Automatic Assessment of Motivational Interview with Diabetes Patients

By

Xizi Wei

A thesis submitted to
the University of Birmingham
for the degree of
DOCTOR OF PHILOSOPHY

Department of Electrical, Electronic and Computer Engineering
University of Birmingham
August 2020

ABSTRACT

Diabetes cost the UK NHS £10 billion each year, and the cost pressure is projected to get worse. Motivational Interviewing (MI) is a goal-driven clinical conversation that seeks to reduce this cost by encouraging patients to take ownership of day-to-day monitoring and medication, whose effectiveness is commonly evaluated against the Motivational Interviewing Treatment Integrity (MITI) manual. Unfortunately, measuring clinicians' MI performance is costly, requiring expert human instructors to ensure the adherence of MITI. Although it is desirable to assess MI in an automated fashion, many challenges still remain due to its complexity.

In this thesis, an automatic system to assess clinicians adherence to the MITI criteria using different spoken language techniques was developed. The system tackled the challenges using automatic speech recognition (ASR), speaker diarisation, topic modelling and clinicians' behaviour code identification.

For ASR, only 8 hours of in-domain MI data are available for training. The experiments with different open-source datasets, for example, WSJCAM0 and AMI, are presented. I have explored adaptative training of the ASR system and also the best training criterion and neural network structure. Over 45 minutes of MI testing data, the best ASR system achieves 43.59% word error rate. The i-vector based diarisation system achieves an F-measure of 0.822. The MITI behaviour code classification system with manual transcriptions achieves an accuracy of 78% for Non Question/Question classification, an accuracy of 80% for Open Question/Closed Question classification and an accuracy of 78% for MI Adherence and MI Non-Adherence classification. Topic modelling was applied to track whether the conversation segments were related to 'diabetes' or not on manual transcriptions as well as ASR outputs. The full automatic assessment system achieve an Assessment Error Rate of 22.54%.

This is the first system that targets the full automation of MI assessment with reasonable performance. In addition, the error analysis from each step is able to guide future research in this area for further improvement and optimisation.

ACKNOWLEDGMENTS

First, I would like to express my sincere gratitude towards my supervisors, Professor Martin Russell and Dr. Peter Jančovič, for their patience, technical expertise and great guidance throughout my PhD. It is Martin, my supervisor for my undergraduate project, who motivated my interest in research and encouraged me to take the challange for pursuing a PhD. Thank you, Martin and Peter.

I would like to thank Claudia Baur, Cathy Chua, Johanna Gerlach, Manny Rayner and Helmer Strik from University of Geneva and Radboud University Nijmegen. It has been a rewarding experience for me to work with them on the 1^{st} and 2^{nd} editions of Spoken CALL shared task.

I would like to thank Melvyn Hunt for his supervision during my internship at Apple Siri group. His insights in phonetics have strengthened my understanding in speech processing techniques. I thank my colleagues, Adrian Skilling, John Bridle, Paul Coles, Siddharth Sigtia, Xiao Zhang, for their kind support and guidance during my internship.

I am also grateful to Professor Mark Gales for his supervision during my research assistant work at the ALTA institute, University of Cambridge. I have learned a lot from the work with him. Also I thank my colleagues form the Machine Intelligence Laboratory, Kate Knill, Yu Wang, Linlin Wang, Xixin Wu and Konstantinos Kyriakopoulos, for the warmth, hospitality and support they offered me.

Many thanks to my colleagues from the speech group in the University of Birmingham, Linxue Bai, Evangelia Fringi, Chloe Seivwright, Maryam Najafian, Roozbeh Nabiei, Yikai Peng, Guy Coop, Philip Weber and especially Mengjie Qian for our collaboration on the shared task competitions. They have made my 4 years' PhD an unforgettable memory.

I would also like to thank my lovely housemate Fan Yang for her years of company and Yunchu Liu for her encouragement when I was lost. A big thank you to all my friends for being in my life.

I thank my parents Xiaoping Wei and Xiaoyan Wu for their support for my PhD and for everything, and also thank my brother Wei Wei. I could never accomplish what I have achieved without a family with full of love.

Finally, I must thank my fiancé Hongyan Xia for being a maintainer of my computer facilities and a tutor for my programming skills.

Table of Contents

Page

1 Introduction 1
- 1.1 Research background 1
- 1.2 Scope of this thesis 3
- 1.3 Thesis outline 4

2 Literature review 6
- 2.1 Introduction 6
- 2.2 Speech processing 9
  - 2.2.1 Mel-frequency cepstral coefficients (MFCC) 9
  - 2.2.2 Robustness to environment and channel effect 10
- 2.3 Gaussian mixture model (GMM) 11
  - 2.3.1 Maximum likelihood parameter estimation for GMM 11
  - 2.3.2 GMM in speaker modelling 12
  - 2.3.3 GMM in automatic speech recognition (ASR) 13
- 2.4 Deep neural network (DNN) 13
  - 2.4.1 Feed-forward deep Neural Network 13
  - 2.4.2 DNN training 14
  - 2.4.3 Recurrent neural network 16
  - 2.4.4 Long short-term memory 17
- 2.5 Automatic speech recognition (ASR) 20
  - 2.5.1 Introduction 20
  - 2.5.2 HMM-based acoustic modeling 21
  - 2.5.3 Training criteria for GMM-HMM ASR 27
  - 2.5.4 Training criteria for DNN-HMM ASR 28
  - 2.5.5 Language model 30
  - 2.5.6 Decoding 31
  - 2.5.7 Adaptation 32

2.6 Speaker diarisation 34
2.6.1 Introduction 34
2.6.2 GMM MAP adaptation 36
2.6.3 Supervector 37
2.6.4 i-vector 37
2.7 Text processing 39
2.7.1 Introduction 39
2.7.2 TF-IDF 40
2.7.3 Word2Vec 41
2.7.4 Doc2Vec 42
2.8 Classification 44
2.8.1 Introduction 44
2.8.2 Support vector machine (SVM) 44
2.8.3 K-Nearest Neighbor (KNN) 46
2.8.4 Word mover distance (WMD) 46
2.8.5 Dynamic time warping (DTW) 49
2.9 Topic modelling 50
2.9.1 Introduction 50
2.9.2 Latent semantic analysis (LSA) 51
2.9.3 Latent Dirichlet allocation (LDA) 52
2.10 Summary 53

3 Motivational Interviews and data 54
3.1 Introduction 54
3.2 MITI guideline 54
3.2.1 Global impression measures 55
3.2.2 Local behavior identification 56
3.3 MI data and annotation 57
3.4 Open-source speech corpora 58
3.4.1 WSJCAM0 corpus 58
3.4.2 AMI meeting corpus 59

4 Automatic speech recognition for Spoken CALL shared task 60
4.1 Introduction 61
4.2 Spoken CALL Shared Task Data 62
4.2.1 Spoken CALL data for 1st edition 62

4.2.2 Spoken CALL data for 2^{nd} edition 62
4.3 Baseline DNN-HMM ASR for 1^{st} edition 63
4.3.1 Front-end speech processing 63
4.3.2 Language model and dictionary 63
4.3.3 GMM-HMM monophone system 63
4.3.4 GMM-HMM triphone system 64
4.3.5 DNN-HMM system 64
4.3.6 Experiments results 65
4.4 Development of ASR for the 2^{nd} edition CALL Shared Task 66
4.4.1 DNN-HMM baseline trained on fMLLR transfromed features 66
4.4.2 Long Short-Term Memory 67
4.4.3 Sequence discriminative training 68
4.5 Summary 69

5 Automatic speech recognition for MI 70
5.1 Introduction 70
5.2 Dictionary and Language Model 71
5.3 Acoustic modeling baseline 71
5.3.1 Triphone-LDA-MLLT 71
5.3.2 Speaker adaptative training 72
5.3.3 DNN-HMM 73
5.3.4 Experiments results 73
5.4 Quality of alignment 74
5.5 fMLLR adaptation 75
5.5.1 fMLLR adaptation to a AMI + MI trained model 75
5.5.2 fMLLR adaptation to a AMI trained model 76
5.5.3 Experiment results 77
5.6 DNN adaptation with Cross-Entropy 78
5.7 DNN adaptation with sequence descriminative training 79
5.8 **Final ASR system** 80
5.9 Summary 81

6 Speaker diarisation 82
6.1 Introduction 82
6.2 GMM i-vector system 85
6.3 DNN i-vector 86

6.4 I-vector clustering and re-segmentation 87
6.4.1 Hierarchical clustering 87
6.4.2 GMM based clustering 88
6.4.3 HMM smoothing - Viterbi decoding 88
6.5 Evaluation criterion 90
6.6 Experiments 91
6.6.1 Clustering and re-segmentation 91
6.6.2 CMN for UBM training 92
6.6.3 T matrix training 93
6.6.4 I-vector Configurations 93
6.6.5 Temporal segmentation 95
6.6.6 DNN-ivector 97
6.6.7 Evaluation of ASR combined with speaker diarisation information . . 98
6.7 Summary 99

7 Topic modeling 100
7.1 Introduction 100
7.2 Global topic vector 101
7.3 Diabetes topic detection 103
7.4 Summary 105

8 Behaviour code classification 106
8.1 Introduction 106
8.2 Text data 108
8.2.1 Data with MITI code 108
8.2.2 Online diabetes data 108
8.3 Text processing 109
8.3.1 TF-IDF 109
8.3.2 In-domian word2vec 109
8.3.3 Generic word2vec 110
8.4 Classification method 110
8.5 Evaluation criterion 111
8.6 Experiments 112
8.6.1 Question vs Non-Question 112
8.6.2 Closed Question vs. Open Question 113
8.6.3 Motivational Interview Adherence vs Motivational Interview Non-Adherence115

8.7 Summary 115

9 Automatic assessment system 117

9.1 Introduction 117

9.2 Semi-automatic assessment system 118

9.3 Full automatic assessment 121

9.4 Summary 124

10 Conclusion 125

10.1 Overview 125

10.2 Contribution 126

10.2.1 ASR system 126

10.2.2 Speaker diarisation system 128

10.2.3 Topic modelling 129

10.2.4 Behaviour code classification 129

10.2.5 Automatic assessment of MI 130

10.3 Future work 130

References 132

List of Figures

1.1 An overview of the automatic assessment system of motivational interview. 5

2.1 Block diagram for Mel-frequency cepstral coefficients (MFCC). 9

2.2 A feed-frward neural network. 14

2.3 A recurrent neural network. 16

2.4 Long short-term memory. The three orange circles f_t, i_t and o_t are forget gate, input gate and output gate, respectively. The black circle stands for element-wise multiplication. 17

2.5 An outline of a brief ASR system 20

2.6 A left-to-right HMM with five states. 22

2.7 Forward and backward probabilities: $\alpha_j(t)$ and $\beta_j(t)$ 24

2.8 DNN-HMM hybird system 27

2.9 Structure of the Skip-gram word2vec model. 41

2.10 Structure of the CBOW word2vec model. 42

2.11 Structure of the PV-DM Doc2Vec model (left) and PV-DBOW model (right). 43

2.12 Top: The outgoing flow from word i in ducument d to d'. Bottom: The incoming flow to word j in ducument d'. 47

2.13 Top: The outgoing flow from word i in ducument d to d'. Bottom: The incoming flow to word j in ducument d'. 50

2.14 A graphical model for LDA topic modelling. 52

3.1 Histogram of the length of the utterances in MI_train. 58

4.1 An overview of the system. 61

5.1 Generation of the baseline features. 72

5.2 Training procedure for Model S4 76

5.3 Training procedure for Model S5 76

5.4 Diagram for the final ASR system. 80

6.1 A flowchart for the i-vector based speaker diarisation. 83

6.2 A diagram for i-vector extraction. 85

6.3 A diagram for incorporating ASR DNN acoustic model into i-vector extraction. 87

6.4 Two state Markov process. 89

6.5 Diagram for the final ASR system. 90

6.6 recall precision and f score with different self-loop probabilities. 92

7.1 Results of LDA-based topic modelling. The global distribution of topics in MI and AMI documents (left) and the most frequent words in the four most frequent topics (right). 102

7.2 DET curve for automatic detection of 'diabetes' topic using the true transcription of MI_test data and ASR output. 104

9.1 Errors analysis made by the M-A-Assess system. 120

9.2 Alignment between the true decision sequence (top) and the A-A-Ass decision sequence (bottom), and three sorts of errors: substitution, deletion and insertion. 122

9.3 Error distribution in system with different level of automation. (a) Error distribution in NQ/Q classification system with manual speaker diarisation and automatic transcription: classification made the most of the errors. (b) Error distribution in NQ/Q classification with fully automatic assessment system: speaker diarisation made the most of the errors. 123

List of Tables

4.1 *Performance of the Spoken CALL Shared Task baseline ASR systems for* 1^{st} *edition on ST1_test, WER(%)* 65

4.2 *Evaluation results (WER%) of Spoken CALL Shared Task DNN-HMM baseline ASR systems for* 2^{nd} *edition, when using different amount of AMI-ihm data* 66

4.3 *Evaluation results (WER%) of Spoken CALL Shared Task LSTM ASR systems for* 2^{nd} *edition, when using different amount of AMI-ihm data and features.* . . 67

4.4 *Evaluation results (WER%) of Spoken CALL Shared Task sMBR ASR systems for* 2^{nd} *edition, when training with different iterations and learning rate* . . . 68

5.1 *Evaluation results on MI_test, WER (%), for acoustic modeling with different training corpora.* 74

5.2 *Evaluation results on MI_test, WER (%), with different alignments* 75

5.3 *Results, WER (%), for MI training with cross-domain fMLLR adaptation.* . 78

5.4 *Recognition performance, WER (%), on MI_test obtained by DNN adaptation with CE training.* 78

5.5 *Recognition performance, WER (%), on MI_test obtained by DNN adaptation with sMBR training.* 79

6.1 *Evaluation of the speaker diarisation systems on MI_test data, with hierarchical clustering* 91

6.2 *Results of the i-vector speaker diarisation systems on the MI_test data, with different CMN level for UBM and T matrix training. Dimension for UBM and i-vector are 512 and 64.* 93

6.3 *Evaluation of the speaker diarisation systems on MI_test data, with different T matrix training. The dimensions of UBM and i-vector are set to be 1024 and 64.* 93

6.4 *Results of the i-vector speaker diarisation systems on the MI_test data, when every utterance from a given speaker was regarded as different speakers for T matrix training. Utterance level CMN was applied for UBM training and session level CMN was applied for T matrix training. Temporal segmentation with 2-second segment and 1.5-second shift.* 94

6.5 *Results of the i-vector speaker diarisation systems on the MI_test data, when every utterance from a given speaker was regarded as different speakers for T matrix training. Utterance level CMN was applied for UBM training and session level CMN was applied for T matrix training. Temporal segmentation with 3-second segment and 2.5-second shift* 95

6.6 *Results of the i-vector speaker diarisation systems on the MI_test data, when every utterance from a given speaker was regarded as different speakers for T matrix training. Utterance level CMN was applied for UBM training and session level CMN was applied for T matrix training. Temporal segmentation with 3-second segment and 1.5-second shift.* 96

6.7 *Evaluation of the speaker diarisation systems on MI_test data, when statistics for i-vector extraction were collected from ASR DNN acoustic models.* 97

6.8 *Recognition performance, WER (%), of the S6 DNN-adapt (see Table 5.4) when speaker diarisation information is employed for fMLLR adaptation.* . . 98

8.1 *MITI behavior code counts from 8 full session.* 108

8.2 *Class accuracies and Binary accuracies of the Question/Non-Question (236/342) classification tasks on manual transcriptions of MI_8.* 113

8.3 *Class accuracies and Binary accuracies of the Closed Question / Open Question classification tasks on manual transcriptions of MI_8.* 114

8.4 *Class accuracies and Binary accuracies of the MIA/MINA (66/51) classification tasks on manual transcriptions of MI_8.* 114

9.1 *Detailed results of the behavior code classification on ASR outputs* 118

Chapter One

Introduction

1.1 Research background

Although diabetes is a physiological disorder, patients also face emotional challenges that can prevent them from the behavioural changes that are needed to achieve self-management. Psychological intervention is playing an increasingly important role as a treatment to address these challenges and to improve biomedical outcomes for diabetes patients (Upsher et al. 2020; Hadjiconstantinou et al. 2020; Chew et al. 2017; Huffman et al. 2015). Motivational Interview (MI) is a type of goal-driven clinical conversation between a clinician and a patient that seeks to facilitate and engage a patient's intrinsic motivation to change behaviour. MI has been found to be more effective than traditional advice-giving in both type 1 and 2 diabetes patient treatment (Alvarado-Martel et al. 2020; Soderlund 2018; Katie et al. 2012) and in many other clinical conversations (Gaume et al. 2009; McCambridge et al. 2011; Woodin, Sotskova, and O'Leary 2011) as well. Proper use of MI has a positive effect on the success of patients' with addiction, obesity and diabetes in acheiving self-management.

