practical for the LVCSR problem, especially when it is desirable to prune branches with small probabilities, which is similar to what is used in Viterbi decoding. This will be discussed in section 3.5.

For an observation o_{ut} corresponding to time t in utterance u, the output $y_{ut}(s)$ of the DNN for the HMM state s is obtained using the following softmax activation function:

$$y_{ut}(s) \triangleq P(s|O_{ut}) = \frac{exp\{a_{ut}(s)\}}{\sum_{s'} exp\{a_{ut}(s')\}} \tag{3.9}$$

where $a_{ut}(s)$ is the activation at the output layer corresponding to state s. The pseudo log likelihood of state s given observation o_{ut},

$$log\ p(o_{ut}|s) = log\ y_{ut}(s) - log\ P(s) \tag{3.10}$$

where $P(s)$ is the prior probability of state s calculated from the training data [Bourlard and Morgan, 1993].

Finally, each label in the output-layer is represented mathematically by a vector that has the same size as the number of classes (CD states). Each label will have the value of one for the correct class and zero everywhere else. This is also known as one-hot vector. The CE loss function is calculated between the one-hot vector and the softmax output. Given that, we are dealing with a typical multi-class classification problem, where it is common to use the negative log posterior as the objective:

$$\mathcal{F}_{CE} = -\sum_{u=1}^{U}\sum_{t=1}^{T_u} log\ y_{ut}(s_{ut}) \tag{3.11}$$

where s_{ut} is the reference state label at time t for utterance u. This is also the expected CE between the distribution represented by the reference labels and the predicted distribution $y(s)$.

A very good review of neural network and related examples can be found in [Nielsen, 2015]. In an excellent study by Mohamed et al. [2012], they illustrated, with reasonable depth, the performance of deep belief network (DBN) as a competitive alternative to Gaussian mixture models for relating states of a hidden Markov model to frames of coefficients derived from the acoustic input.

Various neural network architecture

Various acoustic neural architectures have shown good results on LVCSR. We can list some of the most recent implementations here:

- **Feed-forward neural network:** Hinton et al. [2012] introduced the most recent wave of deep learning in speech recognition, showing a substantial improvement in WER by training FDNN with many hidden layers and trained over a short-window of frames. Their study investigated 3 to 8 hidden layers, and explored 1,024 to 3,072 neurons per layer. Their study reported, in some scenarios in the LVCSR, more than 10% absolute reduction in WER. In a further by, Mohamed et al. [2012] visualised some results and illustrated, for example, some of the gains in DBN. Moreover, Deng et al. [2013] presented experimental evidence that the spectrogram features of speech are superior to MFCC with FDNN, in contrast to the earlier long-standing practice with GMM-HMMs. They also evaluated the multilingual FDNN architecture, which has the input and hidden layers shared by all languages, but separate output layers are made specific to each language. Using their language universal feature extractor, they readily construct a powerful monolingual DNN for any target language.

- **Sequence neural network:** In a study by Graves et al. [2013b], they explored using deep recurrent neural network (deep RNN), and deep long short term memory (deep LSTM). In their study, they relied on end-to-end training, where the RNNs learn to map directly from an acoustic to a phonetic sequence. They studied 1 to 5 RNN hidden layers: Both uni-direction, and bi-direction (BLSTM) layers. Their results were evaluated only on TIMIT phoneme classification task. In their extended study, Graves et al. [2013a] applied the deep BLSTM (DBLSTM) as an acoustic model in a standard neural network-HMM hybrid system. They reported that the DBLSTM-HMM hybrid gives equally good results on TIMIT as their

previous study. It also outperformed the DNN benchmarks on a subset of the Wall Street journal corpus.

Given an input sequence $x = (x_1, ..., x_T)$, a standard RNN computes the hidden vector sequence $h = (h_1, ..., h_T)$ and output vector sequence $y = (y_1, ..., y_T)$ by iterating the following equations from $t = 1$ to T:

$$h_t = \mathcal{H}\ (W_{xh}x_t + W_{hh}h_{t-1} + b_h) \tag{3.12}$$

$$y_t = W_{hy}h_t + b_y \tag{3.13}$$

where the W terms denote weight matrices (e.g., W_{xh} is the input-hidden weight matrix), the b terms denote bias vectors (e.g., b_h is hidden bias vector) and $\mathcal{H}$ is the hidden layer function, which is usually an element-wise application of a sigmoid function. The LSTM architecture [Hochreiter and Schmidhuber, 1997], which uses purpose-built memory cells to store information, is better at finding and exploiting long range context.

In speech recognition, where whole utterances are transcribed at training time, this is possible to exploit future context as well. Bidirection RNNs (BRNNs) [Schuster and Paliwal, 1997] do this by processing the data in both directions with two separate hidden layers, which are then fed forward to the same output layer. Deep RNNs can be created by stacking multiple RNN hidden layers on top of each other. Deep BRNNs can be implemented by replacing each hidden sequence $\mathbf{h}^n$ with the forward and backward sequences $\overleftarrow{\mathbf{h}}^n$ and $\overrightarrow{\mathbf{h}}^n$, and ensuring that every hidden layer receives input from both the forward and backward layers at the level below. If LSTM is used for the hidden layers, we get deep bidirectional LSTM, as illustrated in figure 3.4. Both LSTM and BLSTM will be studied in section 7.2.3.

- **Convolutional neural network:** In a study by Sainath et al. [2013], they studied using deep CNN for LVCSR. They determined the appropriate architecture to make CNNs effective compared to DNNs for LVCSR tasks. Specifically, they focused on how many convolutional layers are needed, the optimal number of hidden units and the best input feature type for CNNs. They reported that CNN offered 13-30% relative improvement over GMMs,

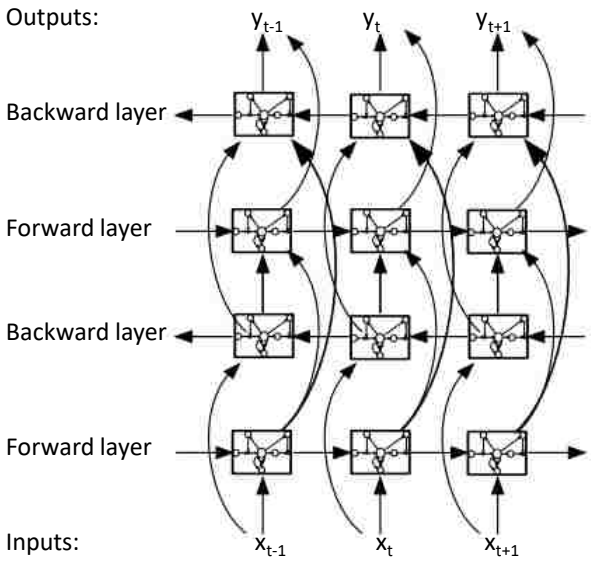


Figure 3.4: Deep bidirectional long short term memory network (DBLSTM).

and 4-12% relative improvement over DNNs, on a 400 hours broadcast news and 300 hours switchboard data.

- **Time delay neural network:** In a study by Peddinti et al. [2015], they proposed a TDNN architecture that models long term temporal dependencies with training times comparable to the standard feed forward DNN. The network uses sub-sampling to reduce computation during training. On the switchboard task, they showed a relative improvement of 6% over the baseline DNN model. They presented results on several LVCSR tasks with training data ranging from 3 to 1800 hours to show the effectiveness of the TDNN architecture in learning wider temporal dependencies in both small and large data scenarios.

  TDNN [Waibel et al., 1989] has proven to be effective in modelling long range temporal dependencies. When processing a wider temporal context, in a standard DNN, the initial layer learns an affine transform for the entire temporal context. However, in a TDNN architecture, the initial transforms are learnt on narrow contexts and the deeper layers process the hidden activations from a wider temporal context. Hence, the higher layers can learn wider temporal relationships. Each layer in a TDNN operates at a different temporal resolution, which increases as we go deeper into the network. The hyper parameters of the TDNN network are the input contexts of each layer

required to compute an output activation, at one-time step. Given that, there are large overlaps between input contexts of activations computed at neighbouring time steps, they can be sub-sampled. Peddinti et al. [2015] splices together adjacent temporal windows of frames at each layer, to allow gaps between the frames. In fact, in the hidden layers of the network, they generally splice no more than two frames. For instance, splicing together frames t–2 through $t+2$ at the input layer, which could be written as context $(-2, -1, 0, 1, 2)$. With the proposed sub-sampling scheme, the overall necessary computation is reduced during the forward pass and backpropagation, due to selective computation of time steps. Another advantage of using sub-sampling is the reduction in the model size. Splicing adjacent frames at hidden layers would require to either have a very large number of parameters, or reduce the hidden-layer size significantly. TDNNs will be studied in section 7.2.3.

- **Various neural network:** There have been many recent architectures for neural acoustic modelling In fact, a recent common trend in ASR modelling is to combine different types of layers [Deng and Platt, 2014, Sainath et al., 2015]. In a study by Cheng et al. [2017], they explored using dropout to improve generalisation in DNN training. They reported that combining TDNN with LSTM (TDNN-LSTM) outperformed BLSTM with about 3% relative gain. In addition to this, it is also much faster to train than BLSTM.

This thesis explores various neural acoustic modelling architectures: TDNN, LSTM, BLTSM, TDNN-LSTM and TDNN-BLSTM. More details are given in chapter 7.

Error function in neural AM

Sequence discriminative training ASR has shown significant reduction in WER in HMM [Woodland and Povey, 2002, Povey and Woodland, 2002, Povey, McDermott et al., 2007]. In the neural framework, more recent work was introduced by Kingsbury [2009], Veselỳ et al. [2013], Su et al. [2013]. Sequence discriminative training of neural networks for ASR has been shown to provide a significant reduction in WER compared to the frame level cross entropy training . In this thesis, we adopt the purely sequence trained neural networks using lattice-free maximum mutual information (LF-MMI) [Povey et al., 2016].

The maximum mutual information (MMI) criterion used in ASR [Bahl et al., 1986] is the mutual information between the distributions of the observation and word sequences. With O_u is the sequence of all observations, and W_u as the word-sequence in the reference for utterance u, the MMI criterion is defined as follows:

$$\mathcal{F}_{MMI} = \sum_{t=1} log \frac{p(O_u|S_u)^k P(W_u)}{\sum_U p(O_u|S)^k P(W)} \tag{3.14}$$

where S_u is the sequence of states corresponding to W_u and k is the acoustic scaling factor. The sum in the denominator is taken over all word sequences in the decoded speech lattice for utterance u.

Computing the denominator in equation 3.14 involves summing over all possible word sequences, i.e., generating lattices, and summing over all words in the lattice. The proposed LF-MMI is denominator-lattice-free, where it does the summation over all possible label sequences on the GPU. To avoid overfitting, it uses a combination of three different regularisation techniques: cross entropy regularisation, output l_2-norm regularisation, and leaky HMM. In Povey et al. [2016], they attempted, to make the LF-MMI computation feasible; they used a phone n-gram language model instead of the word language model. To further reduce its space and time complexity, they computed the objective function using neural network outputs at one third of the standard frame rate. These changes enable the network to perform the computation for the forward-backward algorithm on GPUs. Furthermore, they reduced the output frame-rate which provides a significant speed-up during training. In their study, models trained with LF-MMI provide a relative word error rate reduction of about 11.5%, over those trained with cross entropy objective function. This thesis will use LF-MMI in chapter 7 for the ASR experiments.

3.3 Language modelling

The probabilities assigned to the sequences of words are called language modelling (LM); $p(w)$, as shown in equation 3.2, estimates the prior distribution over a sequence of words $w = [w_1, w_2, ..., w_k]$. The simplest model that assigns a probability to a sequence of words is the n-gram LM [Damerau, 1971]. Where n-gram is a sequence of n words, a 2-gram (or bigram) is a two-word sequence of words

like "dialectal Arabic", "Arabic speech", or "speech recognition", and a 3-gram (or trigram) is a three-word sequence of words like "dialectal Arabic speech", or "Arabic speech recognition" [Jurafsky, 2017].

The n-gram can be expressed as shown in equation 3.15.

$$P(w) = \prod_{k=1}^{K} p(w_k|w_{k-1}, w_{k-2}, ..., w_{k-n+1}) \tag{3.15}$$

This equation has two main hyper-parameters; K is the number of words in W and n is the order of the LM: two for the bigram and three for the trigram LM. Ideally, one can compute LM for an arbitrary order. However, higher order n-gram LM usually leads to unseen word sequences. Therefore, zero probabilities are due to data sparsity reasons, where normally the values of n are typically in the range of two-to-four for speech recognition applications.

An intuitive way to estimate the n-gram probabilities is to use maximum likelihood estimation (MLE). We obtain the MLE estimate for the parameters of an n-gram model by counting the n-gram occurrence in the training text, which can be expressed as follows:

$$p(w_k|w_{k-1}, w_{k-2}, ..., w_{k-n+1}) = \frac{C(w_{k-n+1}, ..., w_k)}{C(w_{k-n+1}, ..., w_{k-1})} \tag{3.16}$$

The frequency of a given word sequence $C(.)$ in the training text for some word sequences may be very low or even zero. To keep the language model from assigning zero probability to these unseen word sequences, we need to reserve some of the probability mass from some more frequent word sequences and allocate it to the unseen word sequences. This process is called smoothing or discounting. This is normally addressed by using two major techniques: (i) back-off or interpolation [Katz, 1987] in which the model will assign the probability mass unevenly to unseen word tokens in proportion to the probability lower than the lower-order n-gram, and (ii) discounting [Kneser and Ney, 1995] in which the smoothing technique relies on assigning some of the probability distribution mass to n-gram sequences unseen in the training text. In this thesis, we use the smoothing technique that was developed by Kneser and Ney [1995].

3.3.1 Neural network language model

With the rise of deep learning, the continuous space language model has become a popular choice compared to the discrete n-gram. Initially, a feed forward neural network was proposed by Bengio et al. [2003]. This was followed by different

architectures of neural networks and has shown better results, such as recurrent neural network (RNN) modelling [Mikolov et al., 2010]. Recent work was introduced using long short term modelling (LSTM) for language modelling [Sundermeyer et al., 2012]. One plausible explanation for the superior performance of the neural language model compared to the n-gram modelling is that the hidden layers in the network hold a better representation of the words. For example, the vector representation for the word-sequence "recognising dialectal Arabic is challenging" would be close enough to the vector for word-sequence "recognising colloquial Arabic is challenging", although the model may have never seen dialectal and colloquial in this context. However, the vector representation for both words, dialectal and colloquial, should be close enough from other contexts, not necessarily in the same exact context. This capability is not available in the n-gram LMs. The superior performance of neural LM comes with the price of high computational cost compared to n-gram, especially the RNN and LSTM LM. Therefore, for practical usage, the neural language model is often used for LM rescoring or reordering the top n ASR results (n-best) with smaller vocabulary size, ideally between 50K-to-100K words. Furthermore, recent work by Liu et al. [2016], Chen et al. [2016b] developed an efficient lattice rescoring methods using recurrent neural network language models. This has been released in CUED-RNNLM [Chen et al., 2016a], which is an open-source toolkit for efficient GPU-based implementation for training and evaluation of RNN LM.

The maximum entropy ME LM can be seen as neural network models with no hidden layer, with the input layer directly connected to the output layer. Such a model has been described in detail in [Xu and Rudnicky, 2000], where it was shown that it can be trained to perform similarly to a Kneser-Ney smoothed n-gram model. Mikolov et al. [2011a] introduced an excellent architecture of recurrent neural network with maximum entropy (RNNME). Their work is based on a hash-based implementation of a maximum entropy ME LM [Rosenfeld, 1996], which has been trained as part of the neural network LM. This has led to a significant reduction in computational complexity of the LM. In their study, they showed that training the RNN model with direct connections can lead to a good performance both on perplexity and word error rate, even if very small hidden layers are used. The model RNNME with only 40 neurons has achieved almost as good performance as RNN model that uses 320 neurons. It is worth noting that RNNME performs completely different to the interpolation of RNN and ME

models. The essential step is to train both models jointly, so that the RNN model can focus on discovering complementary information to the ME model [Mikolov et al., 2011b]. We will use RNNME in our language model rescoring as shown in section 7.2.3.

3.3.2 Evaluating a language model

Since the LM is normally trained using more text than the AM text data, it is common practice to keep some text data out of the training to compare different LMs as well as for evaluating improvements within one LM. Typically, for LM, two values are reported: perplexity (PP) on a test set is the inverse probability of the test set using a language model; simply, the lower the perplexity the better the LM. PP can be expressed as shown in equation 3.17. The second variable is out of vocabulary (OOV), which expressed in the percentage of the unknown words with respect to all the tokens in the test set. Similar to PP, for OOVs, the lower the better.

$$PP = exp\{-\frac{1}{K}\sum_{k=1}^{K} log(p(w_k|w_{k-1}, w_{k-2}, ..., w_{k-n+1}))\} \tag{3.17}$$

3.4 Adaptation and transfer learning

Most modern speech recognition systems will be trained on hundreds or even thousands of hours from a specific domain including many speakers. However, at deployment time, there are often unseen speakers, where the acoustic models μ will have poor performance. In this case, there is a need for speaker adaptation techniques. Another common challenge happens when the deployment domain is different from the training domain. For example, a speech system may be trained on broadcast news data and deployed to recognise a comedy program. In this case, there is a domain mismatch. Both cases are quite common in most of the practical speech recognition systems. The HMM-based systems are often using two main adaptation techniques; maximum a posteriori (MAP) [Gauvain and Lee, 1994] and maximum likelihood linear regression (MLLR) [Leggetter and Woodland, 1995]. One of the successful techniques used for the neural network acoustic models is called the i-vector for speaker and domain adaptation [Saon et al., 2013]. This thesis will be using the i-vector for both speaker and domain adaptation.

The i-vector is widely used in speaker recognition [Dehak et al., 2011a,b] and speaker adaptation [Saon et al., 2013], where a large Gaussian mixture model (GMM), called the universal background model (UBM), is typically trained to act as a prior model of the distribution of speech sounds. The speaker UBM mean components, also known as a supervector, have been found to be an effective representation for the speaker . The i-vector approach models supervector adaptation to a given sequence of frames in a low-dimensional space called the total variability space. In the i-vector framework, each speech utterance can be represented by a GMM supervector, which is assumed to be generated as follows:

$$M = m + Tw \tag{3.18}$$

where m is the speaker independent and channel independent supervector (which can be taken to be the UBM supervector), T is a rectangular matrix of low rank, and w is a random vector having a standard normal distribution prior N(0, 1). The i-vector is a Maximum A Posteriori (MAP) point estimate of the latent variable w adapting the UBM (supervector m) to a given audio file. Recently, the i-vector method has been successfully applied to speaker and channel adaptation in speech recognition. The main idea behind it is adding speaker characteristics to the audio features allowing the neural network to learn more efficiently about the speaker or the channel. One of the common settings for the i-vector is 400-dimensional per sentence and the UBM normally consists of 512 mixture components. This thesis will use the same setup of the i-vector for speech recognition and language identification as shown in chapters 6 and 7.

Moreover, there has been recent work to use deep neural network embedding for speaker adaptation. In a study by Senior and Lopez-Moreno [2014], they proposed providing additional utterance-level features as inputs to a deep neural network to facilitate speaker, channel and background normalisation. They showed that these input features provide the networks with valuable information that, with their proposed regularisation brings roughly 4% relative reduction in word error rate for various model sizes. In another study by Snyder et al. [2017], they showed that long-term speaker characteristics are captured in the network by a temporal pooling layer that aggregates over the input speech. This enables the network to be trained to discriminate between speakers from variable length speech segments. After training, utterances are mapped directly to fixed-dimensional speaker embeddings. Their embeddings outperformed i-vectors for

short speech segments and are competitive on long duration test conditions.

3.5 Decoding

Generating the most-likely word sequence is also known as decoding. Once both the LM and the AM have been trained, a naïve assumption is to obtain and combine their scores from both models to estimate w_h as shown in equation 3.1. However, this poses two major challenges. First, the score of the acoustic model that often overwhelms the effect of the LM, which can be handled by giving different weights to each model as shown in equation 3.19. Ideally, both variables, word insertion penalty μ and language model scale factor κ can be tuned using development data as part of the hyper-parameter optimisation.

$$\begin{aligned} W_h &= \arg\max_{w} log\, p(w|o) \\ &= \arg\max_{w} \{log\, p(o|w;\mu) + \kappa\, log\, p(w)\} \\ &\approx \arg\max_{w} \{log\, (\max_{Q}\, p(O,Q|w;\mu)) + \kappa\, log\, p(w)\} \end{aligned} \tag{3.19}$$

The second challenge comes from combining the models directly, which would lead to a huge search space for all the possible word sequences. This often happens in a typical large vocabulary speech recognition (LVCSR) system. One assumption that has been widely used to make the decoding process more practical is the Viterbi approximation [Viterbi, 1967], which only search for the most likely state sequence by an efficient recursive form. Beam-pruning is often used in Viterbi search, which means we access the frames one by one, and for each frame, we prune away states with low probability.

The recent advances in weighted finite state transducers (WFSTs) [Mohri et al., 2002] have made it possible to build the decoding search space prior to starting recognition statically. A WFST maps the input sequence to an output sequence in which each transition has an input label, an output label and a weight. Table 3.1 shows the four main component used to build WFST for decoding LVCSR.

In Decoding, we deploy the following three WFST operations:

- **Composition ($\circ$):** Combine transducers at different levels. For example, if G is a finite state grammar and L is a pronunciation dictionary then $L \circ G$ transduces a phone string to word strings allowed by the grammar.

	transducer	input sequence	output sequence
$\boldsymbol{H}$	HMM	HMM states	CD phones
$\boldsymbol{C}$	context-dependency	CD phones	phones
$\boldsymbol{L}$	pronunciation lexicon	phones	words
$\boldsymbol{G}$	word-level grammar	words	words

Table 3.1: Main four transducers used in LVCSR decoding.
Note: phones can be replaced with characters if the system is grapheme-based.

- **Determinisation (det):** Ensure that each state has no more than a single output transition for a given input label.
- **Minimisation (min):** Transforms a transducer to an equivalent transducer with the fewest possible states and transitions.

The full process includes composing the acoustic model, lexicon and language model in a single and large network using the HMM topology. This approach is widely used for research and has successfully been adopted in many commercial platforms for offline ASR decoding. The overall decoding graph $\boldsymbol{HCLG}$ is constructed as shown in equation 3.20, this is ideally a much smaller graph than naïvely combining the four transducers shown in table 3.1.

$$HCLG = min(det(H \circ min(det(C \circ min(det(L \circ G)))))) \tag{3.20}$$

The WFST is the main decoding scenario used in the Kaldi software Povey et al. [2011], which will be the main framework for the Arabic ASR decoding in this thesis.

3.6 Evaluation

Word error rate (WER) is the most common metric used to evaluate how well a speech recognition performs. WER is normally used for comparing different systems as well as for evaluating improvements within one system. It can be solved by aligning the recognised word sequence with the reference word sequence using dynamic string alignment. There are three types of errors that can happen in the recognised transcript: deletion (D) when the recognised text missed a word or more in the reference transcription, insertion (I) when the recognised text inserted a word or more which was not in the reference transcription and substitution (S) when the recognised text produced a different word compared

to the word in the reference. Given the total number of words to be recognised in the reference transcript is N, the WER is expressed as in equation 3.21

$$WER = \frac{I + D + S}{N} \tag{3.21}$$

Let the total number of correct words (C). Thus, WER can also be calculated using equation 3.22. It is worth noting that evaluating a speech recognition system is based on the assumption that there is a single ground-truth for speech audio. This thesis will challenge this assumption for dialectal Arabic and equation 3.22 will be the basic for further studies in chapters 8 and 9 for evaluating dialectal speech recognition.

$$WER = \frac{I + D + S}{S + D + C} \tag{3.22}$$

3.7 Arabic speech recognition overview

Building a robust Arabic speech recognition system can be considered as a multidisciplinary effort. In addition to dealing with the standard acoustic pipeline, building a robust Arabic ASR requires various natural language processing (NLP) components to address language challenges. This list can summarise challenges unique to Arabic speech recognition.

1. Arabic has short vowels, which are often ignored in the text as shown in section 2.2.

2. Arabic is a morphologically rich language, and dealing with morphemes is often used to reduce OOV as shown in section 2.4.

3. Arabic has not enough labelled data available for research, more details will be discussed in chapter 5.

4. Arabic has many dialects, where different words are used and pronounced differently.

This thesis, of course, is not the first work to explore Arabic speech recognition. Here, we try to summarise some of the previous work done on Arabic ASR. The 2002 Johns Hopkins Summer Workshop [Kirchhoff et al., 2003] focused on Arabic ASR, more than eight research labs during the summer workshop addressed the first and the second challenges mentioned earlier. They have also

studied the discrepancies between dialectal and formal Arabic. They proposed a novel approach to automatic vowel restoration, morphology-based language modelling and the integration of out-of-corpus language model data, and report significant word error rate improvements on the LDC Arabic Call Home data[1].

3.7.1 GALE Project

The global autonomous language exploitation (GALE) project was funded by Defense Advanced Research Projects Agency (DARPA) to produce a system that is able to automatically take multilingual newscasts, text documents, and other forms of communication, and to make their information available for human queries. The program encompassed three main challenges: automatic speech recognition, machine translation, and information retrieval. The focus of the program was on recognising speech in Mandarin and Arabic and translating it into English. The Arabic speech recognition was mainly concerned with broadcast news and broadcast conversation. This section highlights some of the impact that came out of GALE project.

1- Missing vowels: Several toolkits were developed for Arabic pre-processing such as Arabic tokenisation, diacritization, morphological disambiguation, part-of-speech tagging, stemming and lemmatisation (MADA) [Habash et al., 2005]. A similar tool for Arabic NLP AMIRA [Diab, 2009], where both were studied at Columbia University. Further work was developed for speech recognition pronunciation dictionary using linguistically-based pronunciation rules [Biadsy et al., 2009a].

2- Morphological Analysis and Decomposition: Given that Arabic is a morphologically complex language, a study by El-Desoky et al. [2009] introduced both morphological decomposition and diacritization for Arabic language modelling. In a study by Diehl et al. [2009], they introduced a novel context-sensitive method for morpheme-to-word conversion in language modelling, where they used MADA from decomposition. Both studies reported between 0.5-1.0% absolute reduction in WER.

3- Lack of enough labelled speech data: the linguistic data consortium (LDC) has built a sizable Arabic broadcast corpus for research in the GALE

[1]https://catalog.ldc.upenn.edu/LDC97S45

project, and excellent Arabic recognition systems were built studying different aspect of the language [Al-Onaizan and Mangu, 2007, Gales et al., 2007, Vergyri et al., 2008, Saon et al., 2010, Metze et al., 2010, Kingsbury et al., 2011, Mangu et al., 2011]. Some of these systems were trained on more than 300 hours in the broadcast news domain mainly dominated by modern standard Arabic.

Despite the fact that GALE was a sizable effort that made a very good impact on the state-of-the-art in Arabic ASR and increased awareness of the language, it did not make any of the results available to reproduce by researchers outside the project. There was no publicly available speech lexicon like CMU in American English [Walker et al., 2004] or like British English example pronunciation dictionary (BEEP) [Robinson et al., 1995]. Furthermore, the reported results within GALE were not accessible by researchers not involved in the project. This made it difficult to challenge the achieved results and to improve on the same baseline. It is worth mentioning that five years after the project, LDC has released about 500 hours of transcribed Arabic broadcast speech data from the GALE project, by making it available on their catalog.

Finally, we can acknowledge the progress in Arabic ASR for MSA. We can also see a clear gap in dialectal speech recognition, particularly beyond the broadcast news domain. This thesis will address this gap in detail in chapter 7.

3.8 Summary

We presented a brief overview of the different aspects of modern speech recognition systems. For acoustic modelling, we discussed the standard HMM and focused on recent progress in deep learning approaches using a different cost function such as LF-MMI. The language modelling covered both n-gram approach and the neural network techniques. Finally, the last section shed some light on recent efforts in Arabic ASR.

Chapter 4

Overview of Automatic Language and Dialect Identification

Automatic spoken language identification is defined as the process that determines the identity of the language spoken in a speech audio sample. Its importance can be gagued from the growing interest in automatic speech recognition. A good language recognition system can facilitate labelling the language of a speech segment for many tasks like multilingual speech processing, such as spoken language translation, spoken document retrieval, metadata labelling and multilingual speech recognition [Waibel et al., 2000]. The same principle can be applied on automatic spoken dialect identification that can help reduce the ASR word error rate for dialectal data by training ASR systems for each dialect, or by adapting the ASR models to a specific dialect.

Humans are born with the ability to discriminate between spoken languages as part of human intelligence. A human being with the adequate training is the most accurate language recogniser given that the human listener speaks the language [Li et al., 2013]. It is estimated that there are several thousand spoken languages in the world. The recent edition of the Ethnologue, a database describing all known living languages, has documented 6,909 living spoken languages [Gordon et al., 2009]. Unlike spoken language identification, text-based language identification has traditionally relied on distinct textual features of languages such as words and sub-words: morpheme, stem and characters. Text-based language identification for the Latin-alphabet languages has attained a reasonably good performance, and thus it is considered to be a solved problem [Christopher et al., 2008]. There are special cases in near-by languages are still not solved, e.g re-

cent challenge in discriminating between similar languages (DSL) task covered the following languages: Bosnian, Croatian, and Serbian, Malay and Indonesian, Persian and Dari, Canadian and Hexagonal French, Brazilian and European Portuguese, Argentine, Peninsular, and Peruvian Spanish [Zampieri et al., 2017]. With the rise of Web 2.0, it is easy now to harvest text for many languages or even dialects, which improved the performance on the non-Latin languages significantly for text-based language and dialect detection [Darwish et al., 2014]. In contrast, spoken language identification is far more challenging than text-based language identification because there is no guarantee that the transcription from the speech recognition engine will be error-free. Intuitively, one can argue that once the system knows what a person is saying, its language is obvious. Therefore, the type of perceptual cues that human listeners use is always the source of inspiration for automatic spoken language identification [Zhao et al., 2008]. Similar to most of the machine learning challenges, the objective of spoken language identification is to replicate the human's ability through computational means.

The recent advances in signal processing, cognitive science and machine learning have improved the state of the art in spoken language identification considerably. This thesis is concerned with Arabic dialect identification, which is a closely related problem to the language identification problem. An excellent overview of the spoken language identification can be found in [Li et al., 2013].

4.1 Front-end feature extraction

The vocal apparatus of a human being is capable of producing a wide range of sounds. The physical sound can be referred to as acoustics, while the pattern of the sound can be referred to as phonotactics [Li et al., 2013]. Listeners can often make subjective judgments regarding unknown languages, e.g., this audio sounds like Arabic, it is tonal like Chinese, or it has a stress pattern like German or English. This is of course difficult when a listener tries to distinguish between nearby languages like Portuguese and Spanish without lexical knowledge.

Researchers concluded four groups of features: acoustic, phonotactic, prosody, and lexical features [Muthusamy et al., 1994]. The prosodic features such as stress, duration, rhythm, and intonation are challenging to extract automatically, and did not show great gain as standalone features in spoken language identification compared to phonetic features [Ng et al., 2010]. Therefore, this thesis is

concerned with the remaining three features: acoustic, phonotactic and lexical features extracted automatically from an audio file.

Figure 4.1 shows the most popular techniques used for digitising the audio file for the task of language recognition. By looking top-down, starting at section (a), which has the simplest and the less feature engineering to the audio, in addition to the raw audio file, the spectrogram is a visual representation of the spectrum of frequencies in the audio file. The motivation to list the spectrogram is based on the recent success of using convolution neural networks for image processing classification challenges. In a recent study by Bartz et al. [2017], they investigated using the spectogram features in a language identification task. Their method was applied in the image domain by using convolutional recurrent neural network (CRNN).

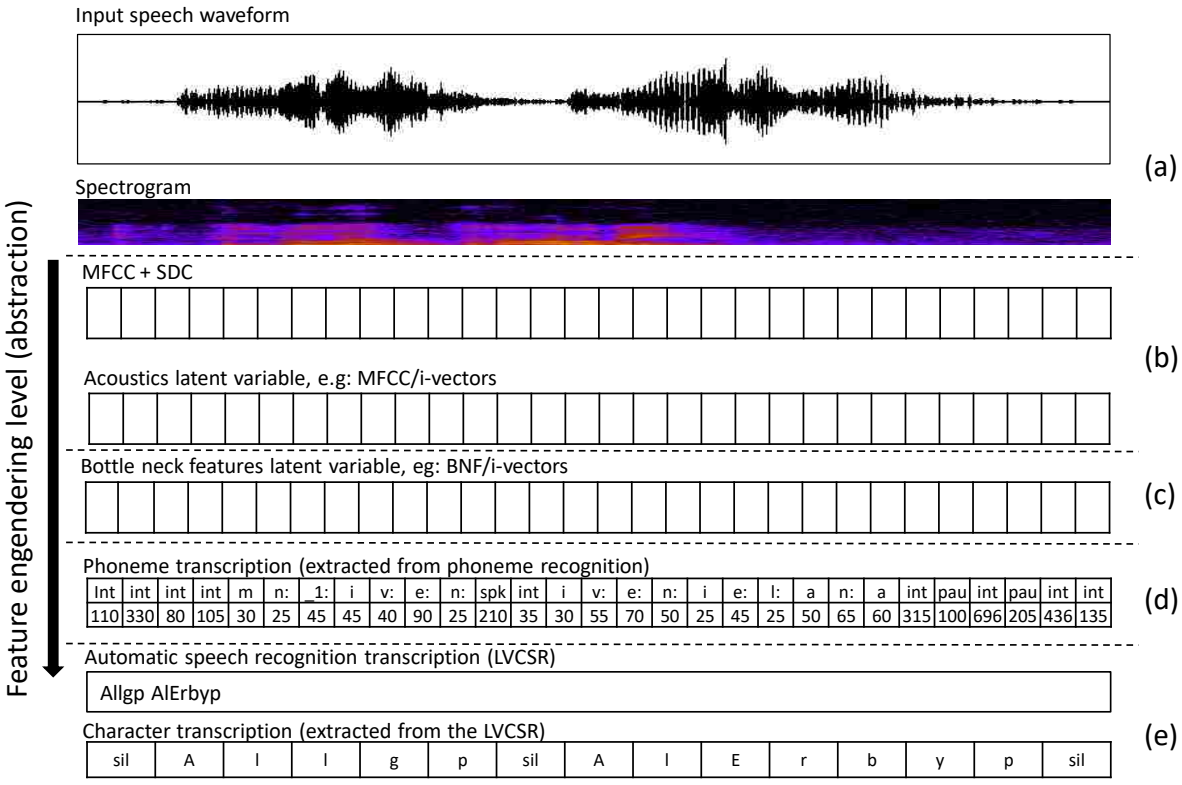


Figure 4.1: Various levels of feature extraction for an audio file for spoken language identification.

4.1.1 Acoustic features

The Mel-frequency cepstral coefficients are effective in most speech recognition tasks as discussed in section 3.1. The first- (Δ) and the second- ($\Delta\Delta$) order MFCC derivatives capture short-term speech spectral dynamics and do not capture longer term variation in speech reflected in high level *language features*, such as prosodic, phonetic and linguistic. The shifted-delta-cepstral (SDC) coefficients as a means of incorporating additional temporal information about the speech into the acoustic feature vectors showed better performance in language recognition [Torres-Carrasquillo et al., 2002, Kohler and Kennedy, 2002].