The quality of clinician conducting MI is measured by two psychologists through video/audio recording of the sessions. The Motivational Interviewing Treatment Integrity (MITI) manual (Moyers et al. 2016) provides a coding system to measure a clinician's competence and skills. MITI assessment measures both local criteria, such as the clinician's use of open rather than closed questions, and global criteria, such as the *spirit* and *empathy* of the conversation. Specifically, Woodin, Sotskova, and O'Leary (2011) found that a higher proportion of *reflection* and *open* questions had a positive effect on behaviour change. A study has shown that the competence in some psychological techniques could be obtained after training with assessment (Magill et al. 2018). The conventional assessment carried out by two psychologists independently coding of the tape-recordings according to MITI, with discrepancies resolved by an expert, is labour-intensive and costly in terms of time and money, and at present is not possible to fully automate the assessment process.

Recently, there has been some progress towards partially automating the process of assessing some of the MITI components from the speech or the language. In terms of global measurement, Xiao et al. (2015) investigated how speech rate entrainment was related to therapist's *empathy*. The hypothesis was that if there was *empathy* between the clinician and patient then some aspects of their spoken language would start to converge. This convergence is referred to as 'entrainment' and can be used as a measure of *empathy*. Gibson et al. (2016) proposed a deep learning approach to learn the local behaviour code, and subsequently to predict the counsellor's session level *empathy* in the motivation interviews for addiction counselling. Chakravarthula et al. (2019) proposed a neural-network based system for automatic behaviour annotation using speech and language. These research projects relied either on manual segmentation and annotation (Gibson et al. 2016) or automatic speech-text alignment to obtain the timing information of a manual session level transcript (Xiao et al. 2015; Chakravarthula et al. 2019). Unfortunately, they are far from fully automating the assessment process, and manually obtaining these annotations is not only costly but time consuming as well.

Researchers have demonstrated impressive speech recognition performance from deep neural network - hidden markov model (DNN-HMM) automatic speech recognition (ASR) systems (Hinton et al. 2012; Peddinti, Povey, and Khudanpur 2015a). However, performance often suffers considerably when the ASR system is used in natural scenarios. The challange can include multiple factors, such as the limited size of in-domain dataset, the accent of speakers, background noise, the speaking style (read or spontaneous) or the use of different types of microphones. For example, Peddinti, Povey, and Khudanpur (2015a) reported the performance of the state-of-the-art Time-Delayed Deep Neural Network (TDNN) ASR system on different database, and the Word Error Rate (WER) varies from 2.30% to 21.03%. Among the dataset they have been using, the Fisher speech dataset, a collection of 1800 hours of conversational telephone speech, achieves a WER of 21.03% for their TDNN ASR system. Although there are many off-the-shelf ASR systems, the recording condition in clinical conversation is very different from the training data that the existing ASR systems have been trained on, and there are medical terminologies that are rarely used in general speech datasets. The mismatch will degradate performance.

Researches on fully automatic analysis of clinical conversations (Pan et al. 2020; Moore 2015; Mirheidari et al. 2016; Mirheidari et al. 2017) have demonstrated the challenges of applying speech technology to such applications. Mirheidari et al. (2016) presented their work to show the fidelity of the automatic conversation transcription for diagnosing people with dementia. Their best ASR system achieved 40.6% Word Error Rate (WER). Later, they developed a fully automatic diagnosis system by including speaker diarisation (Mirheidari et al. 2017). In Pan et al. (2020), the WER was further decrease to 32.3% with a TDNN ASR system. Their linguistic-based and acoustic-based information extracted from the ASR system have both shown to be effective in detecting dementia.

1.2 Scope of this thesis

The objective of the research in this thesis is to develop and to integrate a set of speech and language technologies to construct an automatic assessment system for MI to assess the clinicains' adherence to MI guildlines. This system can be used to automatically provide feedback for a clinician conducting MI with diabetes patients so as to train the clinicians to obtain better MI skills. Currently most of the MI research were conducted based on manully separating the two speakers in the conversation and manul transcription, this is the first system that integrates speaker diarisation and ASR to fully automate the process, which will significantly improve the efficiency of training the clinicians with better MI skills.

An overview of the automatic assessment system is shown in Fig 1.1. First, the system automatically diarises (who is speaking and when) to separate the clinicians' and patients' speech with the speaker diarisation system. Then automatic speech recognition (ASR) system is applied to transcibe the conversation and finally the clinician's competence with respect to the MITI guidelines are analysed with topic modeling and bahaviour code classification. For example, Nature Language Processing (NLP) technologies are applied to topic traction to evaluate whether the clinician is on topic or not during the whole conversation. Classification technologies on clinicians' utterances automate the process of two independent psychologists manually counting the clinicians' use of behaviour code. The spirit and empathy are two significant general measurement on the clinician's MI skills, curently given by the two psychologists based on a general impression over the whole conversation. Automatically predicting these two scores is out of the scope of this thesis, but the automatic assessment system can be used as a baseline to extract language-based and acoustic-based features to predict spirit and empathy for a given MI session. The techniques in this thesis can also be applied to automate the process of evaluating the patients' self-care behavior and the interaction between two speakers. The error analysis in this thesis also provides a meaningful insight on what is needed for a better assessment system.

1.3 Thesis outline

This thesis will be presented as follows: Chapter 2 provides a literature review on related work and introduces the techniques used for the thesis, including data pre-processing, feature normalisation for ASR and speaker diarisation system, and the methodology for building

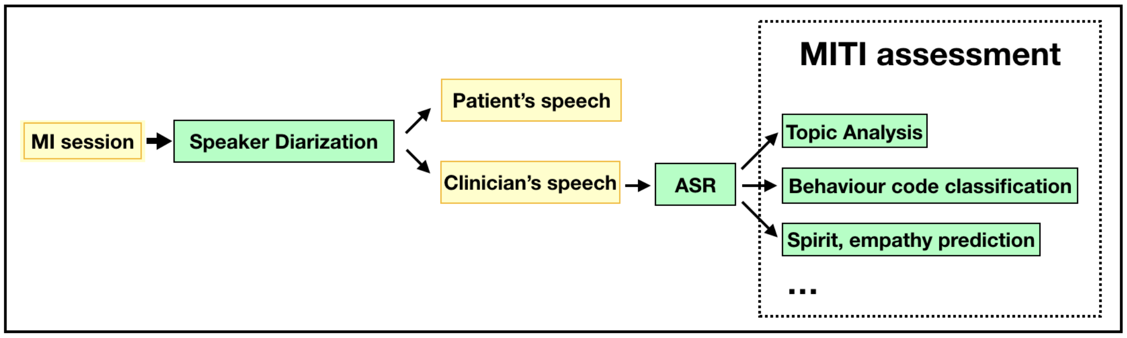


Figure 1.1 An overview of the automatic assessment system of motivational interview.

ASR, speaker diarisation , behavior code classification and topic modelling. Chapter 3 introduces the MI corpus and the MITI guidelines. ASR experiments for Spoken CALL shared task are shown in Chapter 4 and ASR experimental results on MI are presented in Chapter 5. Experiments on acquiring good-quality data alignments are presented in Section 5.4, multi-domain fMLLR adaptation in Section 5.5 and DNN adaptation in Section 5.6 and Section 5.7. Next,the development of the speaker diarisation system to segment the conversation into clinician's and patient's speech is introduced in Chapter 6. The best speaker diarisation system is used to replace the manual speaker information to conduct speaker level normalisation for ASR in Section 6.6.7, and the result outperforms the evaluation with no speaker information. The topic modelling experiments are presented in Chapter 7. Then, the behaviour code classification were explored in Chapter 8. Due to the limited labelled utterances, only classifications for non-question vs. question, open question vs. closed question and MI Adherence and MI Non-Adherence are explored. Finally,the three sub-systems to form a fully automated assessment system are integratedin Chapter 9. Errors from different levels of automation are analysed. Conclution and discussion on the future work based on the error analysis are given in in Chapter 10.

Chapter Two

Literature review

2.1 Introduction

The most widely used tools for assessing the clinicians' fidelity to the MI intervention are the 'motivational interviewing skills code' (MISC) (Miller et al. 2003), and the 'motivational interview treatment integrity' (MITI) (Moyers et al. 2016). The coding includes the counting of the use of local behaviour codes (such as reflection, question, ect.), and global scores based on the general impression over the whole session, such as empathy (details refer to Chapter 3). There are some research efforts towards the automatic assessment of MI based on the linguistic information from the manual transcription. Pérez-Rosas et al. (2017) proposed to use n-grams, semantic and syntactic based features extracted from the clinicians' language, and the linguistic similarity between the clinicians' and clients' speech as inputs to predict reflections and questions for each utterance using Support Vector Machine (SVM) classifier. They have achieved an F score of 0.84 for classifying reflection versus other MITI codes, and an F score of 0.82 for classifying Question versus other MITI codes. Gibson et al. (2015) have used similar language features to predict a global measurement empathy and achieved 75.28% Unweighted Average Recall (UAR). Later Gibson et al. (2016) proposed a deep learning approach and their best model reached a 79.6% UAR. Their proposed system first mapped the vector that represented the language use of the speaker to a target k-hot vector that represented the MISC codes used for each utterance with Recurrent Neural Network (RNN), and then mapped the estimated behaviour code vectors to a session level empathy score. They have found that local behaviour code vectors carries important information for predicting empathy. Then in Gibson et al. (2017), they employed Long Short Term Memory (LSTM) networks with an attention mechanism to predict behaviour codes. Gibson et al. (2019) proposed a multi-label multi-task deep learning approach for automatic behavioural coding.

There are also works towards automating the analysis of clients' behaviour change. For example, Tavabi et al. (2021) developed a neural network to classify the clients' utterances into 3 classes: 3 MISC codes indicating the clients' willness to change. They have used textual features as well as speech features as inputs. The textual features were extracted from the pretrained language model Bidirectional Encoder Representations from Transformers (BERT). The speech feature vector was a representation of prosody. They have shown that for unimodal prediction, textual input outperformed speech input. The textual inputs (what was said) from either clinicians or clients used in all of the researches above still rely on manual transcriptions, automatically transcribing MI to get the vectors to represent their language is yet to be explored.

There are many research efforts to apply DNN-HMM hybrid ASR to clinical conversations for automatic transcriptions. The main challange is that the in-domian speech traning dataset is relatively small compared to most of the open-source dataset for building ASR systems, and the speech is spontaneous and may be interrupted by background noise.

Transfer learning proposed by Ghahremani et al. (2017) has shown to be effective to address this issue, the idea is that the lower-level layers of DNN learn representations of input, which can be pre-trained on a larger dataset and then only higher-level layers or the whole network are retrained for a different task with a smaller in-domian dataset. Transfer learning has been applied to many other tasks to train the ASR system for natural scenarios, which is refered to as acoustic model adaptation in some publications. Pan et al. (2020)'s work on automatic detection of dementia have applied transfer learning to a base TDNN ASR system with their limited in-domian dataset and achieved a WER of 32.3%. Takashima, Takiguchi, and Ariki (2020) developed ASR systems for dysarthric speakers by adapting a baseline non-dysarthric model to the speech of target speakers, and they have shown significant absolute WER reduction (varies from 11.3% to 19.17%) on different target dysarthric speakers. A variety of ways have been explored to adapt a pre-trained ASR model to specific conditions, e.g., Fakhan and Arisoy (2020), Srinivasamurthy et al. (2017), Qian et al. (2017), Yao et al. (2012), and Woodland et al. (2015), but this is still a major ongoing ASR research topic.

For the hybrid DNN-HMM ASR system, a GMM-HMM system is usually needed to provide alignments that align each audio frame to a possible target. The adaptation in the research mentioned above only focused on the training of neural network, the input features and alignments were unchanged during the transfer learning. This thesis explored speaker adaptation for GMM-HMM system (ref to Section 2.5.7) to obtain better input features and alignments. The effect of adapted input features and quality of alignments on the performace of the DNN-HMM ASR system were also explored in this thesis.

In the following sections of this chapter, state-of-the-art speech and language processing techniques that are used for the automatic assessment system of motivation interview in the experimental chapters will be introduced with more details.

2.2 Speech processing

In the first stage of the automatic assessment, the acoustic signal from each MI session is converted to a sequence of acoustic feature vectors, this is called 'front-end analysis'. The front-end analysis for ASR should preserve all the phonetic distinctions that used by human listerners as much as possible while not sensitive to acoustic variations or distortion caused by, for example, different speakers, environments, transmission methods. For speaker diarisation, the feature vectors should be able to capture the characteristics of different speakers (different from ASR where the speaker information are to be suppressed). Moreover, the feature vectors should match with the distributional assumptions made by the acoustic models. For example, if diagonal covariance Gaussian distributions are used for the state-output distributions then the features should be designed to be Gaussian and uncorrelated. Cepstrum-based features, like mel-frequency cepstral coefficients (MFCCs) have proved to be effective for both ASR and speaker diarisation. In ASR, there are many successful speaker normalisation techniques to remove the variations between different speakers. Details on speaker adaptation techniques are introduced in Section 2.5.7.