The SDC features are specified by a set of 4 parameters, N, d, P and k, where:

- N: number of cepstral coefficients computed at each frame.
- d: represents the time advance and delay for the delta computation.
- P: time shift between consecutive blocks.
- k: number of blocks whose delta coefficients are concatenated to form the final feature vector.

For cepstral coefficient $c(t)$, the SDC vector at frame time t is given by the concentration from $i = 0$ to $k - 1$ blocks of all the $\Delta c(t + iP)$, where:

$$\Delta c(t + iP) = \frac{\sum_{d=-D}^{D} dc(t + iP + d)}{\sum_{d=-D}^{D} d^2} \tag{4.1}$$

The commonly-used configuration for N-d-P-k is 7-1-7-3 along with the MFCCs per 20/25 ms sliding window over the speech signal and a 10 ms overlap. This concludes for 56 feature vectors per frame. More detail about MFCC-SDC for language recognition can be found in [Torres-Carrasquillo et al., 2002, Zazo et al., 2016]. Both MFCC and MFCC-SDC are commonly used as acoustic representation for a language identification task.

One of the factor analysis techniques, i-vector, which has been introduced in 3.4, has become very popular in speaker recognition and speaker adaptation. It has also shown very good results in language identification [Dehak et al., 2011b]. Similar to speaker adaptation, the i-vector provides an efficient way to compress the GMM supervectors by combining mainly all the language variabilities into a low dimensional subspace, referred to as the total variability space. Section (b) in figure 4.1 shows a sample for the two most common approaches for the acoustic

features; MFCC+SDC and using them to construct an i-vector. It is worth noting that the acoustic-based i-vector in language identification is commonly used to represent a speech sentence; ideally 10-30 seconds, which is different than its usage in speaker adaptation, where, in some scenarios like live speech recognition, the i-vector is calculated per frame.

4.1.2 Bottleneck features

The probabilistic features were introduced to speech recognition in TANDEM feature extraction [Hermansky et al., 2000]. Normally, these features are extracted by training a neural network to predict context-independent monophone states. The neural network typically has one narrow hidden layer placed in the middle of the network, this is the bottleneck feature. Since the neural network has the ability of nonlinear compression of the input features, the bottleneck output from the hidden layer represents the underlying speech well and is more compact than the input features. Bottleneck features have shown to be effective in improving the accuracy in automatic speech recognition [Grézl et al., 2007] and recently in automatic language identification [Richardson et al., 2015b].

A common setup for using the bottleneck features in language identification is shown in figure 4.2. In the context of language identification, the neural network is used to extract features to be used by a secondary classifier. The bottleneck features have information about both the acoustics and the phonetics since the network is trained to predict sub-phonetic units or "senones" for each input frame. The neural network is a multi-layer perceptron with more than two hidden layers; typically between three-to-five layers with stochastic gradient descent. The speech input is commonly to be a stacked set of standard MFCCs extracted from 20/25 ms sliding window over the speech signal and a 10 ms overlap similar to the setup discussed in section 3.1.

4.1.3 Phonotactic features

Spoken language identification can be characterised by phonological variations between different languages. The phone units are widely used in most of the speech recognition systems. Tokenizing the speech audio into a phonetic sequence is done using a phone recogniser, wherein, no language model is used, practically open phone loop or null grammar is used for decoding the HMM models to allow

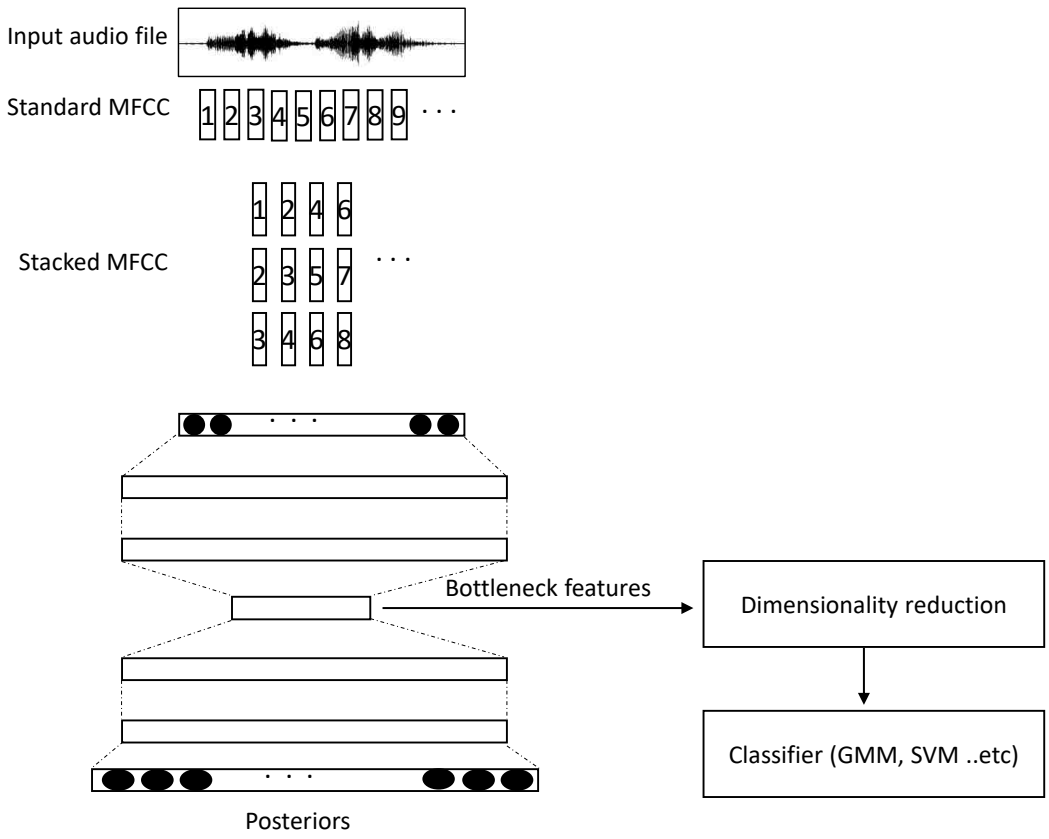


Figure 4.2: Example for bottleneck feature in language recognition

the transition from one phone to another to be equally probable. The outcome from this process is often known as a phoneme sequence. The number of phonemes used in a language ranges from about 15 to 50, with the majority having around 30 phonemes. For example, Arabic phoneme-based speech recognition systems contain 29 consonants and 6 vowels.

Phonetic repositories differ from one language to another, although they may share some common phonemes. Ideally, it is better to use a phone recogniser of the target language. However, practically, robust phone recogniser for other languages have shown superior results. For example, the Hungarian phone recogniser based on a long temporal context [Matejka et al., 2005] has been widely used to discriminate between various languages and dialects not including Hungarian. The intuition here is that a robust phone recogniser is capable of extracting an accurate phonotactic pattern for the recognised language. It is worth noting that most of that common practice in phoneme based systems consists of multiple phone recognisers and combine them to enrich the feature vectors used for classification. In general, a phone recogniser system can produce the phone sequence

along with the duration for each phone, which is one of the prosodic features. Most of the phonotactic-based approaches discard the duration and use the sequence of the sound units. This thesis will study the duration in the context of dialectal Arabic versus standard Arabic.

4.1.4 Lexical features

The LVCSR system will ideally convert a raw audio file into the most likely spoken word sequence given that the speech system knows the language of the spoken audio, which is missing in the context of language recognition. Therefore, it is not a surprise that the word sequence feature vector is not commonly used in spoken language recognition. On the contrary, text-based language recognition system mainly relies on lexical features, word sequence and derivative features; phrases, grammars and character features. This thesis is concerned with spoken Arabic dialect identification; therefore, it is plausible to assume that the Arabic audio input can be recognised by a generic Arabic LVCSR system trained on multi-dialectal speech data. The recognised word sequence from such a system and derivative features can be used to build a lexical-based feature vector to be used for identifying the dialect of the spoken audio [Malmasi et al., 2016, Zampieri et al., 2017].

4.2 Statistical modelling

As shown in section 4.1, there are different kinds of features used in spoken language identification, and based on the information source, the model will vary. A wide spectrum of approaches have been proposed for modelling the characteristics of languages. The commonly used features can be summarised into two groups: (*i*) acoustic-based features, e.g., MFCC-SDC, filterbank and bottleneck features, and (*ii*) sound pattern features, e.g., phonotactic, lexical and derivative features such as character sequence. During the training phase, speech utterances are analysed and models are built using training data given the language labels. The models are intended to represent some language-dependent characteristics seen on the training data. During testing, each utterance will apply the same pre-processing steps for training; the likelihood for each utterance will be computed afterwards for each language. Finally, the language with the highest score

is typically assigned to this utterance. It is worth noting that recent approaches in language recognition explored using a filter-bank for feature extraction [Zazo et al., 2016].

4.2.1 Acoustic-based modelling techniques

Similar to speech recognition, given the extracted acoustic feature vector from an audio file is $o = o_1, o_2, ..., o_T$, where o_t represent the acoustic feature vector for frame t, and there are N possible languages where and all of them are equally probable. The language identification problem can be described as follows:

$$\mathbf{L_h} = \arg\max_{l} p(\mathbf{o}|\mathbf{L_l}) \tag{4.2}$$

Equation 4.2 follows the maximum-likelihood (ML) criterion where $\mathbf{L_h}$ is the most likely language label. The language recognition and speaker recognition are closely related in both modelling and evaluation. The universal background model GMM (UBM-GMM) is often used in acoustic-based modelling. There are two main differences to speaker recognition: (*i*) The UBM is a background model representing all the languages, where typically the UBM model is trained with a sample from all the N languages in the training data, e.g., 5-10 hours from each language; and (*ii*) the GMM is adapted using the MAP adaptation as mentioned in section 3.4 using all the available data from each language in the training data to build specific GMM for each language; practically the GMM has between 512-1024 Gaussians.

Recently, the i-vector has shown to be effective for language recognition similar to the success in speaker recognition. Ideally the outcome from the latent variable is of a fixed length per utterance, practically 100-400 dimensional feature vector per utterance. It is worth noting that the compact i-vector representation yields better results when dimensionality reduction such as linear discriminant analysis (LDA) is used. The extracted vector representation is also often used in vector space modelling (VSM), and the similarity between the vectors can be measured using cosine distance [Salton and Buckley, 1988, Chu-Carroll and Carpenter, 1999]. This will be discussed further in section 6.4.

4.2.2 Sound-pattern modelling techniques

Sound patterns can be referred to as the constraints that determine permissible syllable structures in a language [Li et al., 2013]. This can include phoneme sequence, word sequence, and character sequence. The most common pattern is the phoneme sequence, which is also known as the phonotactic system. This section will use the phonotactic as an example to highlight the common techniques; however, same approaches are applicable to character and to word sequences as well. Given the extracted phoneme sequence from an audio file is $x = \{x_1, x_2, ..., x_T\}$, where x_t represents the spoken phoneme sound at time t, and assuming that there are N languages and all of them are equally probable, the language identification problem can be described as follow:

$$\mathbf{L_h} = \arg\max_{l} P(\mathbf{x}|\mathbf{L_l}) \tag{4.3}$$

The main difference between equation 4.3 and equation 4.2 is that $P(\mathbf{x}|\mathbf{L_l})$ is a discrete probability model for the phoneme occurrence and co-occurrences. One of the widely used techniques is the phoneme n-gram modelling. The method is based on using the phone recognition followed by language modelling (PRLM). The used language for the phone recognition can be one of the target languages, as it can also be a disjoint language to any of the target N languages. One common practice for using the PRLM is to build an n-gram language model using the phoneme sequence for each of the N languages in the training data. During testing, the same pipeline should be applied to extract the phoneme sequence, and measure the perplexity across all the N n-gram language models. A lower perplexity shows that a phone n-gram matches the phone sequence better; in other words, the phone sequence is more predictable.

Figure 4.3 shows a scenario for using the PRLM. Following the same assumption, more than one phone recogniser can be deployed followed by an n-gram language model. The intuition here is that multiple phone recognisers should provide different perspectives for the statistics of the test audio file. This Parallel PRLM (PPRLM) can be seen as a system combination to fuse the results from multiple recognisers. Also, the output from the multiple recognisers is often used by another classifier, e.g., logistic regression or support vector machine (SVM), where both are examples of discriminative models used regularly in a spoken language identification task.

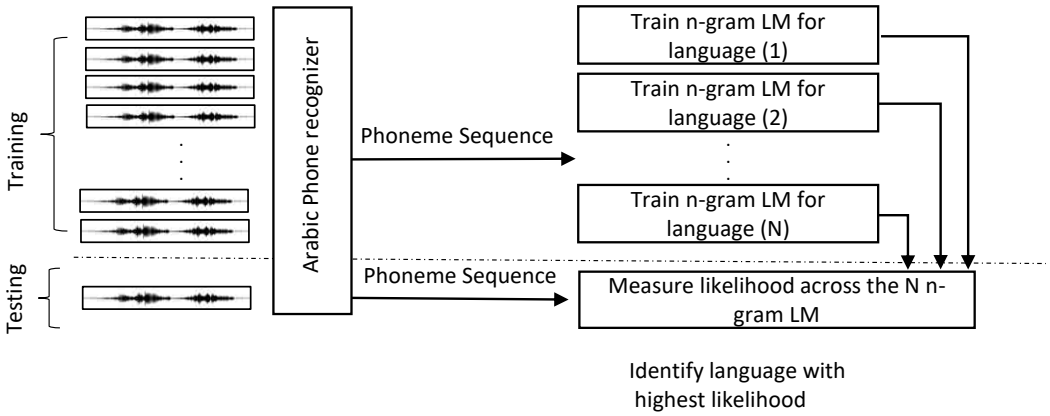


Figure 4.3: Example for PRLM using Arabic phone recogniser.

4.3 Overall system

Most of the modern spoken language identification systems will benefit from system combination. The combination approaches have traditionally been tuned to have higher accuracy than the best subsystem. Generally, system combination includes one or more of the acoustic and of the phonotactic sub-systems. There are various approaches to combine systems. These can be grouped into two groups: (i) combine systems at the feature level and use single classifier in the testing phase, and (ii) build individual sub-systems and combine scores from different each sub-system using some balanced voting. Ideally in the second approach, each system will have different weight that needs to be tuned using development data as part of the hyper-parameter optimisation. This thesis concerns the second approach in system combination and combines scores from different sub-systems.

4.4 Evaluation metric

Since the language identification task is a standard statistical classification problem, this thesis will report precision (positive predictive value), recall (false negative value) and overall accuracy on the test set for evaluation and comparison. Precision, recall and overall accuracy are calculated using true positives t_p, true negative t_n, false positives f_p and false negative f_n. Equations 4.4, 4.5 and 4.6 show how to calculate them. It is worth noting that evaluation is one of the shared items between the speaker and the language recognition challenge. The National Institute of Standards and Technology (NIST) has adopted another evaluation

metric; equal error rate (EER), as the default method for evaluating language and speaker recognition. The EER is a different score to measure the accuracy of a statistical classification, where the value for which the false acceptance errors and false rejection errors are equal. In this thesis, we will report precision, recall and overall accuracy.

$$Precision = \frac{t_p}{t_p + f_p} \tag{4.4}$$

$$Recall = \frac{t_p}{t_p + f_n} \tag{4.5}$$

$$Accuracy = \frac{t_p + t_n}{t_p + t_n + f_p + f_n} \tag{4.6}$$

4.5 Arabic dialect identification overview

The previous sections in this chapter gave an overview of language identification, which is a closely related problem to spoken Arabic dialect identification. One can argue that dialect identification is a harder problem since there is no single view on how many spoken dialects there are in Arabic. This section will summarise some of the previous work in dialectal Arabic identification. Biasdy et al. studied spoken dialectal Arabic in depth and the impact on Arabic ASR [Biadsy and Hirschberg, 2009, Biadsy et al., 2009b, 2010, Biadsy, 2011]. They studied spoken dialect recognition, and their main focus was phonotactic, prosodic and acoustic modelling. They also studied the classification among four dialects: Gulf, Iraqi, Levantine, and Egyptian. For classification, they studied kernel and HMM for modelling. Biasdy reported EER for the shared results. They concluded from their experiments that phonetic features alone carry significant and nearly sufficient information to distinguish dialects. Furthermore, they have shown that, if it is possible to cluster the ASR acoustic training and testing data based on dialect labels and then use it to train dialect-specific models, this will improve ASR on dialectal Arabic.

In a study by Akbacak et al. [2011], they studied dialect-specific and cross-dialectal phonotactic models, using both language models and an SVM classifier. Their system was deployed as a stand-alone and in combination with a cepstral system with joint factor analysis (JFA). The system was evaluated using four

dialects: Levantine, Iraqi, Gulf, and Egyptian. The speech sample was telephony sampled at 8Khz with 30-seconds fixed length each. They achieved 2% average EER for pairwise classification. In Liu et al. [2010], a systematic assessment of the differences between the acoustic characteristics of spontaneous and read speech and their effects on dialect identification performance was applied. They reported that each spans different dialect spaces and with distinct characteristics that need to be addressed respectively. From this comparison, they proposed a novel feature extraction technique.

4.6 Summary

In this chapter, we presented a brief overview of the different aspects of spoken language and dialect identification. For the acoustic features, we discussed the three most commonly used features: MFCC+SDC, bottleneck features, and sound pattern features such as phonotactic and words/character extracted from LVCSR speech recognition system. For statistical modelling, we highlighted the most common techniques, such as VSM, and the commonly-used classifiers, PRLM and PPRLM, SVM, and logistic regression. This was followed by a quick analysis of previous work in spoken dialectal Arabic identification. In the subsequent sections, we will study how to use crowdsource efficiently to build dialectal corpus and our effort in building dialectal Arabic classifier across the five Arabic dialects: Egyptian, Levantine, Gulf, North African, and modern standard Arabic.

Part II

Dialect Identification

This is the second section of the thesis and it is concerned with detecting spoken Arabic dialect. It has the following two chapters:

Chapter 5 introduces our effort in building a dialectal Arabic corpus using a crowd source approach to be used in the dialect identification work.

Chapter 6 introduces our efforts in building dialectal Arabic identification systems using data from broadcast domain.

Chapter 5

Crowd-Sourcing Dialectal Arabic

This chapter is based on [Wray and Ali, 2015] published at Interspeech 2015 and concerns using crowdsourcing to build a dialectal Arabic corpus for Arabic dialect identification.

5.1 Introduction

Arabic is a language with great dialectal variety, with modern standard Arabic (MSA) being the only standardised dialect. Spoken Arabic is characterised by frequent code-switching between MSA and dialectal Arabic (DA). DA varieties are typically differentiated by region, but despite their wide-spread usage, they are under-resourced and lack viable corpora and tools necessary for speech recognition and natural language processing. Existing DA speech corpora are limited in scope, consisting of mainly telephone conversations and scripted speech.

In this chapter, we describe our efforts for using crowdsourcing to annotate a multi-dialectal speech corpus collected from Al Jazeera. We obtained utterance-level dialect labels for 57 hours of high-quality audio from Al Jazeera consisting of four major varieties of DA: Egyptian, Levantine, Gulf, and North African. Using speaker linking to identify utterances spoken by the same speaker, and measures of label accuracy likelihood based on annotator behavior, we automatically labelled an additional 94 hours. The complete corpus contains 850 hours with approximately 18% DA speech.

As mentioned in chapter 2, Arabic consists of numerous varieties. MSA is the standardised dialect of news media and schooling, and the varieties of DA that characterise day-to-day usage can be very roughly categorised into four broad cat-

egories based on region of usage: Egyptian, Levantine (spoken in Syria, Lebanon, Jordan, and Palestine), Gulf (spoken in Saudi Arabia, Qatar, the United Arab Emirates, Bahrain, Oman, Kuwait, Yemen, and Iraq), and North African (spoken in Morocco, Libya, Tunisia, Algeria, and Mauritania.)

Existing Arabic speech corpora are dominated by MSA, and the few colloquial resources (with notable exceptions: 20 hours of Egyptian [Wray, 2016], 45 hours of Levantine [Technologies et al., 2005], 32 hours of Gulf, Levantine, and Egyptian [Almeman et al., 2013], 15 hours of Gulf [Elmahdy et al., 2014]) consist of narrow bandwidth telephone conversations. More detail about Arabic can be found in chapter 2.

Crowdsourcing has become a standard method for accessing large numbers of participants who are demographically diverse and harbor a number of skillsets that can be utilised for collection and annotation of data in various speech and language processing studies, such as text corpus construction [Dolan and Brockett, 2005] and acquisition of translations [Zaidan and Callison-Burch, 2011]. Within the specific domain of speech data, crowdsourcing has been used effectively for transcription of speech [Marge et al., 2010], and collection of speech via prompts [Lane et al., 2010, Davel et al., 2012, Novotney and Callison-Burch, 2010b], among other tasks. Numerous studies have investigated the development of quality control mechanisms which can be used to obtain expert-level quality data at a much lower cost [Snow et al., 2008, Novotney and Callison-Burch, 2010a], making crowdsourcing a viable method for efforts of speech corpus building and labelling.

In this chapter, we present a multi-dialectal speech corpus of DA created from high-quality broadcast, debate, and discussion programs from Al Jazeera, and as such containing a combination of spontaneous and scripted speech. We utilise human computation by means of crowdsourcing, and we develop methods for selecting representative utterances for each speaker to minimise the necessity of complete human annotation for the whole corpus. The chapter is organised as follows: in Section 5.2, we describe the process of collecting speech data from Al Jazeera and selecting representative samples from each speaker for manual classification. Section 5.3 presents the role of human annotation and development of best practices for obtaining reliable classifications. Then, annotator behavior and implications for perception are described in Section 5.4. Section 5.5 shows the results of generalising dialect information for all speech data based on annotations for representative samples.

5.2 Speech data

The Qatar Computing Research Institute (QCRI) has worked closely with Al Jazeera to develop a transcription queue which allows journalists and editors at Al Jazeera to choose episodes to be automatically transcribed by the QCRI advanced transcription system (QATS) [Ali et al., 2014c]. All videos processed by QATS appear on Al Jazeera's Arabic site aljazeera.net. The transcriptions have been formatted into SRT[1] and Distribution Format Exchange Profile (DFXP) subtitles and have been uploaded to the Brightcove video platform. The audio that makes up the corpus in the current study was pulled from videos in the transcription queue in the time period between June 2014 and January 2015, with an average of 33 videos per day. In total, there were more than 8,500 video files, which contain approximately 850 hours of speech. The audio is a mix of programs, reports, and conversational debates. The data is 44.1 kHz with the highest quality that has been uploaded directly from Al Jazeera to Brightcove. After downloading the video files, we ran ffmpeg[2] to downsample to 16 kHz, and then ran each audio file through a pre-processing pipeline before submitting it to annotators.

The pre-processing stage consisted of the following steps. First, for each episode, we ran voice activation detection (VAD) to remove as many non-speech segments (such as music or white noise) as possible. Then, speaker diarization was performed to determine who speaks when, and to assign each segment a speaker ID. All the aforementioned data pre-processing was carried out using the LIUM system speaker diarization [Meignier and Merlin, 2010]. The output from the LIUM segmentation is typically small chunks of audio files containing information about speaker ID, speaker gender and duration of utterance. Table 5.1 shows some statistics about segment duration, we can see that most of the segments are between 10-20 seconds.

Less than 5	5-10	10-20	20-30	More than 30
0.3%	15.4%	39.4%	18.4%	26.4%

Table 5.1: Distribution of segment duration in seconds.

[1]Subtitle, also referred to as closed captioning
https://en.wikipedia.org/wiki/Subtitle_(captioning)
[2]https://www.ffmpeg.org/

5.2.1 Segmentation and speaker linking

As a result of processing the data using the LIUM system, the audio was split into 167,000 segments. Then, we ran a second step in which we concatenated consecutive segments from the same speaker if a one-second or less period of silence or non-speech separated them. The aim of this step was to reduce the number of segments to submit for manual labelling. At this stage, we also discarded any segment less than three seconds long as we felt dialect assessment would be too difficult for the annotator in such a short span of time. After concatenation, 121,000 segments remained. These 121,000 represent the *Expanded* data set which contains every utterance.

The LIUM system also provided speaker linking information in which different speech segments produced by the same speaker were assigned to the same ID within the same file. From the 121,000 segments, two segments per speaker per video were selected, typically the first and the last segments, resulting in a total of 47,696 segments of unknown dialects to be labelled by human annotators. This subset represents the *Sample* data set. The assumption was that labels for the Sample set can be generalised to segments from the same speaker in the Expanded data set. The crowdsourced labelling of the Sample set is described in Section 5.3 and the process of expansion of the Sample set to the Expanded set is evaluated in Section 5.5.

5.3 Crowd-sourcing task

Crowdsourced classifications were obtained via CrowdFlower (henceforth CF)[3], a service that utilises various worker channels including other microworking and rewards sites. Workers can also be targeted by country of user origin. The service also employs optional verification stages in which gold standard data can be used to verify the contributor answers as they are submitted. Additionally, it also makes use of a dynamic judgment system in which more annotators are recruited for items for which the inter-annotator agreement is low.

The output of CF tasks takes various forms. First, the service outputs the full collection of contributor answers. CF also aggregates these answers together, so each task item is assigned a single answer based on inter-annotator agreement

[3] `http://www.crowdflower.com`

scores and contributor trust calculated from their performance in previous tasks. The combined trust and inter-annotator agreement calculation is quantified in a value called *confidence*[4] which is also released with the aggregate data. Thus, each item in the aggregate data set is assigned an answer and a confidence value for that answer. CF also outputs a list of contributors who worked on the task with information about which worker channel they were recruited from and location data.

5.3.1 Task anatomy

The task was restricted to users in the Arab world. All directions for the task were written in modern standard Arabic. Contributors were directed to listen to the short speech segments described in Section 5.2.1 and to determine which dialect they thought the speaker was speaking. Contributors were asked to listen only as long as necessary to determine the dialect being spoken. Compensation for this task was USD 0.03 per page of 10 items.

Dialect judgment was answered by a seven-way forced-choice between Modern Standard Arabic (MSA), Levantine Arabic (LAV), Egyptian Arabic (EGY), North African Arabic (NOR), Gulf Arabic (GLF), non-Arabic speech, and non-speech. The non-Arabic speech in the data included foreign speakers who were not dubbed over. The non-speech included white noise, music, and other non-speech sounds such as traffic and gunfire, which were mislabelled by LIUM as speech data. For each regional variety of DA, contributors were explicitly instructed which countries belonged to which dialect groups.

5.3.2 Development of quality measures

Existing CF quality control options were utilised to reduce the amount of noisy data and post-crowdsource cleanup necessary. Twenty-five audio files were manually annotated to create a gold standard data set in order to use CF automatic quality control. These files were selected to be unambiguous and clear, and the answers distributed across categories with little potential for dispute, such as non-speech, non-Arabic, MSA, in addition to clear examples of DA. Pilot testing confirmed that the gold standard items were appropriately unambiguous.

[4]The confidence is calculated and reported by the CrowdFlower platform.

Live quality control was accomplished in two ways. First, CF optional Quiz Mode was engaged, which required contributors to answer five gold standard items before entering the main portion of the task. Second, for every five items, contributors were presented with a gold standard item that was not discernible from the task items. Contributors had to maintain an accuracy of at least 65% on these hidden gold standard items or else their participation in the task was ended. Although this cutoff point may appear too forgiving, pilot work showed that spammy annotators had an average accuracy of 31% on test questions, whereas the remainder of annotators had an average of 94% accuracy. In addition to utilising live quality control, efforts were also made to reduce the amount of data with low inter-annotator agreement. Recall that CF features a built-in mechanism for fetching additional contributors to provide judgments for items with low inter-annotator agreement. Recall also that each item is assigned a confidence value based on inter-annotator agreement and contributor trust. To determine the most effective way to utilise this feature, an experiment was performed on a random sample of 500 segments. This sample was submitted for contributor judgments three times on CF with different manipulations of both confidence thresholds and maximum number of contributors per item. Suggested settings for the dynamic judgments feature which automatically submits low-agreement items are to resubmit an item with lower than 70% to one additional contributor. This feature was tested on the 500 set, as well as a higher threshold of 75%, and two maximum contributor-per-item numbers: 7 and 9. This experiment demonstrated a gain in total percentage of high-confidence items, as the threshold was made stricter and the number of annotators higher. These results are summarised in table 5.2. After determining best practices for dynamic judgments and quality control, the 47,696 sample files representing 404 hours of speech were classified over a period of three weeks, costing a total of USD 971.

Threshold	Minimum contributors	Maximum contributors	% items above 70% confidence
70%	3	4	79%
75%	3	7	89%
75%	3	9	92%

Table 5.2: Percentage of high-confidence answers for 500 segments annotated with three dynamic judgment options.

5.3.3 Contributor demographics

A total of 2,053 users contributed to the labelling task, with 39% of contributors hailing from Egypt, the single highest country by contributor count. Complete contributor counts by country[5] are shown in Table 5.3. In comparing the numbers

Egypt	795	Saudi Arabia	80	Oman	10
Algeria	422	Palestine	51	Bahrain	4
Tunisia	303	Yemen	45	Qatar	3
Jordan	177	UAE	33		
Morocco	117	Kuwait	13	Total	2053

Table 5.3: Contributor count by country.

of contributors based on their dialect group, North African speakers contributed the highest total percentage to the task. Lowest participation by number of contributors was from countries in the Gulf. Percentages of total contributors per dialect group are shown in the map in Figure 5.1.

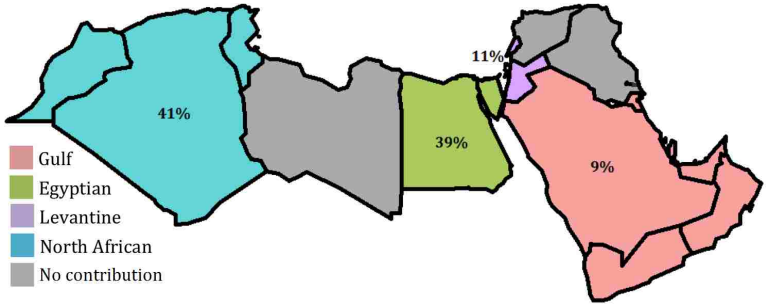


Figure 5.1: Map of contributor origin by dialect group.

Note in Figure 5.1 that although the Gulf region is a large multi-national group, it contributed a minority of the participants. Potential implications for this and other contributor origin-related phenomena are discussed in the following section.

[5]At the time of this study, CF was not available for residents of Iraq, Syria, Libya, and Lebanon. Although the task was available for users in South Sudan, no contributors participated from this country.

5.4 Dialect perception

Although the aim of this chapter is primarily concerned with resource improvement and data collection through crowdsourcing, insights on human perception were also investigated based on contributor behavior. We considered the possibility of annotator bias during the process of labelling, and explored implications of labels which regularly co-occurred.

5.4.1 Contributor bias

Overall, of the four major DA varieties, labels assigned to Egyptian had the overall highest average confidence value and labels for Gulf exhibited the lowest average confidence value. Percentages for confidence values for items are shown by label in Figure 5.2. Items were binned according to three confidence thresholds: less than 50% confidence, between 50% and 75% confidence, and finally above 75%.

Figure 5.2: Distribution of confidence by dialect

As for the relation between annotator origin and label assigned, Zaidan and Callison-Burch [2014] present evidence of annotator bias in a task identifying dialectal content in text mined from comments on on-line news articles. They found that annotators were biased towards selecting their own native dialect when asked to provide dialect judgments. Thus, Egyptian speakers often mistakenly annotated non-Egyptian comments as being Egyptian, Levantine speakers over-annotated sentences as Levantine, and so forth. This raises the question of the current study: Is there any evidence that contributors were biased towards selecting their own dialect when presenting with speech of unknown origin? To determine this, we also presented annotators with twenty-five manually-annotated items per DA category to compare behavior across origins of annotator.

A chi-square test of independence was performed to determine whether an

annotator's dialect of origin affected their selection and therefore whether DA selections were equally distributed. Annotators chose their own dialect 22 ± 3.7% of the time, which was not significantly different from their probability of choosing other dialects 22.2 ± 2.7% of the time;($(x^2(12)=12.7, p=0.4)$). Thus, contributors did not exhibit a bias to their own native dialect group in the process of making dialect judgments.

Although it may be difficult in certain contexts to determine the dialect of a written comment if it contains graphemic cognates common across multiple dialects of colloquial Arabic and even modern standard Arabic, this ambiguity is absent in spoken utterances. The specific cues that lead to differences in dialect confusability across written and spoken modalities are beyond the scope of this chapter, but merit further investigation.

5.4.2 Interdialectal confusability

Recall that a label is assigned to an item based on the judgments of several annotators and in the event an item exhibited low inter-annotator agreement, more annotators would automatically be obtained to provide additional judgments. Each label then is the product of judgments from 3-9 different annotators. However, what was the cause of low agreement in the first place, and was there a pattern to contributor disagreement? To investigate the rates of confusability between dialects and the amount of ambiguity which led to high competition between multiple dialect judgments for one item, we counted each judgment provided to each label. Percentages are shown in Table 5.4.

	Percentage of total judgments				
Label	EGY	GLF	LAV	NOR	MSA
EGY	79.6%	1.3%	2.6%	1.4%	15.1%
GLF	1.4%	61.3%	11.9%	5.2%	20.2%
LAV	1.7%	6.8%	73.8%	3.6%	14.1%
NOR	0.6%	5.1%	5.3%	70.7%	18.3%

Table 5.4: Percentages of judgments by each dialect label with respect to the five-dialects. i.e., each row sums up to 100%.

Results suggest that Egyptian is easily distinguished from other varieties of DA, likely due to its wide-spread representation in media consumed through-

out the Arabic-speaking world. This interpretation is consistent with the high confidence values for EGY labels as shown in section 5.2. Although the GLF label exhibits the highest percentage of competition between GLF judgments and MSA when compared to other DA varieties (20.2% of GLF labels contained MSA judgments, whereas 15.8 ± 2.2% of EGY, LAV and NOR labels contained MSA judgments), a chi-square test of independence shows this difference was not significant (($x^2(1)$=0.91, p=0.3)).

5.5 Expansion results

Recall that the annotated audio set was a subset of the larger audio set. In the process of linking annotated Sample files to the Expanded set in order to generalise contributor judgments, we explored three possible confidence threshold levels for expansion. First, we started with no threshold. All Sample items were eligible for expansion, and whatever answer was selected based on highest inter-annotator agreement and contributor trust was linked to the other files in the Expanded set. The second threshold was set at 50% confidence. At this threshold, any item with at least 50% confidence contributed dialect labels to the files it was linked to in the Expanded set. Items with less than 50% confidence were discarded. Finally, the strictest threshold was the 75% confidence level.

5.5.1 Validating the expansion process

In order to compare the three possible thresholds of expansion, a sample of randomly-selected previously-unseen 200 items per confidence threshold per dialect from the expanded sets were submitted to CF for manual annotation. The purpose of this was to determine if propagating labels from the Sample set to the Expanded set resulted in accurate labels. Table 5.5 shows the results of manual annotation of the selected sample of items from each confidence threshold. Common sense would predict that discarding items that were labelled with low confidence values even after multiple additional annotators improves the total percentage of dialect data during the expansion process, and the manually annotated results confirm this: the total percentage of predicted dialect increases as the confidence threshold becomes more restrictive.