2.2.1 Mel-frequency cepstral coefficients (MFCC)

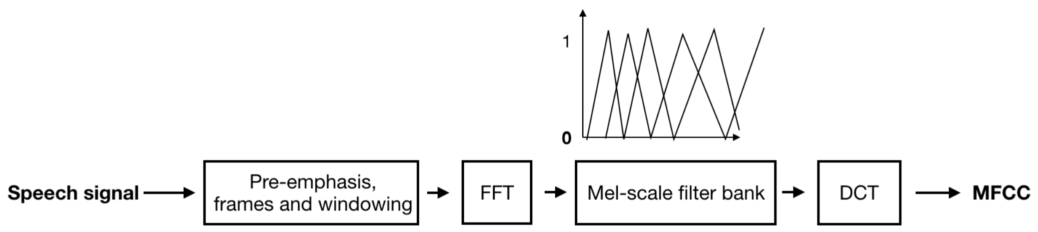


Figure 2.1 Block diagram for Mel-frequency cepstral coefficients (MFCC).

Figure 2.1 shows a block diagram for MFCC process. The vocal tract shape tends to be constant over short intervals (around 10-30ms), thus frames are taken every 10ms (giving a frame rate of 100 frame/s) using an overlapping 25ms window (usually a Hamming or Hanning window). First, fast Fourier analysis (FFT) is applied to the window of speech. Since in a source-filter model, speech can be seen as a convolution of the excitation signal and a vocal tract filter in time domain, which is equivalent to the product in spectral domain after FFT. After taking logarithms of the spectrum, the product components became additive. This results in a representation of the spectrum where it is possible to separate the influences of the vocal tract filter (encoded in the low-order cepstral coefficients) and the excitation (encoded in the high-order cepstral coefficients).

The perception of the human ear against sound frequency does not follow the linear scale, it pay more attention to certain frequency components. This can be modeled using a bank of mel scale filters to compute weighted averages of the spectral components. The following formula is applied to convert frequency f into Mel scale:

$$M(f) = 1125ln(1 + \frac{f}{700}) \tag{2.1}$$

Thus the features in this scale match more closely to human hearing. MFCCs are then generated by applying a discrete cosine transform (DCT). The variation of the separate coefficients tend to be uncorrelated, which helps to improve the diagonal covariance approximation. Typically the first 12-18 MFCCs and its' time derivatives are used for ASR or speaker diarisation.

2.2.2 Robustness to environment and channel effect

In real speech processing applications, the speech may be corrupted with additive noise (external noise present in the signal) or convolutional noise (spectral distortion during transmission from the talkers' mouth to the speech recognizer).

The additive noise can be dealt with in the linear spectral domain, since the noise's spectral components can be seen as being added to the speech components. The convolutional noise is easier to cope with in the log spectral domain, since the convolution in the time domain becomes one of addition in the log spectral domain. This kind of noise tends to be constant over a reasonably long utterance, thus simply subtracting the mean feature vector can effectively remove the convolutional distortion without removing useful speech information. This method is known as cepstral mean normalization (CMN).

2.3 Gaussian mixture model (GMM)

A GMM is a weighted sum of M Gaussian components. The probability density for each component i is a normal distribution with mean vector μ_i and covariance matrix Σ_i. The mixture probability density is given by

$$p(x) = \sum_{i=1}^{M} w_i \mathcal{N}(x; \mu_i, \Sigma_i) \tag{2.2}$$

where w_i is the weight of each component, and $\sum_i^M w_i = 1$, also $0 \leq w_i \leq 1$. The parameter set for a GMM is denoted by λ:

$$\lambda = \{w_i, \mu_i, \Sigma_i\} \; i = 1, ..., M. \tag{2.3}$$

The mixture of Gaussian components makes GMM a powerful tool to model arbitrary densities.

2.3.1 Maximum likelihood parameter estimation for GMM

When a GMM is used to model a particular pattern, the parameter set λ is estimated to best match the distribution of feature vectors. The maximum likelihood (ML) estimation is to find the parameters λ that likelihood of training data being generated by the GMM is maximized.

For a sequence of T training vectors $X = x_1, ..., x_T$, the GMM log likelihood is computed by:

$$logP(X|\lambda) = \sum_{t=1}^{T} logp(x_t|\lambda) \tag{2.4}$$

The log likelihood is usually divided by T to normalize the effect of sequence length. Because the training data is unlabelled, the feature vector x_t at time t being generated by which component is unknown. It's not possible to maximise the likelihood directly, expectation-maximization (EM) algorithm (Dempster, Laird, and Rubin 1977) is applied to achieve ML estimation iteratively.

In the Expectation-step, one has to learn the latent variable from a rough estimate of the parameters. In the M-step, the learned values is used to update the estimate that gives an improved model. In the case of GMM, the component occupancy γ_{it} (the probability of using component i to generate feature vector x_t at time t) is calculated with the current values of w_i, μ_i, Σ_i in the Expectation-step (E-step) by:

$$\gamma_{it} = \frac{w_i \mathcal{N}(x_t; \mu_i, \Sigma_i)}{\sum_{j=1}^{M} w_j \mathcal{N}(x_t; \mu_j, \Sigma_j)} \tag{2.5}$$

Then the new estimates of parameters of the i^{th} GMM component $[\bar{w_i}, \bar{\mu_i}, \bar{\Sigma_i}]$ are given in the Maximization-step (M-step) as below:

$$\bar{\mu_i} = \frac{\sum_{t=1}^{T} \gamma_{it} x_t}{\sum_{t=1}^{T} \gamma_{jt}} \tag{2.6}$$

$$\bar{\Sigma_i} = \frac{\sum_{t=1}^{T} \gamma_{it} (x_t - \mu_i)(x_t - \mu_i)^T}{\sum_{t=1}^{T} \gamma_{it}} \tag{2.7}$$

$$\bar{w_i} = \frac{1}{T} \sum_{t=1}^{T} \gamma_{it} \tag{2.8}$$

The E-M steps are executed successively until the data likelihood reaches a local maximum.

2.3.2 GMM in speaker modelling

A speaker's voice can be characterized by a set of acoustic classes representing some phonetic sounds events like vowels, nasals or fricatives. The mean vector of each Gaussian component corresponding to clusters of feature vectors, which intuitively correspond to phone categories or sub-categories. Thus GMM have become the dominant approach in text-independent speaker modeling for many years. More details about the application of GMM in speaker modeling will be introduced in Section 2.6.

2.3.3 GMM in automatic speech recognition (ASR)

The distribution of the speech features that derived from the same part of a same word by a same person in the same way can be expected to have a mode with deviation. Thus it is reasonable to model this distribution with Gaussian distribution. When different speaker are taken into account in a ASR system, the distribution can be multi-modal. Thus GMM is widely used in HMM based ASR as the state emitting probability distribution.

2.4 Deep neural network (DNN)

2.4.1 Feed-forward deep Neural Network

A feed-forward DNN is an emulation to the biological neural systems in animal brains. A topology of a feed-forward DNN with a stack of fully-connected $L+1$ layers is illustracted in Figure 2.2. It consists of a input layer ($l=0$), a number of hidden layers($0<l<L$) and an output layer ($l=L$). The feature vector is forward propagated from the input layer to the output layer, and the output layer represents the target of the DNN.

The input to a hidden layer l, the excitation vector z^l, is the result of the output from the previous layer v^{l-1} times weight matrix W^l and add bias b^l:

$$z^l = W^l v^{l-1} + b^l, \quad \text{for } 0 < l \leq \ L \tag{2.9}$$

The output of each hidden layers, the activation vector v^l, is obtained by activating the excitation vector z^l using an activation function $f()$:

$$v^l = f(z^l) \tag{2.10}$$

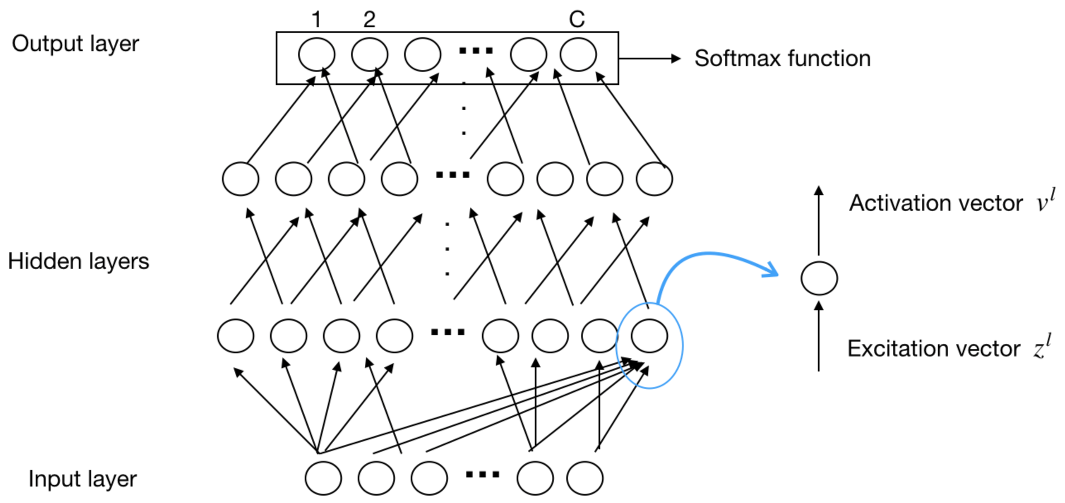


Figure 2.2 A feed-frward neural network.

The activation function on a hidden units performs like a “switch”, and it add non-linearty to the network. Without activation function, the DNN is just consisting of multiple affine transformations, which is euqaliant to a single affine transformation. Sigmoid function is often used as the activation function, thus:

$$v^l = \sigma(z^l) = \frac{1}{1+e^{-z^l}}, \quad \text{for } 0 < l < L \tag{2.11}$$

A softmax function is often used to normalize the output units:

$$v_i^L = softmax_i(z^L) = \frac{e^{z_i^L}}{\sum_{j=1}^{C} e^{z_j^L}} \tag{2.12}$$

If every unit at the output layer is represented a class, the v_i^L with softmax can be viewed as the posterior probability that the observation vectors belongs to class i.

2.4.2 DNN training

The DNN model parameters $\{W, b\}$ are estimated using error back-propagation algorithm (Rumelhart, Hinton, and Williams 1986) with a training set, $S = \{(O^m, y^m)|0 \leq m < M\}$, where M is number of samples, O^m is the m^{th} feature vector and y^m is the target output vector. The discrepancy between the target outputs and the outputs from the model can be measured using different cirteria as introduced in Section 2.5.4 for DNN training in ASR. Then the descrepancy is sent backwards. By calculating the derivative for each weight, the weights can be updated to make the model outputs closer to the target outputs.

Generative pre-training

Instead of initialising the parameters with small random values, a generative model Restricted Boltzmann Machine (RBM) is introduced to initialize each layer of DNN (Hinton, Osindero, and Teh 2006) to model the structure in the input data. No information about the targets is using at the stage of RBMs training. A Deep Belief Network (DBN) is a stack of RBMs, and it is used as the pre-training for DNN, which provides a better weight initialisation. A DBN-DNN is created by adding a softmax layer onto the DBN structure as the output layer. Then the error back propogation is used to fine-tune the weights of the whole network. (Hinton et al. 2012)

A RBM is a special type of Markov random field that has one layer of hidden units h and one layer of visible units v, and all visible units are connected to all hidden units with no lateral interactions. The joint probability $P(v, h)$ is determined by an energy function as below:

$$P(v,h) = \frac{e^{-E(v,h)}}{\sum_{v,h} e^{-E(v,h)}}, \tag{2.13}$$

and the energy is determined by the RBM weights W_{rbm}, visible layer bias a_{rbm}, and hidden layer b_{rbm} linearly. In the case of a Bernoulli-Bernoulli RBM:

$$E(v,h) = -a_{rbm}^T v - b_{rbm}^T h - h^T W_{rbm} v \tag{2.14}$$

As hidden layer nodes take binary values, the conditional probability of h given v is as below:

$$P(h_i = 1|v) = \frac{1}{1 + e^{-(W_{rbm,i}^T v + b_{rbm,i})}} \tag{2.15}$$

It has the same form as the logistic sigmoid activation function, so the weights of the RBM can be used to initialise a hidden layer of DNN with sigmoid activation function.

Regularisation- early stopping

During the fine-tuning, over-fitting is when 'loss' (the discrepancy between the targets and DNN outputs) on training dataset consistently decreases, but the loss on a 'held-out' validation dataset increases in later interations. To prevent neural networks from over-fitting to the training data one popular way is to apply early stopping. The parameters are trained only base on the training data. The training process terminates when the validation loss begins to increase or the loss on validation dataset is below some threshold.

2.4.3 Recurrent neural network

The conventional feed-forward DNN only uses fixed-length input layer and output layer. Sequential data like speech data often has flexible length of data. Recurrent neural network (RNN) (Williams and Zipser 1989) is introduced to resolve this issue. A RNN with one hidden layer is illustrated in Figure 2.3. The delayed activation vector v_{t-1} at time $t-1$,

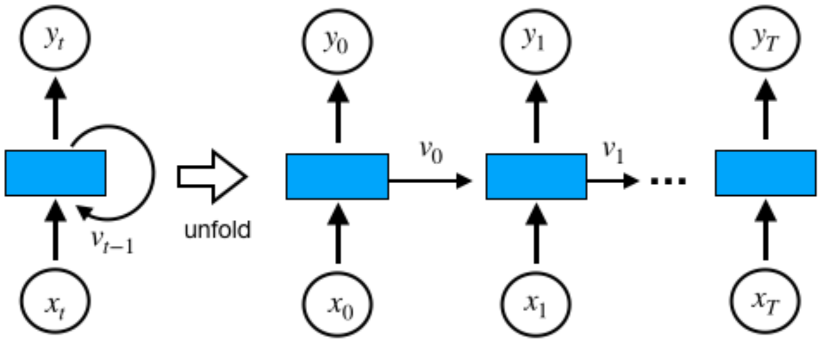


Figure 2.3 A recurrent neural network.

i.e. the hidden vector h_{t-1}, is incorporated to the excitation vector z_t to the hidden layer at time t as below:

$$z_t = W^T x_t + R^T v_{t-1} + b \tag{2.16}$$

$$= W^T x_t + R^T h_{t-1} + b \tag{2.17}$$

where R is the weight matrix for the recurrent connections. Thus the output of the hidden layer contains information from the current feature vectors and the previous feaure vectors. Time delayed error back propogation is applied to train the parameters of RNN.

2.4.4 Long short-term memory

Conventional LSTM

One challenge in training RNN is the vanishing gradients problem. The magnitudes of gradients turn to be very small in long sequences. (Bengio, Simard, and Frasconi 1994) Long short-term memory (LSTM) (Hochreiter and Schmidhuber 1997) is introducted to address this issue. LSTM is a special case of RNN, capable of avoid long-term dependency problem. A diagram of LSTM with one hidden layer is shown in Figure 4.3. LSTM uses cell state and three gates to effectively control history information over a long period.

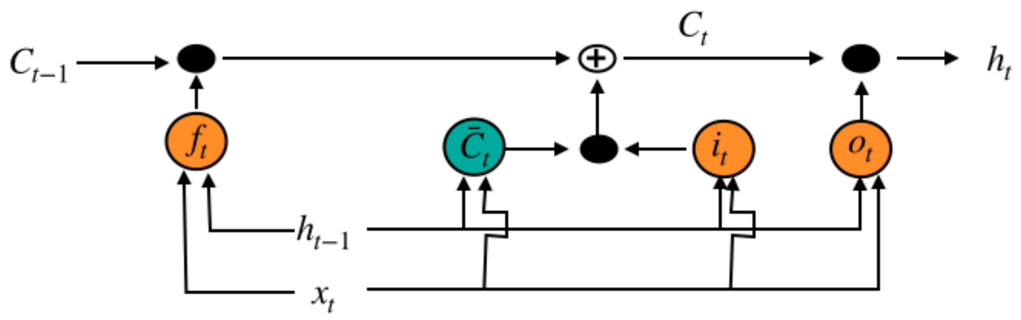


Figure 2.4 Long short-term memory. The three orange circles f_t, i_t and o_t are forget gate, input gate and output gate, respectively. The black circle stands for element-wise multiplication.

The gate is usually a sigmoid function, it ouputs numbers between zero and one to determine how much of each component should be perserved. The three gates are forget gate f_t, input gate i_t and output gate o_t.

The forget gate is to decide which information are to be throw away from the cell state c_{t-1}:

$$f_t = \sigma(W_f^T x_t + R_f^T h_{t-1} + b_f) \tag{2.18}$$

The input gate decide which value to be update:

$$i_t = \sigma(W_i^T x_t + R_i^T h_{t-1} + b_i) \tag{2.19}$$

$\bar{C}_t$ denote the new candidate values that could be added to the cell state:

$$\bar{C}_t = tanh(W_C^T x_t + R_C^T h_{t-1} + b_C) \tag{2.20}$$

Then the new cell state C_t can be update by forgetting with $f_t * C_{t-1}$ and adding with $i_t * \bar{C}_t$:

$$C_t = f_t * C_{t-1} + i_t * \bar{C}_t \tag{2.21}$$

where $*$ stands for element-wise multiplication. The output gate is to decide what to output:

$$o_t = \sigma(W_o^T x_t + R_o^T h_{t-1} + b_o) \tag{2.22}$$

$$h_t = o_t * tanh(C_t) \tag{2.23}$$

Variants on LSTM

Based on the conventional LSTM described above, there are many different LSTM version. For example, peephole connnections are adding to the LSTM by Gers and Schmidhuber (2000). Adding peepholes to all the gates means that the gates are decided by the cell states as well. The input gate, forget gate and ouput gate are now computed as below:

$$i_t = \sigma(W_{ix}^T x_t + W_{ih}^T h_{t-1} + W_{ic}^T C_{t-1} + b_f) \tag{2.24}$$

$$f_t = \sigma(W_{fx}^T x_t + W_{fh}^T h_{t-1} + W_{fc}^T C_{t-1} + b_f) \tag{2.25}$$

$$o_t = \sigma(W_{ox}^T x_t + W_{oh}^T h_{t-1} + W_{oc}^T C_t + b_f) \tag{2.26}$$

In order to reduce the total number of paramters to be trained for LSTM, based on the LSTM with peepholes described above, Sak, Senior, and Beaufays (2014) introduce a recurrent projection layer and an optinal non-recurrent projection layer. The projection layers contains less units than the hidden layer. Instead of directly connect the cell state outputs h_t to the gates and cell input ($\bar{C}_t$), h_t is projected to a recurrent projection layer r_t and an non recurrent projection layer p_t, the r_t is connected to the gates and cell input. Thus the mapping from an input sequence $x = x_1, ..., x_T$ to an output sequence $y = y_1, ..., y_T$ in the LSTM is computed as below:

$$i_t = \sigma(W_{ix}^T x_t + W_{ir}^T r_{t-1} + W_{ic}^T C_{t-1} + b_f) \tag{2.27}$$

$$f_t = \sigma(W_{fx}^T x_t + W_{fr}^T r_{t-1} + W_{fc}^T C_{t-1} + b_f) \tag{2.28}$$

$$o_t = \sigma(W_{ox}^T x_t + W_{or}^T r_{t-1} + W_{oc}^T C_t + b_f) \tag{2.29}$$

$$\bar{C}_t = tanh(W_C^T x_t + R_C^T r_{t-1} + b_C) \tag{2.30}$$

$$C_t = f_t * C_{t-1} + i_t * \bar{C}_t \tag{2.31}$$

$$h_t = o_t * tanh(C_t) \tag{2.32}$$

$$r_t = W_{rh} h_t \tag{2.33}$$

$$p_t = W_{ph} h_t \tag{2.34}$$

$$y_t = W_{yr} r_t + W_{yt} p_t + b_y \tag{2.35}$$

2.5 Automatic speech recognition (ASR)

2.5.1 Introduction

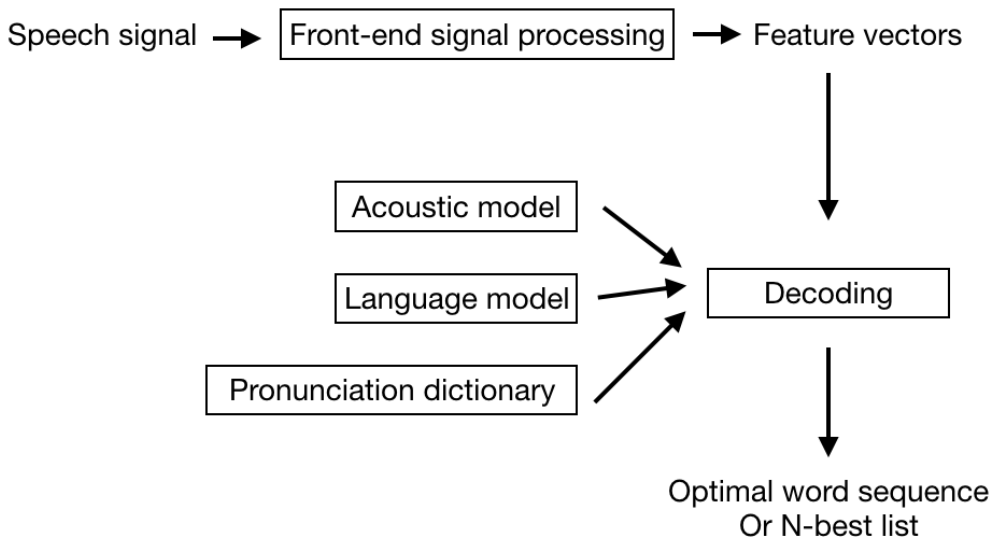


Figure 2.5 An outline of a brief ASR system

As shown in Figure 2.5, automatic speech recognition for transcription is a process of finding the best matching word sequence for an input audio waveform. The audio waveform is converted into a sequence of acoustic feature vectors Y. For the simplest case, if each word is spoken separately and the boundary of the word are known, finding the best matching word is to find the one whose model most likely to generate the observed sequence of feature vectors. The basic unit of sound is phone, thus the acoustic model for a word is obtained by concatenating phone models as defined by a pronunciation dictionary. For speech transcription, the task is to find a word sequence W, and it can be derive from the one with the maximum probability $P(W|Y)$ among all word sequences W as follows:

$$\hat{W} = \underset{W}{argmax} P(W|Y) \tag{2.36}$$

$P(W|Y)$ can be modeled directly based on descriminative models, also known as end-to-end models, such as connectionist temporal classification (CTC) (Graves et al. 2006), encoder-decoder model (Cho et al. 2014) and attention-based model (Chorowski et al. 2015).

Alternatively, $P(W|Y)$ can also be transformed according to Bayes' rule:

$$P(W|Y) = \frac{P(Y|W)P(W)}{P(Y)} \tag{2.37}$$

Then the most probable W is the one which maximizes the product of $P(Y|W)$ and $P(W)$ ($P(Y)$ can be ignored because it does not affect the choice of word):

$$\hat{W} = \underset{W}{argmax} P(Y|W)P(W) \tag{2.38}$$

$P(W)$ is given by a language model (typically a N-gram) that provides the probability of the word given its $N-1$ predecessors, and it can be trained with a text corpus independent of the acoustic information. $P(Y|W)$ is given by the a generative acoustic model, such as Hidden Markov Model (HMM). HMM based acoustic modeling is the standard and contributes to most of the successful speech recognition systems (Hinton et al. 2012; Peddinti, Povey, and Khudanpur 2015a; Chai et al. 2021) to date. Especially in low-resource data scenario (Fakhan and Arisoy 2020). More details about HMM based acoustic modeling will be presented in Sec 2.5.2. More details about language model is given in Sec 2.5.5

2.5.2 HMM-based acoustic modeling

As mentioned before, the word model w can be composed by concatenating phone models. Each phone is modeled by a standard left-to-right five states (with two non-emitting state) HMM as illustrated in Figure 2.6. Two types of parameters are associated with this HMM model:

- state transition probability: a_{ij}, the probability of the transition from state s_i to state s_j.

$$\sum_{j=1}^{N} a_{ij} = 1 \tag{2.39}$$

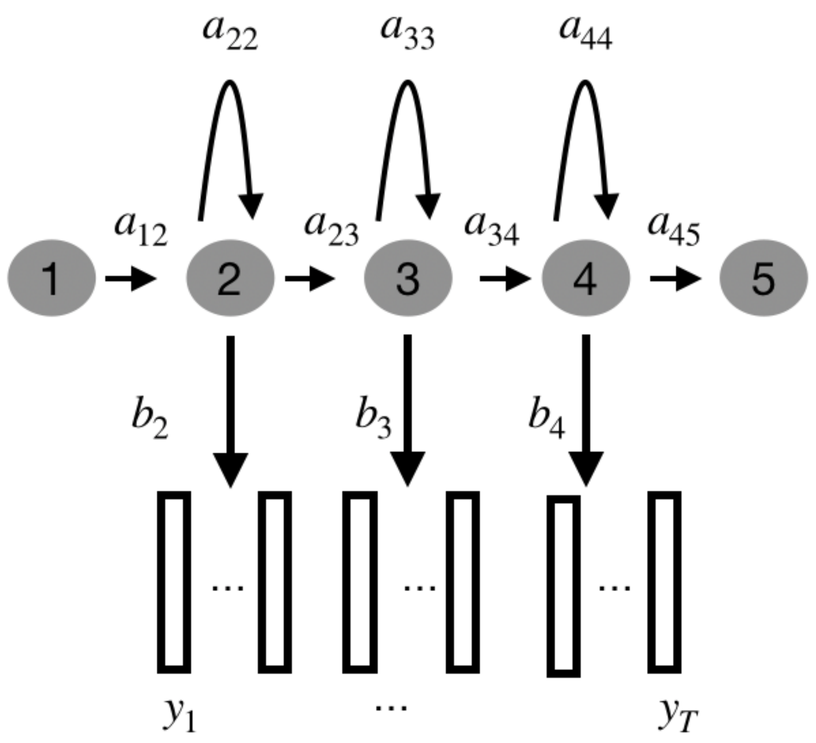


Figure 2.6 A left-to-right HMM with five states.

- state emitting probability: $b_j(y)$, the density PDF generating a feature vector from state j.

For every time step t, the feature vector y_t is observed, but the state s_t on which it depends is unobserved. This form of HMM based on the assumption that:

- Observations y_t only depends on the current state s_t.
- The transition probability to state s_{t+1} only depends on the current state s_t

Thus the acoustic likelihood is given by:

$$p(Y|w) \simeq \sum_S (\prod_{t=1}^{T} b_{s_t} a_{s_{t-1}s_t}) \tag{2.40}$$

Where $S = s_1, s_2, ..., s_T$ is a sequence of the hidden states though the composite model w, the summation goes through every possible state sequence. The parameters $\lambda = \{a_{ij}, b_{s_j}\}$ are to be estimated from a training corpus using expectation-maximisation(EM) with Maximum Likelihood (ML) training cirteria or discriminative training criteria. The state emitting probability can be modeled using a GMM (Section 2.3) or with DNN (Section 2.4), denoted as GMM-HMM and DNN-HMM, respectively. Details about GMM-HMM and DNN-HMM will be introduced later in this section.

Parameter estimation for HMM

Starting from an initial estimate of the parameters λ^0, the basic idea for HMM-based acoustic model training with successive iterations of EM is as follow:

- **E-step** For any given utterance r in the training corpus, corresponding composite HMM is constructed by concatenating the phone model according to the phone-level transcription.

  The forward-backward algorithm (also known as Baum-Welch algorithm) is applied to compute the probability of the model occupying state s_j at time t, so-called occupation probabilities $\gamma_j(t)$. The forward probability $\alpha_j(t)$ and the backward probability $\beta_j(t)$ are calculated as shown in Figure 2.7. The probability of the model occupying state s_j at time t and that the whole of the utterance is aligned with the composite HMM (or something similar) is given by the normalized product of the $\alpha_j(t)\beta_j(t)$ as below:

$$\gamma_j(t) = \frac{\alpha_j(t)\beta_j(t)}{\sum_{i=1}^{N} \alpha_i(t)\beta_i(t)} \tag{2.41}$$

  In contrast to the forward-backward algorithm takes all possible path into account, Viterbi training uses the most likely path. The most likely path is found by calculating the value of $\alpha_j(t)$ for all states and frames, keep track of the most probable state for each frame and tracing back from the final state. The resulting path is also used as alignment that align the audio to the reference trancripts with the most current acoustic model.

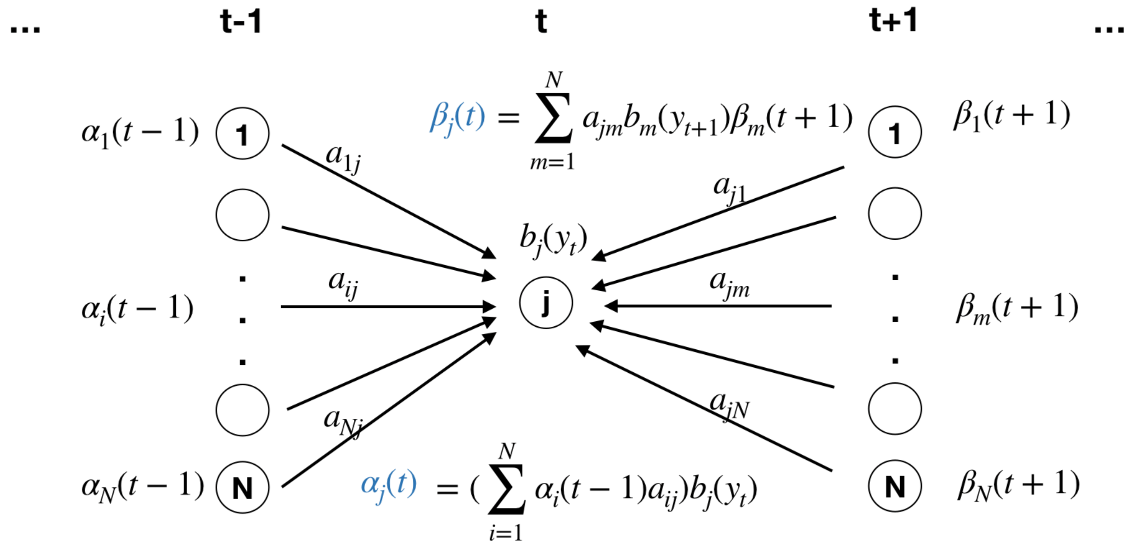


Figure 2.7 Forward and backward probabilities: $\alpha_j(t)$ and $\beta_j(t)$

- **M-step** ML criteria that maximising the likelihood of the training data or discriminative training criteria that minimising the error rate (for GMM-HMM in Section 2.5.3 and for DNN-HMM in Section 2.5.4) is applied to re-estimate transition probabilities and the parameters of the emitting probability distribution, given these state occupancies (also called alignments).