However, as shown in Table 5.5, given that even a strict threshold of 75%

Confidence threshold	Expected Dialect	Hours linked	Confirmed % of sample
None	EGY	32h 59m	17%
	GLF	27h 11m	25%
	LAV	55h 42m	19%
	NOR	27h 02m	16%
50%	EGY	31h 31m	36%
	GLF	22h 17m	39%
	LAV	50h 30m	31%
	NOR	24h 32m	36%
75%	EGY	26h 37m	65%
	GLF	12h 30m	41%
	LAV	38h 49m	53%
	NOR	18h 24m	69%

Table 5.5: Results of manually-annotated expansion sets.

does not produce full coverage of the predicted dialect, a question presents itself: what other speech is contained in the files and what makes it so easily confused with the predicted dialect?

5.5.2 Codeswitching

In looking at the results of the highest confidence threshold and the manually annotated dialectal labels versus the expected dialect labels, it is clear that using an sample-expansion system does not result in completely generalisable labels. However, a closer look reveals that this could be due to the nature of codeswitching. Arabic as a language is characterised by frequent bi-dialectal codeswitching, meaning a speaker alternates between their native dialect and MSA [Ferguson, 1959, Elfardy and Diab, 2012, Elfardy et al., 2013, Solorio et al., 2014, Elfardy et al., 2014]. Because of this fact, much of the remaining percentage of expected dialect data is in fact MSA, as shown in Table 5.6. (Remaining percentages belonged to Non-Arabic and Non-Speech categories.)

For speakers whose samples were labelled as a particular DA variety, the majority of their speech was indeed in that variety, with a minority being in MSA. The exception to this is the Gulf variety. It is therefore possible that Gulf speakers in the corpus used more MSA in their speech than their native dialect, but a comprehensive account of the differences in codeswitching for different DA varieties is warranted.

Expected Dialect	EGY	GLF	LAV	NOR	MSA
EGY	65%				32%
GLF		41%	4%		53%
LAV	1%	1%	53%		39%
NOR	1%			69%	28%

Table 5.6: Percentages of expected dialect (from expansion) of segment by actual dialect (from manual annotation).

5.6 Conclusions

This chapter presented our efforts to create a multi-dialectal corpus of Arabic speech[6] using audio from Al Jazeera. We showed that using CrowdFlower to label samples from each speaker at the beginning and at the end of an audio segment results in labels for all of that speaker's speech and that results are suggestive of a regular practice of code-switching between one's native dialect and MSA. The corpus has been automatically transcribed, and utterances determined as DA have also begun to be manually transcribed using crowdsourcing. The data with confidence 0.75 or higher will be used for testing and development in the Arabic dialect identification studies.

[6]The corpus can be accessed at http://alt.qcri.org/resources/aljazeeraSpeechCorpus/

Chapter 6

Arabic Dialect Identification

This chapter is based on [Ahmed et al., 2016, Khurana et al., 2017, Shon et al., 2017] published at Interspeech 2015, InterSpeech 2016 and ASRU 2017 respectively. This chapter will cover my contribution to the three papers. The main focus of the three aforementioned papers is Arabic dialect identification. This chapter will also discuss some experiments that have not been published yet.

6.1 Introduction

In this chapter, we investigate different approaches for Arabic dialect identification (ADI) in broadcast speech. These methods are based on phonotactic and lexical features obtained from a speech recognition system, and acoustic features using the i-vector framework. We studied both generative and discriminative classifiers, and we combined these features using a multi-class support vector machine (SVM), a deep neural network (DNN), and a convolutional neural network (CNN). We validated our results on an Arabic/English language identification task. We also evaluated these features in a binary classifier to discriminate between modern standard Arabic (MSA) and dialectal Arabic (DA). We further report results using the proposed methods to discriminate between the five most widely used dialects of Arabic: namely Egyptian, Gulf, Levantine, North African, and MSA.

We discuss dialect identification errors in the context of dialect codeswitching between DA and MSA, and compare the error patterns between labelled data, and the output from our classifier. All the data used in our experiments have been released to the public as a dialect identification corpus.

The task of dialect identification (DID) is a special case of the more general problem of language identification (LID). LID refers to the process of automatically identifying the language class for a given speech segment or text document. DID is arguably a more challenging problem than LID, since it consists of identifying the different dialects within the same language class. As discussed in section 2.5, Arabic dialects are sufficiently distinctive, and it is reasonable to regard the DID task in Arabic as similar to the LID task in other languages. Table 2.1 shows two phrases across the different dialects. It is clear from this example that there are lexical variations across the different dialects that motivate us to consider it.

Two broad LID approaches have been investigated in the literature: low-level acoustic features, and high-level phonetic and lexical features. In the lexical area, words, roots, morphology, and grammars [Reynolds et al., 2008, Ambikairajah et al., 2011] have been studied. Acoustic features such as shifted delta cepstral coefficients [Dehak et al., 2011b] and prosodic features [Martínez et al., 2012] using Gaussian mixture models (GMMs), i-vector representations and support vector machine (SVM) classifiers [Dehak et al., 2011b] have been shown to be effective for LID. More recent work explored the use of frame-by-frame phone posteriors (PLLRs) [Plchot et al., 2014] as new features for LID. New subspace approaches based on non-negative factor analysis (NFA) for GMM weight decomposition and adaptation [Bahari et al., 2014] were also applied to both LID and DID tasks. GMM weight adaptation subspaces seem to provide complementary information to the classical i-vector framework. Finally, phoneme sequence modelling and its n-gram subspace have been studied for both Arabic DID [Soltau et al., 2011] and LID [Soufifar et al., 2012].

In this chapter, we investigate three vector subspace models (VSMs) for ADI based on 1) lexical, 2) phonotactic, and 3) acoustics. We conduct a thorough feature selection study of these models to better understand their interaction. A further contribution of this chapter is the release of an ADI system, so others can extend and improve DID performance on this task.[1]. It is worth noting that Arabic dialect identification was introduced as a challenge at VarDial 2016 [Malmasi et al., 2016] and 2017 [Zampieri et al., 2017] and at MGB-3 [Ali et al., 2017b]. There have been more than 30 different submissions addressing challenges in dialectal Arabic in these three competitions.

[1]https://github.com/qcri/dialectID

6.2 Data corpus

The data for the ADI task comes from a multi-dialectal speech corpus created from Arabic broadcast, debate and discussion programs from Al Jazeera and other Arabic channels. The development and testing data were labelled using the crowdsource platform CrowdFlower, with the criteria to have a minimum of three judges per file and up to nine judges, or 75% inter-annotator agreement (whichever comes first). It is worth noting that this is a subset of the sampled data not the expanded corpus. More detail about the testing and development data can be found in chapter 5. On the other hand, recording the training data was done using satellite cable sampled at 16kHz directly from the broadcast speech from many Arabic broadcast channels.

Training data were manually segmented and labelled. Although the testing and the development data sets came from the same broadcast domain, the recording setup is different, which could potential lead to channel mismatch as we will discuss later in section 6.4. The training, development and testing data are well balanced across the five dialects studied in this chapter; EGY, LAV, GLF, and NOR, as well as in MSA. Table 6.1 shows some statistics about the ADI training, development and testing datasets.

		Training			Development			Testing		
Dialect	**Dialect**	**Ex.**	**Dur.**	**Words**	**Ex.**	**Dur.**	**Words**	**Ex.**	**Dur.**	**Words**
Egyptian	EGY	3,093	12.4	76	298	2	11.0	302	2.0	11.6
Gulf	GLF	2,744	10.0	56	264	2	11.9	250	2.1	12.3
Levantine	LAV	2,851	10.3	53	330	2	10.3	334	2.0	10.9
MSA	MSA	2183	10.4	69	281	2	13.4	262	1.9	13.0
North African	NOR	2,954	10.5	38	351	2	9.9	344	2.1	10.3
Total		13,825	53.6	292	1524	10	56.5	1492	10.1	58.1

Table 6.1: The ADI data: examples (Ex.) in utterances, duration (Dur.) in hours, and words in 1000s.

6.3 Features

This section investigates different features to identify spoken Arabic dialect. For the acoustic features, we studied the MFCC-SDC in section 4.1.1 and bottleneck features using i-vector as a latent variable representation in section 4.1.2. For the linguistic features, we studied phonetic and lexical features.

6.3.1 Acoustic representation

Recently, bottleneck features extracted from an ASR DNN-based model were applied successfully to language identification [Song et al., 2013, Matejka et al., 2014, Richardson et al., 2015a]. In this chapter, we used a similar bottleneck feature configuration to the ASR-DNN system for MSA speech recognition [Cardinal et al., 2015]. This system is based on two successive DNN models. Both DNNs use the same setup of 5 hidden sigmoid layers and 1 linear BN layer, and they were both based on tied states as target outputs. The senone labels of dimension 3,040 are generated by forced alignment from an HMM-GMM baseline trained on 60 hours of manually transcribed Al Jazeera MSA news recordings [Ali et al., 2014b]. The input to the first DNN consists of 23 critical-band energies that are obtained from Bark scale. Pitch and voicing probability are then added. With 11 consecutive frames are then stacked together. The second DNN is used for correcting the posterior outputs of the first DNN. In this architecture, the input features of the second DNN are the outputs of the BN layer from the first DNN. Context expansion is achieved by concatenating frames with time offsets of -10, -5, 0, 5, and 10. Thus, the overall time context seen by the second DNN is 31 frames. For modelling, we use the i-vector approach which has been introduced in section 3.4. The universal background model (UBM) – typically a large GMM with 2,048 components, the bottleneck features were used as input feature, and the i-vectors were of 400-dimensional for each utterance. In order to maximise the discrimination between the different dialect classes in the i-vector space, we combine linear discriminant analysis (LDA). Both SVM and DNN were studied for the ADI classification task.

Unlike the bottleneck features, which combined information from acoustic and phonetic information, we studied pure acoustic features, we parametrised the speech signal by extracting *Mel-frequency cepstral coefficients* (MFCCs) per 25 ms sliding window over the speech signal, having a 10 ms overlap. The MFCC feature vector is enriched using *shifted delta cepstral coefficients* (SDCs) [Torres-Carrasquillo et al., 2002]. We use the configuration 7-1-3-7 for extracting the MFCC-SDC features, similar to the one used in [Zazo et al., 2016], more details about SDC settings can be found in section 4.1.1. The aforementioned approach gives us a sequence of feature vectors for each spoken utterance. We use Kaldi [Povey et al., 2011], a publicly available Automatic Speech Recognition toolkit,

for feature extraction. Similar to the bottleneck features, we used the i-vector framework and we applied LDA for dimensionality reduction.

6.3.2 Lexical representation

The word sequences are extracted using a state-of-the-art Arabic speech-to-text transcription system built as part of the multi-genre broadcast challenge (MGB-2) [Ali et al., 2016]. The system is a combination of a time-delayed neural network (TDNN), a long short-term memory recurrent neural network (LSTM) and bidirectional LSTM (B-LSTM) acoustic models, followed by 4-gram and recurrent neural network (RNN) for language model rescoring. Our system used a grapheme lexicon. The acoustic models are trained on 1,200 hours of Arabic broadcast speech. More details about the ASR system are covered in chapter 7.

Each utterance is represented using vector space modelling (VSM). The word-based utterance VSM ($\mathbf{U_w}$) is constructed as follows. An ASR system is used to extract the word sequence for each utterance in the speech database. Given that the ASR system is not tailored to any specific dialect, there are often many out-of-vocabulary dialectal words. In an attempt to capture this pattern, we kept the *unknown* word that is produced by the ASR indicating the possibility of an out-of-vocabulary word. Each speech utterance ($\mathbf{u}$) is then represented as a high-dimensional sparse vector ($\vec{u}$):

$$\vec{u} = (A(f(u, w_1)), A(f(u, w_2)), \ldots, A(f(u, w_{d'}))) , \tag{6.1}$$

where $f(u, w_i)$ is the number of times a word w_i occurs in the speech utterance u and A is the scaling function. The identity scaling function and $tf.idf$ scaling function, commonly used in the field of natural language processing [Ramos, 2003] to downweight the contribution of the words that occur in almost all documents (in utterances in our case), these words do not provide enough discriminative information across the other documents (utterances). The vocabulary size was 55K words.

The vector space is then represented by the matrix, $\mathbf{U_s} \in \mathbb{R}^{d \times N}$ (see Fig 6.1). This approach and the notation used to define a VSM is directly inspired by the seminal works in the area of VSM of natural language processing in [Salton et al., 1975, Lowe et al., 2000, Padó and Lapata, 2007] and in language identification in Li et al. [2007].

$$\mathbf{U_W} = \begin{array}{c} \\ w_1 \\ w_2 \\ \vdots \\ w_d \end{array} \begin{array}{c} \begin{array}{cccc} u_1 & u_2 & ... & u_N \end{array} \\ \begin{bmatrix} A(f(w_1,u_1) & A(f(w_1,u_2) & \dots & A(f(w_1,u_N) \\ A(f(w_2,u_1) & A(f(w_2,u_2) & \dots & A(f(w_2,u_N) \\ \vdots & \vdots & \ddots & \vdots \\ A(f(w_d,u_1) & A(f(w_d,u_2) & \dots & A(f(w_d,u_N) \end{bmatrix} \end{array}$$

Figure 6.1: *Word-based utterance VSM. The column vectors of the matrix correspond to the speech utterance vector representation formed using equation 6.1, d is the size of the word dictionary, and N is the total number of speech utterances in the dialectal speech database.*

As an extension for the lexical features, we experiment with character-based features, which are extracted from the same word sequence using the same LVCSR system. Similar to the lexical features, we kept the *unknown* from the ASR, and we replaced it with a special character to keep the out-of-vocabulary (OOV) information. Space was inserted between all characters including the word boundaries. This led to 38 characters as the vocabulary size for the character-based system.

6.3.3 Phonotactic representation

The phonotactic approach was based on multiple phoneme recognisers as introduced in section 5.3. Initially, we experimented with Arabic phoneme recognition, in which a phoneme recogniser is used to extract the n-gram phone sequence. The phoneme sequence is obtained by automatic vowelization of the training text, followed by vowelization to phonetization (V2P). The 36 chosen phonemes cover all the dialectal Arabic sounds. Further details about the speech recognition pipeline, training data, and phoneme set are given in [Ali et al., 2014b]. For the phoneme sequence, we process the phoneme lattice, and obtain the one-best transcription, ignoring silences as well as noisy silences. As mentioned earlier, the pipeline included V2P, which is based on vowelizing the input text. For this purpose, MADA [Habash et al., 2005] was used for automatic vowelization.

The Arabic phoneme recognition was mainly based on MSA data, mainly because the automatic vowelization did not perform as well on dialectal data. Therefore, we limit the training for the 60 hours MSA Broadcast speech corpus from the multi-genre broadcast MGB-2 Arabic. This will be discussed more in

chapter 7. As mentioned in section 4.1.2, a robust phoneme recognisers for other languages has shown superior results. For example, the Hungarian phoneme recogniser based on long temporal context has been widely used to discriminate between various languages and dialects not including Hungarian. We explored the Czech, Hungarian and Russian using narrowband model, and English using a broadband model [Matejka et al., 2005]

Phoneme duration: In an attempt to study the speech pattern in dialectal Arabic, we investigated phoneme duration in all Arabic dialects. We also ran the same experiment using English speech data: 10 hours randomly selected from MIT open courses corpus [Glass et al., 2007]. We investigated phoneme duration features, using both the average phoneme duration for each dialect, and the duration of the consecutive phonemes. Sampled information for average phoneme duration for each dialect as well as English can be found in table 6.2. Both the phoneme sequence and the phoneme duration[2] are extracted using the aforementioned Arabic phoneme recogniser. There were only negligible differences in average phoneme duration across different Arabic dialects. However, average phoneme duration is a more useful feature, when English and Arabic (both MSA and dialectal Arabic) are compared. More interestingly, phoneme duration does differ between MSA and dialectal Arabic. One plausible explanation for this is that MSA is nobody's native dialect, which means that MSA is a formal speech scenario used in: news, lectures, and formal presentations. However, DA is more widely used in day-to-day communication. It can be argued that the difference between MSA and DA is mainly the pattern of talking rather than yet another dialect.

Phone	EGY	GLF	LAV	NOR	**MSA**	**English**
b	90	90	91	91	**85**	**97**
i	45	46	45	43	**39**	**52**
m	80	81	82	81	**75**	**92**
z	130	137	139	141	**122**	**159**
Z	140	131	141	135	**117**	**153**

Table 6.2: Sample phoneme duration of Arabic phoneme recognition across the major Arabic dialects and English.

[2]It is worth noting that phoneme duration is not normalised with respect to speaker and environment factors.

We also investigated individual phoneme duration features, in which a VSM feature vector could potentially look like: *g_10 T_20 S_200 a_30 A_ 50*. We binned the durations using 40 bins from 0 to 400 ms with a 10 ms window, thus avoiding a long tail of phonemes with longer durations, resulting in 1,440 unique phoneme features in total (40 per phone for 36 phones). We also investigated using unlabelled phone durations, so the previous feature vector will look like *10 20 200 30 50*, resulting in 40 unique features. In both cases, discriminating between MSA and DA yielded 100% accuracy using a simple logistic classifier. Also, 100% accuracy was achieved for discriminating Arabic speech from English.

6.4 Experiments

In this section, we explored several models to study the acoustic- and linguistics-based features. Given that the training and the development have been released for the VarDial and the MGB competitions, but not the test set, we used the training data to train different models and the development data to report results. The final results in the system combination in section 6.5 used the test data.

Best Classifier: We studied the best classification approach for the ADI task from a set of two generative models: n-gram language model [Roark et al., 2007] and Naive Bayes [Frank and Bouckaert, 2006], and two discriminative classifiers: linear SVM [Drucker et al., 1999] and Maximum Entropy [Nigam et al., 1999]. We also explored using deep neural networks for classification. The DNN has an embedding layer of 256 dimensions followed by two fully connected feed forward layers 64 and 32 dimensions using a $ReLU$ activation function. The DNN has a dropout of 0.2. We measured the performance of each model on the word-based vector space model, which was constructed using the approach mentioned in section 6.3.2, using identity scaling function A, and performing no dimensionality reduction. Hence, the dimensionality of an utterance vector, $\vec{u}$, is the same as the size of the lexicon, which in our case was 55k.

Finally, inspired by the recent success of using convolutional neural networks (CNNs) for text classification [Kim, 2014], we explored using CNN for dialect identification, and here we introduce a brief comparison with the other classification techniques. More detail about the deployed CNN is depicted in section 6.4.2. The results can be seen in table 6.3. Both, CNN and SVM with linear ker-

nel perform the best, with slightly better results when using the CNN classifier. Therefore, it is our choice that both will be used as back-end classifiers for the rest of the experiments in this chapter. However, for the acoustic methods, we decided to consider the DNN for classification as the internal layers can help in reducing the dimensionality of the acoustic feature vector and can be comparable with the LDA combined with SVM. In section 6.5, we studied system combination, and this was done on the score level to achieve the best results in the overall ADI system.

Model	Precision	Recall	Overall Accuracy
n-gram Language Model	0.44	0.44	0.42
Naive Bayes	0.41	0.53	0.39
Maximum Entropy	0.44	0.43	0.42
Two layers neural network	0.44	0.45	0.44
Support Vector Machine	**0.49**	**0.48**	**0.47**
Convolutional Neural Networks	**0.51**	**0.50**	**0.50**

Table 6.3: *Performance of different classifiers using lexical features with lexicon size of 55K. All the models were evaluated on the development data set.*

6.4.1 Acoustic methods

In this section, we studied two acoustic representations: the bottleneck features and the MFCC-SDC features. The i-vector latent variable is deployed as a final representation for both features. Each utterance is finally represented as a 400-dimensional feature vector.

Modelling the bottleneck i-vector features: The bottleneck features were extracted as explained in section 6.3.1. The i-vector representation has been widely used in speaker recognition and language recognition, and in this chapter we adopt the i-vector approach as explained in section 3.4. Similar to equation 3.12, we use the i-vector as low dimension representation for each sentence.

$$M = m + Tw \tag{6.2}$$

The main differences here compared to using the i-vector for speaker recognition are the following: (i) m is the dialect independent supervector, and (ii) w is the

factor that describes each utterance. In our experiments, the UBM is a GMM comprised of 2,048 components, and the i-vector dimension is 400. In order to maximize the discrimination between the different dialect classes in the i-vector space, we combine LDA as dimensionality reduction for the 400-dimensional i-vector to 4 dimensions. Table 6.4 shows the results before and after dimensionality reduction for both the SVM and DNN classifiers, which shows the superior performance of the bottleneck representation compared to the lexical representation in table 6.3. There is very little difference between the SVM and the DNN. We can see small gain from the LDA projection in terms of precision for both systems. Figures 6.2 and 6.3 show the LDA projection for the five dialects.

It is clear that MSA is different from the rest of the dialects, and also NOR is separate enough from EGY, GLF and LAV; this is indeed true from a linguistic point of view. We can see a similar pattern for the the testing and the development data. However, figure 6.4 shows a different pattern for the training data in the same LDA space; this can be justified as a result of the channel mismatch between the training data versus the testing and the development data, which is expected to be visible in the acoustic space.

Bottleneck i-vector features	Classifier	Precision	Recall	Overall Accuracy
400 dimensions	SVM	0.61	0.58	0.57
4 dimensions LDA projection	SVM	0.62	0.58	0.58
400 dimensions	DNN	0.60	0.56	0.57
4 dimensions LDA projection	DNN	0.62	0.59	0.58

Table 6.4: Evaluating bottleneck i-vector features across the five dialects.

Modelling the MFCC-SDC features: We parameterised the speech signal using the MFCC-SDC as described in section 6.3.1. Similar to the bottleneck features, the i-vector representation is deployed for each sentence using a 400-dimensional feature vector. We also experimented to classify the five dialects using the raw 400 MFCC-SDC and also applying the LDA dimensionality reduction. Table 6.5 shows the results before and after dimensionality reduction, which shows better accuracy compared to both lexical and phonetic VSM, but significantly worse than the bottleneck, which is expected since the bottleneck representation holds information about both acoustic and linguistic features. Therefore, the bottleneck features are our choice in the final system combination.

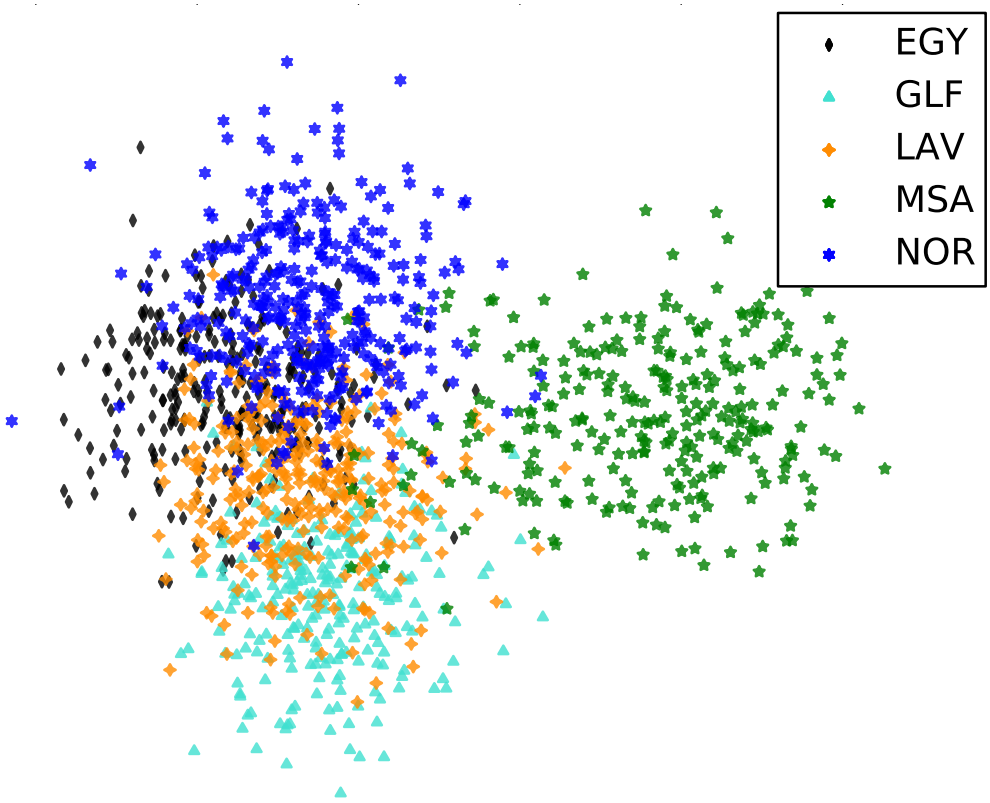


Figure 6.2: LDA projection for the BNF development data.

MFCC-SDC i-vector features	Classifier	Precision	Recall	Overall Accuracy
400 dimensions	SVM	0.50	0.49	0.48
4 dimensions LDA projection	SVM	0.52	0.50	0.49
400 dimensions	DNN	0.51	0.50	0.50
4 dimensions LDA projection	DNN	0.51	0.50	0.52

Table 6.5: Evaluating MFCC-SDC i-vector features across the five dialects.

6.4.2 Lexical methods

Given that the dialectal data used in this chapter have not been manually transcribed, we used standard automatic speech recognition to extract the word sequence corresponding to each utterance. Word sequences are extracted as described in section 6.3.2. We kept the *unknown* word that indicates the possibility of an out-of-vocabulary words. In this section, we explored the word sequence as well as the character sequence. Table 6.6 shows some linguistic statistics for the training, the development and the testing datasets, including the frequency of the *unknown* word. One interesting finding here is that the frequency of the *un-*

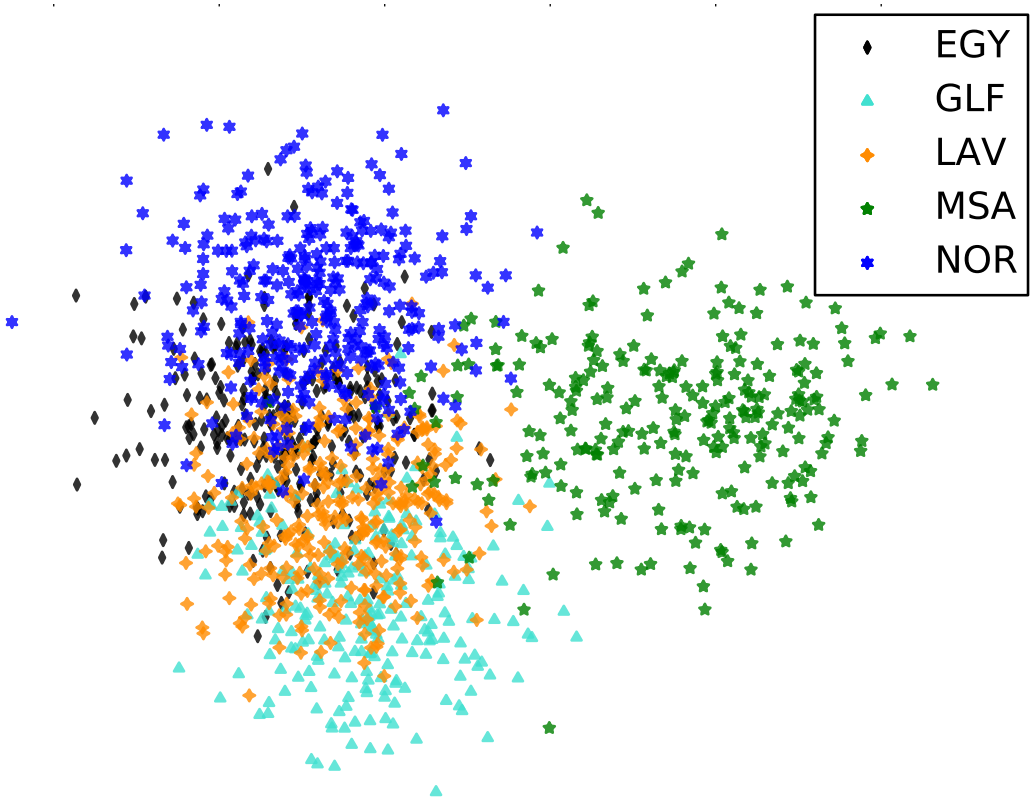


Figure 6.3: LDA projection for the BNF testing data.

known is small in MSA compared to the other four dialects, which is expected as the speech recognition training data is mainly dominated by MSA. More details about the speech recognition chapter will be explained in chapter 7.

As part of the linguistic feature ablation study, we deployed the SVM classifier twice; for the word features and for the character features. Table 6.7 shows the results across the five dialects. The hyper-parameters for the SVM classifier were tuned for each system. The bigram context was enough for the word-based system, while the character-based system benefited from larger context; we found no gain beyond 5-gram context for the character-based system. Finally the dictionary size for the character-based system was slightly bigger than the word-based system 294K versus 230K. The confusion matrix from the character based and the word-based, and the error pattern was similar, we therefore decided to explore different classification techniques. In the following section, we explore using a convolutional neural network for classification using the lexical features.

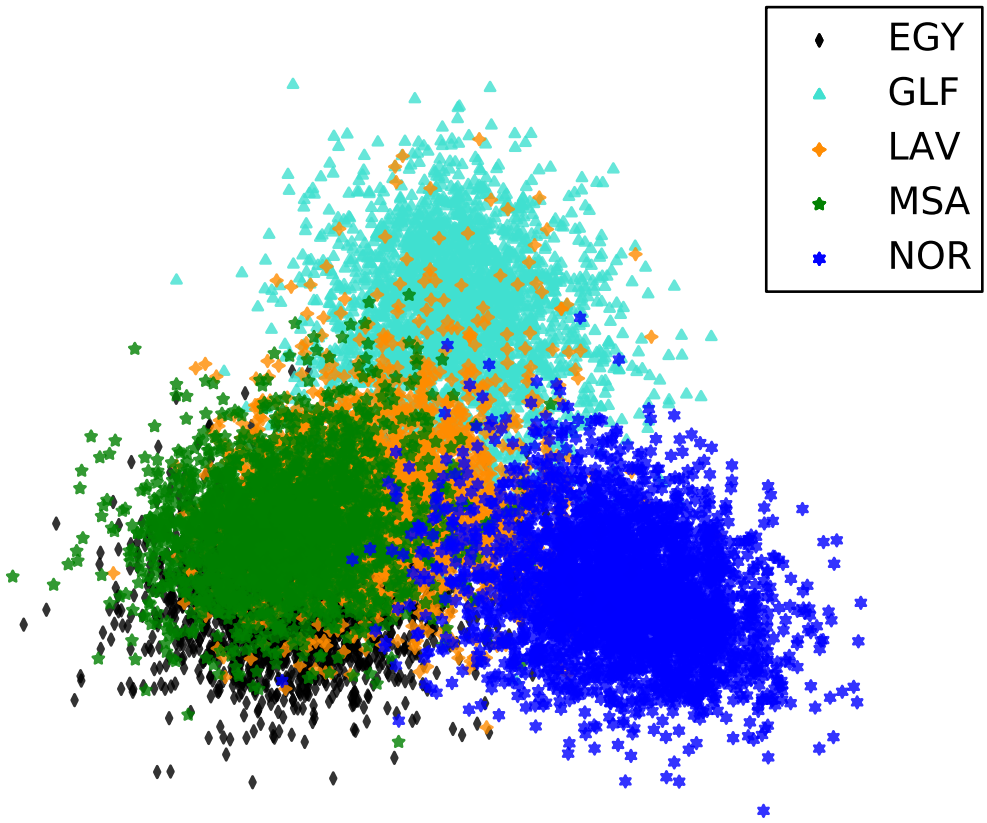


Figure 6.4: LDA projection for the BNF training data.

CNNs with lexical features

Inspired by the recent success of using CNNs for text classification [Kim, 2014, Zhang et al., 2015], we used lexical features for CNN-based classification. In this section, we explore the word sequence as well as the character sequence. The input word sequences were trimmed to a maximum of 100 words for the long sentences, and we padded shorter sentences with zeros. However, a maximum of 200 dimensions were used for the character-based CNN experiments. This was followed by an embedding layer of a dimension of 256. Followed by three convolutional layers in parallel to each other with the same number of filter: 512 each, and $ReLU$ activation function. The filters' sizes were different for each convolution layer: 3, 4 and 5, respectively. The three-convolutional layers were then merged into a single tensor. This was followed by a fully-connected hidden layer with a dropout of 0.2 to get the final representation, which is fed to softmax layer that outputs the final prediction. The architecture is same as the one used in [Oswal, 2016], which is shown in figure 6.5.

	Training			Development			Testing		
Dialect	**Char.**	**UNK.**	**Words**	**Char.**	**UNK.**	**Words**	**Char.**	**UNK.**	**Words**
EGY	420	701	76	60	117	11.0	63	127	11.6
GLF	308	584	56	66	85	11.9	69	115	12.3
LAV	290	550	53	56	96	10.3	59	122	10.9
MSA	398	**287**	69	77	**90**	13.4	75	**58**	13.0
NOR	207	583	38	55	106	9.9	57	102	10.3
Total	1623	2705	292	314	494	56.5	323	524	58.1

Table 6.6: The ADI data: Characters in 1000s, *unknown*, and words in 1000s.

Lexical Features	Context	Dictionary Size	Precision	Recall	Overall Accuracy
Words	bigram	230K	0.51	0.49	0.48
Characters	five-gram	294K	0.52	0.51	0.53

Table 6.7: Evaluating SVM with characters and word features.

We evaluated the CNN for words and characters as described before, and we obtained the results as shown in table 6.8. It is worth noting that both systems were trained with a maximum of 50 epochs, with an early stopping criterion to avoid overfitting. The character-based system stopped after 17 epochs and the word-based after 9 epochs. We can see from this result that the word-based CNN's overall accuracy outperforms the character-based CNN system and, furthermore, the word-based CNN is slightly better than the word-based SVM system. We found that the character-based SVM achieved the best overall accuracy in the lexical representation. In the overall ADI system, we will combine the words and the characters system for both the SVM and CNN. The combination will be done at the system-level, and not at the feature-level; this will be discussed more in section 6.5.

CNN Classification	Input Dimension	Precision	Recall	Overall Accuracy
Words	100	0.51	0.50	0.50
Characters	200	0.47	0.47	0.46

Table 6.8: Evaluating CNN for characters and word features.

6.4.3 Phonotactic methods

In this section, we study phoneme representation; the Arabic phoneme sequence and four non-Arabic phoneme sequence extracted using a phone recogniser based

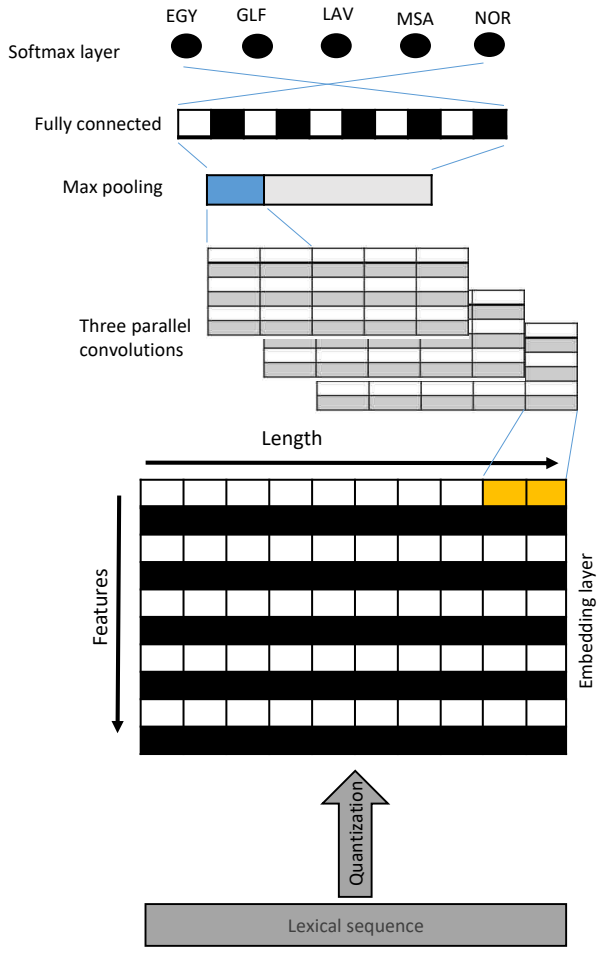


Figure 6.5: Lexical based CNN architecture used for dialect classification.

on long temporal context; namely English, Hungarian, Russian and Czech. The five phoneme recognitions systems have been compared using an SVM classifier. We used the training data for training the SVM and the development data for reporting results to be consistent with all the previous results.