EM algorithm is a standard learning algorithm for HMM, because in the case of HMM the state sequence is hidden.

GMM-HMM

In GMM-HMM model, the emiting probability for state j is defined by a GMM. In monophone, a single model is used to represent a phoneme in all contexts. Then the training task is to optimize the parameters of GMMs for each state, and the transition model of HMM. The monophone GMM-HMM model is usually the first step of a ASR system training. It is used as an initial acoustic model to produce alignments.

In real speech, the pronunciation of a single phone can be affect by the context (the proceeding and forwarding phones), so-called co-articulation effect. Co-articulation dictates that the acoustic realisation of a phone will depend on the neighboring phones because of the constraints of the articulation system. This context dependency extends beyong the phone's immediate neighbours. However, for computational considerations and to restrict the number of model parameters it is sufficient to model the contextual effects due to the immediately neighbouring phones.

This lead to triphones. A triphone models a phoneme in a particular left and right context, which denoted as 'L-X+R', where 'L' phone precedes 'X' and the 'R' phone follows the 'X'. This results in a logical triphone set. Even with triphones, there are potentially too many parameters, and creats a data insufficiency problem. As many triphones may not exist in the training corpus or exist with limited samples, GMMs for the states from those could not be trained properly. Also, states in different triphones of the same phone may be acousticaly very similar.

By clustering and tying acoustically similar triphone states according to decision tree with a range of phonetic questions, logical triphone set can be mapped to a set of physical models, which mitigates this data sparsity issue (Young, Odell, and Woodland 1994). The states in each subset are tied to share data. The leaves of the decision tree are the final set of HMM states (physical models), and are also referred to as senones in DNN-HMM. The number of the leaves largely depends on the size of the training corpus.

There are three main limitations of the GMM-HMM framework (**problem_asr**):

- The HMM assumption that observations only depend on the current state is not completely true. The speech frame usually takes 10ms interval, but dependence persists for 50-100 ms. For this reason, segmental models or Buried Markov models have been examined.
- Training with ML cirteria estimate the parameters in a way that only maximises (locally) the likelihood of the training data. This is not guaranteed to create models that are optimal for discriminating between the correct word-sequence and any other possible word-sequence. However for ASR, the ultimate goal is to minimising the error rate. Discriminative training is introduced to address this issue.
- The duration distribution for acoustic unit is not well defined in the structure of a HMM state. The phonetic duration information can be important for some languages such as Japanese, Finnish, Estonian and Arabic for which length is phonemic. Hidden Semi-Markov Models have been proposed to improve the weak duration modeling (Russell and Moore 1985; Oura, Nankaku, and Tokuda 2006). And more recently, a neural network based duration modeling is proposed in (Wei, Hunt, and Skilling 2019).

DNN-HMM

As shown in Figure 2.8, in a DNN-HMM hybird system, a DNN is trained to compute the posterior probabilities of HMM states given the current acoustic input. The neural network estimate the posteriori probability $P(s_i|y)$, given a feature vector y and a possible state s_i. Viterbi algorithm can be applied to find the best state sequence to generate the observed feature vector sequence. The posteriori probability can be convert to a likelihood $p(y|s_i)$ according to Bayes' rule.

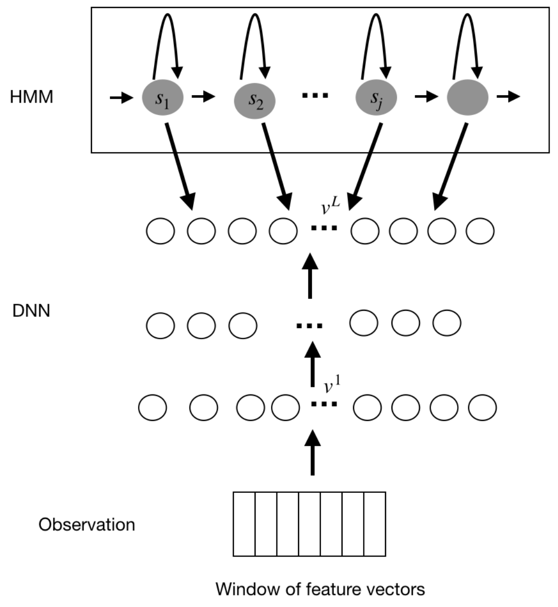


Figure 2.8 DNN-HMM hybird system

2.5.3 Training criteria for GMM-HMM ASR

This section introduce the basic idea about these training criteria for GMM-HMM, details for the parameters updating with these training criteria can be found in Povey (2003).

Maximum likelihood estimation (MLE)

The likelihood of the HMM-based model generating the observed feature sequences Y can be computed as:

$$F_{ml}(\lambda) = \frac{1}{R}\sum_{r=1}^{R} log(P(Y^r|W^r_{ref};\lambda)) \tag{2.42}$$

where the summation is over all the utterances R in the training dataset and the likelihood for a given utterance r is computed with the forward and backward probabilities as below:

$$P(Y^r|W^r_{ref};\lambda) = \sum_{i=1}^{N} \alpha_i(t)\beta_i(t) \tag{2.43}$$

where t can be any frame from the observations (usually the last frame) and W_{ref} is the correct word sequence given by the transcription of that utterance. Parameters re-estimation with ML criteria is to maximise this likelihood given the correct word sequences.

Discriminative training

In discriminative training, the objective is to train the model that increase the likelihood of the model generating the correct word sequence and decrease the likelihood for all other possible word sequences. In maximum mutual information (MMI), the following form is maximised:

$$F_{mmi}(\lambda) = \frac{1}{R}\sum_{r=1}^{R} log(P(W^r_{ref}|Y^r;\lambda)) \tag{2.44}$$

where the posterior of the correct word sequence is computed as below:

$$P(W^r_{ref}|Y;\lambda) = \frac{P(Y^r|W^r_{ref};\lambda)P(W_{ref})}{\sum_W P(Y|W;\lambda)P(W)} \tag{2.45}$$

It equates to maximising the mutual information between the correct word sequences and the models.

Sequence-discriminative training

To achieve the best word error rate, one might want to optimise the likelihood given the word sequence with acceptable error rate compared to the reference W_{ref}. Thus the posterior function to be maximized is modified into:

$$P(W^r_{ref}|Y;\lambda) = \frac{\sum_W P(Y^r|W;\lambda)P(W)A(W,W^r_{ref})}{\sum_W P(Y|W;\lambda)P(W)} \tag{2.46}$$

where $A(W, W^r_{ref})$ is the transcription accuracy of W given the reference W^r_{ref}. If the transcription accuracy is computed between phone level sequence, it is called Minimum phone error (MPE) training. Or state-level minimum Bayes risk (sMBR), if state level sequence is used.

2.5.4 Training criteria for DNN-HMM ASR

In a DNN-HMM hybrid ASR system, a GMM-HMM is usually pre-trained to provide alignments that align each frame to a possible state (also called senones). Then DNN is trained with error-back propagation. The 'error' can be expressed in cross-entropy (CE) criteria or discriminative training criteria.

Cross-Entropy

A standard DNN training criterion is Cross-Entropy (CE) training. It's a discriminative training at the frame level. The softmax function at the output layer makes the posterior for each class sum to one, thus increases the posterior probability for the correct class and decreases the probabilities for the rest. It uses the negative log posterior as the objective function as below:

$$F_{CE} = -\sum_{u=1}^{U}\sum_{t=1}^{T_u} log y'_{ut}(y_{ut}) \tag{2.47}$$

where y_{ut} is the reference state label at time t for utterance u and y'_{ut} is the predicted distribution. Parameters (weights of the DNN) are updated to minimise the difference between the distribution for the reference label and the predicted distribution for each frame.

Sequence discriminative training

Given the success of discriminative training in GMM-HMM, DNNs can also be trained with a sequence level discriminative training criterion, where the objective function operating at the word sequence level. MMI, MPE and sMBR can also be used as the objective function for DNN training. The weights of the DNN is trained to minimise the difference between the best alignment and predicted state sequence and to minimise the probabilities for unlikely alignments in the lattice at the same time.

Usually a CE-trained DNN is needed to generate alignments and lattice for sequence training DNN. For any objective function, the key quantity is the gradient with respect to the units at the output layer, details in terms of the gradient computations can be found in Ghoshal and Povey (2013).

2.5.5 Language model

The prior probability of a word sequence W=$w_1, ..., w_K$ is approximated using an N-gram language model. In an N-gram language model the probability of each word w_k is conditioned only on its N-1 predecessors, where N is usually in the range of 2 to 4:

$$p(w) \simeq \prod_{k=1}^{K} p(w_k|w_{k-1}, ..., w_{k-(N-1)}) \tag{2.48}$$

The N-gram language models with N=1,2,3 are called uni-gram, bigram and trigram language models respectively. The N-gram probabilities are estimated by counting N-gram occurrences for maximum likelihood parameter estimates. For example, the probability of w_k conditioned on the word predecessors $w_{k-2}w_{k-1}$ is:

$$P(w_k|w_{k-1}, w_{k-2}) \approx \frac{C(w_{k-2}w_{k-1}w_k)}{C(w_{k-2}w_{k-1})} \tag{2.49}$$

where $C(w_{k-2}w_{k-1}w_k)$, $C(w_{k-2}w_{k-1})$ denote the number of occurrences of the word sequence $w_{k-2}w_{k-1}w_k$ and $w_{k-2}w_{k-1}$ respectively.

Back-off language model

Data sparsity is a major problem with this simple ML estimation. For a small training corpus, the probabilities of unseen word sequences and word sequences with insufficient occurrences might not be estimated accurately. Discounting and back-off language model is applied to address this issue. The trigram probability can be estimated as below:

$$p(w_k|w_{k-1}, w_{k-2}) = \begin{cases} \frac{C(w_{k-2}w_{k-1}w_k)}{C(w_{k-2}w_{k-1})} & C(w_{k-2}w_{k-1}w_k) > C' \\ d\frac{C(w_{k-2}w_{k-1}w_k)}{C(w_{k-2}w_{k-1})} & 0 < C(w_{k-2}w_{k-1}w_k) \leq C' \\ \alpha(w_{k-2}, w_{k-1})p(w_k|w_{k-1}) & otherwise \end{cases} \tag{2.50}$$

where C' is a count threshold, d is a discount coefficient and α is a normalisation constane. Thus when the N-gram counts is bellow the threshold C', the discounted ML estimate is applied. The discounting coefficient is determined by the Turing-Good estimate $d = (r +$

$1)n_{r+1}/rn_r$, where n_r is the number of N-grams that occur r times in training set [98]. When the word sequence $w_{k-2}w_{k-1}w_k$ is not observed in the training data, then we use a back off probability based on the occurrence count of a shorter context $w_{k-2}w_{k-1}$.

2.5.6 Decoding

As said in the introduction (section 2.5.1) , speech recognition is to find the most probable word sequence $\hat{W}$ given a sequence of feature vectors Y. It is found by searching all possible state sequences for the sequence which is most likely to have generated the observation Y. This process is often referred to as decoding.

A multi-level network is used to incorporate the acoustic modelling, pronunciation and language modelling, etc. For the first level, triphone is represented by a network of HMM states. Then for the second level. a model for a word is formed by a network of triphone according to the pronunciation dictionary. For the final level, a sentence is made up by a network of words. where the connections between word corresponds to language model probabilities.

Viterbi algorithm is applied to decoding, and is described in this section later. In Viterbi decoding, no decision will be made until all probabilities being evaluated for all paths at all levels. But in practical, especially for large vocabulary speech recognition, the paths with low likelihood score are pruned out to save the computation cost.

Different structures have been applied to the Viterbi decoding for large vocabulary speech recognition. The weighted finite state transducers (WFSTs) has shown to effectively compose all of the required information in to a highly optimised network.

Decoding HMM state sequence with Viterbi algorithm

Given an observation sequence $O = o_1, ..., o_T$, decoding with Viterbi algorithm is to find the HMM state sequence that the likelihood of the observation data is maximized. The probability of being in state j at time t and having been through the most probable path is denoted as $\hat{\alpha}_j(k)$, and $\hat{\alpha}_j(k)$ is computed as below:

$$\hat{\alpha}_j(t) = \max_{1 \leq\ i\ \leq\ N}(\hat{\alpha}_i(t-1)a_{ij})b_j(o_t)) \text{ for } 1 < k \leq T \tag{2.51}$$

Let $\hat{\pi}_j(t)$ keep track of the most probable previous state, i.e., the most probable state at $k-1$ for being state j at t.

$$\hat{\pi}_j(t) = \underset{1 \leq\ i\ \leq\ N}{argmax}(\hat{\alpha}_i(t-1)a_{ij}b_j(o_t) \tag{2.52}$$

Defining $\hat{\alpha}_F(T)$ as the probability of all the segments being aligned to the most probable state sequence $q_1, ..., q_T$, the value is given by:

$$\hat{\alpha}_F(T) = \max_{1 \leq\ i\ \leq\ N}(\hat{\alpha}_i(T)) \tag{2.53}$$

Thus the optimal state sequence S_k is obtained by tracking back through $\hat{\pi}$ starting at the final state F ar time T.

ASR evaluation criterion

The performance of the ASR system are evaluated by the Word Error Rate (%WER). It compares the ASR output and the true transcription, and count three types of errors: deletions (D), insertion (I) and substitutions (S). %WER is computed by the proportion of these error among the total number of words (N) as below:

$$\%WER = \frac{D+S+I}{N} \times 100\% \tag{2.54}$$

2.5.7 Adaptation

However the model is trained to maximum the likelihood of training data, it might not work well if the condition in which the model is used is different from which the model is trained. The mismatch between training and testing data includes: different accents, ages and genders between speakers, the nature of the speech (e.g. read speech or spontaneous speech) or the different recording environment. The mismatch will significantly affect the recognition performance.