Best phoneme sequence: We evaluated the five systems using an SVM. The hyper-parameters for the SVM were tuned separately for each system. Table 6.9 shows the results for the five phoneme recognisers. The Hungarian phoneme recognition achieved the best results, which is even better than the Arabic system. This can be due to the fact that the Arabic phoneme recognition system was built using only MSA data, and also the Hungarian phone recogniser based on long temporal context is more robust and is capable of extracting more accurate phonotactic patterns for the recognised dialects. This aligns with recent prac-

tice that Hungarian phone recogniser based on long temporal context [Matejka et al., 2005] has been widely used to discriminate between various languages and dialects, not including Hungarian [Li et al., 2013]. The Hungarian system will be used for the final system combination.

Phone Recognisers	Context	Dictionary Size	Precision	Recall	Overall Accuracy
Arabic	trigram	47K	0.44	0.45	0.45
Czech	bigram	2K	0.45	0.45	0.45
Hungarian	**trigram**	**48K**	**0.47**	**0.47**	**0.48**
Russian	trigram	49K	0.46	0.47	0.47
English	trigram	31K	0.33	0.33	0.34

Table 6.9: Evaluating five phoneme recognition features using SVM classifier.

We explored using the CNN classifier with a similar architecture to the lexical system and we deployed it on the best phoneme sequence, which is the Hungarian in this case. Unlike the lexical system, the input phoneme sequences were trimmed to a maximum of 300 for the long sentences, and we padded shorter sentences to 300. This is intuitive since the number of phonemes is expected to be much bigger than the number of words for the same utterance. We used early stopping as well, and we found no gain beyond 10 epochs. Table 6.10 shows the comparison between the SVM and CNN, and it is clear that the CNN is outperforming the SVM classification results on all measures. Therefore, we decided to use the Hungarian CNN classifier for the phoneme score in the overall ADI system.

Classifer	Precision	Recall	Overall Accuracy
CNN	0.51	0.51	0.50
SVM	0.47	0.47	0.48

Table 6.10: Benchmarking CNN and SVM using Hungarian phoneme sequence.

6.5 System combination

We fused the scores of the best system from the three vector representations; (*i*) acoustics, (*ii*) lexical, which comprised of two sub-systems: the word-based and character-based, and finally (*iii*) Phonotactic. Since we have limited data (10 hours only for training data), the final system used the training and development data for training and testing data to report results. The score from B_1 and B_2

were combined and the average score is used to represent the lexical features B. The scores from the three systems A, B and C we fused with weight of 0.7, 0.2, 0.1 respectively. The intuition here is that the acoustic based system is considerably outperforming the lexical and the phonetic system, so it would make sense to increase the weight for the acoustic system in the final contribution. Also, the weight for the lexical representation is higher than the Phonotactic. Combining the three systems improved the final results to be 0.73, 0.73, 0.73 for precision, recall and overall accuracy, respectively.

Features	Classifier	Precision	Recall	Overall Accuracy
Acoustics:				
bottleneck ivectors with LDA (4 dimensions)	SVM	0.62	0.63	0.62
(A) bottleneck ivectors (400 dimensions)	DNN	0.67	0.67	0.67
Lexical:				
Words	SVM	0.53	0.55	0.54
(B_1) Words	CNN	0.54	0.54	0.54
(B_2) Characters	SVM	0.59	0.58	0.59
Characters	CNN	0.56	0.56	0.55
Phonotactics:				
Hungarian phoneme sequence	SVM	0.52	0.52	0.53
(C) Hungarian phoneme sequence	CNN	0.53	0.53	0.53
overall system: $0.7 * A + 0.2 * B + 0.1 * C$		**0.73**	**0.73**	**0.73**

Table 6.11: Train on training + development and test on testing data.

Looking at the confusion matrix in figure 6.6, it can be inferred that Gulf is the most confused dialect, most often with LAV and MSA. The second most confused dialect is NOR, most often with LAV. It can be inferred that it is difficult to distinguish between the following three dialects: GLF, LAV and NOR.

We hypothesize that the reason our system performs worst in the case of Gulf and North African dialects is codeswitching [Elfardy and Diab, 2012, Auer, 2013, Elfardy et al., 2013, 2014, Solorio et al., 2014], where the same speaker alternates between two dialects in the context of a single conversation. This is similar to our findings in the crowd-sourcing task in figure 5.6. For more detail see section 5.5.2. We perform further investigation into the error patterns for utterances of different durations. Assuming that the speech is spoken in a single dialect, the ADI accuracy should increase as the duration of the speech utterances increases. GLF and NOR do not follow the aforementioned pattern as shown in figure 6.7, unlike other dialects. This leads us to believe that there is codeswitching in the

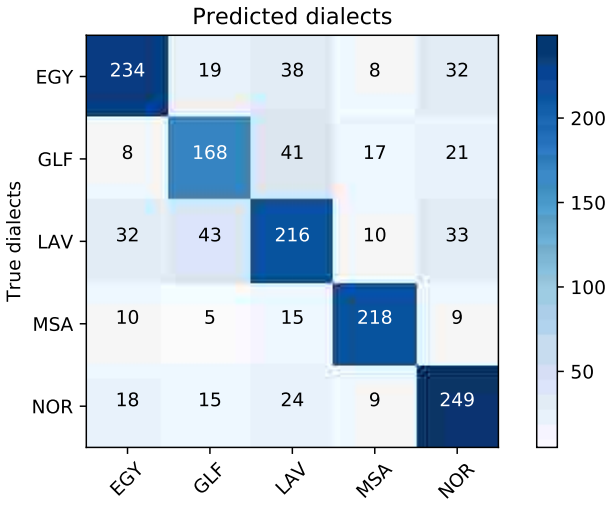


Figure 6.6: ADI Confusion matrix for the final combined system.

spoken utterances of these two dialects which make them the two most difficult dialects to recognise.

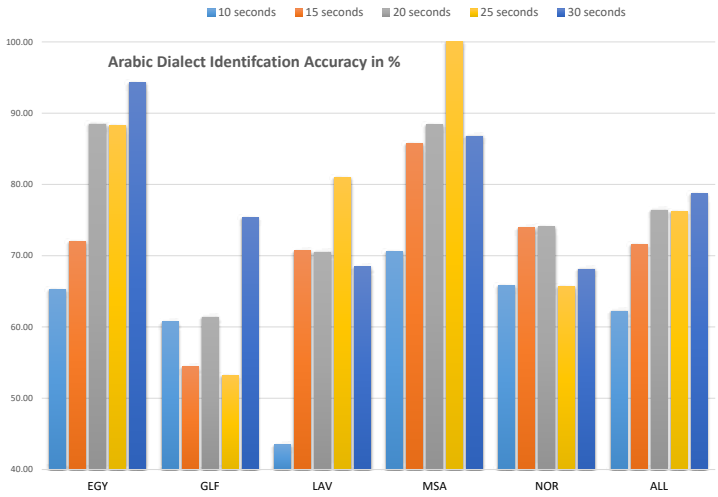


Figure 6.7: ADI accuracy over 10-30 seconds bins.

6.6 Conclusions

This chapter has presented our efforts in automatic dialect identification for Arabic broadcast speech. We have demonstrated a dialect classifier with an overall accuracy of 73% using system combination. We highlighted the codeswitching patterns between Arabic dialects, which can be considered as dialect diarization in spoken Arabic. Finally, the work in this chapter has created a baseline for the VarDial 2016 [Malmasi et al., 2016] and 2017 [Zampieri et al., 2017] and the MGB-3 [Ali et al., 2017b] challenges. This has led to more than 30 different submissions addressing challenges in dialectal Arabic in these three competitions.

Part III

Automatic Speech Recognition

This is the third section of the thesis and it has one chapter

Chapter 7 introduces our effort in building Arabic speech recognition, and creating an open research community to advance it. In this chapter, we have two main goals:

- Creating a framework for Arabic speech recognition that is publicly available for research. We present our efforts in building multi-genre broadcast challenges: MGB-2 and MGB-3.
- Building a state-of-the-art Arabic ASR system, and reporting WER for different techniques.

Chapter 7

Arabic Speech Recognition

This chapter concerns Arabic speech recognition. Various parts of this chapter were published in [Ali et al., 2014a,b, Khurana and Ali, 2016, Ali et al., 2016, 2017b]. Most of the results in this chapter have been reproduced to ensure rational comparison. We will also discuss some experiments that have not been published yet. Two sections in this chapter are not directly my contributions and have been borrowed from our previous submissions:

- Light alignment implementation in section 7.2.1.
- Recurrent neural network implementation for language model rescoring in section 7.2.3.

7.1 Introduction

This chapter describes our efforts in creating a public framework for an Arabic speech recognition challenge, a multi-genre broadcast (MGB) competition. The MGB challenge is a core evaluation of speech recognition, speaker diarization, lightly supervised alignment, and dialect identification using TV recordings from the BBC and Aljazeera, as well as YouTube videos.

My contribution in the MGB is designing the tasks, leading the investigation of the competition, developing the baseline, and creating the public framework. I have also developed a robust system for the MGB-2 (see section 7.2).

MGB-1 Challenge: First edition of the MGB challenge is the MGB-1 for the 2015 Automatic Speech Recognition and Understanding (ASRU-2015) [Bell et al.,

2015]. MGB-1 focuses on BBC English TV output across four channels. A total of 1,600 hours of broadcast audio is provided for acoustic modelling and several hundred million words of BBC subtitle text is shared for language modelling.

MGB-2 Challenge: The second edition of the MGB challenge is the MGB-2 for the 2016 Spoken Language Technology (SLT-2016) [Ali et al., 2016]. MGB-2 has an emphasis on handling the diversity in broadcast news domain in Arabic speech. Audio data comes from 19 distinct programmes from the Al Jazeera Arabic TV channel. A total of 1,200 hours have been released with lightly supervised transcriptions for the acoustic modelling. For language modelling, we made available over 130M words crawled from Al Jazeera Arabic website `Aljazeera.net`.

MGB-3 Challenge: The third edition of the MGB challenge is the MGB-3 for ASRU-2017 [Ali et al., 2017b]. MGB-3 focuses on dialectal Arabic (DA) using a multi-genre collection of Egyptian YouTube videos. Seven genres were used for the data collection. A total of 16 hours of videos, split evenly across the different genres, were divided into adaptation, development and evaluation data sets. The MGB-3 has three targets: a) dealing with languages which do not have well-defined orthographic systems, Egyptian Arabic in particular, b) Multi-genre scenarios; seven different genres are included in the challenge, and c) low-resource scenarios; only 16 hours of in-domain data was provided.

The rest of the chapter focuses on the MGB-2 challenge (see section 7.2) and the MGB-3 challenge (see section 7.3). We summarise the results of the MGB-3 in appendix A.

7.2 MGB-2 framework

The second round of the Multi-Genre Broadcast MGB [Bell et al., 2015] challenge is a controlled evaluation of Arabic speech to text transcription. The MGB-2 used a multi-dialect dataset, spanning more than 10 years of Arabic language broadcasts. The total amount of speech data crawled from Al Jazeera using the QCRI Advanced Transcription System (QATS) [Ali et al., 2014c] was about 3,000 hours of broadcast programmes, whose durations ranged from 3 to 45 minutes. For the purpose of this evaluation, we only used those programmes with transcription

on their Arabic website, Aljazeera.net. These textual transcriptions contained no timing information. The quality of the transcription varied significantly: the most challenging were conversational programmes in which overlapping speech and dialectal usage were more frequent.

7.2.1 MGB-2 Data

The Arabic MGB-2 Challenge used more than 1,200 hours of broadcast videos recorded during 2005–2015 from the Al Jazeera Arabic TV channel. These programmes were manually transcribed, but not in a verbatim fashion. In some cases, the transcript includes re-phrasing, the removal of repetition, or summarization of what was spoken, in cases such as overlapping speech. We found that the quality of the transcription varied significantly. The WER between the original transcribed text from Al Jazeera to the verbatim version is about 5% on the development set. It is worth noting that the WER in the MGB-2 is much lower than the English MGB-1 [Bell et al., 2015], owing to varying subtitle (closed captions) time-lags, and transcript reliability, owing to differences in the subtitle creation process (prerecorded (offline) or live (re-speaking)). Also, the text here did not include the filled pauses, indication of music and sound effects, or indications of the way the text has been pronounced.

Metadata challenges

We selected Al Jazeera programmes that were manually transcribed (albeit without timing information). A total of 19 programmes series were collected, which have been recorded over 10 years. Most, but not all, of the recorded programmes included the following metadata: programme name, episode title, presenter name, guests' names, speaker change information, date, and topic. The duration of an episode is typically 20–50 minutes, and the recorded programmes can be split into three broad categories: **conversation** (63%), where a presenter talks with more than one guest discussing current affairs, **interview** (19%), where a presenter speaks with one guest, and **report** (18%), such as news or documentary. Conversational speech, which includes the use of multiple dialects and overlapping talkers, is a challenging condition and is the typical scenario for political debates and talk show programmes.

Much of the recorded data used in MGB-2 was Modern Standard Arabic

(MSA): we estimate that more than 70% of the speech is MSA, with the rest in dialectal Arabic (DA). English and French language speech is also included, where typically the speech is translated and dubbed into Arabic. This is not marked in the transcribed text.

The original transcription has no clear metadata structure that would enable domain classification, so we decided to perform classification based on the keyword tags that were provided for the 3,000 episodes to define 12 domain classes, namely: politics, economy, society, culture, media, law, science, religion, education, sport, medicine, and military. Because some domains have a very small number of programmes, we merged them to the nearest domain to have a coarse-grained classification as shown in table 7.1, where the politics domain is the most frequent class.

Domain	Politics	Society	Economy	Media	Law	Science
Percentage	76%	9%	8%	3%	2%	2%

Table 7.1: MGB-2 coarse-grained domain distribution.

Data processing and light alignment

Removing programmes with damaged aligned transcriptions resulted in a total of about 1,200 hours of audio, which was released to the MGB-2 participants. All programmes were aligned using the QCRI Arabic LVCSR system [Ali et al., 2014b], which is grapheme-based with one unique grapheme sequence per word. It used LSTM acoustic models and trigram language models with a vocabulary size of about one million words. The same language model and decoding setup was used for all programmes. For each programme, the ASR system generated word-level timings with confidence scores for each word. This ASR output was aligned with the original transcription to generate small speech segments of duration 5–30 seconds suitable for building speech recognition systems.

As shown in [Braunschweiler et al., 2010] and based on the Smith–Waterman algorithm [Smith and Waterman, 1981], we identified matching sequences by performing local sequence alignment to determine similar regions between two strings. We addressed two challenges when aligning the data:

- The original transcription did not match the audio in some cases owing to edits to enhance clarity, paraphrasing, the removal of hesitations and disfluencies, and summarisation in cases such as overlapping speech.

- Poor ASR quality in cases such as noisy acoustic environments, dialectal speech, use of out-of-vocabulary words, and overlapped speech.

We applied two levels of matching to deal with these challenges: exact match (where the transcription and the ASR output are identical), and approximate match (where there is a forgiving edit distance between words in the transcription and the ASR output).

To evaluate the quality of the alignment between the ASR output and the transcription, we calculated the "anchor rate" for each segment as follows:

$$AnchorRate = \frac{\#MatchedWords}{\#TranscriptionWords} \tag{7.1}$$

The AnchorRate across all segments came with the following: 48% exact match, 15% approximate match, and 37% with no match. More details about the AnchorRate distribution is shown in figure 7.1.

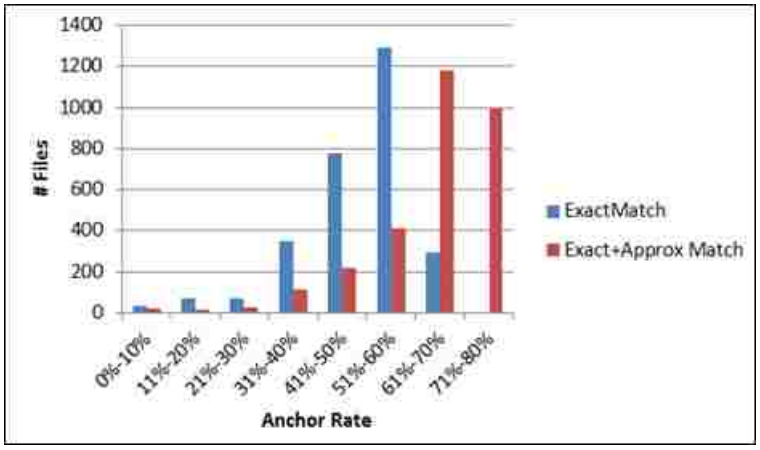


Figure 7.1: Anchor rate distribution for all programs.

To assign time for non-matching word sequences, we used linear interpolation to force-align the original text to the remaining speech segments.

After aligning the whole transcript, each audio file was acoustically segmented into speech utterances, with a minimum silence duration of 300 milliseconds. The metadata for aligned segments includes timing information obtained from the ASR, speaker name, and text obtained from the manual transcription. For each segment, the average word duration in seconds (AWD), the phoneme matching error rate (PMER), and word matching error rate (WMER) are stored in the given meta-data. Both PMER and WMER were calculated as traditional error rates but are described as matched error rates since there are not accurate transcripts

to be used as reference. For more details about PMER and WMER, see Bell et al. [2015].

Overall, more than 550,000 segments with a total duration of 1,200 hours were made available together with the aligned transcription and metadata. Figure 7.2 shows segment distribution according to the AWD value, and figure 7.3 shows segment distribution according to a cumulative AWD value. Figure 7.4 shows cumulative duration for the grapheme and the word matching error rate.

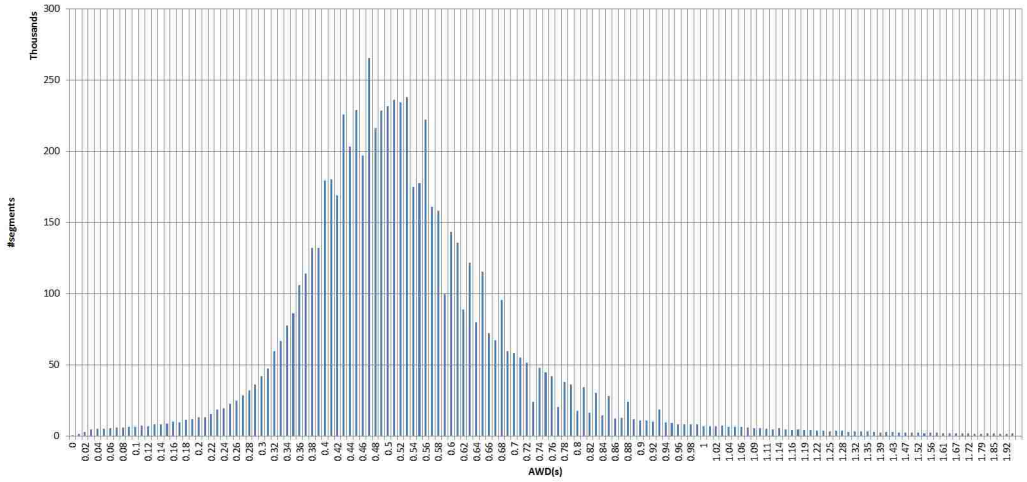


Figure 7.2: Distribution of average word duration (AWD).

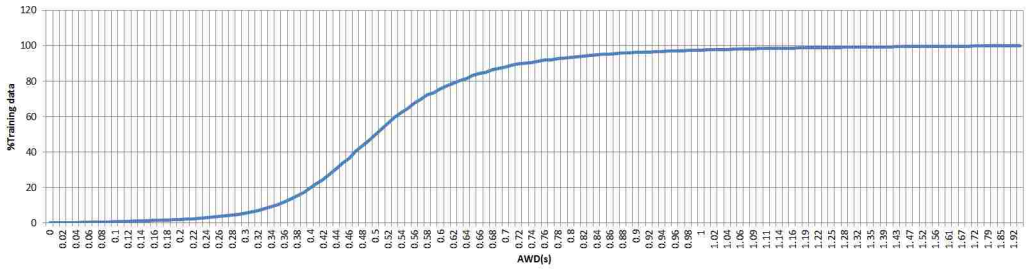


Figure 7.3: Cumulative average word duration (AWD).

During data preparation, we removed about 300 hours, mainly coming from very short audio clips with the corresponding full transcription. These audio clips are just the highlights of other programmes. No further filtering was applied to the 1,200 hours data.

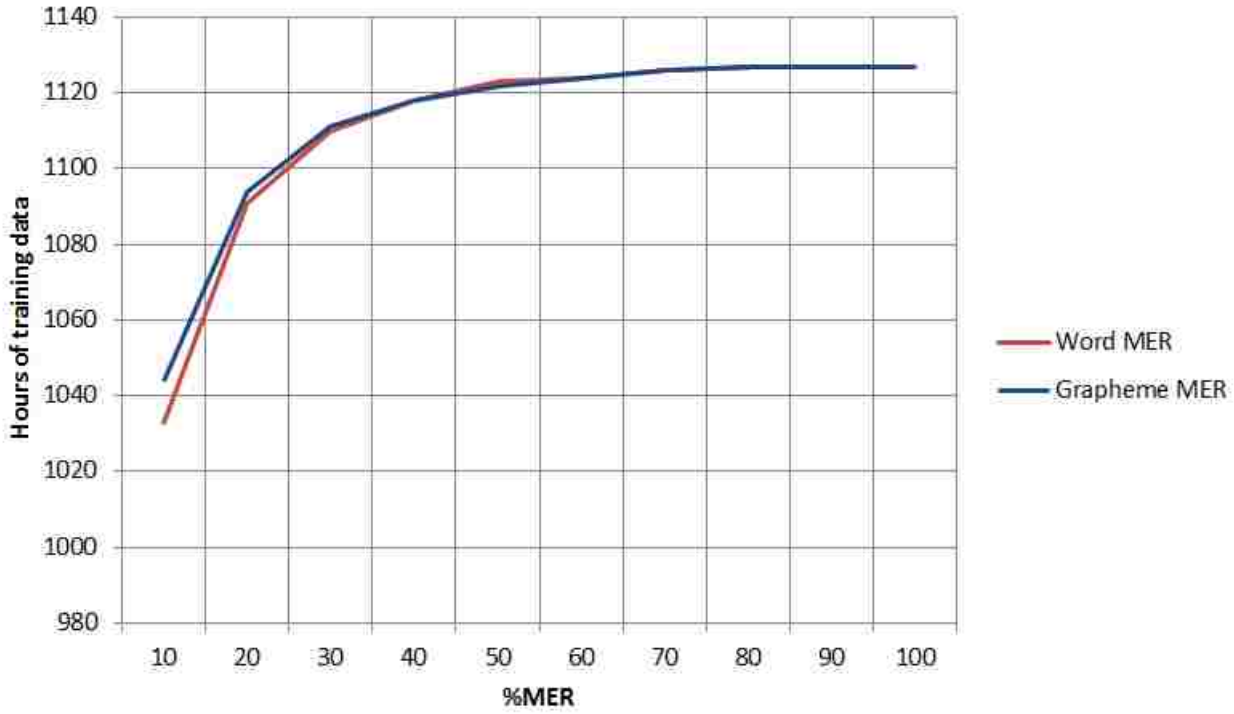


Figure 7.4: Cumulative duration for word (red line), and grapheme (blue line) matching error rate (MER).

Lexicon

Two lexicons were made available for participants in the challenge: a grapheme-based lexicon[1] with more than 900K entries with one unique grapheme sequence per word, and a phoneme lexicon[2] with more than 500K words with an average of four phoneme sequence per word using our previous vowelization to phonetization (V2P) pipeline [Ali et al., 2014b]. Participants could also choose any lexicon outside the provided resources.

Evaluation conditions

This speech transcription task operates on a collection of whole TV shows drawn from diverse Arabic dialectal programmes from Al Jazeera TV channel. Scoring required ASR output with word-level timings. Segments with overlapped speech were scored but not considered in the main ranking. Overlapped speech was defined to minimise the regions removed – at the segment level where possible. As the training data comes from only 19 series, some programmes from the same

[1]http://alt.qcri.org/resources/speech/dictionary/arar_grapheme_lexicon_20160209.bz2
[2]http://alt.qcri.org/resources/speech/dictionary/arar_lexicon_20140317.txt.bz2

series appeared in the training, the development and the evaluation data. Each show in the development and in the evaluation set was processed independently, so no speaker linking across shows was given. The data were carefully selected to cover different genres, and being diverse among the five dialects. Development and evaluation came from the last month of 2015 to avoid being seen in the training data. The duration of each file was between 20 minutes and 50 minutes with a total duration of 10 hours for each set. These files were transcribed verbatim, and manually segmented for speech/silence/overlapped-speech with segment lengths between three and ten seconds. Words with hesitation or correction were also marked by adding a special symbol at the end of these words.

Four WER scores were reported for the speech-to-text task:

- Scoring the original text as being produced by manual transcription, which might have punctuation/diacritization.
- Scoring after removing any punctuation or diacritization.
- Scoring using the Global Mapping File (GLM), which is the official result for the competition. This deals with various ways of writing numbers, and common words with no standard orthography.
- Scoring after normalising the Alef, Yaa, and Taa Marbouta characters in the Arabic text, i.e., assuming that these kinds of differences are not considered as errors because they can be easily corrected using a surface spelling correction component.

  The WER report in this thesis will follow the official result using the GLM.

7.2.2 MGB-2 baseline system

We provided an open-source baseline system for the challenge, via a GitHub repository[3]. The data was shared in XML format[4]. The baseline system included data pre-processing, data selection, acoustic modelling (AM), and language modelling (LM), as well as decoding. This allowed participants to focus on more advanced aspects of ASR and LM modelling. A Kaldi toolkit [Povey et al., 2011] recipe for the MGB2, and for language modelling the SRILM [Stolcke et al.] toolkit was used. The baseline system was trained on 250 hours sampled from the training data, which comes from 500 episodes. This system uses a standard MFCC multi-pass decoding:

[3]https://github.com/qcri/ArabicASRChallenge2016
[4]http://xmlstar.sourceforge.net/

- The first pass uses a GMM with 5,000 tied states, and 100K total Gaussians, trained on features transformed with FMLLR.
- The second pass is trained using a DNN with four hidden layers, and 1,024 neurons per layer, sequence trained with the MPE criterion.
- A tri-gram language model is trained on the normalised version of the sample data text (250 hours).

The baseline results were reported on 10 hours of verbatim transcribed development set: 34% (8.5 hours) for the non-overlap speech and 73% (1.5 hours) for the overlap speech.

7.2.3 MGB-2 ASR system

In this section, we describe our speech transcription system using the MGB-2 challenge data. Our system is a combination of five purely sequence-trained recognition systems that achieved the lowest WER of 13%[5]. The key features of our transcription system are the following: purely sequence trained acoustic models using the lattice free maximum mutual information (LF-MMI) modelling framework, language model rescoring using a four-gram and a recurrent neural network with MaxEnt connections (RNNME) for language modelling, and finally a system combination using minimum Bayes risk (MBR) decoding criterion. The whole system is built using the Kaldi speech recognition toolkit.

Training data

AM training data: Initially, we used about 250 hours of training data for the AM experiments. This is the same amount of data as for the baseline system. This is called MGB-2 sample data through the rest of the chapter. The MGB-2 sample data are those segments with word MER less than 80%, and limited to the first 500 programs. The full AM training data comes from all segments with MWER less than 80%, which summed up to more than 370K segmented across the 2,214 programs, creating more than 1,200 hours of speech segments. The development and the evaluation data are coming from diverse 10 hours each that have not been used in the training data. The program title itself may have been seen, but not these particular episodes. Table 7.2 shows more details about the

[5]The experiments here were done after the MGB-2 evaluation deadline. The MGB-2 test data is the data from the MGB-2 evaluation which is public.

AM data.

Type	Duration	Programs	Segments
Sample Data	250h	500	83
Training	1200h	2214	370
Development	10h	17	5.8
Evaluation	10h	17	5.6

Table 7.2: *Data used for acoustic model training, development and evaluation duration in hours and segments in 1000s.*

LM training data: We used the provided Buckwalter format for the transcription as well as the 130M words crawled from Al Jazeera. The data did not have any punctuation or dicraization, and we did not use any text normalisation like normalising Alef, Yaa, and Taa marbouta in the given text. Table 7.3 shows more details about the LM data.

Type	Tokens	Vocabulary
Transcription Text	8M	200k
Background Text	130M	1.3M

Table 7.3: *Transcription text refers to the training transcripts and Background text refers to the extra Arabic language modelling text provided for the challenge.*

Acoustic modelling experiments

Grapheme versus phoneme ASR system: In an attempt to evaluate both the grapheme and the phoneme approaches for the Arabic ASR, we built time-delay neural network acoustic-models in both systems. The phoneme system used the phonetic lexicon shared by QCRI, while the grapheme lexicon used the same word list with 1:1 word-to-character mapping, which means that the vocabulary size is the same as the lexicon size. Table 7.4 shows more details about both systems. We can see here that the phoneme-based system outperforms the grapheme-based system with about 0.5% absolute reduction in WER on both the dev and the test sets. However, this comes with a price of almost four times the size of the lexicon.

The out-of-vocabulary for this lexicon on the dev set is 3.3% and 1.9% on the test set, which is relatively high.

Given that Arabic is a phonologically complex language [Pasha et al., 2014], increasing the lexicon size considerably to reduce the out-of-vocabulary (OOV) is needed. Therefore, the final ASR systems used the grapheme approach. The final lexicon was constructed using the word list in the shared phoneme and grapheme lexicon in addition to the most frequent words in background text and the *Arabic giga-word corpus*[6]; we considered any word that occurred more than twice in the lexicon. The final lexicon size was 1.3M words. Our acoustic units will represent the character in the surface form of the words instead of phone units.

	Phoneme	Grapheme
Word-to-pronunciation	1:3.8	1:1
Lexicon size	2M	520K
Dev data WER	22.9%	23.5%
Test data WER	22.6%	23.1%

Table 7.4: *Comparison between grapheme and phoneme systems trained on sample MGB-2 data.*

Acoustic modelling training setup

Basic recipe for acoustic modelling: The AM development was started by first training a monophone Gaussian Mixture Model (GMM) from the 10,000 shortest utterances in the corpus with the highest confidence in the transcription (WMER= 0). This was followed by three tri-phone models that were built in succession; first a regular GMM model, then a GMM model on top of features transformed with linear discriminant analysis (LDA) and lastly a speaker adaptive trained (SAT) GMM model [Anastasakos et al., 1996]. The tri-phone models were trained using all the 1200 hours in table 7.2. This SAT model was used to generate state-frame-alignment for the neural network acoustic modelling. The training of the neural network acoustic models included volume perturbation and three-way speed perturbation of the training data [Ko et al., 2015] and the training of an LDA-based i-vector extractor.

[6]https://catalog.ldc.upenn.edu/ldc2006t02

The i-vectors were extracted for two utterances at a time at most to provide training variability. Mainly, as the shared data did not include speaker information, the i-vectors were extracted per utterance in decoding. From the final GMM model, alignment lattices (which contain multiple alignments per utterance) were generated. Together with the i-vectors, the alignments and high-dimensional MFCC features were joined in a neural network training examples with the amount of context applicable for the used network. The acoustics feature vector is a concatenation of 40 dimensional high-resolution MFCC features (MFCC hires) and 100 dimensional i-vectors [Dehak et al., 2011a] for each frame.

Recurrent neural networks acoustic models have shown tremendous improvements in recognition performance by reducing the WER significantly. Being inspired by the recent progress in RNN and TDNN [Povey et al., 2014, Peddinti et al., 2015, Povey et al., 2016], we explored the following five neural network architectures:

- Unidirectional long short term memory (LSTM)
- Bidirectional LSTM (BLSTM)
- Time-delay neural network (TDNN)
- TDNN layers along with LSTM layers (TDNN-LSTM)
- TDNN layers followed by BLSTM layers (TDNN-BLSTM)

For the objective function, we focus on the lattice-free MMI (LF-MMI) [Povey et al., 2016] models because they are several times faster to train and yield better performance than standard cross-entropy (CE)-trained systems. For more discussion about LF-MMI, see section 3.2.1.

LSTM AM: Similar to [Sak et al., 2014, 2015], the AM is trained using concatenated 40 dimensional high-resolution MFCC features (MFCC hires) and 100 dimensional i-vectors for each frame. While the cross entropy model would have used 3 LSTM layers with a delay of -1,-2 and -3 at each layer, in our adopted purely sequence trained model the delay at each layer is chosen to be -3. We use the LSTM architecture with recurrent and non-recurrent projection layers. An output label delay of 5 is also used. A major component in an LSTM model is the memory block, that consists of the following:

- input (i ●) the input gate controls the flow of input activations into the memory cell.

- output (o ●) the output gate controls the output flow of cell activations into the rest of the network.
- forget (f ●) the forget gate prevents the LSTM from processing continuous input streams that are not segmented into subsequences.

In addition, the modern LSTM architecture contains peephole connections from its internal cells to the gates in the same cell to learn precise timing of the outputs. The architecture of the LSTM memory is similar to the originally proposed in [Sak et al., 2014]. Figure 7.5 shows the hyperparametrs of the LSTM memory block.

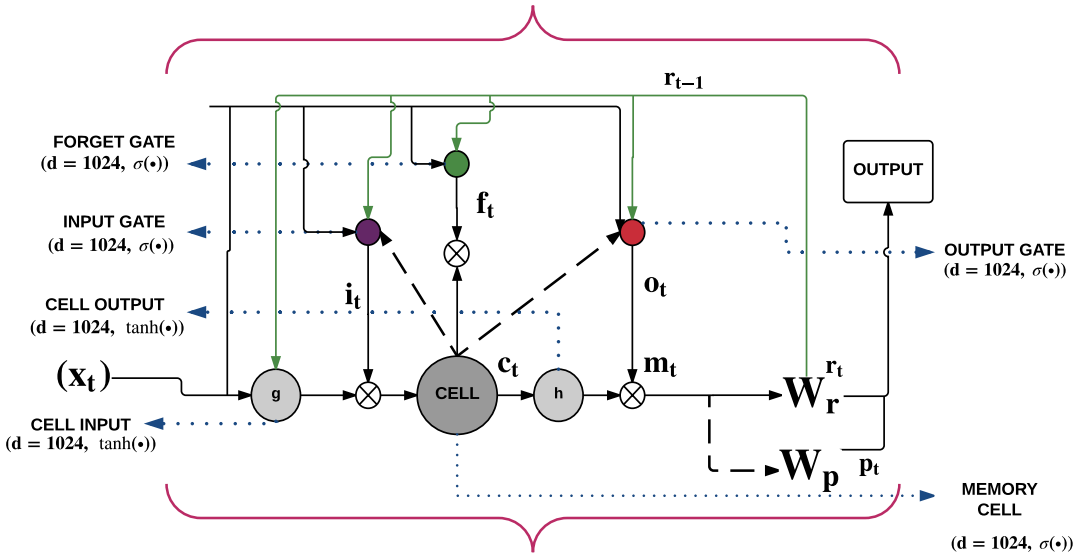


Figure 7.5: LSTM architecture used in acoustic modelling.
$\mathbf{W_r}$ and $\mathbf{W_p}$ refer to the recurrent and the non-recurrent projection spaces respectively, which are of dimensions 256. The dimension of the memory cell is 1024 and the activation function is showed next to the dotted lines.

BLSTM AM: The acoustic modelling architecture in BLSTM is similar to LSTM, except that the training occured in both directions; left-to-right and right-to-left. For more discussion about LSTM and BLSTM AM, see section 3.2.1.

TDNN AM: The time-delayed neural network acoustic is trained using the same concatenated 40-dimensional high-resolution MFCCs with 100-dimensional i-vectors for each speech frame. TDNNs require less training time than sequence

models such as LSTMs, while attempting to capture the long-term temporal dependencies just like a sequence model. We use the same TDNN architecture as given in Peddinti et al. [2015], except with a different configuration of splicing indexes. The splicing indexes used are -1,0,1 -1,0,1, -3,0,3 -3,0,3 -6,-3,0. The initial -1,0,1 means that the first layer sees 3 consecutive frames of input; the -3,0,3 means that most hidden layers see 3 frames of the previous layer, separated by 3 frames. Since these differ by multiples of 3 and we only evaluate the output at multiples of 3 frames, most hidden layers only need to be evaluated every 3 frames, like the output, which is efficient. All the TDNN layers have fixed dimensions of 450 for each layer. For more discussion about TDNN and splicing, see section 3.2.1.

TDNN-LSTM AM: The TDNN-LSTM had three recurrent layers interleaved with the six TDNN layers as follow: TDNN_2-LSTM_1-TDNN_3 TDNN_4-LSTM_2-TDNN_5 TDNN_6-LSTM_3. The three LSTM layers have 128 recurrent and 128 non-recurrent with 512 dimensions for each layer in the TDNN and the LSTM models, respectively.

TDNN-BLSTM AM: The TDNN-BLSTM had three forward and three backward layers following three TDNN layers. The three forward and three backward layers have 256 recurrent and 256 non-recurrent, with a 1024 dimensions for each layer in TDNN and B-LSTM.

It is worth noting that the number of layers as well as the hyper-parameters for the neural acoustic model were not fully tuned for the Arabic MGB-2 data. However, most were borrowed from similar tasks in the Kaldi recipes [Povey et al., 2011].

Table 7.5 shows the AM results for various model architectures. It is clear from the results that TDNN-LSTM and TDNN-BLSTM are giving the best results across the five models. At this stage, we decided to use the five acoustic models for LM model rescoring, and further system combination.

LM modelling experiments

***N*-gram language model:** We train two n-gram language models (LMs); big-four-gram LM (bLM4), which is trained using the spoken transcripts and the

Data	Dev	Test
TDNN	22.3%	23.7%
LSTM	20.0%	21.1%
BLSTM	19.7%	20.8%
TDNN-LSTM	19.4%	20.3%
TDNN-BLSTM	18.2%	19.1%

Table 7.5: AM results for different acoustic models architecture.

background text as shown in table 7.3. This language model was pruned to small-four-gram LM (sLM4). The LM was built and pruned using pocolm [7]. The small LM is used for first-pass acoustic decoding to generate lattices. These lattices are then rescored using bLM4. The pruned LM technique is similar to one used in [Stolcke et al.], with the difference in taking into account the change in the likelihood of the backed-off-to-state. The sLM4 was pruned with a limit of about 2 million n-grams target. The SRILM baseline LM were trigram trained with the SRILM toolkit, roughly the same size, but pruned differently. Table 7.6 shows some analysis for the n-gram LM.

LM	sLM4	bLM4
Total n-gram	2.1M	10.4M
Size on disk	13M	85M
Dev set OOV	0.78%	
Dev set Perplexity	1417	1121
Test set OOV	0.64%	
Test set Perplexity	1389	1089

Table 7.6: *N*-gram LM analysis.

Recurrent network language model: We trained a recurrent neural network language model with maxEnt connections (RNNME) using RNNLM-Toolkit [Mikolov et al., 2011b]. RNNLM-Toolkit is arguably the first toolkit publicly released to construct RNN language models. As the training procedure in this toolkit is CPU-based, it takes a considerable amount of time to train an LM, and hence we go straight to building an RNNME LM. This has been shown to

[7] https://github.com/danpovey/pocolm

perform better than RNN LM without direct connections (MaxEnt) between the input and the ouput layer [Mikolov et al., 2011b].

RNNME refers to an RNN architecture, which along with recurrent connections, also has non-recurrent or direct connections between the input and the output layer. These direct connections are known as MaxEnt connections, which derives its name from maximum entropy language model. This kind of RNN architecture provides a way to jointly train an n-gram LM and an RNN LM. RNNME has been shown to perform better than the conventional RNN LM. In this work, we train a class-based RNNME LM, with hyperparameter settings as follows; **class dimensions:** 200, **input-layer-size:** 40k, which is also the language model vocabulary, which is restricted to the top 40k most frequent words, **hidden-dimension:** 300, **hidden-activation function:** sigmoid, **direct-connections:** 2000M, which are the number of weights used for direct connections between the input and the output layer, ***n*-gram order:** 3, which is referred to as the direct-order in the RNNLM toolkit. Fig 7.6 shows the RNNME architecture along with the hyperparameter settings used.

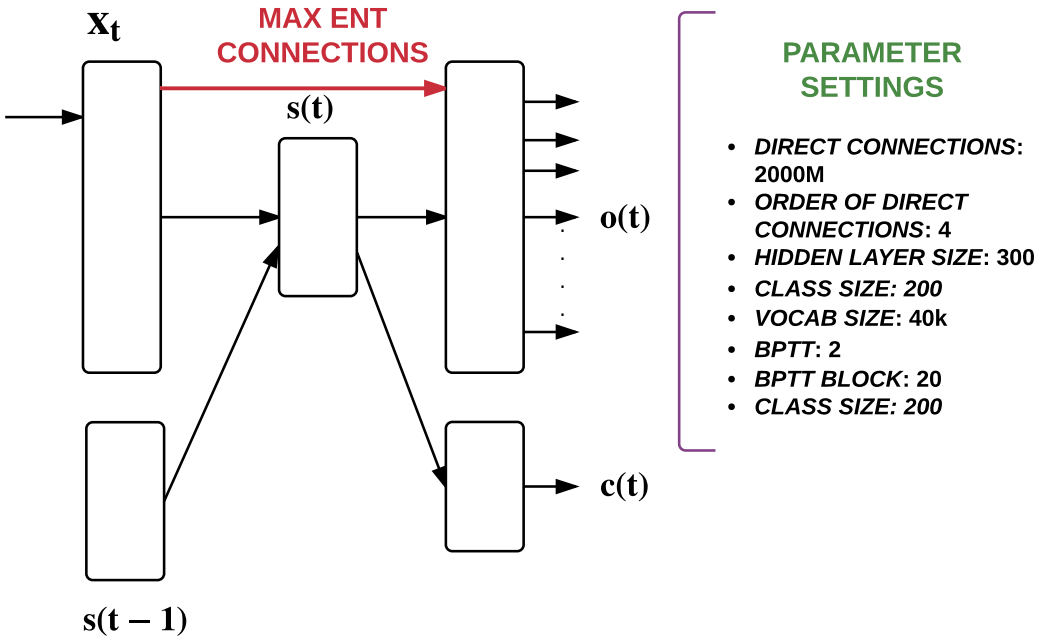


Figure 7.6: RNNME LM architecture and the hyperparameter settings.

Overall experiments and results

GMM-HMM baseline system: We train a GMM-HMM recognition system that provides frame vs HMM-state alignments that are used to train the neural network acoustic models. The GMM-HMM system is built using whitened (Mean Normalised) spliced MFCC features that are transformed using LDA and MLLT, followed by speaker adaptive training (SAT) [Matsoukas et al., 1997]. We used Kaldi to build the baseline system, which is explained in [Povey et al., 2011]. The %WER, using the baseline GMM-HMM system, is 40.2 on the dev set. No further filtering was applied on the 1,200 hours. Our intuition here was to avoid filtering as the dialectal data will contain challenging utterances that may be seen as poor transcription quality. We decided to keep all data for training the neural acoustic modelling.

Data augmentation: We use the audio augmentation technique proposed in [Ko et al., 2015]. We perform audio speed perturbation with speed factors of $0.9, 1.0, 1.1$. This gives us three times the original speech utterances. The speed-perturbed data is followed by volume perturbation with volume factors that are uniformly sampled from the interval $[\frac{1}{8}, 2.0]$. The same data augmentation approach was also used by [Peddinti et al., 2015].

Decoding: Table 7.5 gives the recognition performance on the dev and on the test set using the five acoustic modelling recognition systems: the decoding results used the small-four-gram LM (sLM4), and no lattice-rescoring was applied until this stage. The best results as expected are coming from TDNN-LSTM and TDNN-BLSTM models.

Four-gram LM rescoring: The decoding lattices obtained from LF-MMI trained recognition systems from the previous step are rescored using the big-four-gram LM (bLM4), which is built using the same data with less pruning. Table 7.6 shows details for both LM. The language model rescoring assigned a new graph score to each alternated hypothesis path in the lattice by scoring it using the bLM4. Table 7.7 shows improvements in recognition results due to n-gram LM rescoring.

RNN LM rescoring: We rescore the bLM4 rescored lattices obtained from the

Data	Dev	Test
TDNN	21.1%	22.2%
LSTM	19.1%	20.0%
BLSTM	19.0%	20.1%
TDNN-LSTM	18.6%	19.7%
TDNN-BLSTM	17.7%	18.5%

Table 7.7: AM results after *n*-gram bLM4 LM lattice rescoring. The baseline without rescoring is shown in table 7.5.

previous step, using an RNNME LM. Full lattice rescoring is inefficient using RNN LMs, and hence we extract the N-best hypotheses for each utterance and rescore the N-best list. In our case, N is 1000. We found out that the interpolation of the scores that RNNME LM assigns to the hypotheses with the score assigned by the bLM4 language model gives us the best recognition performance. The interpolation parameters are 0.3 and 0.7 for the bLM4 LM score and RNNME LM score respectively. These parameters are optimized on the dev set. Thus, Table 7.8 shows the results of N-best rescoring for the test set only. Clear improvements in the recognition results can be seen after performing N-best list rescoring.

Model	%WER(bLM4)	%WER(bLM4 +RNNME)
TDNN	22.2	21.4%
LSTM	20.0	19.3%
BLSTM	20.1	19.3%
TDNN-LSTM	19.7	19.0%
TDNN-BLSTM	18.5	17.9%

Table 7.8: *Recognition results on the test set after performing interpolated bLM4 and RNNME LM rescoring. Interpolation parameters are 0.3 for bLM4 and 0.7 for the RNNME.*

Overall speech recognition system

The overall Arabic speech recognition system, which is illustrated in figure 7.7, is the combination of the five LF-MMI trained recognition systems that are rescored

using four-gram and RNNME language models, i.e., we combine the five recognition systems mentioned in Table 7.8.

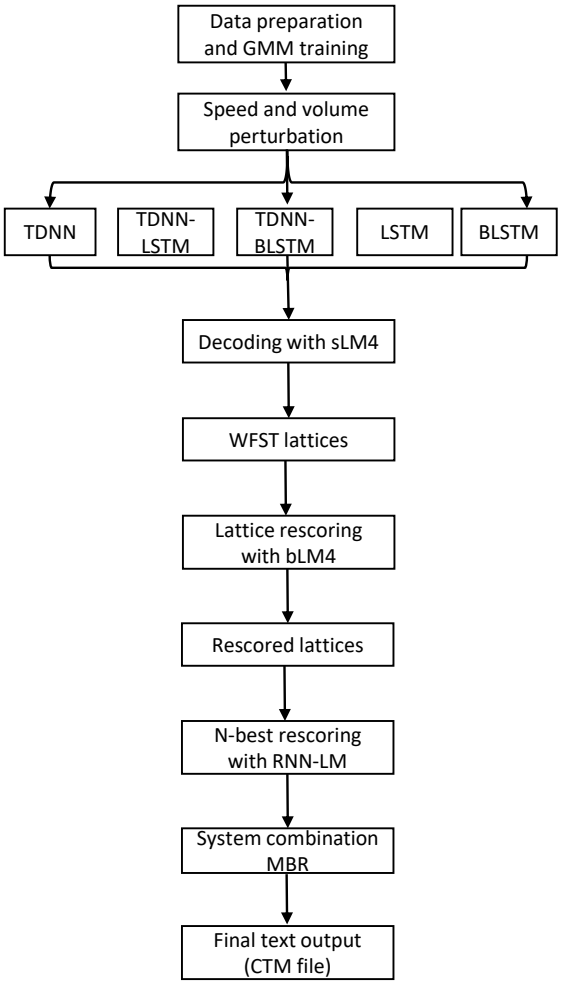


Figure 7.7: *System Description of the final Arabic MGB-2 speech recognition system.*

The three sets of output lattices are combined to form a union lattice, which is then used as an input to the minimum Bayes risk (MBR) decoding pipeline to get the final recognition output on the evaluation and on the development set. The best WER achieved on the full 5,655 sentences test set is 17%. Moreover, for our results to be comparable to the MGB-2 reported results, we excluded sentences with overlap speech, so the official test set comprised 5,002 non-overlapping speech segments. **Our official WER is 13%, which has the lowest WER reported on this task**. More detail about other submissions for this task can be found in [Ali et al., 2016, 2017b].

7.3 MGB-3 framework

Similar to the MGB-2, the MGB-3 Arabic Challenge was a controlled evaluation of speech-to-text transcription and dialect identification, focused on Egyptian dialect speech obtained from YouTube.

While the previous MGB challenges used mainstream broadcast media (BBC in MGB-1, Al Jazeera in MGB-2), the Arabic MGB-3 challenge used YouTube recordings for two reasons: it is a platform that enables dialectal recordings to be harvested easily, and it also allows the collection of videos across different genres. Thus, the MGB-3 Arabic Challenge extends the diversity of the data compared to previous MGB Challenges. This results in a relatively high baseline word error rate (WER) for MGB-3 Arabic, compared to MGB-2 Arabic. We, thus, targeted the following aspects in MGB-3 Arabic:

- Dealing with languages which do not have well-defined orthographic rules; Egyptian Arabic in particular.
- Multi-genre scenarios: seven different genres are included in MGB-3 Arabic.
- Low-resource scenarios: only 16 hours of in-domain data was provided, split into adaptation, development, and testing data.

The MGB-3 Arabic data comprised 16 hours of Egyptian Arabic speech extracted from 80 YouTube videos distributed across seven genres: comedy, cooking, family/kids, fashion, drama, sports, and science talks (TEDx [8]).

We assume that the MGB-3 data is not enough by itself to build robust speech recognition systems, but could be useful for adaptation, and for hyper-parameter tuning of models built using the MGB-2 data. Therefore, we reused the MGB-2 training data in this challenge, and we considered the provided in-domain data as (supervised) adaptation data.

7.3.1 MGB-3 data

To build the MGB-3 corpus, YouTube clips from various Egyptian channels were selected. The various genres are shown in Table 7.9. Across the seven different genres, a total of 80 videos were selected.

From each video, the first 12 minutes were selected. Manually-identified non-speech segments were removed. The resulting clips were then distributed into

[8] TEDx Talks includes prepared talks of up to 18 minutes duration; the chosen TEDx talks are in Egyptian dialect: `https://www.youtube.com/user/TEDxTalks`

	Adapt	Dev	Test
Comedy	0.6/3	0.6/3	1.0/5
Cooking	0.6/3	0.8/4	0.6/3
FamilyKids	0.8/4	0.6/3	1.0/5
Fashion	0.6/3	0.6/3	0.8/4
Drama	0.6/3	0.8/4	0.8/4
Science	0.6/3	0.8/4	1.0/5
Sports	0.8/4	0.6/3	0.8/4
Total overlap speech segments*	0.6	0.3	0.5
Total non-overlap speech segments*	4.0	4.1	5.3
Overall data	**4.6/23**	**4.8/24**	**6.0/30**

Table 7.9: MGB3 data distribution across the three classes, duration in hours/number of programs (12 minutes each). * is duration in hours across all speech segments.

adaptation, development, and testing groups, with the test set being a little larger than the other two sets. Details can be seen in Table 7.9. The table also summarizes how much of the overall data contains overlapping speech (more than one speaker talking simultaneously) and how much data contains non-overlapping speech.

It can be argued that Egyptian Arabic is a language with no orthographic rules [Ali et al., 2017a]. Given that dialectal Arabic does not have a clearly defined orthography, different people tend to write the same word in slightly different forms. Therefore, instead of developing strict guidelines to ensure a standardised orthography, we allow for variations in spelling. Thus, we decided to have multiple transcriptions, allowing transcribers to write the transcripts as they deemed correct. This can be addressed in evaluation by using a multi-reference WER estimation (MR-WER) [Ali et al., 2015]. The idea behind MR-WER is to align the ASR results with multiple transcriptions, which is similar to the multi-reference BLEU score [Papineni et al., 2002] used to evaluate Machine Translation (MT). For more discussion about MR-WER, refer to chapter 8.

Table 7.10 shows the inter-annotator disagreement on the development data. This table shows two numbers: the raw Word Error Rate (WER) and the WER after applying surface normalisation[9]. This indicates that there is about 13%

[9]Surface orthographic normalisation for three characters; alef, yah and hah, which are often mistakenly written in dialectal text. This normalisation is standard for dialectal Arabic pre-

	ref2	ref3	ref4
ref1	23/17	17/14	15/11
ref2	–	19/15	20/16
ref3	–	–	8/7

Table 7.10: Word-level inter annotator disagreement on the development data across the four different human references before/after normalisation (in %).

disagreement between the annotators for the MGB-3 data. We will report results for the MGB-3 data after normalisation only.

7.3.2 MGB-3 baseline

The ASR baseline system was trained using the full MGB-2 data; 1,200 hours of audio. This data was augmented by applying speed and volume perturbation [Ko et al., 2015], increasing the number of training frames by a factor of 3. The code recipe is available on the Kaldi repository[10]. The acoustic modelling is similar to the QCRI submission to the MGB-2 Challenge [Khurana and Ali, 2016]. The lexicon was grapheme-based, covering 950,000 words collected from a set of shared lexicons, as well as the training data text. The systems used a single-pass decoding with a trigram LM, along with a purely sequence trained TDNN acoustic model [Povey et al., 2016]; i-vectors were used for speaker adaptation. We report results for the MGB-2 development set (5,002 non-overlapping speech segments) on which we achieve a WER of 22.6% without LM rescoring. This is a fairly reasonable MGB-2 baseline. We also report results for the MGB-3 development set explained in table 7.9 using the MGB-2 baseline system, without adaptation to Egyptian Arabic using the MGB-3 data.
Table 7.11 shows the results for all 1,279 non-overlapping speech segments across the four annotators. We can observe that the MGB-3 baseline WER is high, which is to be expected as the system was not adapted to the changed characteristics of the MGB-3 data.

processing and reduces the sparseness in the text.
[10] https://github.com/kaldi-asr/kaldi/tree/master/egs/gale_arabic/s5b

	WER1	WER2	WER3	WER4	AV-WER	MR-WER
Comedy	59.5	59	58.8	60.4	59.4	53.6
Cooking	72.0	71.3	71.5	71.2	71.5	67.5
FamilyKids	50.2	48.4	48.3	48.3	48.8	43.5
Fashion	82.2	81.4	82.0	81.2	81.7	78.0
Drama	68.8	68.5	68.8	68.2	68.6	64.5
Science	59.3	57.7	59.5	57.2	58.4	51.4
Sports	54.6	54.9	55.0	54.4	54.7	49.4
Overall WER	**63.8**	**62.9**	**63.3**	**62.8**	**63.20**	**58.0**

Table 7.11: Baseline results in % for the development data after applying surface text normalisation. WERs are given for each of the four references (produced by different transcribers), as well as average WER (AV-WER) and multi-reference WER (MR-WER) across the four references.

7.3.3 MGB-3 submissions and results

We assume that the MGB-3 data is not enough by itself to build robust dialect-dependent speech recognition systems, but could be useful for adaptation, and for hyper-parameter tuning of models built using the MGB-2 data. Therefore, we reused the MGB-2 training data in this challenge, and considered the provided in-domain data as (supervised) adaptation data.

Our focus in the MGB-3 was mainly to build the framework, design the tasks and to take a particular care of the evaluation, i.e., introduce new evaluation metric MR-WER (more details in chapter 8). Therefore, we did not make a submission ourselves. Appendix A highlights some of the major features in the submitted systems.

7.4 Conclusions

This chapter presented our efforts in building public framework for researchers to advance the state of the art in Arabic speech recognition. We focused on two areas: broadcast news transcription and dialectal multi-genre broadcast. Our experiments have shown promising results in building robust acoustic model using deep neural network with more than thousands hours with no-need for verbatim transcription. For the language model, we explored the standard n-gram approach as well as recurrent neural language modelling to train models with more than 100 million tokens and 1.3 million-words vocabulary. We achieved the best results using system combination. Our code and data are publicly available for the research community to build on it and advance the shared results.

Part IV

Speech Recognition Evaluation

This is the fourth section of the thesis and it addresses evaluating speech recognition, with special focus on dialectal speech

Chapter 8 introduces a novel approach of using multiple references to deal with the non-orthographic rules in dialects to report more appropriate evaluation metric for dialectal speech recognition.

Chapter 9 builds on chapter 8 and introduces a new method which relies on a single reference and learn from social media particularly tweets, multiple lexical variations.

Chapter 10 addresses how to estimate the WER with no need for reference transcription and introduces robust quality estimation for the LVCSR system.

Chapter 8

Multi Reference Word Error Rate: MR-WER

This chapter is based on [Ali et al., 2015] published at ASRU 2015 and concerns evaluating dialectal speech recognition using multiple-references.

8.1 Introduction

WER has continued to be the most commonly used metric for evaluating ASR. The metric simply relies on comparing the recognised text to a reference of a manual transcription to the speech signal. This approach has always been seen as sufficient for an effective evaluation of ASR, since transcription of the speech signal is deterministic and one manual transcription should be a sufficient reference. However, in recent years, some interest has been directed towards ASR for dialects and rural languages. Some of these languages suffer from the absence of unified orthographic rules; non standard orthographic languages (NSOL). DA is an example for NSOL. Although DA is not a rural language, its variants are spoken by 350 million people, and there is no unique writing system for it as explained in chapter 2. This creates a challenge for evaluating an ASR output, since one reference transcription may cover only a few of many valid forms of the spoken words.

Unlike English, where *enough* is a correct word and *enuf* is an incorrect spelling, NSOL can have many valid written forms for the same word. Table 2.4 has an example of spoken Egyptian sentence transcribed in Arabic and Buckwalter, the table highlights the variations when writing a NSOL.

In this chapter, we propose an evaluation methodology for ASR, which accepts the presence of multiple transcription references. The methodology is inspired by the evaluation of machine translation (MT) systems, where multiple translation references could be used. Similarly for some languages, multiple spellings and forms could be accepted as a transcription for a word or a phrase. We introduce *multi-reference* WER (MR-WER), which is a modified version of WER that uses multiple reference transcriptions. We describe the process of aligning the multiple references (that can be of different lengths), and we show how MR-WER is calculated. We examine our new metric over two different datasets of DA, namely, Egyptian and North African Arabic, that both have no standardised orthography. For each dialect, we collected a set of five different transcriptions using a crowdsourcing platform, and we compared the performance of WER to MR-WER for these dialects. We provide our scripts and code for calculating MR-WER for the research community for usage and potential future contributions [1].

8.2 ASR for NSOL

Several studies have investigated applying ASR to under-resourced languages. Under-resourced languages are those lacking the basic components to have a decent ASR system, such as sufficient labelled speech data for training, a lexicon, and a natural language processing (NLP) pipeline for phonetic systems. Moreover, they can be NSOL. DA is considered one of the largest under-resourced languages that is highly used by millions of people in daily conversation and in social media, while lacking most of the required resources for creating an effective ASR. More details about DA can be found in chapter 2. For more discussion about resources in DA, refer to section 5.2.

In a study by Habash et al. [2012], they presented conventional orthography for dialectal Arabic (CODA), explaining the design principles of CODA, and using the Egyptian dialect as an example, which has been presented mainly for the purpose of developing DA computational models. Similar work by Ali et al. [2014a] studied the best practices for writing Egyptian orthography. They released guidelines for transcribing Egyptian speech for what is called augmented conventional orthography for dialectal Arabic (augmented-CODA). They also reported a gain in Egyptian speech recognition when augmented-CODA is followed

[1] https://github.com/qcri/multiRefWER

in transcribing Egyptian speech data.

In this chapter, we propose a more robust solution for handling the variations in orthography when no rules exist by using multiple reference transcriptions for evaluations. This leads to less-biased evaluation to a given form of writing. In addition, it is a language-independent approach that could be applied to any NSOL. This has been the main motivation for us not to apply any text normalisation or pre-processing for the text.

8.3 Multi-reference evaluation for ASR

8.3.1 Multi-references alignment to recognised speech text

The initial step for an ASR multi-reference evaluation is to have alignment between each recognised word and the corresponding reference words from all references. Our approach extends the current alignment used when performing ASR evaluation between the recognised text and one reference text to allow for alignment between the recognised text and N references.

For a recognised text $Rec = (w'_1, w'_2, ... w'_{|Rec|})$, and a set of N references: $Ref_1 = (w_{11}, w_{12}, ... w_{1|Ref_1|})$ to $Ref_N = (w_{N1}, w_{N2}, ... w_{N|Ref_N|})$, we perform the following steps:

- For each word in Rec, list all words in Ref_1 to Ref_N that are aligned to it. Note that some references may not include any corresponding word for some of the words in Rec, which is counted as an insertion. The output of this process will be an array of size N of reference words for each recognised word.

- The previous step effectively captures insertions, substitutions, and correct recognitions. However, deletions would not be handled, since there is no corresponding word in the Rec to the deleted words in the reference. In addition, a different number of deletions could exist across different references. To map deletions effectively across multiple references, for each reference, we map any non-aligned word to the recognised text to a *deletion pointer* (<**DEL**>) with a counter to the position of the last aligned word in Rec. For example, if two deletions are detected for one reference after three aligned words with Rec, the words in the reference would be

mapped to {*03-01 <DEL>*, *03-02 <DEL>*} in the *Rec*. If another deletion is detected after the fifth word in *Rec*, it will be mapped to *05-01 <DEL>*. For deletion pointers that are mapped to some of the references only, those references that have nothing deleted would be assigned to *NULL*.

Table 8.1 shows the output of alignment of a recognised DA sentence with four different references that disagree on the spelling of many words and the number of words itself. As shown, each word in the recognition is aligned to N references, which maximises the likelihood of finding a possible match that is accepted by one of the references.

8.3.2 Calculating MR-WER

Using the multi-aligned references, the number of correct, insertions, substitutions, and deletions are calculated as follows:

- **C** (Correct): is the number of recognised words that has a match in any of the aligned reference words.
- **S** (Substitutions): is the number of recognised words that has alignment to at least one reference words, but none of them matches it.
- **I** (Insertions): is the number of recognised words that are not aligned to any reference word, i.e., all corresponding alignments are *<INS>*.
- **D** (Deletions): is the number of *<DEL>* instances in the *Rec* that has no *NULL* alignment in any of the references. The main reason for not counting deletions that have no corresponding word in one of the references is that if one of the reference transcriptions decided not to write such a word, then the ASR should not be penalised for missing it.

$$WER = \frac{I + D + S}{S + D + C} \times 100\% \tag{8.1}$$

As shown in Table 8.1, the length of the transcription varies from one reference to another, which means that the deletion count is different between different transcriptions. The WER per reference ranged between 65% to 81%, which demonstrates the challenge of using a single reference for evaluation. One solution is to average the WER across different references, which is 75%. However,

Index	*Rec*	*Ref*1	*Ref*2	*Ref*3	*Ref*4
(00-1)	<DEL>	NULL	NULL	nEm	NULL
(00-2)	<DEL>	nEm	nEm	nEm	nEm
(01)	>ETY	Ah	Ah	Ah	hw
(02)	b<n	TbyEy	TbyEy	hw	TbyEY
(03)	dA	<n	dA	TbyEy	dA
(04)	>SlA	dp	>SlA	dh	>SlA
(05)	yEny	>SlAF	yEny	ASlA	yEnY
(06)	<HnA	<HnA	>HnA	AHnA	nHn
(07)	fy	fy	fY	fy	fy
(08)	wDE	wDE	wDE	wDE	wDE
(09)	gyr	gyr	gyr	gyr	gyr
(10)	qAnwny	qAnwny	qAnwny	qAnwny	qAnwnY
(11)	bAlmr	bAlmrp	bAlmrp	bAlmrh	bAlmrh
(12)	gyr	gyr	gyr	gyr	gyr
(13)	dstwry	dstwry	dstwry	dstwry	<INS>
(14)	bAlmr	<INS>	<INS>	<INS>	<INS>
(15)	wADH	<INS>	<INS>	<INS>	<INS>
(16)	>h	<INS>	bAlmrp	<INS>	dstwrY
(17)	fyh	bAlmrp	Ah	bAlmrh	bAlmrh
(18)	AnqlAb	wDE	wDE	wDE	wDE
WER	MR:53%	75%	65%	82%	81%

Table 8.1: Alignment applied between a recognised text (*Rec*) and four different references.

the MR-WER words achieved 53% calculated as shown in equation 8.1 using the alignment in table 8.1, which is a more realistic measure for this type of orthography. Our approach is similar to finding the best path through a lattice of references. However, we leave the exact comparison for future research.

8.4 Experiments

Our experiments were done using the crowdsourced dialectal data collected in chapter 5, we chose two dialects, namely Egyptian (EGY) and North African (NOR). For each dialect, we asked for five transcriptions for each speech segment (utterance), with an average length of 4-6 seconds per utterance. EGY had 2,087

utterances, totaling 3.6 hours, and NOR had 1,088 utterances with 3.1 hours. The data was transcribed using CrowdFlower[2], a crowdsourcing platform with a large user base in the Arab world. Quality control was performed using the best practices described by Wray and Ali [2015], Wray et al. [2015]. For more details about the crowdsourced data, see chapter 5.

8.4.1 Inter-reference agreement

An initial necessary step before evaluating the effectiveness of our evaluation methodology is to measure the difficulty of the problem. Here we measure the agreement on the transcriptions with different references. We measure the WER between each two references and we apply this to all references for all segments. We found that the median WER between different references for EGY is 59% and for NOR is 78.5%. We also calculated the percentage of exact-match transcriptions among references. The percentage was only 2.2% and 1.3% for EGY and NOR, respectively. These values were astonishing to us, therefore, we looked at many examples and determined that this was due to valid variation in the transcription. Our intuition for this diversity is that using a crowdsourcing platform for transcription limits the chance of having detailed training and in-depth guidelines for annotators. Consequently, we can see many disagreements in transcribing repeated words or unintelligible words, in addition to many transliterated words. In our further study to automate the multi-reference WER by harvesting dialectal data, we used a more controlled environment for transcription. Chapter 9 will discuss this in more detail. In general, this highlights the severe issue for these languages and confirms that the evaluation of ASR systems with only one reference would be highly biased.

8.4.2 MR-WER results

We evaluated the ASR output using 1 to 5 reference transcriptions. We used all the combinations between reference transcriptions in cases when $N > 1$ to validate our findings. As shown in table 8.2, for each experiment, we report the minimum, the maximum and the average MR-WER for each number of transcriptions we use. We conclude two findings from these experiments:

[2]http://www.crowdflower.com

	# Ref	One	Two	Three	Four	Five
EGY	Min.	69.1%	52.3%	45.9%	42.2%	39.70%
	Av.	71.4%	53.4%	46.4%	42.3%	
	Max.	74.0%	55.1%	47.3%	42.7%	
	# Exp.	5	10	10	5	1

	# Ref	One	Two	Three	Four	Five
NOR	Min.	78.9%	59.1%	51.8%	48.1%	45.9%
	Av.	80.2%	60.4%	52.8%	48.7%	
	Max.	80.7%	62.2%	53.9%	49.2%	
	# Exp.	5	10	10	5	1

Table 8.2: MR-WER for various number of references per experiment.

1. The WER is reduced considerably when we increase the number of transcriptions, and there may be a potential to reduce the WER more if there are more transcriptions (although we can see the reduction in MR-WER between four and five references is minor). The MR-WER has reduced the error from 71.4% to 39.7% in EGY, and from 80.1% to 45.9% in NOR. This could be happening due to various ways of writing DA and not due to bad ASR.

2. The variance in WER is reduced noticeably when the number of references increase. This is due to the fact that multi-reference is capable of capturing some of the variations in transcription, which makes the reported error rate more robust to actual mistakes.

8.4.3 Applying voting with multi-references

In the standard WER, the algorithm will loop over a single reference and check each word: insertion, deletion, substitution or correct. However, in the MR scenario, someone can argue that the algorithm in acting like cherry-picking and looking for a correct word in any of the references to make the WER look better rather than validating these findings. To address this concern, we explore the impact in MR-WER when the algorithm asks for more than one evidence that a word is correct, i.e., the same word occurred in the same position in more than one reference. We evaluated correct word counting in 1+ (standard), 2+ and 3+

		One	Two	Three	Four	Five
EGY	1+	71.4%	53.4%	46.4%	42.4%	39.7%
	2+	NA	78.3%	63.3%	55.5%	50.7%
	3+	NA	NA	83.7%	69.6%	61.6%
		One	Two	Three	Four	Five
NOR	1+	80.2%	60.4%	52.8%	48.7%	45.9%
	2+	NA	84.5%	69.7%	61.6%	56.7%
	3+	NA	NA	88.9%	76.0%	67.5%

Table 8.3: MR-WER with one or more voting for acceptance.

occurrences. Obviously, we apply N number of times seeing the word correct if there is N number of references or more.

As we can clearly see in Table 8.3, the proposed MR-WER reports that while asking for more than one proof in the reference for each correct word, the MR-WER still outperforms the standard WER when we average it over five references.

8.5 Conclusions

In this chapter, we presented a novel way for measuring ASR performance in non-standard orthographic languages: *multi-reference* Word Error Rate (MR-WER). Our results were based on two Dialectal Arabic corpora: Egyptian and North African. We were able to report 39.7% and 45.9% MR-WER, respectively, using five reference transcriptions collectively, while for the same test set the average WER was 71.4%, and 80.1%, respectively, when we used the same five references individually.

Chapter 9

Dialectal Word Error Rate: WERd

This chapter is based on [Ali et al., 2017a] published at ASRU 2017 and concerns dialectal word error rate **WER*d***, which evaluates dialectal speech recognition using social text spelling. We study the problem of evaluating automatic speech recognition (ASR) systems that target dialectal speech input.

9.1 Introduction

Automatic Speech Recognition (ASR) has shown fast progress recently, thanks to advancements in deep learning. As a result, the best systems for English have achieved a single-digit word error rate (WER) for some conversational tasks [Saon et al., 2017]. However, this is different for dialectal ASR, for which the WER can easily go over 40% [Ali et al., 2017b]. Chapter 7 shows more details about high WER in dialectal speech in the MGB-3 challenge.