Speaker adaptation for GMM-HMM ASR

In terms of the speaker variance during training and testing, a speaker-dependent (SD) ASR system usually outperform speaker-independent (SI) systems if a sufficient amount of data is available. If the adaptation data is limited, speaker adaptation (SA) training is applied to make use of the parameters of the generic SI model and obtained a SA model with the limited adaptation data. The mismatch between the speakers in training data and testing data can be compensate by some speaker adaptation method. Adaptation techniques fall into two main categories: 1) feature space adaptation: transform the data to make it more like the data on which the model is trained. 2) model space adaptation: the parameters of the model are adjusted to improve the modelling of the new data.

Maximum a posteriori (MAP) speaker adaptation is incorporating an priori distribution of model parameters $p_0(\lambda^0)$ (the parameters for the generic SI model) to a ML training cirterion as below:

$$\lambda = \underset{\lambda}{argmax} P(Y|\lambda)p_0(\lambda^0) \tag{2.55}$$

During MAP training, if the occupancy of the components in the adaptation data is small, the MAP estimated parameters will remain close to the initial model. On the other hand, if the components is well presented in the data, the MAP estimate is shifted more towards to the ML estimate over the adaptation data.

Adaptation with linear transforms assumes that they capture general relationships between the generic SI model and the adaptation data, so that parameters with limited occurrence can be adapted as well. This overcome the drawback of MAP adaptation. Maximum likelihood linear regression (MLLR) (Gales and Woodland 1996) uses linear transformation of Gaussian model parameters including mean vectors and covariance matrices to adapt to new data, and the transforms on mean and covariance are unrelated to each other. If the same transformation matrix is used for both mean and variance transform, this method is called constrained MLLR (CMLLR). CMLLR can be implemented as a transform of observation features, thus it can be called feature-space MLLR (fMLLR). CMLLR is easier to implement than MLLR, due to the huge runtime space requirement of MLLR. SAT with CMLLR can be seen as a type of speaker normalization during training time, since maximising the likelihood of transformed data from different speakers given the SI model can be seen as the data being adapted to the SI model. The GMM parameters for the SI model and CMLLR transforms are estimated during training time. During recognition, CMLLR transforms are learned for each speakers but there is no need to adapt the model parameters. Thus the mismatch between training and testing data can be compensated by adapting them to a generic model.

2.6 Speaker diarisation

2.6.1 Introduction

The origin of the key techniques that used for speaker diarisation is in speaker recognition. In speaker recognition, a voice sample is either to be verified as from a person who he or she claims (speaker verification) or to be identified as one of known speakers (speaker identification). Speaker diarisation decides 'who speaks and when'. It starts from temporal segmentation, followed by extracting vector representations from these segments. The diarisation results are produced by clustering the vector representations into a number of sets. For these tasks, researchers need to represent a speech segment as a fixed dimensional vector. The representation of the speech segment should be able to preserve the speaker-dependent characteristics that can distinguish one speaker from another.

A GMM-based speaker model was first introduced by Reynolds and Rose (1995). Later, a Gaussian Mixture Model-Universal Background Model (GMM-UBM) was developed by Reynolds (1997). In this system, UBM is a large GMM trained on all the training data for all speakers with EM algorithm. Given data from a particular speaker, maximum a posteriori (MAP) adaptation is applied to the speaker-independent UBM to estimate the speaker-dependent GMM. But if the speaker's data is so sparse that it may not span all of the phonetic events, the corresponding GMM components will not be updated by MAP adaptation. Therefore, a low-dimension vector is introduced for speaker representation by Kenny, Boulianne, and Dumouchel (2005). This vector is referred to as i-vector.

I-vector is usually combined with clustering techniques for speaker diarisation systems, for example, variational Bayesian Gaussian model (Shum et al. 2013), probabilistic linear discriminant analysis(PLDA) (Sell and Garcia-Romero 2014). With the development of deep learning, researchers have explored the use of DNN to extract i-Vector (Guo et al. 2018) or directly extracting embedding vector representation from the bottleneck layer output of a DNN, so called x-vector (Snyder et al. 2017). The x-vector based system has won the first DIHARD speech diarisation challenge (Sell et al. 2018).

The most widely used metric in speaker diarisation is diarisation error rate (DER) (Fiscus et al. 2006), which measures three different error types: False alarm (FA) of speech, missed detection of speech and confusion between speaker labels.

$$DER = \frac{FA + Missed + Speaker - confusion}{TotalDurationofTime} \tag{2.56}$$

Given the nature of the MI data used in this thesis, there are very little silence between the two speakers, the metric used in this thesis focuses more on the system correctly detecting the speaker change point (ref to Section 6.5).

More details about the GMM-UBM supervector and i-vector will be discussed later in this section.

2.6.2 GMM MAP adaptation

MAP is close to ML, but incorporates a prior distribution of the model parameters. The parameters are estimated as below:

$$\widehat{\lambda} = \underset{\lambda}{argmax}[p(X|\lambda)p(\lambda)] \tag{2.57}$$

In Reynolds (1997), UBM was use to provide the prior distribution. Given a UBM and feature vectors from the target speaker, $X = x_1, ..., x_T$, the first step is identical to the E-step of EM algorithm introduced in Section 2.3.1, the component occupancy γ_{it} is computed for every component i in UBM. Then, the occupancy and feature vector x_t are used to compute sufficient statistics as below:

$$0^{th}order : n_i = \sum_{t=1}^{T} \gamma_{it} \tag{2.58}$$

$$1^{st}order : E_i(x) = \frac{1}{n_i} \sum_{t=1}^{T} \gamma_{it} x_t \tag{2.59}$$

$$2^{nd}order : E_i(x^2) = \frac{1}{n_i} \sum_{t=1}^{T} \gamma_{it} x_t x_t^T \tag{2.60}$$

These are the basic statistics needed to compute the mixture weight, mean and variance of a GMM if speaker data is sufficient. In MAP adaptation, the parameters are updated as below:

$$\bar{w_i} = [\alpha_i^w n_i / T + (1 - \alpha_i^w) w_i]\gamma \tag{2.61}$$

$$\bar{\mu_i} = \alpha_i^m E_i(x) + (1 - \alpha_i^m)\mu_i \tag{2.62}$$

$$\bar{\sigma_i^2} = \alpha_i^v E_i(x^2) + (1 - \alpha_i^v)(\sigma_i^2 + \mu_i^2) - \bar{\mu_i^2}. \tag{2.63}$$

where $\alpha_i^w, \alpha_i^m, \alpha_i^v$ are the adaptation coefficients for the weights, mean and variance respectively, and γ is the scaling factor to ensure the mixture weights sum to 1. The data-dependent adaptation coefficients $\alpha_i^\rho, \rho = w, m, v$ are defined as:

$$\alpha_i^\rho = \frac{n_i}{n_i + r^\rho} \tag{2.64}$$

where r^{ρ} is a fixed factor for parameter ρ. If the probabilistic count for component i (0^th order statistics), n_i, is low, which means the component is less represented in the data, then the adapted parameters of this component will not be changed too much. If n_i is high, the parameters will be updated relying more on the sufficient statistics computed from the adaptation data.

2.6.3 Supervector

Given a MAP adapted GMM model with M components defined on a D dimensional vector space, then the GMM-supervector S is the $M \times D$ dimensional vector obtained by concatenating the mean vectors (Campbell et al. 2006). The problem with GMM-supervectors is their high dimensions. The MAP-adapted GMM supervectors couldn't work well if the data from a speaker is sparse. Researchers have applied dimension reduction techniques like principal component analysis (PCA) to map the high-dimensional supervector to a low-dimensional space with minimum reconstruction error. However, PCA requires an accurately-estimated covariance matrix, which means it is not possible if the dimension of supervectors is much larger than the number of supervectors.

2.6.4 i-vector

Dehak et al. (2011) defined a total variability matrix T, i.e., a linear mapping from a low-dimensional 'total variability subspace' into the supervector space. T is estimated directly from training data. The GMM supervector can be decomposed as $S = m + Tw$, where m is UBM and w is an 'i-vector' sampled from a standard normal distribution. The T matrix and w are estimated to maximise the posterior probability given the data. The posterior distribution is a Gaussian distribution and the mean corresponds to the i-vector. The sufficient statistics are computed from the UBM as shown in Section2.6.2. Given T and these sufficient statistics, the i-vector for a speech utterance u is computed as below:

$$w = (I + T^t\Sigma^{-1}N(u)T)^{-1}.T^t\Sigma^{-1}\widetilde{F}(u) \tag{2.65}$$

where $N(u)$ is a diagonal matrix of dimension $MD \times MD$ with diagonal blocks n_iI $(i = 1, ..., M)$ and n_i is the 0^{th}-order sufficient statistics. $\widetilde{F}(u)$ is a vector of dimension $MD \times 1$ obtained by concatenating the centralised 1^{st}-order sufficient statistics and Σ is the covariance matrix of the supervectors.

So far, the supervector originates from a GMM. If there are a set of symbols representing feature vectors and posterior probability of each symbol can be calculated, then i-vectors can be build on this set of symbols. Lei et al. (2014) have used senones and its output posterior from a DNN that has been trained from ASR to extract i-vectors for speaker recognition.

2.7 Text processing

2.7.1 Introduction

In text processing, a document is a sequence of words. In the case of topic modelling, a whole MI session is regarded as a document. In the case of behaviour code classification each utterance is seen as a document. In order to apply vector space classification methods, such as SVMs and KNN, to documents, it is necessary to represent a document as a vector. Since documents have different lengths, this is non-trivial. The simplest way to represent a document d as a vector is to create a document vector whose i^{th} element is the TF-IDF weight of the i^{th} term relative to the document.

More recently a number of methods have been proposed where the document vector is created using a neural network, known as Doc2Vec. The Doc2Vec originates from a word vector representation Word2Vec.

In terms of KNN, Euclidean distance based on TF-IDF document vector is the standard approach for the computation of dissimilarity between documents. Word Mover Distance and Dynamic Timing Warping based on Word2Vec embedding are explored.

Word2Vec is introduced in Section 2.7.3 and then Doc2Vec in Section 2.7.4.

2.7.2 TF-IDF

Term frequency-inverse document frequency (TF-IDF) (Salton and Buckley 1988) document vector is a simple method to represent document as a collection of words. It is a vector with the dimension of the vocabulary size. The t entry is the TF-IDF weight of the term t relative to the document.

If term t occurs c_t times in document d, the relative term frequency is given by the frequency of the term that occurs in the document d:

$$tf(t,d) = \frac{c_t}{\sum_{i=1}^{n} c_i} \tag{2.66}$$

The inverse document frequency (IDF) provide information on the term's occurances across all documents in the corpus D. It is an inverse function of the frequency of documents in which the term occurs:

$$idf(t,D) = log\frac{N}{|\{d \in D : t \in d\}|} \tag{2.67}$$

where N is the total number of documents in the corpus and $|\{d \in D : t \in d\}|$ is the number of documents that contain the term. TF-IDF is the product of the term frequency and inverse document frequency:

$$tfidf(t,d,D) = tf(t,d) * idf(t,D) \tag{2.68}$$

A word with high TF-IDF indicates that it occurs many times in a ducument but not in all documents.

Since the length of each sentence can be short, this method will lead to a really sparse vector (most of the values are zero). And it only takes the occurrence of the words into account, disregarding the semantic and syntactic meaning and word order. For example, two semantically closer documents are equally distant as two irrelavent documents only if they use different words. Also because the word order is lost, two different documents can have same vector representation if same words are used.

2.7.3 Word2Vec

Word2vec is introduced as a semantically meaningful word vector representation method (Mikolov et al. 2013a; Mikolov et al. 2013b). Given enough data, it trains a two-layer neural network that gives the feature vectors of the words in the corpus. Each word vector is trained to maximize the probability of neighboring words in a corpus, or the vectors of the context are trained to predict the current word. The two method are called Skip-Gram or Continuous Bag Of Words (CBOW), respectively. Different from the conventional DNN, there is no activation function at the hidden layer (also called the projection layer). The only non-linearty is the softmax function at the output layer. The network is usually trained using stochastic gradient descent with error backpropagation. Thus, semantically similar words are mapped to a closer position in the Word2Vec vector space.

The structure of the Skip-Gram is shown in Figure 2.9. The input vector is an V dimensional one-hot representation of the current word $w(t)$, where V is the size of the vocabulary. The hidden layer is of dimension E, and is used as the word embedding. The dimension of the softmax output layer is V and each value represents the probability of the word being the neighboring words of the current words within a given context length.

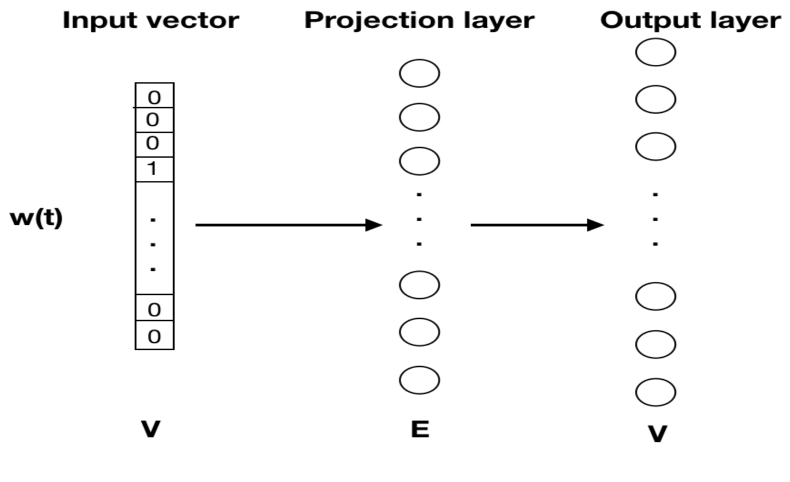


Figure 2.9 Structure of the Skip-gram word2vec model.

As shown in Figure 2.10, in CBOW word2vec model, the dimension of the hidden layer and output layer is same with Skip-gram word2vev. The input are contanated by the context word vectors for the current word (i.e. a 4V dimensional input vector for 4 context words, with 2 on each side). Each word vector will be projected to the E dimensional hidden layer, and the embeddings are averaged or concatenated to obtained the final output of the hidden layer. Each value in the output layer represents the probability of the current word $w(t)$.

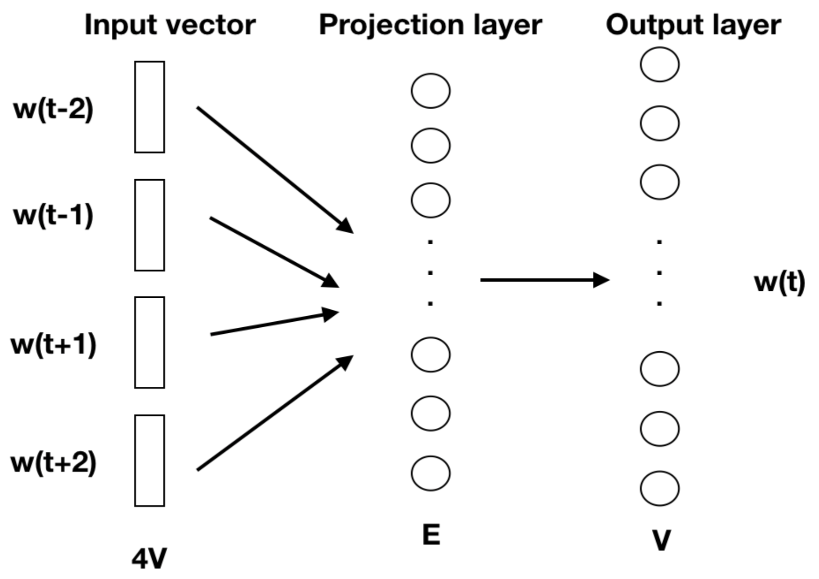


Figure 2.10 Structure of the CBOW word2vec model.