As mentioned in chapter 8, in a standardised language such as English, we know that *enough* is a correct spelling, while *enuf* is not. However, we cannot be sure about the correct spellings of dialectal words; at best, we would know what a preferred or a dominant spelling is. This is because dialects typically do not have an official status and thus their spelling is not regulated, which opens the door widely to orthographic variation.[1]

Table 2.4 shows some examples of spelling variation in DA. We can see that clitics (pronouns and negations) can be written concatenated or separated from the verb, and the definite article can undergo different spelling variations due

[1]Note that here we target primarily intra-dialectal variation. Yet, there is also inter-dialect variation, e.g., between the different dialects of Arabic.

to coarticulation with the following word. Long vowels can become short, and thus be dropped as they are typically not written in Arabic, etc. While some variations can happen in standardised languages such as English, e.g., *healthcare* vs. *health care*, or *organise* vs. *organise*, this is much less common, and in ASR it is easily handled with simple rules, e.g., using the Global Mapping file[2] in SCLITE [Rosenfeld and Clarkson, 1997, Fiscus, 1997].

The examples in table 2.4 partially explain the high WER for dialects. While they suffer from a lack of training resources, the main problem is their informal status, which means that their spelling is rarely regulated. This makes training an ASR system for dialects much harder as there is no single gold standard towards which to optimise at training time.

More importantly, it is hard to evaluate such a system and to measure progress as multiple possible text outputs for the same speech signal could be considered correct by different people. Thus, there is a need for an evaluation measure that would allow for common spelling variations. In this chapter, we propose to mine such variations from dialectal Arabic tweets and to incorporate them as spelling variants as part of a more adequate ASR evaluation measure for dialects.

In chapter 8, we addressed this challenge using the multi-reference word error rate (MR-WER) [Ali et al., 2015], which is similar to the multi-reference BLEU score [Papineni et al., 2002] used to evaluate Machine Translation (MT). However, obtaining multiple references is expensive. Moreover, it could take many human annotators to get good coverage of the possible orthographic variants of the transcription of a speech recording. Thus, we propose to use a single reference, but to perform matching using spelling variants that could capture some of the variation.

This was applied to MT, e.g., for parameter optimisation [Madnani et al., 2007], where additional synthetic references are generated for tuning purposes, or for phrase-based MT, where paraphrasing is applied to the source side of the phrase table [Callison-Burch et al., 2006], of the training bi-text [Nakov, 2008b], or both [Nakov, 2008a, Nakov and Ng, 2011, Wang et al., 2012, 2016]. Paraphrasing has also been used for evaluating text summarisation [Zhou et al., 2006].

More relevant to the present work, in MT evaluation, paraphrasing was applied to the output of an MT system [Kauchak and Barzilay, 2006]. It was also

[2]The global mapping file can help for handcrafted variants like *color/colour* and *ten/10* in English. However, it is not applicable to dialectal Arabic, where multiple spelling variants are acceptable; we use 11M pairs.

incorporated in measures such as TERp [Snover et al., 2009], which is a translation edit rate metric with paraphrases. Indeed, here we borrow ideas from TERp for dialectal ASR, with a paraphrase table (in our case, a spelling variants table), which we mine automatically from a huge collection of tweets in an unsupervised fashion. Our experiments and our manual analysis show that this is a very promising idea.

Our contributions are as follows: (*i*) We propose a method for automatically collecting spelling and tokenisation variations for dialectical Arabic (and, presumably, other languages and language variants) from Twitter data; (*ii*) We further incorporate these spelling variants in an evaluation metric, WERd, which is a variation of TERp, and we demonstrate its utility for dialectal Arabic ASR.

9.2 Method

We propose a method for evaluating dialectal ASR, which consists of two steps: (*i*) collecting a large number of spelling variants, which we mine from social media in an unsupervised manner, and (*ii*) using these spelling variants, with associated probabilities, into an MT-inspired evaluation measure (together with standard unit-cost word insertions, deletions, and substitutions).

9.2.1 Mining spelling variants from social media

We use social media to mine dialectal spelling variants from a collection of half a billion dialectal Arabic tweets. Our approach is language-independent, scalable, and unsupervised, as it assumes no prior knowledge about the language, its dialects, or the data.

We build a list of pairs of spelling variants with probabilities using the following steps (as shown in figure 9.1):

First, we collect Arabic tweets. Then, we normalise hashtags, URLs, emoticons. We further drop Arabic diacritics and elongation, and we reduce letter repetitions to a maximum of three. Our pipeline is an extension to the previous work done in Arabic language processing for microblogs [Darwish et al., 2012].

Next, we extract all the *n*-grams of lengths 5–8. In each *n*-gram, we consider the first two and the last two words as a context, and the 1–4 words in the middle as a *target* for this context. For example, for a 5-gram con-

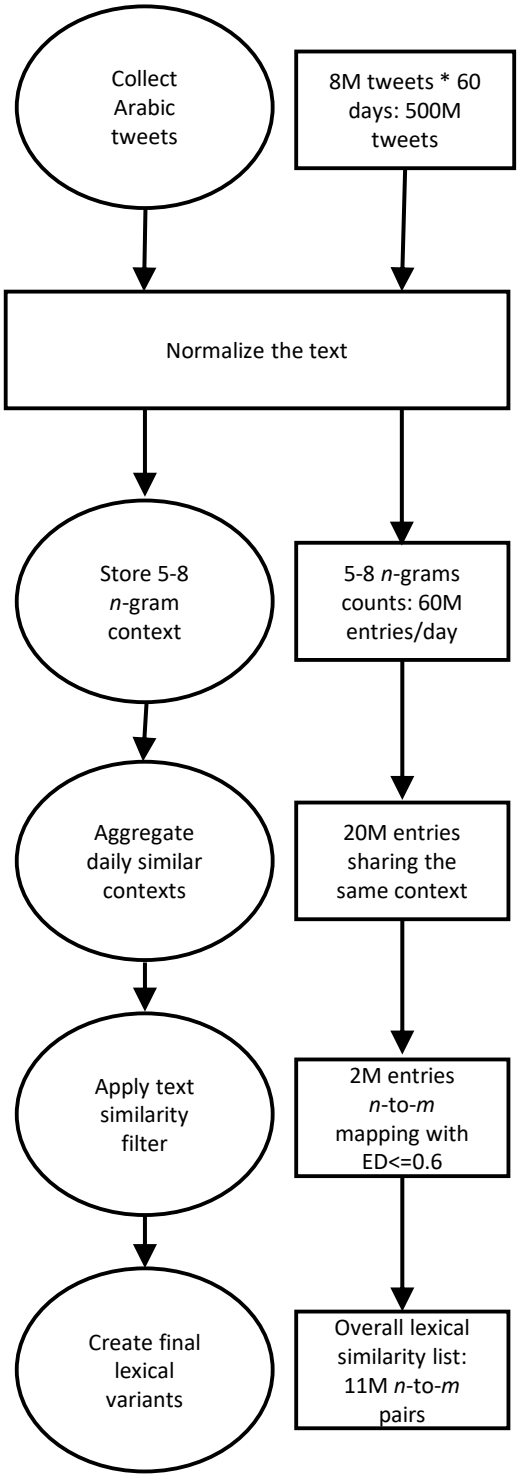


Figure 9.1: Diagram of our pipeline for extracting dialectal spelling variants from Twitter.

text, we will have $< L_1, L_2, t_1, R_1, R_2 >$, while for an 8-gram we will have $< L_1, L_2, t_1, t_2, t_3, t_4, R_1, R_2 >$, where L_i and R_i represent the left and the right

context words, and t_j are the target words in the middle ($1 \leq i \leq 2$, $1 \leq j \leq 4$).

Next, we generate pairs of potential spelling variants for targets that share the same contexts. This is subject to the constraint that the normalised Levenshtein distance between the targets is less than t, measured in characters. We tried values between 0.1 and 0.6 for t, and we manually inspected the resulting pairs of spelling variants. Ultimately, we set $t = 0.6$. With normalisation in mind, we further impose a constraint that in each pair of spelling variants, one of the targets is extracted in the same contexts at least N times more frequently than the other one (we set N to 3). Finally, with each pair of spelling variants, we associate a score: the average of the two Levenshtein distances. The resulting scored pairs of spelling variants form a spelling variant table for WERd.

Here are two examples from this final table of n-to-m spelling variant pairs with corresponding frequencies and normalised edit distance (shown in Buckwalter):

mAfy	mAAfy	752	75	0.25
lwny w DAEt	lwny wDAEt	32	8	0.1

The first column (yellow) contains the frequent form, which is the target **mAfy**. The second column (green) contains the source **mAAfy**, which is a less frequent term. The next column is the frequency of the target, e.g., the word **mAfy** occurred 752 times. The following column is the frequency of the source in the same context, e.g., **mAAfy** occurred 75 times. Finally comes the normalised edit distance.

Related approaches for paraphrase extraction have used random walks [Hassan and Menezes, 2013], pairwise similarity [Han et al., 2012], and continuous representations [Sridhar, 2015, Sproat and Jaitly, 2016]. Unlike that work, we mine pairs of spelling variants for ASR evaluation, not for modeling; we further allow many-to-many mappings, and we do not target canonical gold normalisation.

9.2.2 Using the spelling variants for evaluation: WERd

We borrow ideas from an evaluation measure for MT evaluation, namely *Translation Edit Rate Plus* or *TERp* [Snover et al., 2006]. TERp allows block alignment

of words, called *shifts* within the hypothesis as a low cost edit, a cost of 1, the same as the cost for inserting, deleting or substituting a word. TERp uses a greedy search and shift constraints to both reduce the computational complexity and to model the quality of translation better. The metric further supports tuned weights for the edit operations, a paraphrase table, synonym/hypernym-based matching using WordNet, etc.

The main motivation for using paraphrases in TERp for MT evaluation is to capture some lexical variation, e.g., *(controversy over, polemic about), (by using power, by force), (brief, short), (response, reaction)*. In contrast, we focus on capturing spelling variation in a dialect as shown in section 2.4.

In this work, we only use the paraphrasing capability of TERp. We restrict the matching to monotonic, i.e., no reorderings and no shifts. The only additional operation that we allow, compared to WER, is mapping between the hypothesis and the reference using a pair of spelling variants from our spelling variants table, which can span up to four words on either side of the pair of spelling variants as we have explained above. This monotonic version of TERp, with no reordering but with spelling variant matching capabilities gives rise to our metric for dialectal ASR evaluation, which we will call WERd (or *WER for dialects*).

9.3 Experiments and evaluation

9.3.1 Dialectal data

Speech data. We collected two hours of Egyptian Arabic Broadcast news [Wray and Ali, 2015] speech data, which we split into 1,217 segments, each 3-10 seconds long. The data are a subset of the Egyptian dialectal data collected in chapter 5. Since Egyptian Arabic has no established orthographic rules, it is difficult to develop standard transcription guidelines covering orthography. Therefore, we decided to have multiple transcriptions, but to let transcribers write the transcripts as they deemed correct, while trying to be as verbatim as possible. All the transcribers are native speakers of the chosen dialect with no linguistic background[3]. It is worth noting that, transcription in this chapter has been carried out in carefully controlled environment rather than using CrowdFlower as described

[3]The transcribers were asked to follow these transcription guidelines: `http://alt.qcri.org/resources/MGB-3/Arabic_Transcription%20_Guidelines_20170330.pdf`

in chapter 8 to make sure we have relatively high quality control. In an attempt to understand why the difference in variance is much smaller here compared to the crowdsourced data in chapter 8, we looked at some examples and we observed the following:

- Due to the nature of crowdsourcing, we are limited on how detailed guidelines could be. This is not the case here where we were able to give detailed instructions (as detailed in section 9.3.1) and to provide some on-boarding to the annotators. Therefore, we noticed many cases where hesitation and unintelligible words were not consistent across annotations. These are common in the DA speech.

- The diversity of workers in CrowdFlower was high, with transcribers coming from all over the Arab region. Regional differences in style and spelling were manifested. This led to higher sparseness.

- We found a small percentage of incorrect transcription that was not caught by our quality control mechanisms.

Table 9.1 shows the overlap agreement between the annotators, at the segment level, for their original transcription and after applying surface normalisation for *alef*, *yah* and *hah*, which is standard for Arabic. In Table 9.1, the first number is for the original text, and the second number is for the normalised text. We can see that even after normalisation,[4] there are about 15% differences between most of the annotators.

Social media data: We further collected dialectal Arabic tweets in order to extract spelling variants. In particular, we issued queries using `lang:ar` against the Twitter API[5]. Note that we did not try to control the location where the tweets originated from, but only the language they were written in. We collected two months of tweets (from December 2015 and January 2016), with about eight million tweets per day on average, which yielded a total of half a billion tweets containing over seven billion word tokens.

ASR system: For our experiments, we used the speech-to-text transcription system built using the 2016 Arabic Multi-Dialect Broadcast Media Recognition (MGB-2) as discussed in chapter 7.

[4]Below, we will report results after normalisation only.

[5]`http://dev.twitter.com/`

	ref2	ref3	ref4	ref5
ref1	77/86	80/84	78/86	80/87
ref2	—	74/83	71/85	72/85
ref3	—	—	77/84	78/84
ref4	—	—	—	91/93

Table 9.1: Pairwise overlap of the five human references before/after normalisation (in %).

	WER	TER	WERd	MR-WER
ref1	46.2	37.4	34.3	—
ref2	42.9	38.7	35.7	—
ref3	48.9	41.9	38.3	—
ref4	46.2	39.0	35.6	—
ref5	46.0	38.3	34.9	—
ALL refs	—	—	—	25.3

Table 9.2: WER vs. TER vs. WERd vs. MR-WER, after normalisation (in %).

9.3.2 Experimental results

We first evaluated the ASR system on our two-hour dialectal Arabic test dataset using WER with respect to each of the five references. The results are shown in Table 9.2. We can see that the WER is much higher on our dialectal Arabic dataset, ranging in 40–50%.

We further calculated MR-WER for our ASR system using all five references, achieving a score of 25.3%. This number is much lower than when evaluating with respect to any individual reference, which is to be expected, as we allow more matching options.

Table 9.2 reports TER[6] and WERd scores calculated with respect to each of the five references and shows for both metrics a strong correlation with WER. We can also see that the scores for WERd are halfway between WER and MR-WER

[6]The difference between TER and WER arises owing to the costs associated with deletion (D), insertion (I), and substitution (S) in the TER framework are 1.4, 0.25 and 1.6 respectively. This is an unusual setup for the ASR. We plan to use the same unified cost for I, D and S in our future study to be able to compare WER with WERd.

	No Variants	ED $\leq$ 0.1	ED $\leq$ 0.2	ED $\leq$ 0.3	ED $\leq$ 0.4	ED $\leq$ 0.5	ED $\leq$ 0.6
ref1	37.4	37.1	36.6	35.5	34.8	34.6	34.3
ref2	38.7	38.4	37.9	36.9	36.3	36.1	35.7
ref3	41.9	41.5	40.9	39.7	38.9	38.7	38.3
ref4	39.0	38.6	38.1	36.9	36.2	36.0	35.6
ref5	38.3	37.9	37.3	36.2	35.6	35.3	34.9

Table 9.3: WERd using pairs of spelling variants extracted using different maximum edit distances (ED).

(e.g., for ref1, it is 34.3 vs. 46.2 and 25.3, respectively), but without the need for additional human references.

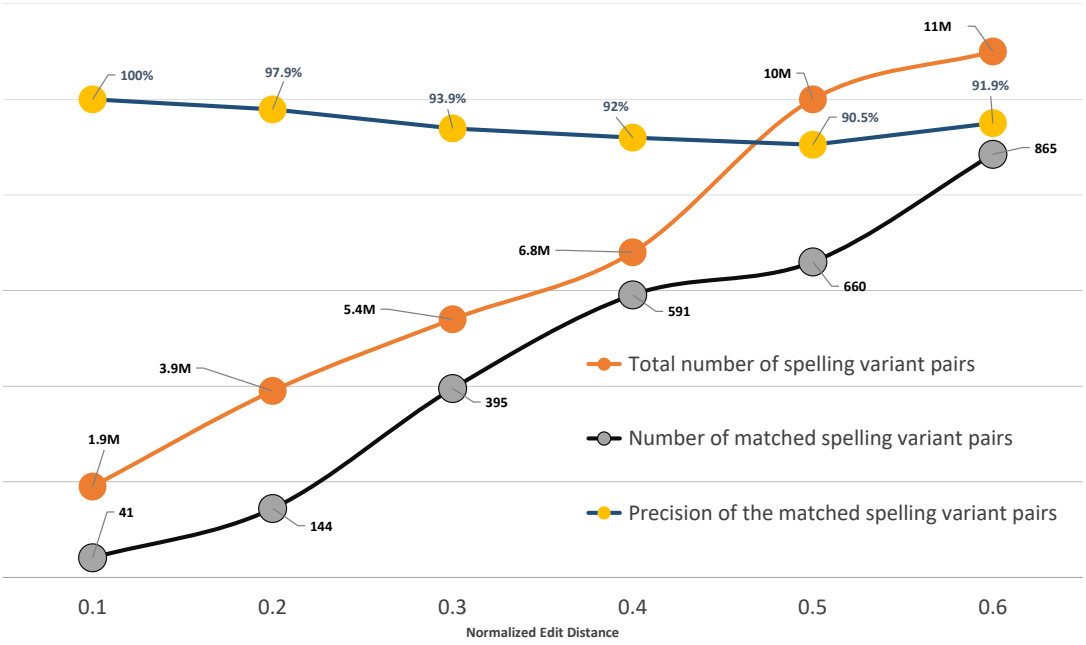


Figure 9.2: Analysis of the total number of spelling variants, the number of matched variants, and the precision for different thresholds on the edit distance (with respect to ref1). Note that the y axis shows different units for each of the three curves.

There are two reasons for MR-WER to be considerably lower compared to the other metrics. First, the way foreign words and codeswitching is handled by different annotators, e.g., words like *BBC* can be written in either Arabic (بي بي سي *by by sy*) or Latin characters. Some annotators would use Arabic while others would prefer English, which would allow matching either of them when evaluating the ASR output with multiple references. Second, in dialectal

Arabic, there are many filler words such as يعني *yEny*, اصل *ASl* and زي *zy*, which some annotators would skip and some would keep.

9.4 Discussion

WERd for different thresholds. Table 9.3 shows the performance of WERd when using pairs of spelling variants with different maximum edit distances: 0.1–0.6. As the threshold increases, WERd decreases, e.g., for ref1, it goes from 37.4 to 34.3, or 8% relative reduction. The difference is due to the number of matched spelling variant pairs, e.g., 865 for ref1.

Analysis of the pairs of spelling variant matches. Next, we study the relationship between the threshold on the maximum edit distance vs. the spelling variant table size, the number of spelling variants matches, and the accuracy of these matches. This is shown in figure 9.2, where we focus on the best reference, ref1 (according to native speakers of Egyptian Arabic who have a linguistic background). We can see that the threshold has a major impact on the spelling variant table size: going from 0.1 to 0.6 yields a six times larger table. It also yields a 21 times larger number of spelling variant matches on the test dataset: from 41 to 865.

Of course, this comes at a cost: while all 41 spelling variant matches at threshold of 0.1 are correct, there are 8% errors among the 865 matches at threshold of 0.6. We believe this is a relatively small price to pay, given the advantage of being able to identify 791 additional correct matches, which we capture without the need for having multiple references.[7]

Pearson correlation. We further measured the correlation between WER/MR-WER vs. TER/WERd. For the 1,217 test utterances, we calculated the scores for WER/MR-WER/TER/WERd in isolation, and then we calculated the Pearson correlation using the corresponding lists of utterance-level scores. The results are shown in Table 9.4. We can see that WERd correlates better than TER with both WER and also with MR-WER. Overall, we can conclude that WERd is a promising measure for evaluating ASR systems that target dialectal speech input.

Closer look at the spelling variants used in test. Finally, we had a closer look at the 865 spelling variant pairs that were matched and used when calcu-

[7]We were unable to measure the recall as it requires manual evaluation of all the possible candidates for spelling variants in the references.

Metrics Compared	Correlation
WER vs. TER	0.44
WER vs. WERd	0.47
MR-WER vs. TER	0.36
MR-WER vs. WERd	0.39

Table 9.4: Pearson correlations.

lating WERd for the test set of 1,217 segments, when using edit distance of 0.6. Our analysis shows three types of word-level changes:

1. *Word splitting*: 3% of the pairs
   e.g., مفيش (mfy$) → ما فيش (mA fy$).
2. *Word merging*: 16% of the pairs
   e.g., زي ما حنا (zy mA HnA) → زي ماحنا (zy mAHnA).
3. *Word substitution*: 81% of the pairs
   e.g., الامريكان (AlAmrykAn) → الاميركان (AlAmyrkAn).

These statistics show that we learn many useful spelling variants, i.e., more than 80%, rather than just splitting and merging words. Moreover, note that these word-level substitutions are actually small character-level transformations inside words. Tables 9.5 and 9.6 show some examples of correct and wrong spelling variant pairs that were matched when calculating WERd for our Dialectal Arabic test set.

Table 9.7 further shows how spelling variant pairs affect hypothesis scoring for an example test sentence. There are three spelling variant pairs that match the input ASR hypothesis ما فيش → مفيش, الامريكيه → الاميركيه, and finally علشان → عشان. The former involves word splitting, while the latter two are about substitution. We can see that the WER after using spelling variant substitutions would go down to 30%, (the actual WERd score would be slightly higher as it needs to take the cost of the spelling variant substitutions into account), while the initial WER was 61.5%.

English Gloss	Spelling Variants		Operation
Netanyahu	نتانياهو	ntAnyAhw	word substitution
	نتنياهو	ntnyAhw	
as we are	زي ما حنا	zy mA HnA	word merging
	زي ماحنا	zy mAHnA	
talking	بتتكلم	bttklm	word substitution
	تتكلم	ttklm	
like (as if)	يعني	yEny	word splitting
	يعني ب	yEny b	

Table 9.5: Correctly accepted spelling variants in test.

English Gloss	Spelling Variants		Operation
some	بعد	bEd	word substitution
after	بعض	bED	
principal	رئيسي	r}ysy	word substitution
president	رئيس	r}ys	

Table 9.6: Wrongly accepted spelling variants in test.

Hypothesis (before): مفيش هم من مصر من الولايات المتحده الاميركيه عشان
mfy$ hm mn mSr mn AlwlAyAt AlmtHdh AlAmyrkyh E$An
WER: 61.54 [8/13; 0 insertions, 4 deletions, 4 substitutions]
Hypothesis (after): ما فيش هم من مصر من الولايات المتحده الامريكيه علشان
mA fy$ hm mn mSr mn AlwlAyAt AlmtHdh AlAmrykyh El$An
WER: 30.77 [4/13; 0 insertions, 3 deletions, 1 substitutions]
Reference: ما فيش زيهم جم من مصر وجم من كل الولايات المتحده الامريكيه علشان
mA fy$ zyhm jm mn mSr wjm mn kl AlwlAyAt AlmtHdh AlAmrykyh El$An

Table 9.7: Extra word matches due to using spelling variants. Shown is an ASR hypothesis for a test utterance, and the impact of hypothesis matching on the number of insertions, deletions and substitutions, as well as on the overall WER score.

9.5 Conclusions

In this chapter, we have addressed the evaluation of ASR systems that target dialectal speech input, where a major problem is the natural variation in spelling due to the unofficial status and the lack of standardisation of the orthography. We have proposed a new metric, WERd (or *WER for dialects*), a variation of TERp, for which multiple text outputs for the same speech signal can be acceptable given a single reference transcript. Our implementation of WERd was based on mining 11M pairs of spelling variants from a huge dialectal Arabic tweet collection. Our automatic experiments, as well as manual analysis, have shown that this is a highly promising metric that addresses the problems of WER for dialectal speech, and approaches the performance of multi-reference WER.

Chapter 10

Word Error Rate Estimation: e-WER

This chapter is based on [Ali and Renals, 2018] published at ACL 2018 and concerns estimating word error rate with no golden transcription.

10.1 Introduction

Measuring the performance of ASR systems requires manually transcribed data in order to compute the WER, which is often time-consuming and expensive. In this chapter, we propose a novel approach to estimate WER, or e-WER, which does not require a gold-standard transcription of the test set. Our e-WER framework uses a comprehensive set of features: ASR recognised text, grapheme recognition results to complement recognition output, and internal decoder (glass-box) features. We report results for the two features; black-box and glass-box using unseen 24 Arabic broadcast programs.

ASR has made rapid progress in recent years, primarily due to advances in deep learning and powerful computing platforms. As a result, the quality of ASR has improved dramatically, leading to various applications, such as speech-to-speech translation, personal assistants, and broadcast media monitoring. Despite this progress, ASR performance is still closely tied to how well the acoustic model (AM) and language model (LM) training data matches the test conditions. Thus, it is important to be able to estimate the accuracy of an ASR system in a particular target environment.

WER is the standard approach to evaluate the performance of a large vo-

cabulary continuous speech recognition (LVCSR) system. The word sequence hypothesised by the ASR system is aligned with a reference transcription, and the number of errors is computed as the sum of substitutions (S), insertions (I), and deletions (D). If there are N total words in the reference transcription, then the word error rate WER is computed as follows:

$$\text{WER} = \frac{I + D + S}{N} \times 100. \tag{10.1}$$

To obtain a reliable estimate of the WER, at least two hours of test data are required for a typical LVCSR system. In order to perform the alignment, the test data needs to be manually transcribed at the word level – a time-consuming and expensive process. In scenarios such as broadcast monitoring, it may not be practical to create test sets, as new channels may be frequently presented for monitoring. It is, thus, of interest to develop techniques which can estimate the quality of an automatically generated transcription without requiring a gold-standard reference.

Such quality estimation techniques have been extensively investigated for machine translation [Specia et al., 2013], with extensions to spoken language translation [Ng et al., 2015, 2016]. Although there is a long history of exploring word-level confidence measures for speech recognition [Evermann and Woodland, 2000, Cox and Dasmahapatra, 2002, Jiang, 2005, Seigel and Woodland, 2011, Huang et al., 2013], there has been less work on the direct estimation of speech recognition errors.

Seigel and Woodland [2014] studied the detection of deletions in ASR output using a conditional random field (CRF) sequence model to detect one or more deleted word regions in ASR output. Ghannay et al. [2015] used word embeddings to build a confidence classifier which labelled each word in the recognised word sequence with an error or a correct label. They studied both Multi-layer Perceptrons (MLPs) and CRFs for sequence modeling. Tam et al. [2014] investigated the use of a RNN LM with complementary DNN and Gaussian Mixture Model (GMM) acoustic models in order to identify ASR errors, based on the assumption that when two ASR systems disagree on an utterance region, then it is most likely an error.

Ogawa and Hori [2015] investigated using deep bidirectional recurrent neural networks (DBRNNs) to detect errors in ASR results. They explored four tasks for ASR error detection and recognition rate estimation: confidence estimation, out-

of-vocabulary (OOV) word detection, error type classification, and recognition rate estimation. They showed that the DBRNN substantially outperformed CRFs for this task. In an extension to this work, Ogawa et al. [2016] investigated the estimation of speech recognition accuracy based on the classification of error types, in which sequence classification was performed by a CRF. Each word in a hypothesised word sequence was classified into one of three categories: correct, substitution error, or insertion error. Their study did not estimate the presence of deletions, and consequently cannot estimate the WER.

Jalalvand et al. [2016] developed a tool for ASR quality estimation, TranscRater, which is capable of predicting WER per utterance. This approach is based on a large set of extracted features (which do not require internal access to the ASR system) used to train a regression model (e.g., extremely randomised trees), with a mean absolute error (MAE) loss function, and can also rank different transcriptions from multiple sources [Negri et al., 2014, de Souza et al., 2015, Jalalvand and Falavigna, 2015, Jalalvand et al., 2015a,b]. TranscRater provides a WER per utterance, reporting the results as the MAE with respect to a reference transcription. This work did not report WER estimates for complete recordings or test sets, although it is possible that this could be done using utterance length estimates.

In this chapter, we build on these contributions to develop a system to directly estimate the WER of an ASR output hypothesis[1]. Our contributions are: (*i*) a novel approach to estimate WER per sentence and to aggregate them to provide WER estimation per recording or for a whole test set; (*ii*) an evaluation of our approach which compares the use of "black-box" features (without ASR decoder information) and "glass-box" features which use internal information from the decoder; and (*iii*) a release of the code and the data used for this chapter for further research.

10.2 e-WER framework

Estimating the probability of error of each word in a recognised word sequence has been successfully used to detect insertions, substitutions, and interword deletions [Ogawa et al., 2016, Ogawa and Hori, 2015, Ghannay et al., 2015, Jalalvand and

[1]It is worth noting that many commercial systems estimate the quality of an ASR using the average word-based confidence. We will evaluate this approach as a possible future extension.

Falavigna, 2015, Seigel and Woodland, 2014]. However, these local estimates do not provide an estimate of the overall pattern of error, such as the total number of deletions in an utterance.

In our framework, we use two speech recognition systems; a word-based LVCSR system and a grapheme-sequence based system. Following Tam et al. [2014], we assume that when two corresponding ASR systems disagree on a sentence or part of a sentence, there is a pattern of error to be learned. Our architecture also benefits from utterance-based LVCSR decoder features including the total number of frames, the average log likelihood and the duration. Intuitively, we correlate short sentences with less context and assume that LM scoring will not be able to capture long context. Since our approach is looking for the overall error pattern, we are not particularly concerned with the type of the error (insertion, deletion, or substitution). We estimate directly the numerator in equation 10.1, which is the summation of insertion, deletion and substitution errors, which we refer to as $E\hat{R}R$, the estimated total number of errors per utterance. We also estimate directly $\hat{N}$, an estimate of the total number of words in the reference. Therefore, e-WER is defined as follows:

$$\text{e-WER} = \frac{E\hat{R}R}{\hat{N}} \times 100\% \tag{10.2}$$

Our model is required to predict two values for each utterance: $E\hat{R}R$ and $\hat{N}$. Given that each is integer-valued, we decided to frame their estimation as a classification task rather than a regression problem as shown in equations 10.3 and 10.4. Each class represents a specific word count. We limit the total number of classes to a maximum of C in $E\hat{R}R$, with range from 0 to C. However, the total number of classes for $\hat{N}$ is $C - K$ to avoid estimating an utterance length of zero, with a range from K to C. If an utterance has more than C words or less than K words, it will thus be penalised by the loss function,

$$E\hat{R}R = \arg\max_{c_j \in C} P(c_j | x_1, x_2, ..., x_n) \tag{10.3}$$

$$\hat{N} = \arg\max_{k_j \in C-K} P(k_j | x_1, x_2, ..., x_n) \tag{10.4}$$

Table 10.3 shows that fewer than 5% of the sentences have more than 20 words, and it is very unlikely to have an utterance with fewer than 2 words. We trained our system with $C = 20$ and $K = 2$. Since our approach predicts $E\hat{R}R$ and $\hat{N}$ for

	LVCSR
Output	brnAmj AlwAqE AlErby
Normalised	b r n A m j A l w A q E A l E r b y
	Grapheme ASR
Output	q p SIL A l j w A q A l E d y
Normalised	q p A l j w A q A l E d y

Table 10.1: Normalised sequences from the two systems used to provide the grapheme alignment error rate. GER=50.00%; G_ERR=9; G_N=18; G_I=0; G_D=5; G_S=4.

each sentence, it is possible to aggregate each of the two values across the entire test set in order to estimate the overall WER, as shown in section 10.4.

10.2.1 Speech recognition system

The LVCSR system was trained using the second Multi-Genre Broadcast challenge data, MGB-2 as described in chapter 7. For acoustic modelling, we used the TDNN-LSTM models as described in section 7.2.3 and for LM, we use the small-four-gram LM (sLM4) as described in section 7.2.3. More details about the AM and LM data can be found in chapter 7. The results used in this chapter are limited to the one-best ASR results from the first-pass decoding to ensure that the decoder features match the single pass decoding results. Therefore, no system combination, nor LM rescoring was applied to the ASR results in this chapter.

10.2.2 e-WER features

To estimate e-WER, we combine features from the word-based LVCSR system with features from the grapheme-based system. By running both word-based and character-based ASR systems, we are able to align their outputs against each other as shown in table 10.1, assuming the LVCSR system provides the ground truth. We extract six features from the grapheme alignment: (1) grapheme error rate (GER); (2) total grapheme errors (G_ERR); (3) grapheme count (G_N); (4) grapheme insertion count (G_I); (5) grapheme substitution count (G_S); and (6) grapheme deletion count (G_D). We split the studied features into the following four groups:

- *L*: lexical features – the word sequence extracted from the LVCSR;

- G: grapheme features – character sequence extracted from the grapheme recognition;
- N: numerical features – basic features about the speech signal, as well as grapheme alignment error details;
- D: decoder features – total frame count, average log-likelihood, total acoustic model likelihood and total language model likelihood.

Similar to previous research in ASR quality estimation, we refer to $\{L,G,N\}$ as the black-box features, and $\{L,G,N,D\}$ as the glass-box features, which are used to estimate the total number of words $\hat{N}$, and the total number of errors $E\hat{R}R$ in a given sentence. It is worth noting that the glass-box features assume a single pass ASR decoding and not a multi-stage or system combination. Table 10.2 shows more detail about the studied features and their dimensions.

Category	Feature Type	# of Features
(L) Lexical (unigram/bigram)	word sequence	14K/58K
(G) Grapheme (unigram/bigram)	grapheme sequence	37/1.1K
(N) Sentence Duration	general	1
(N) Word count	lexical	1
(N) Character count	lexical	1
(N) Grapheme alignment	GER details	6
(D) Frames count	decoder features	1
(D) Average loglikelihood	decoder	1
(D) Total AM loglikelihood	decoder	1
(D) Total LM loglikelihood	decoer	1
Total: 14,450 unigrams and 59,513 bigrams.		

Table 10.2: Features used for WER estimation.

10.2.3 Classification Back-end

We deployed a feed-forward neural network as a backend classifier for e-WER. The deployed network in this work has two fully-connected hidden layers (ReLU activation function), with 128 neurons in the first layer and 64 neurons in the second layer followed by a softmax layer. A minibatch size of 32 was used, and the number of epochs was up to 50 with an early stopping criterion. Figure 10.1 illustrates the deployed DNN system architecture. We report mean absolute error

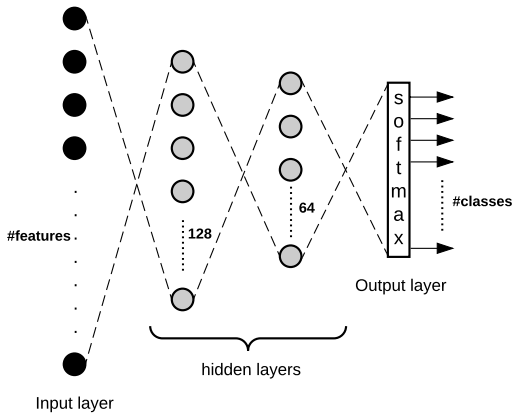


Figure 10.1: DNN architecture.

(MAE) and root mean square error (RMSE) for estimating the total number of errors ($E\hat{R}R$) and the total number of words ($\hat{N}$) per sentence.

10.3 Data

The e-WER training and development data sets are the same as the Arabic MGB-2 development and evlauation sets [Ali et al., 2016], which is comprised of audio extracted from Al-Jazeera Arabic TV programs recorded by Brightcove in the last months of 2015. The e-WER training and development sets come each comprise 10 hours of audio that were not used in the MGB-2 training data. (Other episodes of the same program may have been included in the training set). For more details about the MGB-2 development and evaluation, see section 7.2.1. To test whether our approach generalises to test sets from a different source, and not tuned to the MGB-2 data set, we validated our results on another three hours of test set collected by BBC Monitoring during November 2016, as part of the SUMMA project[2]. The SUMMA data is referred to as the test set. All data was manually segmented and labelled. Table 10.3 shows more details about the data used for these experiments.