2.7.4 Doc2Vec

Based on the structure of CBOW Word2Vec model, instead of just using the context words to predict the current word, a numeric vector representation of the document is input to the network (Le and Mikolov 2014). The document vector and word vectors can have different dimensions and are concatenated (or averaged if they have same dimension) to predict the current word. Thus the Doc2Vec is trained together with the Word2Vec using stochastic gradient descent and the gradient is from error backpropagation. The Word2Vec embeddings are shared across different document. The document vector is shared for all contexts generated from the same document. For a new document, the Doc2Vec vector is obtained by gradient descent with the fixed word embedding and weight matrix. This Doc2Vec is known as Distributed Memory version of Paragraph Vector (PV-DM) and is illustrated in the left Figure 2.7.4.

Another document vector method is to predict words randomly sampled from the document given the document vector, this version is known as the Distributed Bag of Words version of Paragraph Vector (PV-DBOW). In this simpler model, only softmax weights are stored, while in PV-DM model both softmax weights and word vectors are stored.

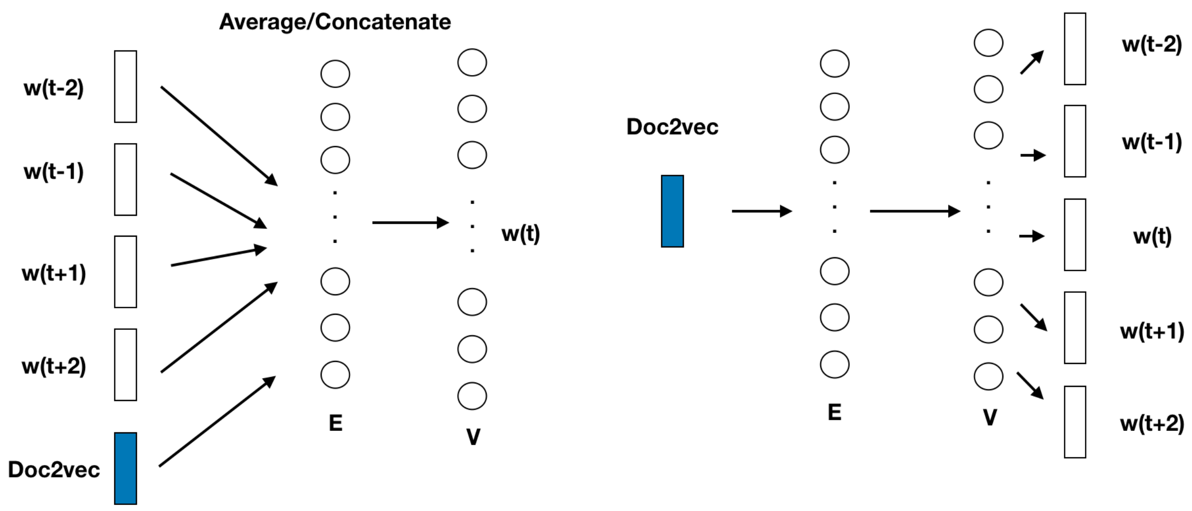


Figure 2.11 Structure of the PV-DM Doc2Vec model (left) and PV-DBOW model (right).

2.8 Classification

2.8.1 Introduction

Section 3.2.2 introduced five behaviour codes that could be assigned to clinicians' utterances. The counts are used as a local measurement of the clinicians' performance. For example, it is better to use more Open questions than Closed questions and more MI Adherent than MI Non-Adherent.

The utterances can be represented as vectors using various document representation methods introduced in Section 2.7. These can be used as the input to a classifier whose task is to assign the utterance into one of the behavior codes.

The two classification methods, Support vector machine (SVM) and K-Nearset Neighbor (KNN) will be presented in this section. Since KNN is a distance based method, different distance functions, for example, Word Mover Distance (WMD) and Dynamic Time Warping (DTW), are introduced.

2.8.2 Support vector machine (SVM)

This thesis will only discuss the support vector machine (SVM) (Cortes and Vapnik 1995) in binary classification problems. Given a set of samples $X = \{x_i\}$ with class labels $Y = \{y_i\}, y_i \in [-1, +1]$. The purpose of a SVM is to find the hyperplane that separate the input space into two half space with the maximum margin so that any x_i with label $+1$ is on the positive side and any x_i with label -1 is on the negative side.

For a linear SVM, the hyperplane is defined by:

$$w^T x - b = 0, \tag{2.69}$$

where w^T is the normal vector to the hyperplane. Thus the hyperplane is the subspace of vectors that are orthogonal to $\mathbf{w}$ and which is at distance $\frac{b}{||w||}$ from the origin. The hyperplane is subjected to:

$$y_i(w^T x_i - b) \geq +1 \tag{2.70}$$

Intuitively, the hyperplane not only separate the samples to be on the right side, but also keep some distance for better generalization. Support vectors are the data points that are closest to the hyperplane and decide the margin of the classifier. The total margin is $\frac{2}{||w||}$.

If the data is not linearly separable, the hyperlane with soft-margin that incurs the least error is introduced. The error can be if the sample is on the wrong side, or it is on the right side but lies in the margin. The hinge loss function is introduced to define the error:

$$max(0, 1 - y_i(w^T x_i - b)) \tag{2.71}$$

Regularization parameters are introduced to balance the margin maximization and error minimization. SGD is usually applied to computed the parameters.

Kernel Trick

Kernel functions are introduced to map the data points linearly or non-linearly into a higher dimensional vector space where the classes are more easily separated. Commonly used kernel functions are: linear, radial basis function (RBF), and sigmoid. The idea of kernel machines is to compute the high-dimensional relationship between pairs of data directly in the original input space rather than mapping the data and then doing a dot product. These kernel functions on a pair of data points $\{x_i, x_j\}$ are defined as below:

$$Linear: \quad K(x_i, x_j) = x_i^T x_j + c \tag{2.72}$$

$$RBF: \quad K(x_i, x_j) = exp(-\gamma \left\| x_i - x_j \right\|^2) \tag{2.73}$$

$$Sigmoid: \quad K(x_i, x_j) = tanh(\alpha x_i^T x_j + c) \tag{2.74}$$

where the c, γ and α are the hyperparameters.

2.8.3 K-Nearest Neighbor (KNN)

In KNN, for each sample from testing data, its distances to every sample from the training data are computed and then classified by the majority vote of its k neighbors. Word mover distance (WMD) (Kusner et al. 2015) and dynamic time warping (DTW) can be used as the distance measurement, denoted as WMD-KNN and DTW-KNN respectively.

2.8.4 Word mover distance (WMD)

Word mover distance (WMD) was first introduced as a distance function to measure the dissimilarity between text documents that incorporates the semantic similarity between individual word pairs (Kusner et al. 2015). It is based on Euclidean distance of the word2vec embeddings and constrained by each word's local co-occurrences in sentences.

Given a word2vec model with a vocabulary of n words, let x_i denoted the word2vec embedding for the i^{th} word. Text documents are represented as a vector of term frequency d, namely, if word i occurs c_i times in the document, $d_i = \frac{c_i}{\sum_{j=1}^{n} c_j}$. The dimension of d is the size of the vocabulary.

Word travel cost

The distance between word i and word j is computed by the Eculidean distance in the word2vec space, $c(i, j) = \|x_i - x_j\|_2$. This is refer to as the word travel cost from one word to another.

Document distance

Let d and d' denote the term frequency vector representation of two text documents. Each word i in d can transport into any word j in d', the travel cost is denoted as T_{ij} ($T_{ij} \geq 0$).

The WMD between the two documents is given by the minimun cumulative cost of moving all words from d to d', i.e.:

$$WMD(d,d') = \min_{T \geq 0} \sum_{i,j=1}^{n} T_{ij} c(i,j), \tag{2.75}$$

with the following constraints:

$$\sum_{j=1}^{n} T_{ij} = d_i \quad \forall i \in 1, ..., n \tag{2.76}$$

$$\sum_{i=1}^{n} T_{ij} = d'_j \quad \forall j \in 1, ..., n \tag{2.77}$$

Each word i in document d can transform to more than one word in document d', and each word j in document d' can come from more than one word in document d. The constraints ensure that when transforming d into d' the sum of the entire outgoing flow from word i equals the relative term frequency d_i, and the sum of the incoming flow to word j equals d'_j. Figure 2.12 illustrates the transportation from ducument d to d'.

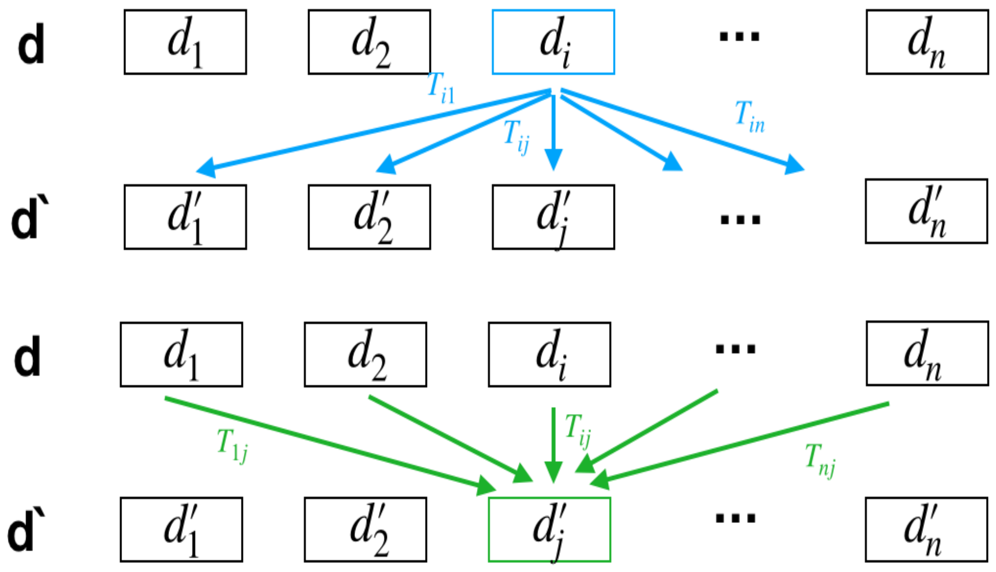


Figure 2.12 Top: The outgoing flow from word i in ducument d to d'. Bottom: The incoming flow to word j in ducument d'.

The WMD derives from Earth Mover Distance (EMD). An analogy can be created for the WMD by having a set of n locations and two sets of buckets - red buckets and blue buckets. The n locations correspond to the different vocabulary items. Each bucket (red and blue) is placed on a location and the red buckets are filled with sand. The red buckets can be represented as a n dimensional vector d, where d_i is the capacity of the red bucket at location i (which is 0 if there isn't a red bucket at location i). The total capacity of the red buckets is the same as the total capacity of the blue buckets (in WMD this corresponds to the relative frequencies sum to 1). T_{ij} is the amount of sand moved from the i^{th} red bucket to the j^{th} blue bucket. Thus the two constraints ensure that all of the sand in the red buckets must be moved to the blue buckets. $c(i, j)$ is the physical distance from the location of the i^{th} red bucket to the location of the j^{th} blue bucket, corresponding to the word travel cost between words in d and d'. If the physical distance is small between the two locations, EMD will tend to move more sand from i to j (i.e. increasing T_{ij}). In WMD this follows because if word i in document d and word j in document d' are semantically closer (low Euclidean distance in the word2vec space $c(i, j)$) than any other words in d', to achieve a minimum cumulative cost, T_{ij} should be relatively greater than any other pairs $(T_{i1}, T_{i2}, .., T_{in})$. The set of T_{ij}s that minimises the cumulative cost of moving can be found using linear programming.

The rules ensure that the words that are semantically closer are more likely aligned to each other during document similarity computation, there is no constraint on the word ordering in the documents.

Nasir et al. (2019) adopted this method to measure the dissimilarity in language used in multiple consecutive speaker turns in the clinical psychology domain. And their extended WMD method has shown to have correlation with the therapist's empathy towards their patient in motivational interviewing.

2.8.5 Dynamic time warping (DTW)

Dynamic time warping (DTW) is a efficient method to find the best alignment between two documents that gives the minimal cost, i.e. the cumulative distance along the optimum path. The alignment enforces that the first words and final words of the two documents must be aligned. This minimal cost is used as the distance between document for the implement of KNN. The distance between word i in document d and word j in document d' is the Euclidean distance $c(i,j)$ in the word2vec embedding space.

As shown in Figure 2.13, the path always goes forward. If point (i,j) is on the optimum path, it can only come from three immediately preceding points: $(i-1,j)$, $(i-1,j-1)$ and $(i,j-1)$ The best way to get to the point (i,j) is to get to one of the three immediately preceding points with the best way. Let $D(i,j)$ denoted the cumulative distance along the optimum path from the beginning to point (i,j),thus:

$$D(i,j) = min[D(i-1,j), D(i-1,j-1), D(i-1,j-1)] + c(i,j) \tag{2.78}$$

with $D(1,1) = d(1,1)$. Thus $D(n,m)$ is the minimal cumulative distance between two document d and d', with length of n and m, respectively.

The alignment from DTW ensure that if the i^{th} word in d is aligned with j^{th} word in d', the words before the i^{th} word won't be aligned to the words after the i^{th} word in string d', and vice versa. This monotonicity in DTW incorporates the word order information into distance measurement, which might be a major factor when classifying questions from non-questions.

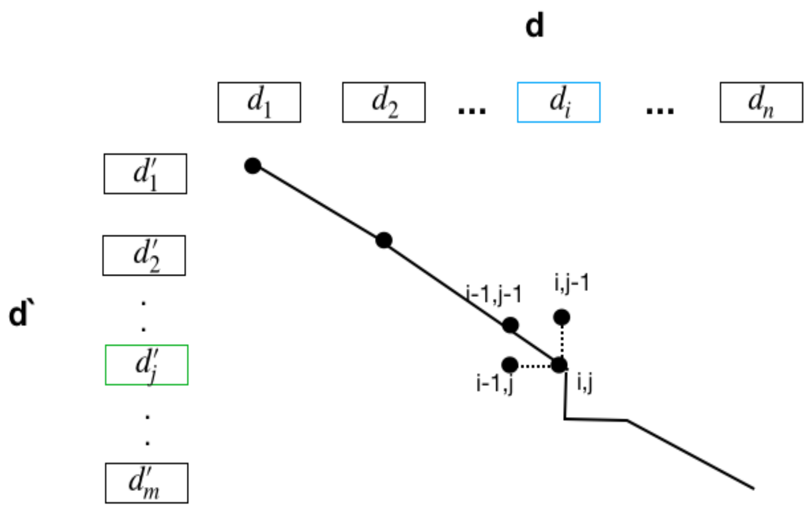


Figure 2.13 Top: The outgoing flow from word i in ducument d to d'. Bottom: The incoming flow to word j in ducument d'.