10.4 Experiments and discussions

We trained two DNN systems to estimate $\hat{N}$ and $E\hat{R}R$ separately. We explored training both a black-box based DNN system (without the decoder features)

[2]http://summa-project.eu

	Train	Dev	Test
Number of programs in corpus	17	17	24
Utterances	58K	56K	1.4K
Duration (in hours)	9.9	10.2	3.2
2-20 words sentences	96%	95%	96%
Word count (N)	**75K**	**69K**	**20K**
ASR word count (hyp)	58K	60K	18K
WER	42.6%	33.1%	28.5%
Sentence Error Rate (SER)	88.7%	89.1%	86.0%
Total INS	1.9K	1.8K	130
Total DEL	19.1K	10.2K	2.6K
Total SUB	11.1K	10.8K	2.9K
ERR count (ERR)	**32.1K**	**22.8K**	**5.7K**

Table 10.3: Analysis of the train, dev and test data.

	MAE/Dev			**MAE/Test**		
	$E\hat{R}R$	$\hat{N}$	e-WER	$E\hat{R}R$	$\hat{N}$	e-WER
glass-box	1.60	1.75	13.78	1.65	1.69	12.29
black-box	1.81	2.16	28.38	1.97	2.32	24.68

Table 10.4: MAE per sentence reported for the glass-box and black-box features.

and a glass-box system using the decoder features. Overall, four systems were trained: two glass-box systems and two black-box systems. We used the same hyper-parameters across the four systems. Tables 10.4 and 10.5 present the e-WER performance in terms of MAE and RMSE per sentence for $E\hat{R}R$, $\hat{N}$ and the estimated WER for the dev and test sets with reference to the errors computed using a gold-standard reference. As expected, the glass-box features help to reduce MAE and RMSE for both $E\hat{R}R$ and $\hat{N}$. Although the difference between the black-box estimation and the glass-box results is not big for $E\hat{R}R$ and $\hat{N}$, we can see that the impact becomes substantial on the estimated WER per sentence, which is almost double the error in both MAE and RMSE per sentence.

Table 10.6 reports the overall performance on the dev and on the test set. Across the 17 programs in the MGB-2 dev data, the actual WER is 33.0%, and the glass-box e-WER is 29.32%, while the black-box e-WER is 30.87%. Evaluating the same models on the 24 programs test data results in an actual WER of 28.51%, while the glass-box e-WER is 25.35%, and the black-box e-WER is 30.25%.

	RMSE/Dev			RMSE/Test		
	$E\hat{R}R$	$\hat{N}$	e-WER	$E\hat{R}R$	$\hat{N}$	e-WER
glass-box	2.20	2.14	18.32	2.32	2.15	16.88
black-box	2.43	2.71	36.13	2.6	2.88	35.01

Table 10.5: RMSE per sentence reported for the glass-box and black-box features.

	Actual/estimated WER		
Data	Reference	glass-box	black-box
Dev	33.03%	29.32%	30.87%
Eval	28.51%	25.35%	30.25%

Table 10.6: Overall WER across dev and eval data set.

Tables 10.4 and 10.5 show the glass-box features outperformed the black-box features in predicting both $E\hat{R}R$ and $\hat{N}$. Furthermore, the performance of the estimated WER per sentence in the glass-box is substantially better than the black-box for both development and test sets. Table 10.6 (and also figure 10.2) indicates that the glass-box estimate is systematically lower than the black-box estimate. To further visualise these results, figure 10.2 plots the cumulative WER and e-WER across the three hours of test set. This plot indicates that the glass-box estimate is continually lower than the black-box estimate. The large difference during the first 30 minutes arises owing to significant over-estimates of the WER by the black-box system during the first 2-3 programs, as can be seen in figure 10.2.

We estimate $\hat{N}$ and $E\hat{R}R$ separately. Therefore, our system is capable of estimating the WER at different levels of granularity.

Since the sentence is the smallest unit in the proposed framework, we tried to visualise the performance of our prediction at the sentence level on the dev set. Here, we plot the confusion matrix heat-map for predicting the total number of errors and total number of words for each sentence. It is noticeable that most of the heat is around the diagonal axis and when there is a shift, it is not far from the right zone. Figure 10.3 shows the heat-map for predicting $E\hat{R}R$, where we can see most of the intensity in the diagonal with less than ten words. This is expected since the overall WER is less than 30%. On the other hand, figure 10.4 shows the heat-map for word count $\hat{N}$, where we can see most of the intensity between 5-15 words. The test set has a similar pattern for the confusion matrix.

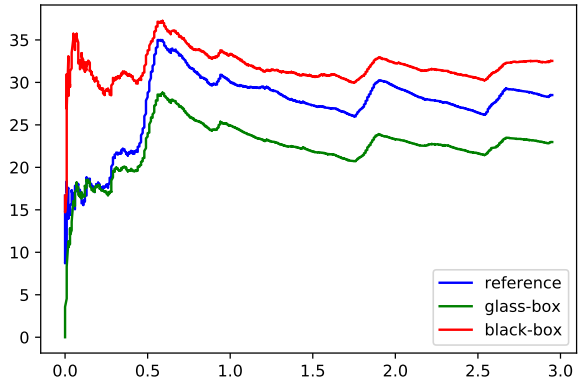


Figure 10.2: Test set cumulative WER over all sentences, x-axis is duration in hours and y-axis is WER in %.

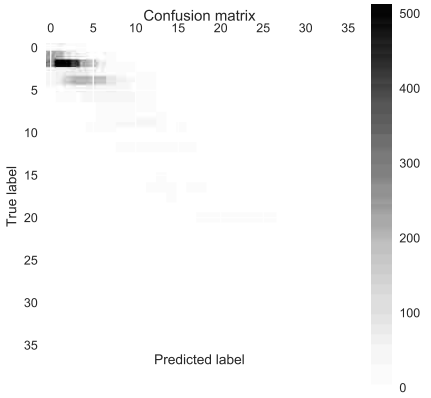


Figure 10.3: Confusion matrix heatmap for total error estimation per sentence.

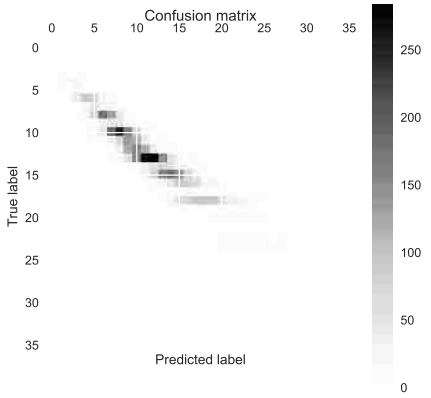


Figure 10.4: Confusion matrix heatmap for word count estimation per sentence.

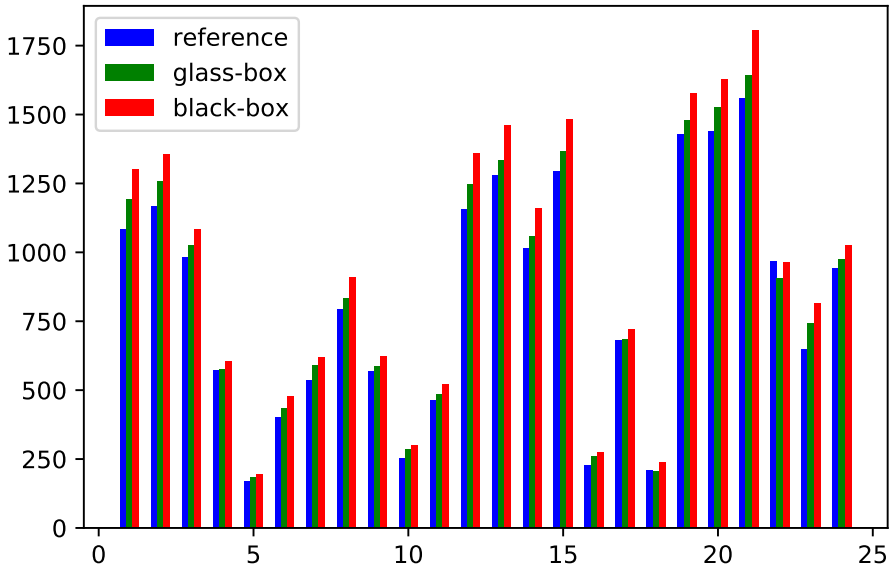


Figure 10.5: Total word count estimated over 24 programs in the test data.

Finally, we visualise the prediction per program, and we can see that predicting the total word count $\hat{N}$ in figure 10.5 shows the glass-box results are slightly better than the black-box. Similarly, in figure 10.6, the glass-box prediction outperforms the black-box in predicting total error count $E\hat{R}R$ per program. In scenarios such as media-monitoring, where the main objective is to have a robust monitoring system for specific programs, we plot the WER across the 24 programs in the test set, and we can see in figure 10.7 that both the glass-box and black-box estimation are following the gold-standard WER per program. However, unlike predicting word count $\hat{N}$ or error count $E\hat{R}R$, we can see that the black-box, in general, over-estimates the WER, while the glass-box system under-estimates WER similar to figure 10.2. It is not very clear which system is better. One can argue from figure 10.7 that the decoder features are not helping in programs with high WER. We found both systems to be useful for reporting WER per program.

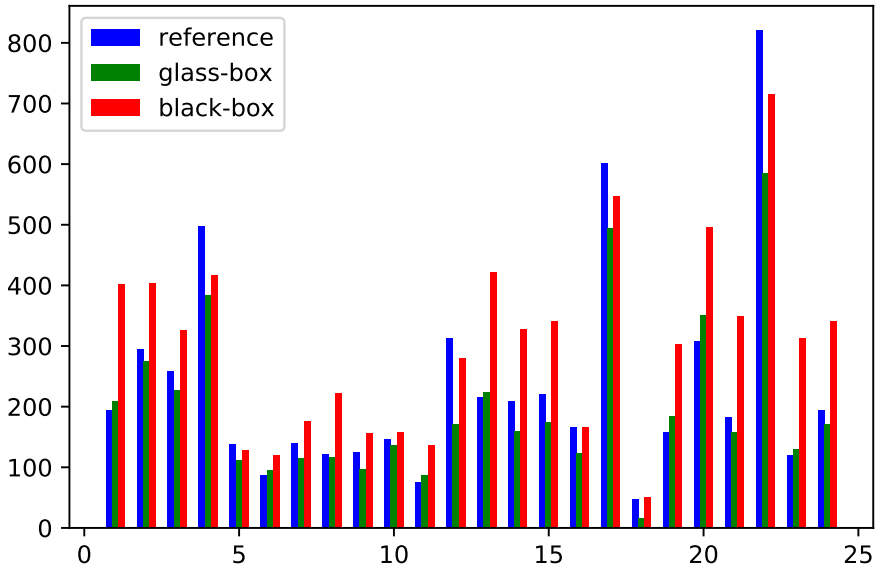


Figure 10.6: Total error count estimated over 24 programs on the eval data.

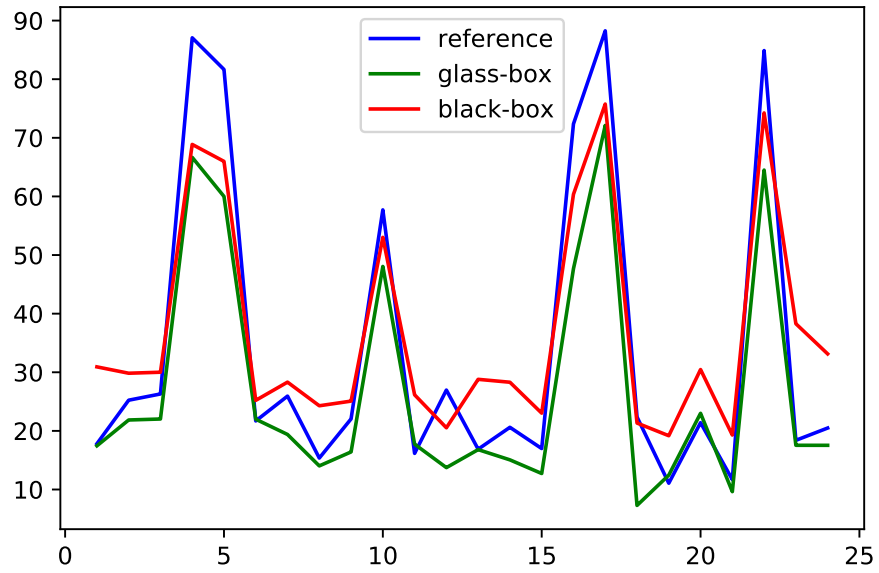


Figure 10.7: WER estimated over 24 programs on the eval data.

10.5 Conclusions

This chapter presented our efforts in predicting speech recognition word error rate without requiring a gold-standard reference transcription. We presented a DNN-based classifier to predict the total number of errors per utterance and the total word count separately. Our approach benefits from combining word-based and grapheme-based ASR results for the same sentence, along with extracted decoder features. We evaluated our approach per sentences and per program. Our experiments have shown that this approach is highly promising to estimate WER per sentence and we have aggregated the estimated results to predict WER for complete recordings, programs or test sets without the need for a reference transcription. For future work, we shall continue our investigation into approaches that can estimate the word error rate using convolutional neural networks. In particular, we would like to explore combining the DNN numerical features with the CNN word embedding features. Another line of research worth exploring is the possibility to build a language-independent module for word error rate estimation.

Chapter 11

Conclusions

11.1 Overview of contributions

This thesis investigated multi-dialect Arabic automatic speech recognition (ASR) with no prior knowledge about the spoken dialect. There are three main challenges in dialect Arabic ASR: (1) finding labelled dialectal Arabic speech data, (2) building robust dialectal speech recognition with limited labelled data and (3) evaluating speech recognition for dialects with no orthographic conventions. Below, we summarise our contributions and findings, and we explore possible future work.

11.1.1 Arabic dialect identification

We developed a dialectal Arabic (DA) corpus and we investigated different approaches to build robust Arabic dialect identification (ADI) system using acoustic and linguistic features. Chapter 5 featured our effort in labelling the multi-dialectal speech corpus, which we collected from Al Jazeera TV channel. The corpus contains 850 hours with approximately 18% DA speech. We used crowdsourcing to annotate a multi-dialectal speech corpus. We obtained utterance level dialect labels for 57 hours consisting of four major varieties of DA, namely: Egyptian, Levantine, Gulf, and North African. Using speaker linking to identify utterances spoken by the same speaker and label accuracy based on annotator behavior, we automatically labelled an additional 94 hours. We showed that using crowdsourcing to label samples from each speaker at the beginning and at the end of an audio segment resulted in labels for all of that speaker's speech and the

results are suggestive of a regular practice of codeswitching between one's native dialect and modern standard Arabic (MSA) – as shown in table 5.6. Chapter 6 was dedicated to approaches designed and optimised for ADI for Arabic broadcast speech. In our work, we proposed Arabic as a five-class dialect challenge comprising of the previously mentioned four dialects as well as MSA. We investigated different approaches for ADI in broadcast speech. These methods are based on phonotactic and lexical features obtained from a speech recognition system, and acoustic features using the i-vector framework. We studied both generative and discriminative classifiers, and we combined these features using a multi-class support vector machine (SVM), deep neural network (DNN) and convolutional neural network (CNN). We validated our results on an Arabic/English language identification task, with an accuracy of 100%. We also evaluated these features in a binary classifier to discriminate between MSA and DA, with an accuracy of 100%. We reported results using the proposed methods to discriminate between the five most widely used dialects of Arabic with an overall accuracy of 73%. We discussed dialect identification errors in the context of dialect codeswitching between DA and MSA, and compared the error pattern between labelled data, and the output from our classifier. All the data used in our experiments have been released to the public as a dialect identification corpus.

Future work for ADI: Our dialect identification study has two limitations: First, we restricted Arabic to five major dialects, while many more exist. A potential future direction for ADI is to increase the granularity of the task to account for more dialects – often multiple dialects per country. For example, we have seen annotators often disagree on speech from countries like Jordan to be classified as Gulf or Levantine. In a preliminary study, we found Youtube to be a good platform to harvest lots of speech data in a semi-supervised way per country. Second, we can address the DA codeswitching challenge as a dialect diarization problem as opposed to a classification problem. Speakers regularly codeswitch between one's native DA and MSA. As we saw in table 5.6 and figure 6.7, long utterance yields worse accuracy.

11.1.2 Arabic speech recognition

We introduced our effort in building Arabic ASR, and created an open research community platform to advance it. We have two main contributions: first, cre-

ating a framework for Arabic ASR that is publicly available for research; second, building a robust Arabic ASR system with limited labelled data leading to competitive WER results, which can be used as a potential benchmark to advance the state of the art in Arabic ASR.

- Creating a framework for Arabic ASR that is publicly available for research. We addressed our effort in building two multi-genre broadcast (MGB) challenges. MGB-2 for the 2016 Spoken Language Technology (SLT-2016) conference [Ali et al., 2016]. The second version of the MGB challenge emphasised the handling of diversity in the broadcast news domain in Arabic speech. Audio data comes from 19 distinct programmes from the Al Jazeera Arabic TV channel between March 2005 and December 2015. The programmes are split into three groups: conversations, interviews, and reports. A total of 1,200 hours have been released with lightly supervised transcriptions for the acoustic modelling. For language modelling, we made available over 130M words crawled from Al Jazeera Arabic website `Aljazeera.net` for a 10 year duration 2000-2011. Two lexicons were provided; one phoneme based and one grapheme based. The MGB-3 for the 2017 Automatic Speech Recognition and Understanding (ASRU-2017) [Ali et al., 2017b], which is the third version of the MGB challenge, emphasised DA ASR using a multi-genre collection of Egyptian YouTube videos. Seven genres were used for the data collection, namely: comedy, cooking, family/kids, fashion, drama, sports, and science (TEDx). A total of 16 hours of videos, split evenly across the different genres, were divided into adaptation, development and evaluation data sets. The MGB-3 has three targets; (a) dealing with languages which do not have well-defined orthographic systems, Egyptian Arabic in particular, (b) Multi-genre scenarios: seven different genres are included in the challenge, and (c) low-resource scenarios: only 16 hours of in-domain data was provided. Overall, thirteen teams submitted ten systems to the challenge. We outlined the approaches adopted in each system, and summarised the evaluation results.

- Building a robust Arabic ASR system and reporting a competitive word error rate (WER) to use as a potential benchmark to advance the state of the art in Arabic ASR. Our overall ASR system for the MGB-2 is a combination of the five acoustic models (AM); named as unidirectional long short term

memory (LSTM), bidirectional LSTM (BLSTM), time delay neural network (TDNN), TDNN layers along with LSTM layers (TDNN-LSTM), and finally TDNN layers followed by BLSTM layers (TDNN-BLSTM). The AM was trained using the purely sequence trained neural networks lattice-free maximum mutual information (LF-MMI). We also performed data augmentation using speed perturbation with speed factors of $0.9, 1.0, 1.1$, followed by volume perturbation uniformly sampled from the interval $[\frac{1}{8}, 2.0]$. This gave us three times the original training data. Given that Arabic is a phonologically complex language, the lexicon size was 1.3M words. Our acoustic units represent the character in the surface form of the words instead of phone units. As for the language model, we deployed n-gram LM for decoding; 4-gram for first pass decoding and LM rescoring. In addition to recurrent neural network language model with MaxEnt connections RNNME for n-best rescoring. The word error rate (WER) for the final system is 13%, which is the lowest WER reported on this task.

11.1.3 Dialect speech recognition evaluation

The standard word error rate (WER) assumes a single reference is sufficient for a single speech utterance, which is not true for languages and dialects lacking orthographic conventions. We examined appropriate methods for evaluating dialectal speech recognition:

- ***Multi-reference* word error rate (MR-WER):** We proposed a novel approach for evaluating ASR using multi-references. For each recognised speech segment, we asked n (five in our study) different users to transcribe the speech. We combined the alignment for the multiple references, and we used the combined alignment to report a modified version WER. This approach is in favor of accepting a recognised word if any of the references typed it in the same form. Our approach showed promising results for two dialects, namely Egyptian and North African. The proposed MR-WER approach is similar to BLEU in machine translation (MT) evaluation.

- **Dialectal word error rate (WERd):** In this study, we continued our effort in evaluating dialectal ASR with no orthographic rules. We automated the multi-reference in MR-WER by learning different writing from

Twitter data. We automatically mined more than 500M tweets in an unsupervised fashion to build more than 11M n-to-m lexical pairs, and we proposed a new evaluation metric inspired by the MT community, namely the translation edit rate metric with paraphrases TERp. Indeed, WERd (or *WER for dialects*) borrowed ideas from TERp for dialectal ASR, with a paraphrase table (in our case, a spelling variants table). Our experiments and our manual analysis show that this is a very promising idea.

Future work for dialectal ASR evaluation: We plan experiments with other dialects and non-standardised language varieties. We also want to incorporate word embeddings in the process of computation, e.g., character-based, which can naturally tolerate some spelling variation [Bojanowski et al., 2016]. We further want to explore using weighted finite state transducers (WFST) as an alternative way to allow using multiple spelling variants for both references and hypotheses. Another line of research that is worth exploring is integrating MR-WER and WERd for ASR systems such as a new technique for discriminative training objective function.

11.1.4 Word error rate estimation (e-WER)

Measuring the performance of ASR systems requires manually transcribed data in order to compute the WER, which is often time-consuming and expensive. In chapter 10, we proposed a novel approach to estimate WER, or e-WER, which does not require a gold-standard transcription of a test set. Our e-WER framework used a comprehensive set of features, namely: ASR recognised text, grapheme recognition results to complement recognition output, and internal decoder (glass-box) features. We reported results for the two features sets: black-box and glass-box using unseen 24 Arabic broadcast programs. Our system achieved 12.3% WER mean absolute error (MAE) and 16.9% WER root mean square error (RMSE) across 1,400 sentences. The estimated overall WER e-WER was 22.9% for the 3 hours test set, while the actual WER was 28.5%. For future work, we shall continue our investigation into approaches that can estimate the word error rate using convolutional neural networks. In particular, we would like to explore combining the DNN numerical features with the CNN word embedding features. Another line of research worth exploring is the possibility of building a language independent module for word error rate estimation.

11.2 Future work

Our investigations concerned both dialectal Arabic and modern standard Arabic in the broadcast domain. Future work can apply the sequence-to-sequence modelling, which provides simple and elegant architecture. In such a system, acoustic models, language models, and pronunciation dictionary are mingled in one single network. Chan et al. [2015] introduced the attention-based sequence-to-sequence model, namely listen, attend and spell (LAS) approach, which has showed superior performance compared to other sequence-to-sequence model for a single dialect. Recently, Li et al. [2017] adopted the LAS framework for multi-dialect modelling. The dialect-specific information is incorporated into the model by modifying the training targets by inserting the dialect symbol at the end of the original grapheme sequence and also feeding a one-hot representation of the dialect information into all layers of the model. Similar approach can be applied to multi-dialect Arabic speech recognition.

The variational autoencoder (VAE) has shown to be effective for domain mismatch between training and testing. Hsu et al. [2017] proposed unsupervised domain adaptation for robust speech recognition. Such technique can potentially be used for dialect mismatch between dialect and MSA. Since, there are considerable amount of transcribed data in modern standard Arabic and limited transcribed dialectal data as shown in chapter 7, VAE approach can be used for dialect adaptation in unsupervised way.

Recent advances in CNN and LSTM modelling along with character aware language model [Ling et al., 2015, Kim et al., 2016] seem to be promising approach to deal with the non-orthographic rules in dialectal Arabic for multi dialect Arabic ASR, such as open vocabulary word representation.

Bibliography

Ali Ahmed, Najim Dehak, Patrick Cardinal, Sameer Khurana, Sree Harsha Yella, James Glass, Peter Bell, and Steve Renals. Automatic dialect detection in Arabic broadcast speech. In *Proc. Interspeech*, 2016.

Murat Akbacak, Dimitra Vergyri, Andreas Stolcke, Nicolas Scheffer, and Arindam Mandal. Effective Arabic dialect classification using diverse phonotactic models. In *Proc. Interspeech*, 2011.

Yaser Al-Onaizan and Lidia Mangu. Arabic ASR and MT integration for GALE. In *Proc. ICASSP*, 2007.

Ahmed Ali and Steve Renals. Word error rate estimation for speech recognition: e-WER. In *Proc. NAACL*, 2018.

Ahmed Ali, Hamdy Mubarak, and Stephan Vogel. Advances in dialectal Arabic speech recognition: A study using Twitter to improve Egyptian ASR. In *Proc. International Workshop on Spoken Language Translation (IWSLT)*, 2014a.

Ahmed Ali, Yifan Zhang, Patrick Cardinal, Najim Dahak, Stephan Vogel, and Jim Glass. A complete Kaldi recipe for building Arabic speech recognition systems. In *Proc. IEEE Spoken Language Technology Workshop (SLT)*, 2014b.

Ahmed Ali, Yifan Zhang, and Stephan Vogel. QCRI advanced transcription system (QATS). In *Proc. IEEE Spoken Language Technology Workshop (SLT)*, 2014c.

Ahmed Ali, Walid Magdy, Peter Bell, and Steve Renals. Multi-reference WER for evaluating ASR for languages with no orthographic rules. In *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2015.

Ahmed Ali, Peter Bell, James Glass, Yacine Messaoui, Hamdy Mubarak, Steve Renals, and Yifan Zhang. The MGB-2 challenge: Arabic multi-dialect broadcast media recognition. In *Proc. IEEE Spoken Language Technology Workshop (SLT)*, 2016.

Ahmed Ali, Preslav Nakov, Peter Bell, and Steve Renals. WERd: Using social text spelling variants for evaluating dialectal speech recognition. In *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2017a.

Ahmed Ali, Stephan Vogel, and Steve Renals. Speech recognition challenge in the wild: Arabic MGB-3. In *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2017b.

Khalid Almeman, Mark Lee, and Ali Abdulrahman Almiman. Multi dialect Arabic speech parallel corpora. In *Communications, Signal Processing, and their Applications (ICCSPA)*, pages 1–6. IEEE, 2013.

Eliathamby Ambikairajah, Haizhou Li, Liang Wang, Bo Yin, and Vidhyasaharan Sethu. Language identification: A tutorial. *Circuits and Systems Magazine, IEEE*, 11(2):82–108, 2011.

Tasos Anastasakos, John McDonough, Richard Schwartz, and John Makhoul. A compact model for speaker-adaptive training. In *Proc. ICSLP*, 1996.

Peter Auer. *Code-switching in conversation: Language, interaction and identity*. Routledge, 2013.

El Said Badawi, Michael Carter, and Adrian Gully. *Modern written Arabic: A comprehensive grammar*. Routledge, 2013.

Mohamad Hasan Bahari, Najim Dehak, Lukas Burget, Ahmed Ali, Jim Glass, et al. Non-negative factor analysis for GMM weight adaptation. *IEEE Transactions on Audio Speech and Language Processing*, 2014.

Lalit Bahl, Peter Brown, Peter De Souza, and Robert Mercer. Maximum mutual information estimation of hidden Markov model parameters for speech recognition. In *Proc. ICASSP*, 1986.

Christian Bartz, Tom Herold, Haojin Yang, and Christoph Meinel. Language identification using deep convolutional recurrent neural networks. *arXiv preprint arXiv:1708.04811*, 2017.

Leonard E Baum, Ted Petrie, George Soules, and Norman Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *The annals of mathematical statistics*, 41(1):164–171, 1970.

Peter Bell, MJF Gales, Thomas Hain, Jonathan Kilgour, Pierre Lanchantin, Xunying Liu, Andrew McParland, Steve Renals, Oscar Saz, Mirjam Wester, et al. The MGB challenge: Evaluating multi-genre broadcast media recognition. In *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2015.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3 (Feb):1137–1155, 2003.

Fadi Biadsy. *Automatic dialect and accent recognition and its application to speech recognition*. PhD thesis, Columbia University, 2011.

Fadi Biadsy and Julia Hirschberg. Using prosody and phonotactics in Arabic dialect identification. In *Proc. Interspeech*, 2009.

Fadi Biadsy, Nizar Habash, and Julia Hirschberg. Improving the Arabic pronunciation dictionary for phone and word recognition with linguistically-based pronunciation rules. In *Proc. NAACL*, 2009a.

Fadi Biadsy, Julia Hirschberg, and Nizar Habash. Spoken Arabic dialect identification using phonotactic modeling. In *Proc. EACL*, 2009b.

Fadi Biadsy, Julia Hirschberg, and Michael Collins. Dialect recognition using a phone-GMM-supervector-based SVM kernel. In *Proc. Interspeech*, 2010.

Maximilian Bisani and Hermann Ney. Joint-sequence models for grapheme-to-phoneme conversion. *Speech communication*, 50(5):434–451, 2008.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*, 2016.

Herve A Bourlard and Nelson Morgan. Connectionist speech recognition: A hybrid approach. 1993.

Norbert Braunschweiler, Mark JF Gales, and Sabine Buchholz. Lightly supervised recognition for automatic alignment of large coherent speech recordings. In *Proc. Interspeech*, 2010.

Timothy Buckwalter. Issues in Arabic morphological analysis. *Arabic computational morphology*, pages 23–41, 2007.

Chris Callison-Burch, Philipp Koehn, and Miles Osborne. Improved statistical machine translation using paraphrases. In *Proc. NAACL*, 2006.

Patrick Cardinal, Najim Dehak, Yu Zhang, and James Glass. Speaker adaptation using the i-vector technique for bottleneck features. In *Proc. Interspeech*, 2015.

William Chan, Navdeep Jaitly, Quoc V Le, and Oriol Vinyals. Listen, attend and spell. *arXiv preprint arXiv:1508.01211*, 2015.

Xie Chen, Xunying Liu, Yanmin Qian, MJF Gales, and Philip C Woodland. CUED-RNNLM-An open-source toolkit for efficient training and evaluation of recurrent neural network language models. In *Proc. ICASSP*, 2016a.

Xie Chen, Xunying Liu, Yongqiang Wang, Mark JF Gales, and Philip C Woodland. Efficient training and evaluation of recurrent neural network language models for automatic speech recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(11):2146–2157, 2016b.

Gaofeng Cheng, Vijayaditya Peddinti, Daniel Povey, Vimal Manohar, Sanjeev Khudanpur, and Yonghong Yan. An exploration of dropout with LSTMs. In *Proc. Interspeech*, 2017.

Jan Chorowski, Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. End-to-end continuous speech recognition using attention-based recurrent NN: first results. *arXiv preprint arXiv:1412.1602*, 2014.

Jan K Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio. Attention-based models for speech recognition. In *In Proc. Advances in Neural Information Processing Systems (NIPS)*, 2015.

D Manning Christopher, Raghavan Prabhakar, and SCHÜTZE Hinrich. Introduction to information retrieval. *An Introduction To Information Retrieval*, 151:177, 2008.

Jennifer Chu-Carroll and Bob Carpenter. Vector-based natural language call routing. *Computational linguistics*, 25(3):361–388, 1999.

Alexander Clark, Chris Fox, and Shalom Lappin. *The handbook of computational linguistics and natural language processing*. John Wiley & Sons, 2013.

Stephen Cox and Srinandan Dasmahapatra. High-level approaches to confidence estimation in speech recognition. *IEEE Transactions on Speech and Audio processing*, 10(7):460–471, 2002.

Frederick J Damerau. *Markov models and linguistic theory: an experimental study of a model for English*. Number 95. Mouton De Gruyter, 1971.

Kareem Darwish, Walid Magdy, and Ahmed Mourad. Language processing for Arabic microblog retrieval. In *Proc. CIKM*, 2012.

Kareem Darwish, Hassan Sajjad, and Hamdy Mubarak. Verifiably effective Arabic dialect identification. In *Proc. EMNLP*, 2014.

Marelie H Davel, Charl J van Heerden, and Etienne Barnard. Validating smartphone-collected speech corpora. In *Proc. Spoken Language Technologies for Under-Resourced Languages*, 2012.

Steven Davis and Paul Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE transactions on acoustics, speech, and signal processing*, 28(4):357–366, 1980.

José GC de Souza, Hamed Zamani, Matteo Negri, Marco Turchi, and Daniele Falavigna. Multitask learning for adaptive quality estimation of automatically transcribed utterances. In *Proc. HLT-NAACL*, 2015.

Najim Dehak, Patrick Kenny, Réda Dehak, Pierre Dumouchel, and Pierre Ouellet. Front-end factor analysis for speaker verification. *Audio, Speech, and Language Processing, IEEE Transactions on*, 19(4):788–798, 2011a.

Najim Dehak, Pedro A Torres-Carrasquillo, Douglas A Reynolds, and Reda Dehak. Language recognition via i-vectors and dimensionality reduction. In *Proc. Interspeech*, 2011b.

Li Deng and John Platt. Ensemble deep learning for speech recognition. 2014.

Li Deng, Jinyu Li, Jui-Ting Huang, Kaisheng Yao, Dong Yu, Frank Seide, Michael Seltzer, Geoff Zweig, Xiaodong He, Jason Williams, et al. Recent advances in deep learning for speech research at Microsoft. In *Proc. ICASSP*, 2013.

Mona Diab. Second generation AMIRA tools for Arabic processing: Fast and robust tokenization, POS tagging, and base phrase chunking. 2009.

Frank Diehl, Mark JF Gales, Marcus Tomalin, and Philip C Woodland. Morphological analysis and decomposition for Arabic speech-to-text systems. In *Proc. Interspeech*, 2009.

William B. Dolan and Chris Brockett. Automatically constructing a corpus of sentential paraphrases. In *Proc. Third International Workshop on Paraphrasing (IWP2005)*, 2005.

Harris Drucker, Donghui Wu, and Vladimir N Vapnik. Support vector machines for spam categorization. *Neural Networks, IEEE Transactions on*, 10(5):1048–1054, 1999.

Amr El-Desoky, Christian Gollan, David Rybach, Ralf Schlüter, and Hermann Ney. Investigating the use of morphological decomposition and diacritization for improving Arabic LVCSR. In *Proc. Interspeech*, 2009.

Heba Elfardy and Mona Diab. Token level identification of linguistic code switching. In *Proc. COLING*, 2012.

Heba Elfardy, Mohamed Al-Badrashiny, and Mona Diab. Code switch point detection in Arabic. In *International Conference on Application of Natural Language to Information Systems*, pages 412–416. Springer, 2013.

Heba Elfardy, Mohamed Al-Badrashiny, and Mona Diab. Aida: Identifying code switching in informal Arabic text. *Proc. EMNLP*, 2014.

Mohamed Elmahdy, Mark Hasegawa-Johnson, and Eiman Mustafawi. Development of a tv broadcasts speech recognition system for Qatari Arabic. In *Proc. ELRA*, 2014.

Gunnar Evermann and PC Woodland. Posterior probability decoding, confidence estimation and system combination. In *Proc. Speech Transcription Workshop*, 2000.

Charles Albert Ferguson. Diglossia. *Word-Journal of the International Linguistic Association*, 15(2):325–340, 1959.

Jonathan G Fiscus. A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (ROVER). In *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 1997.

Eibe Frank and Remco Bouckaert. Naive Bayes for text classification with unbalanced classes. *Knowledge Discovery in Databases: PKDD 2006*, pages 503–510, 2006.

Mark Gales and Steve Young. The application of hidden Markov models in speech recognition. *Foundations and trends in signal processing*, 1(3):195–304, 2008.

Mark JF Gales, Frank Diehl, Chandra Kant Raut, Marcus Tomalin, Philip C Woodland, and Kai Yu. Development of a phonetic system for large vocabulary Arabic speech recognition. In *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2007.

J-L Gauvain and Chin-Hui Lee. Maximum a posteriori estimation for multivariate gaussian mixture observations of markov chains. *IEEE transactions on speech and audio processing*, 2(2):291–298, 1994.

Pegah Ghahremani, Bagher BabaAli, Daniel Povey, Korbinian Riedhammer, Jan Trmal, and Sanjeev Khudanpur. A pitch extraction algorithm tuned for automatic speech recognition. In *Proc. ICASSP*, 2014.

Sahar Ghannay, Yannick Esteve, and Nathalie Camelin. Word embeddings combination and neural networks for robustness in ASR error detection. In *Proc. Signal Processing Conference (EUSIPCO)*, 2015.

James Glass, Timothy J Hazen, Scott Cyphers, Igor Malioutov, David Huynh, and Regina Barzilay. Recent progress in the MIT spoken lecture processing project. In *Proc. Interspeech*, 2007.

Raymond G Gordon, Barbara F Grimes, et al. *Ethnologue: Languages of the world*. sil International Dallas, TX, 2009.

Alex Graves and Navdeep Jaitly. Towards end-to-end speech recognition with recurrent neural networks. In *Proc. the 31st International Conference on Machine Learning (ICML-14)*, 2014.

Alex Graves, Navdeep Jaitly, and Abdel-rahman Mohamed. Hybrid speech recognition with deep bidirectional LSTM. In *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2013a.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *Proc. ICASSP*, 2013b.

Frantisek Grézl, Martin Karafiát, Stanislav Kontár, and Jan Cernocky. Probabilistic and bottle-neck features for LVCSR of meetings. In *Proc. ICASSP*, 2007.

Nizar Habash, Owen Rambow, and Ryan Roth. MADA+ TOKAN: A toolkit for Arabic tokenization, diacritization, morphological disambiguation, POS tagging, stemming and lemmatization. 2005.

Nizar Habash, Mona T Diab, and Owen Rambow. Conventional orthography for dialectal Arabic. In *Proc. LREC*, 2012.

Nizar Y Habash. Introduction to Arabic natural language processing. *Synthesis Lectures on Human Language Technologies*, 3(1):1–187, 2010.

Bo Han, Paul Cook, and Timothy Baldwin. Automatically constructing a normalisation dictionary for microblogs. In *Proc. ACL*, 2012.

Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, et al. Deep speech: Scaling up end-to-end speech recognition. *arXiv preprint arXiv:1412.5567*, 2014.

Hany Hassan and Arul Menezes. Social text normalization using contextual graph random walks. In *Proc. ACL*, 2013.

Hynek Hermansky. Perceptual linear predictive (PLP) analysis of speech. *the Journal of the Acoustical Society of America*, 87(4):1738–1752, 1990.

Hynek Hermansky, Daniel PW Ellis, and Sangita Sharma. Tandem connectionist feature extraction for conventional HMM systems. In *Proc. ICASSP*, 2000.

Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N

Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Clive Holes. *Modern Arabic: Structures, functions, and varieties*. Georgetown University Press, 2004.

Wei-Ning Hsu, Yu Zhang, and James Glass. Unsupervised domain adaptation for robust speech recognition via variational autoencoder-based data augmentation. *arXiv preprint arXiv:1707.06265*, 2017.

Po-Sen Huang, Kshitiz Kumar, Chaojun Liu, Yifan Gong, and Li Deng. Predicting speech recognition confidence using deep learning with word identity and score features. In *Proc. ICASSP*, 2013.

Shahab Jalalvand and Daniele Falavigna. Stacked auto-encoder for ASR error detection and word error rate prediction. In *Proc. Interspeech*, 2015.

Shahab Jalalvand, Daniele Falavigna, Marco Matassoni, Piergiorgio Svaizer, and Maurizio Omologo. Boosted acoustic model learning and hypotheses rescoring on the CHiME-3 task. In *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2015a.

Shahab Jalalvand, Matteo Negri, Falavigna Daniele, and Marco Turchi. Driving rover with segment-based ASR quality estimation. In *Proc. ACL*, 2015b.

Shahab Jalalvand, Matteo Negri, Marco Turchi, José GC de Souza, Daniele Falavigna, and Mohammed RH Qwaider. Transcrater: a tool for automatic speech recognition quality estimation. 2016.

Hui Jiang. Confidence measures for speech recognition: A survey. *Speech Communication*, 45(4):455–470, 2005.

Daniel Jurafsky. Speech and language processing: An introduction to natural language processing. *Computational linguistics, and speech recognition*, 2017.

Slava Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE transactions on acoustics, speech, and signal processing*, 35(3):400–401, 1987.

David Kauchak and Regina Barzilay. Paraphrasing for automatic evaluation. In *Proc. NAACL*, 2006.

Sameer Khurana and Ahmed Ali. QCRI advanced transcription system (QATS) for the Arabic multi-dialect broadcast media recognition: MGB-2 challenge. In *Proc. IEEE Spoken Language Technology Workshop (SLT)*, 2016.

Sameer Khurana, Maryam Najafian, Ahmed Ali, Tuka Al Hanai, Yonatan Belinkov, and James Glass. QMDIS: QCRI-MIT advanced dialect identification system. 2017.

Yoon Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. Character-aware neural language models. In *Proc. AAAI*, 2016.

Brian Kingsbury. Lattice-based optimization of sequence classification criteria for neural-network acoustic modeling. In *Proc. ICASSP*, 2009.

Brian Kingsbury, Hagen Soltau, George Saon, Stephen Chu, Hong-Kwang Kuo, Lidia Mangu, Suman Ravuri, Nelson Morgan, and Adam Janin. The IBM 2009 GALE Arabic speech transcription system. In *Proc. ICASSP*, 2011.

Katrin Kirchhoff, Jeff Bilmes, Sourin Das, Nicolae Duta, Melissa Egan, Gang Ji, Feng He, John Henderson, Daben Liu, Mohammed Noamany, et al. Novel approaches to Arabic speech recognition: report from the 2002 Johns Hopkins summer workshop. In *Proc. ICASSP*, 2003.

Reinhard Kneser and Hermann Ney. Improved backing-off for m-gram language modeling. In *Proc. ICASSP*, 1995.

Tom Ko, Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur. Audio augmentation for speech recognition. In *Proc Interspeech*, 2015.

Mary A Kohler and M Kennedy. Language identification using shifted delta cepstra. In *Circuits and Systems, 2002. MWSCAS-2002. The 2002 45th Midwest Symposium on*, volume 3, pages III–69. IEEE, 2002.

Ian Lane, Alex Waibel, Matthias Eck, and Kay Rottmann. Tools for collecting speech corpora via mechanical-turk. In *Proc. NAACL*, 2010.

Christopher J Leggetter and Philip C Woodland. Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models. *Computer Speech & Language*, 9(2):171–185, 1995.

Bo Li, Tara N Sainath, Khe Chai Sim, Michiel Bacchiani, Eugene Weinstein, Patrick Nguyen, Zhifeng Chen, Yonghui Wu, and Kanishka Rao. Multi-dialect speech recognition with a single sequence-to-sequence model. *arXiv preprint arXiv:1712.01541*, 2017.

Haizhou Li, Bin Ma, and Chin-Hui Lee. A Vector Space Modeling Approach to Spoken Language Identification. *IEEE Transactions on Audio, Speech and Language Processing*, 15(1):271–284, 2007. ISSN 1558-7916.

Haizhou Li, Bin Ma, and Kong Aik Lee. Spoken language recognition: from fundamentals to practice. *Proceedings of the IEEE*, 101(5):1136–1159, 2013.

Wang Ling, Tiago Luís, Luís Marujo, Ramón Fernandez Astudillo, Silvio Amir, Chris Dyer, Alan W Black, and Isabel Trancoso. Finding function in form: Compositional character models for open vocabulary word representation. *arXiv preprint arXiv:1508.02096*, 2015.

Gang Liu, Yun Lei, and John HL Hansen. Dialect identification: Impact of differences between read versus spontaneous speech. *EUSIPCO-2010: European Signal Processing Conference, Aalborg, Denmark*, pages 2003–2006, 2010.

Xunying Liu, Xie Chen, Yongqiang Wang, Mark JF Gales, and Philip C Woodland. Two efficient lattice rescoring methods using recurrent neural network language models. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 24(8):1438–1449, 2016.

Will Lowe, Scott Mcdonald, Will Lowe, and Scott Mcdonald. Division of Informatics , University of Edinburgh Institute for Adaptive and Neural Computa-

tion The Direct Route : Mediated Priming in Semantic Space by The Direct Route : Mediated Priming in Semantic Space. (April), 2000.

Mohamed Maamouri, Ann Bies, Tim Buckwalter, and Wigdan Mekki. The penn Arabic treebank: Building a large-scale annotated Arabic corpus. In *Proc. NEMLAR conference on Arabic language resources and tools*, 2004.

Mohamed Maamouri, Ann Bies, Tim Buckwalter, Mona Diab, Nizar Habash, Owen Rambow, and Dalila Tabessi. Developing and using a pilot dialectal Arabic treebank. In *Proc. LREC*, 2006.

Nitin Madnani, Necip Fazil Ayan, Philip Resnik, and Bonnie Dorr. Using paraphrases for parameter tuning in statistical machine translation. In *Proc. of the Second Workshop on Statistical Machine Translation, ACL*, 2007.

Shervin Malmasi, Marcos Zampieri, Nikola Ljubešic, Preslav Nakov, Ahmed Ali, and Jörg Tiedemann. Discriminating between similar languages and Arabic dialect identification: A report on the third DSL shared task. *VarDial 3*, page 1, 2016.

Lidia Mangu, Hong-Kwang Kuo, Stephen Chu, Brian Kingsbury, George Saon, Hagen Soltau, and Fadi Biadsy. The IBM 2011 GALE Arabic speech transcription system. In *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2011.

Vimal Manohar, Daniel Povey, and Sanjeev Khudanpur. JHU Kaldi system for Arabic MGB-3 ASR challenge using diarization, audio-transcript alignment and transfer learning. In *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2017.

Matthew Marge, Satanjeev Banerjee, and Alexander I Rudnicky. Using the amazon mechanical turk for transcription of spoken language. In *Proc. ICASSP*, 2010.

David Martínez, Lukáš Burget, Luciana Ferrer, and Nicolas Scheffer. ivector-based prosodic system for language identification. In *Proc. ICASSP*, 2012.

Pavel Matejka, Petr Schwarz, Jan Cernockỳ, and Pavel Chytil. Phonotactic language identification using high quality phoneme recognition. In *Proc. Interspeech*, 2005.

Pavel Matejka, Le Zhang, Tim Ng, Sri Harish Mallidi, Ondrej Glembek, Jeff Ma, and Zhang Bing. Neural network bottleneck features for language identification. In *Proc. Odyssey*, 2014.

Spyros Matsoukas, Rich Schwartz, Hubert Jin, and Long Nguyen. Practical implementations of speaker-adaptive training. In *DARPA Speech Recognition Workshop*. Citeseer, 1997.

Erik McDermott, Timothy J Hazen, Jonathan Le Roux, Atsushi Nakamura, and Shigeru Katagiri. Discriminative training for large-vocabulary speech recognition using minimum classification error. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(1):203–223, 2007.

Sylvain Meignier and Teva Merlin. LIUM spkdiarization: an open source toolkit for diarization. In *CMU SPUD Workshop*, Dallas (Texas, USA), 2010.

Florian Metze, Roger Hsiao, Qin Jin, Udhyakumar Nallasamy, and Tanja Schultz. The 2010 CMU GALE speech-to-text system. 2010.

Piotr Michalowski. The lives of the Sumerian language. *Margins of writing, origins of cultures*, pages 159–84, 2006.

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. *Proc. Interspeech*, 2010.

Tomáš Mikolov, Anoop Deoras, Daniel Povey, Lukáš Burget, and Jan Černockỳ. Strategies for training large scale neural network language models. In *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2011a.

Tomas Mikolov, Stefan Kombrink, Anoop Deoras, Lukar Burget, and Jan Cernocky. RNNLM-recurrent neural network language modeling toolkit. In *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2011b.

Abdel-rahman Mohamed, Geoffrey Hinton, and Gerald Penn. Understanding how deep belief networks perform acoustic modelling. In *Proc. ICASSP*, 2012.

Mehryar Mohri, Fernando Pereira, and Michael Riley. Weighted finite-state transducers in speech recognition. *Computer Speech & Language*, 16(1):69–88, 2002.

Todd K Moon. The expectation-maximization algorithm. *IEEE Signal processing magazine*, 13(6):47–60, 1996.

Nelson Morgan and Herve Bourlard. Continuous speech recognition. *IEEE signal processing magazine*, 12(3):24–42, 1995.

Yeshwant K Muthusamy, Neena Jain, and Ronald A Cole. Perceptual benchmarks for automatic language identification. In *Proc. ICASSP-94*, 1994.

MMaryam Najafian, Wei-Ning Hsu, Ahmed Ali, and James Glass. Automatic speech recognition of Arabic multi-genre broadcast media. In *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2017.

Preslav Nakov. Improved statistical machine translation using monolingual paraphrases. In *Proc. of the European Conference on Artificial Intelligence (ECAI)*, 2008a.

Preslav Nakov. Improving English-Spanish statistical machine translation: Experiments in domain adaptation, sentence paraphrasing, tokenization, and recasing. In *Proc. of the Third Workshop on Statistical Machine Translation (WMT)*, 2008b.

Preslav Nakov and Hwee Tou Ng. Translating from morphologically complex languages: A paraphrase-based approach. In *Proc. HLT-NAACL*, 2011.

Matteo Negri, Marco Turchi, José GC de Souza, and Daniele Falavigna. Quality estimation for automatic speech recognition. In *Proc. COLING*, 2014.

Raymond WM Ng, Cheung-Chi Leung, Tan Lee, Bin Ma, and Haizhou Li. Prosodic attribute model for spoken language identification. In *Proc. ICASSP*, 2010.

Raymond WM Ng, Kashif Shah, Lucia Specia, and Thomas Hain. A study on the stability and effectiveness of features in quality estimation for spoken language translation. In *Proc. Interspeech*, 2015.

Raymond WM Ng, Kashif Shah, Lucia Specia, and Thomas Hain. Groupwise learning for ASR k-best list reranking in spoken language translation. In *Proc. ICASSP*, 2016.

Michael A Nielsen. Neural networks and deep learning, 2015.

Kamal Nigam, John Lafferty, and Andrew McCallum. Using maximum entropy for text classification. In *Proc. IJCAI workshop on machine learning for information filtering*, 1999.

Scott Novotney and Chris Callison-Burch. Cheap, fast and good enough: Automatic speech recognition with non-expert transcription. In *Proc. HLT-NAACL*, 2010a.

Scott Novotney and Chris Callison-Burch. Shared task: crowdsourced accessibility elicitation of wikipedia articles. In *Proc. HLT-NAACL*, 2010b.

Atsunori Ogawa and Takaaki Hori. ASR error detection and recognition rate estimation using deep bidirectional recurrent neural networks. In *Proc. ICASSP*, 2015.

Atsunori Ogawa, Takaaki Hori, and Atsushi Nakamura. Estimating speech recognition accuracy based on error type classification. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(12):2400–2413, 2016.

Bhavesh Vinod Oswal. CNN-text-classification-keras. `https://github.com/bhaveshoswal/CNN-text-classification-keras`, 2016.

Sebastian Padó and Mirella Lapata. Dependency-Based Construction of Semantic Space Models. *Computational Linguistics*, 33(December 2004):161–199, 2007.

Dimitri Palaz, Ronan Collobert, et al. Analysis of CNN-based speech recognition system using raw speech as input. Technical report, Idiap, 2015.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. In *Proc. ACL*, 2002.

Arfath Pasha, Mohamed Al-Badrashiny, Mona T Diab, Ahmed El Kholy, Ramy Eskander, Nizar Habash, Manoj Pooleery, Owen Rambow, and Ryan Roth. MADAMIRA: A fast, comprehensive tool for morphological analysis and disambiguation of Arabic. In *Proc. LREC*, 2014.

Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur. A time delay neural network architecture for efficient modeling of long temporal contexts. In *Proc. Interspeech*, 2015.

Oldrich Plchot, Mireia Diez, Mehdi Soufifar, and Lukáš Burget. PLLR features in language recognition system for rats. In *Proc. Interspeech*, 2014.

Daniel Povey. *Discriminative training for large vocabulary speech recognition*. PhD thesis.

Daniel Povey and Philip C Woodland. Minimum phone error and I-smoothing for improved discriminative training. In *Proc. ICASSP*, 2002.

Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al. The Kaldi speech recognition toolkit. In *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2011.

Daniel Povey, Xiaohui Zhang, and Sanjeev Khudanpur. Parallel training of deep neural networks with natural gradient and parameter averaging. *arXiv preprint*, 2014.

Daniel Povey, Vijayaditya Peddinti, Daniel Galvez, Pegah Ghahrmani, Vimal Manohar, Xingyu Na, Yiming Wang, and Sanjeev Khudanpur. Purely sequence-trained neural networks for ASR based on lattice-free MMI. 2016.

Lawrence R Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

Juan Ramos. Using TF-IDF to determine word relevance in document queries. In *Proc. of the first instructional conference on machine learning (iCML)*, 2003.

Douglas A Reynolds, William M Campbell, Wade Shen, and Elliot Singer. Automatic language recognition via spectral and token based approaches. In Jacob Benesty, M. Mohan Sondhi, and Yitang Huang, editors, *Springer Handbook of Speech Processing*. Springer, 2008.

Fred Richardson, Douglas Reynolds, and Najim Dehak. A unified deep neural network for speaker and language recognition. In *Proc. Interspeech*, 2015a.

Fred Richardson, Douglas Reynolds, and Najim Dehak. Deep neural network approaches to speaker and language recognition. *IEEE Signal Processing Letters*, 22(10):1671–1675, 2015b.

Brian Roark, Murat Saraclar, and Michael Collins. Discriminative n-gram language modeling. *Computer Speech & Language*, 21(2):373–392, 2007.

Tony Robinson, Jeroen Fransen, David Pye, Jonathan Foote, and Steve Renals. WSJCAMO: a British English speech corpus for large vocabulary continuous speech recognition. In *Pro. ICASSP*, 1995.

Ronald Rosenfeld and Philip Clarkson. CMU-Cambridge statistical language modeling toolkit v2, 1997.

Roni Rosenfeld. A maximum entropy approach to adaptive statistical language modeling. 1996.

Tara N Sainath, Abdel-rahman Mohamed, Brian Kingsbury, and Bhuvana Ramabhadran. Deep convolutional neural networks for LVCSR. In *Proc. ICASSP*, 2013.

Tara N Sainath, Oriol Vinyals, Andrew Senior, and Haşim Sak. Convolutional, long short-term memory, fully connected deep neural networks. In *Proc. ICASSP*, 2015.

Hasim Sak, Andrew W Senior, and Françoise Beaufays. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In *Proc. Interspeech*, 2014.

Haşim Sak, Andrew Senior, Kanishka Rao, and Françoise Beaufays. Fast and accurate recurrent neural network acoustic models for speech recognition. *arXiv preprint arXiv:1507.06947*, 2015.

Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523, 1988.

Gerard Salton, Anita Wong, and Chung-Shu Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975.

George Saon, Hagen Soltau, Upendra Chaudhari, Stephen Chu, Brian Kingsbury, Hong-Kwang Kuo, Lidia Mangu, and Daniel Povey. The IBM 2008 GALE Arabic speech transcription system. In *Proc. ICASSP*, 2010.

George Saon, Hagen Soltau, David Nahamoo, and Michael Picheny. Speaker adaptation of neural network acoustic models using i-vectors. In *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2013.

George Saon, Gakuto Kurata, Tom Sercu, Kartik Audhkhasi, Samuel Thomas, Dimitrios Dimitriadis, Xiaodong Cui, Bhuvana Ramabhadran, Michael Picheny, Lynn-Li Lim, Bergul Roomi, and Phil Hall. English conversational telephone speech recognition by humans and machines. *arXiv preprint arXiv:1703.02136*, 2017.

Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997.

Matthew Stephen Seigel and Philip C Woodland. Combining information sources for confidence estimation with CRF models. In *Prc. Interspeech*, 2011.

Matthew Stephen Seigel and Philip C Woodland. Detecting deletions in ASR output. In *Proc. ICASSP*, 2014.

Andrew Senior and Ignacio Lopez-Moreno. Improving DNN speaker independence with i-vector inputs. In *Proc. ICASSP*, 2014.

Suwon Shon, Ahmed Ali, and James Glass. MIT-QCRI Arabic dialect identification system for the 2017 multi-genre broadcast challenge. In *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2017.

Peter Smit, Siva Gangireddy, Seppo Enarvi, Sami Virpioja, and Mikko Kurimo. Aalto system for the 2017 Arabic multi-genre brodcast challenge. In *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2017.

Temple F Smith and Michael S Waterman. Identification of common molecular subsequences. *Journal of molecular biology*, 147(1), 1981.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. A study of translation edit rate with targeted human annotation.

In *Proc. of association for machine translation in the Americas*, Cambridge, Massachusetts, USA, 2006.

Matthew G. Snover, Nitin Madnani, Bonnie Dorr, and Richard Schwartz. TER-Plus: Paraphrase, semantic, and alignment enhancements to translation edit rate. *Machine Translation*, pages 117–127, 2009.

Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y. Ng. Cheap and fast—but is it good?: Evaluating non-expert annotations for natural language tasks. In *Proc. EMNLP*, 2008.

David Snyder, Daniel Garcia-Romero, Daniel Povey, and Sanjeev Khudanpur. Deep neural network embeddings for text-independent speaker verification. 2017.

Thamar Solorio, Elizabeth Blair, Suraj Maharjan, Steven Bethard, Mona Diab, Abdelati Hawwari, Fahad AlGhamdi, Julia Hirschberg, Alison Chang, et al. Overview for the first shared task on language identification in code-switched data. 2014.

Hagen Soltau, Lidia Mangu, and Fadi Biadsy. From modern standard Arabic to levantine ASR: Leveraging GALE for dialects. In *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2011.

Yan Song, Bing Jiang, YeBo Bao, Si Wei, and Li-Rong Dai. I-vector representation based on bottleneck features for language identification. *Electronics Letters*, 49(24):1569–1570, 2013.

Mehdi Soufifar, Sandro Cumani, Lukáš Burget, and Jan Černocky. Discriminative classifiers for phonotactic language recognition with ivectors. In *Proc. ICASSP*, 2012.

Lucia Specia, Kashif Shah, Jose G.C. de Souza, and Trevor Cohn. QuEst – A translation quality estimation framework. In *Proc. ACL*, 2013.

Richard Sproat and Navdeep Jaitly. RNN approaches to text normalization: A challenge. *arXiv preprint arXiv:1611.00068*, 2016.

Vivek Kumar Rangarajan Sridhar. Unsupervised text normalization using distributed representations of words and phrases. In *Proc. HLT-NAACL*, 2015.

Andreas Stolcke et al. SRILM-an extensible language modeling toolkit.

Hang Su, Gang Li, Dong Yu, and Frank Seide. Error back propagation for sequence training of context-dependent deep networks for conversational speech transcription. In *Proc. ICASSP*, 2013.

Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. LSTM neural networks for language modeling. In *Proc. Interspeech*, 2012.

Yik-Cheung Tam, Yun Lei, Jing Zheng, and Wen Wang. ASR error detection using recurrent neural network language model and complementary ASR. In *Proc. ICASSP*, 2014.

BBN Technologies, (with American University of Beirut a subcontractor), John Makhoul, Bushra Zawaydeh, Frederick Choi, and David Stallard. *BBN/AUB DARPA Babylon Levantine Arabic Speech and Transcripts*. Linguistics Data Consortium, 2005.

Pedro A Torres-Carrasquillo, Elliot Singer, Mary A Kohler, Richard J Greene, Douglas A Reynolds, and John R Deller Jr. Approaches to language identification using gaussian mixture models and shifted delta cepstral features. In *Proc. Interspeech*, 2002.

Zoltán Tüske, Pavel Golik, Ralf Schlüter, and Hermann Ney. Acoustic modeling with deep neural networks using raw time signal for LVCSR. In *Proc. Interspeech*, 2014.

Dimitra Vergyri, Arindam Mandal, Wen Wang, Andreas Stolcke, Jing Zheng, Martin Graciarena, David Rybach, Christian Gollan, Ralf Schlüter, Katrin Kirchhoff, et al. Development of the SRI/Nightingale Arabic ASR system. In *Proc. Interspeech*, 2008.

Karel Veselỳ, Arnab Ghoshal, Lukás Burget, and Daniel Povey. Sequence-discriminative training of deep neural networks. In *Proc. Interspeech*, 2013.

Karel Veselỳ, Baskar Karthick Murali, Mireia Diez, and Karel Beneš. MGB-3 BUT system: Low-resource ASR on Egyptian YouTube data. In *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2017.

Andrew Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE transactions on Information Theory*, 13 (2):260–269, 1967.

Alex Waibel, Toshiyuki Hanazawa, Geoffrey Hinton, Kiyohiro Shikano, and Kevin J Lang. Phoneme recognition using time-delay neural networks. *IEEE transactions on acoustics, speech, and signal processing*, 37(3):328–339, 1989.

Alex Waibel, Petra Geutner, L Mayfield Tomokiyo, Tanja Schultz, and Monika Woszczyna. Multilinguality in speech and spoken language systems. *Proceedings of the IEEE*, 88(8):1297–1313, 2000.

Willie Walker, Paul Lamere, Philip Kwok, Bhiksha Raj, Rita Singh, Evandro Gouvea, Peter Wolf, and Joe Woelfel. Sphinx-4: A flexible open source framework for speech recognition. 2004.

Pidong Wang, Preslav Nakov, and Hwee Tou Ng. Source language adaptation for resource-poor machine translation. In *Proc. EMNLP*, 2012.

Pidong Wang, Preslav Nakov, and Hwee Tou Ng. Source language adaptation approaches for resource-poor machine translation. *Computational Linguistics*, 42(2):277–306, 2016.

Philip C Woodland and Daniel Povey. Large scale discriminative training of hidden markov models for speech recognition. *Computer Speech & Language*, 16(1):25–47, 2002.

Samantha Wray and Ahmed Ali. Crowdsource a little to label a lot: Labeling a speech corpus of dialectal Arabic. In *Proc. Interspeech*, 2015.

Samantha Wray, Hamdy Mubarak, and Ahmed Ali. Best Practices for Crowdsourcing Dialectal Arabic Speech Transcription. In *Proc. of Workshop on Arabic Natural Language Processing, ACL*, 2015.

Samantha Carol Wray. *Decomposability and the effects of morpheme frequency in lexical access*. The University of Arizona, 2016.

Wei Xu and Alexander I Rudnicky. Can artificial neural networks learn language models? 2000.

Xu-Kui Yang, Wen-Lin Zhang, DAN Qu, and Wei-Qiang Zhang. The NDSC-THUEE transcription system for the 2017 multi-genre broadcast challenge. In *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2017.

Omar Zaidan and Chris Callison-Burch. Arabic dialect identification. *Computational Linguistics*, 40.1:171–202, 2014.

Omar F Zaidan and Chris Callison-Burch. Crowdsourcing translation: Professional quality from non-professionals. In *Proc. HLT-ACL*, pages 1220–1229. Association for Computational Linguistics, 2011.

Marcos Zampieri, Shervin Malmasi, Nikola Ljubešić, Preslav Nakov, Ahmed Ali, Jörg Tiedemann, Yves Scherrer, and Noëmi Aepli. Findings of the VarDial evaluation campaign 2017. 2017.

Ruben Zazo, Alicia Lozano-Diez, Javier Gonzalez-Dominguez, Doroteo T Toledano, and Joaquin Gonzalez-Rodriguez. Language identification in short utterances using long short-term memory (LSTM) recurrent neural networks. *PloS one*, 11(1):e0146917, 2016.

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Proc. Advances in neural information processing systems*, 2015.

Jingjing Zhao, Hua Shu, Linjun Zhang, Xiaoyi Wang, Qiyong Gong, and Ping Li. Cortical competition during language discrimination. *NeuroImage*, 43(3): 624–633, 2008.

Liang Zhou, Chin-Yew Lin, and Eduard Hovy. Re-evaluating machine translation results with paraphrase support. In *Proc. EMNLP*, 2006.

Appendix A

MGB-3 Submissions and Results

In this appendix, we highlight some of the major features in the submitted systems in the MGB-3 challenge (introduced in chapter 7). Participants were asked to submit results for the MGB-2 and MGB-3 Arabic test sets. Participants submitted one primary submission and as many contrastive submissions as they wished. We scored and ranked results based on the primary submissions. The test set was manually segmented and only non-overlapping speech was used for scoring.

Aalto University[1]: The novelties of the Aalto ASR system Smit et al. [2017] come from using TDNN-BLSTM acoustic models trained on 1,022 hours filtered from the MGB-2 training data, and adapted using the MGB-3 dialectal Egyptian data. Further improvement came from creating systems using subword and character-based language models (lexicon-free). The final submission was a minimum Bayes risk (MBR)-decoded system combination of over 30 systems using two acoustic models and a variety of language models (character-, subword- and word-based). Aalto achieved the best results 37.5% AV-WER and 29.25% MR-WER.

NDSC-THUEE The NDSC-THUEE system Yang et al. [2017] used a TDNN followed by unidirectional LSTM layers or bidirectional LSTM (BLSTM) layers for the acoustic model. Their overall system makes use of speaker adaptive training, knowledge distillation-based domain adaptation, and MBR for system combination. Finally, they used an RNNLM for rescoring to generate their results. They achieved 40.8% AV-WER and 32.5% MR-WER.

[1] spa.aalto.fi/en/research/research_groups/speech_recognition/

Johns Hopkins University (JHU[2]): The JHU Kaldi system Manohar et al. [2017], Povey et al. [2011] trained seed acoustic models using 982 hours filtered from the MGB-2 training set using speaker diarization and audio-transcript alignment, which was used to prepare lightly supervised transcriptions. They used a TDNN-LSTM acoustic model with a lattice-free (LF) MMI objective followed by segmental MBR (sMBR) discriminative training. For supervision, they fused transcripts from the four independent transcribers into confusion network training graphs. They achieved 40.7% AV-WER and 32.8% MR-WER.

MIT[3]: The MIT system Najafian et al. [2017] used both the MGB-2 and MGB-3 data to train a wide range of acoustic models: DNN, TDNN, LSTM, BLSTM, and prioritized grid LSTM (BPGLSTM) trained using LF-MMI. They used both the Kaldi and the CNTK toolkits. They applied 40 rounds of data augmentation to the MGB-3 data, and combined this with the MGB-2 data for acoustic domain adaptation. They used the full MGB-2 data without data filtering. They achieved 44.9% AV-WER and 36.8% MR-WER.

Brno University of Technology (BUT)[4]: The BUT submission Veselỳ et al. [2017] addressed the task as a low-resource challenge. Their system trained BLSTM-HMM models using 250 hours. They integrated speaker diariazation to improve speaker adaptation. They investigated the integration of the four transcriptions into acoustic model training, by using them serially (including each sentence four times into the training data, once with each transcription). An alternative, parallel, approach consisted of combining all the annotations into a confusion network. They achieved 53.4% AV-WER and 46.8% MR-WER.

Table A.1 summarises the main features of all the submitted systems. We can conclude that the leading teams benefitted from transfer learning and audio adaptation by building background acoustic models using the MGB-2 data and augmenting the five hours of in-domain MGB-3 training data. Also, language modelling approaches, such as lexicon adaptation and higher order n-gram and RNN LM rescoring, also made positive contributions to the overall systems. Only the Aalto team used subword language modeling to deal with the non-orthographic nature of the dialectal speech in the MGB-3 data. Finally, BUT and JHU ex-

[2] `clsp.jhu.edu`
[3] `csail.mit.edu`
[4] `speech.fit.vutbr.cz`

	Aalto	JHU	NDSC-THUEE	BUT	MIT
Used MGB2 data (in hours)	1,022	982	680	250	1,200
MGB3 domain adaptation (transfer learning)	✓	✓	✓	✓	✓
Subword modeling	✓	-	-	-	-
RNNLM rescoring	✓	✓	✓	-	-
Speaker diarization	-	✓	-	✓	-
FST (confusion matrix)	-	✓	-	✓	-
Low-resource	-	-	-	✓	-
AM (NN)					
LSTM	-	-	-	-	✓
BLSTM	-	-	-	✓	✓
BPGLSTM	-	-	-	-	✓
TDNN	✓	-	-	-	✓
TDNN-LSTM	✓	✓	✓	-	-
TDNN-BLSTM	✓	-	✓	-	-

Table A.1: Main features in the submitted systems for Arabic speech-to-text transcription.

	Aalto	JHU	NDSC-THUEE	BUT	MIT
Comedy AV-WER	**51.4**	55.0	54.3	67.7	58.0
Comedy MR-WER	**42.4**	45.7	46.2	61.5	50.0
Cooking AV-WER	**38.2**	43.1	43.8	57.1	46.7
Cooking MR-WER	**30.9**	36.1	37.2	52.0	40.1
FamilyKids AV-WER	**30.6**	35.3	33.9	49.6	38.0
FamilyKids MR-WER	**24.2**	27.7	26.9	44.0	31.3
Fashion AV-WER	40.5	42.2	**40.4**	54.8	45.1
Fashion MR-WER	30.9	31.5	**30.9**	46.9	35.44
Drama AV-WER	**28.7**	32.7	30.7	41.7	34.9
Drama MR-WER	**19.9**	24.2	22.5	34.6	27.0
Science AV-WER	**31.1**	36.6	35.4	48.2	39.4
Science MR-WER	**23.1**	27.7	27.2	41.4	31.6
Sports AV-WER	**45.2**	49.0	48.7	64.2	52.1
Sports MR-WER	**36.0**	39.1	43.9	57.6	42.7
MGB3 AV-WER	**37.5**	**40.7**	**40.7**	**53.4**	**44.9**
MGB3 MR-WER	**29.3**	**32.8**	**32.5**	**46.8**	**36.8**

Table A.2: Error rates (AV-WER and MR-WER over four reference transcriptions) per genre for Arabic speech-to-text transcription for the MGB-3 Egyptian Arabic test set.

plored combining the four transcriptions into a confusion matrix, allowing an alignment process to choose the best transcription.

	WER per transcriber				**Overall**	
	WER1	**WER2**	**WER3**	**WER4**	**AV-WER**	**MR-WER**
Aalto	38.0	37.7	37.4	36.9	**37.5**	**29.3**
NDSC-THUEE	41.5	40.1	40.7	40.8	40.75	32.5
JHU	42.1	42.4	41.4	41.1	40.7	32.8
MIT	45.4	45.4	45.5	44.2	44.9	36.8
BUT	55.0	55.2	54.3	54.4	53.4	46.8

Table A.3: Summary of speech-to-text transcription results for MGB-3 data. WERs are given for each of the four references (produced by different transcribers), as well as average WER (AV-WER) and multi-reference WER (MR-WER) across the four references.

Table A.2 presents the error rates per genre for each of the submitted systems. In this table, we show both AV-WER across the four transcribers per genre, and the MR-WER. The most accurate system is consistently more accurate across all genres (with a small exception for the fashion genre). We also note that the ordering of systems by AV-WER and MR-WER can change, in particular, at higher error rates. For example, results for comedy and science are not consistent between JHU and NDSC-THUEE. The overall ranking is still consistent using the two evaluation metrics. We ranked all the submitted systems with respect to MGB3 AV-WER and MGB3 MR-WER in order. Table A.3 summarises the overall results, sorted by the best results.