2.9 Topic modelling

2.9.1 Introduction

Topic modelling is an unsupervised approach to find the topics present in a collection of documents. A topic is usually represented as a collection of words, or a distribution over words.

A corpus is a collection of documents, and can be represented as a document-word matrix. The columns of the matrix represent different documents. The rows of the matrix represent different words. Each entry (i, j) in this matirx is a raw count of the occurance (or TD-IDF weight) of the word i in the document j. The matrix is a $n \times m$ matirx if the dimension of vocabulary is n and the total number of documents in the corpus is m.

Words with a common term will usually have similar meaning, for example: 'connected' , 'connected', 'connecting', 'connection' and 'connections'. For topic modeling, meaning of each word is more importatant than the tense and structure. The Porter stemmer (Porter

1997) was applied to remove the suffixes from words and leave the basic terms. By preprocessing the text using Porter stemmer, the size of vocabulary was reduced and thus dimension of the document vector was effectively reduced.

Latent semantic analysis (LSA) and latent Dirichlet allocation (LDA) are the two most widely used technologies for topic modelling.

2.9.2 Latent semantic analysis (LSA)

Latent semantic analysis (LSA) (Dumais 2004) learns topics by performing a matrix decomposition on the document-word matrix M using singular value decomposition as below:

$$M = USV^T \tag{2.79}$$

The diagonal S matrix contains the singular values of M. The t largest singular values corresponds to the t most significant dimensions. A rank t approximation to M can be obtained by only keeping the t largest singular values and the first t columns of U and V with the smallest error. Each dimension can be regarded as a latent topic. So t is a hyperparameter to be selected manually and adjusted to reflect the desired number of topics.

$$M \simeq U_t S_t V_t^T \tag{2.80}$$

In the $m \times t$ document-topic matrix U_t, each row represents a document as a vector of topics. In the $n \times t$ topic-word matrix V_t, columns represent topics as vectors of words.

2.9.3 Latent Dirichlet allocation (LDA)

Latent Dirichlet allocation (LDA) (Blei, Ng, and Jordan 2003) models a document d as a multinomial distribution over an underlying set of K topics $\theta^{\mathbf{d}} = [\theta_1^d, ..., \theta_K^d]^T$, with $\theta_k^d \geq 0$ and $\sum_{t=1}^{K} \theta_k^d = 1$. The probabilities θ_k^d correspond to the proportions of different topics and are drawn from a Dirichlet prior with hyperparameter α.

$$\theta^{\mathbf{d}} \sim Dirichlet_K(\alpha) \tag{2.81}$$

Each topic k is represented as a multinomial distribution over M words $\beta^{\mathbf{k}} = [\beta_1^k, ..., \beta_M^k]$, with $\beta_m^k \geq 0$ and $\sum_{m=1}^{M} \beta_m^k = 1$. The probabilities are also drawn from a Dirichlet prior with hyperparameter γ.

$$\beta^{\mathbf{k}} \sim Dirichlet_M(\gamma) \tag{2.82}$$

A graphical model for LDA is shown in Figure 2.9.3.

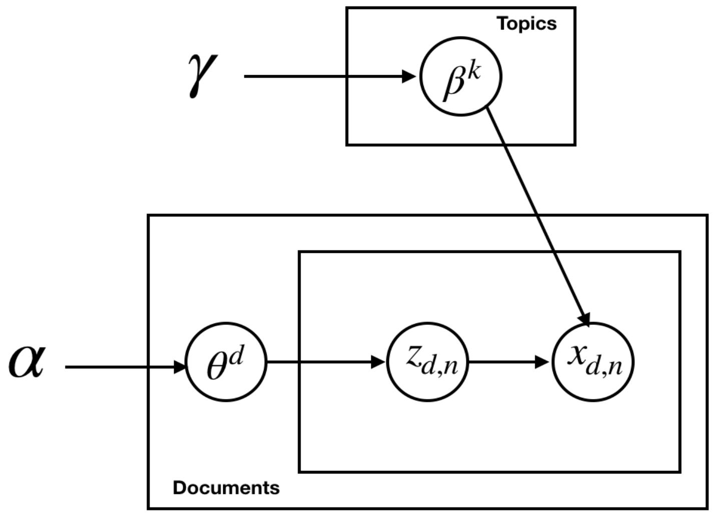


Figure 2.14 A graphical model for LDA topic modelling.

First, for each document d choose a distribution of topics θ^d. Then, for each word m in the document, a topic is choosen from the topic distribution by sampling from θ^d and denoted as $z_{d,n}$. $z_{d,n}$ is a value between 1 and K. The probability of topic $z_{d,n}$ is then given by $\theta^d_{z_{d,n}}$ And then a word is choosen from the word distribution β^k that representing topic $z_{d,n}$. The process is repeated for every word in the document.

From a collection of documents, also defined as a corpus, fitting a LDA model will obtained the topic assignments $z_{d,n}$ for each word, topic distirbution θ^d for each document and word distribution β^k for each topic. Number of topics (t) is pre-defined.

The θ^d is often used to represent a document for similarity computation.

2.10 Summary

This chapter introduced all of the speech and language techniques that are used for the development of the automatic assessment system. First, the acoustic feature vectors for ASR and speaker diarisation are both MFCC features, and was introduced in Section 2.2. A conventional ASR system consists of an acoustic modelling and a language modelling. The decoding process generates the most probable word sequence as the output of the ASR system. HMM is used as a standard generative acoustic model that gives the probability of the model generates the observations. The HMM state emitting probability can be modelled using either a GMM or a DNN, denoted as GMM-HMM and DNN-HMM, respectively. Language model provides a prior probability of a word sequence. For GMM-HMM ASR, different speaker adaptation training techniques were introduced. For DNN-HMM ASR, various DNN structures, for example, RNN and LSTM, were presented.

In the case of the speaker diarisation, the origin of the speaker representation technique i-vector was discussed in Section 2.6. Text processing techniques that represent text into vectors were introduced in Section 2.7. For the assessment purpose, the classification techniques(i.e SVM and KNN) and topic modelling techniques (i.e. LDA and LSA) were presented in Section 2.8 and Section 2.9. Different word2vec based document distance criteria for KNN (i.e. WMD and DTW) were introduced.

Chapter Three

Motivational Interviews and data

3.1 Introduction

Motivational Interviewing is a client-centered counseling style that seek to evoke the clients inner motivation to change by helping the client explore and resolve ambivalence. The counselor elicits clients' behavior change rather than simply using direct persuasion.

This chapter will first introduce the coding system that is widely used for evaluating the clinicians' MI skills and providing feedback. Then in Section 3.3, the in-domain MI data will be indtroduced. Given the in-domain data is limited, open-source speech corpora used for the ASR and speaker diarisation experiments will be introduced in Section 3.4.

3.2 MITI guideline

Motivational Interviewing Treatment Interity (MITI 4.0) provides a behavioural coding system to evaluate the quality of the motivational interview. It only measures clinicians' behaviours and is used as a means of providing feedback for training. In the context of MI assessment relative to the MITI Guidelines, the assessor is often referred to as a "coder". The assessment is made based on the coder's global impresssion over the interview as well as the instances of particular clinician behaviours.

3.2.1 Global impression measures

A single number from a five-point scale is to be assigned by the coder to assess an interview session in five dimensions: Evocation, Collaboration, Autonomy/Support, Direction and Empathy. Evocation, Collaboration and Autonomy/Support are averaged to obtain a Spirit score.

- Evocation: This scale is to assess how well/poorly the clinician leads the client to the change goal. A good interview requires that the clinician constantly concentrates on the clients' language about change, and hold on to the opportunities to encourage the client to change.
- Autonomy/Support: This is a scale that assesses the extent to which the clinician supports the patient's self-control and own choice rather than persuading the patient to change.
- Collaboration: This scale is to ensure that the interview goes on between two equal partners. Both of the two participants need to show understanding with each other and knowledge to achieve the change goal.
- Direction: This scale measures whether the clinicians focus on a specific target behavior or concerns directly related to it.
- Empathy: This scale focuses on the clinicians understanding of the clients perspective and experience as well as expressing that understanding to the client.

3.2.2 Local behavior identification

Instead of making assessment on an overall impression, behaviour counts intend to capture specific behaviours of the clinicians during the interviews. The clinician's utterances are assigned with five primary types of behaviour codes, but not all clinician utterances will be given behaviour codes. The MITI guidelines define an utterance as a complete thought. A new idea or a patient response can always introduce or terminate an interviewer's utterance. The five primary behaviour codes are:

- Giving Information: This is used when clinician gives opinions without advising. For example, when the clinician educates about a topic, provides feedback from assessment instruments or discloses personal information to the patients, it is regarded as Giving Information.
- MI Adherent: This is coded when clinician's behaviours are consistent with a motivational interview approach. It consists of asking permission before giving advice, affirming the patient by saying something positive, emphasizing the patient's ability to control his or her own behaviour.
- MI Non Adherent: This is used when the clinician's behaviour are inconsistent with a motivational interviwe approach.
- Question: This include two sub-categories, Open Question and Closed Question. Open Question is when the clinician asks a question that can be answered in a wide range. Closed Question can be answered with a "yes" or "no" question. It is can also coded when the range of answers are very restricted. Closed Question usually start with word such as: can, could, did, would, should, are, will, have, ect.
- Reflections: This category is coded when the clinician responses to the patient's statement. It can be futher categorized into Simple Reflection and Complex Reflection.

3.3 MI data and annotation

The MI speech corpus was collected by the Institute of Psychiatry, Psychology and Neurosciences (IoPPN) at Kings College London. It contains recordings of MIs between clinicians and diabetes patients, which were recorded using distant microphones in various clinicians' offices. The identities of the clinician and the patient in a particular recording session are unknown.

The corpus currently comprises of speech recordings from 8 full sessions, each session lasting around 20-30 minutes and coded by two psychologists according to the MITI guideline, and 84 extracts (length from 5 minutes to 10 minutes) with accompanying manual transcriptions. Each session's recording contains two channels. We use data from both channels and regard data from different channels as different sessions. For the rest of this thesis, a 'session' means one of the channels.

56 sessions from the MI corpus consisting of 7 hours of speech were manually segmented into utterances, denoted as MI_train. Each utterance is less than 10 seconds long and contains the speech of a single person. While performing the segmentation, we excluded parts of the signal that contained significant noise, no speech or overlap between speakers. Figure 3.1 shows a histogram of the length of the utterances. The testing set, denoted by MI_test, consist of 45-minute speech recordings from 12 sessions.

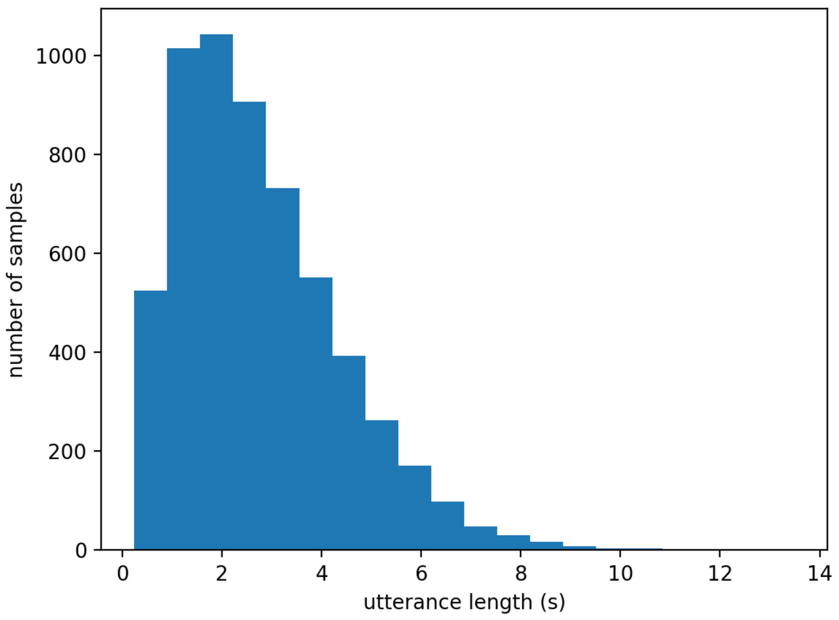


Figure 3.1 Histogram of the length of the utterances in MI_train.

3.4 Open-source speech corpora

3.4.1 WSJCAM0 corpus

The WSJCAM0 corpus (Robinson et al. 1995) is a British English speech corpus, equivalent to the WSJ0, which is a US American English corpus. WSJCAM0 consists of 140 native UK English speakers reading material from Wall Street Journal, split into training, development and evaluaton sets. The minimum age of the speakers is 18 and majority of them are between 18 and 28, and the number of male and female speakers is balanced. For the training set in the WSJCAM0, 90 utterances were read by each of 92 speakers, which contains 15.5 hours data in total. The transcriptions of all the utterances are available. The utterances were recorded at a sampling rate of 16kHZ in a quiet room recording environment using a far-field desk microphone and a head-mounted close-talking microphone.

A British English Example Pronunciation Dictionary (BEEP) has been developed to cover all the words in WSJCAM0 corpus, which contains over 250,000 English word pronunciations.

The phone set in BEEP are composed of 44 phones. In order to train system together with Shared Task data, the recording from WSJCAM0 is down-sampled to 8kHZ.

3.4.2 AMI meeting corpus

The AMI Meeting corpus (Mccowan et al. 2005) consists of 100 hours of meeting recordings. The meetings were recorded in English using three different rooms with different acoustic properties, and most of the speakers were non-native adult speakers. Audio was downsampled from 48kHz to 16kHz for building ASR system for MI and downsampled to 8kHz for Spoken CALL Shared Task. The recordings were recorded simultaneously by a single distant microphone, denoted by AMI-sdm, and by individual headset microphones, denoted by AMI-ihm. High quality manual transcription for each speaker are available.

Chapter Four

Automatic speech recognition for Spoken CALL shared task

Before starting work on the MI data research was conducted to develop an ASR system for the 2017 Spoken CALL Shared Task. The task will be described below. Specifically, a baseline ASR system was developed for the 2017 Shared Task, so that laboratories with expertise in text processing but no ASR could participate in the task using the output from the baseline ASR system. The ASR baseline is a successful application of ASR in the education domain. Based on this ASR system, developed systems were explored and submitted to the 1^{st} edition and 2^{nd} edition challenges. The contributions made for this task also provided a solution for the adaptative training that is also needed for building a MI ASR.

This case study section will be organised as below: Section 4.1 provides an introduction to the shared task, the data collected for this task in 1^{st} and 2^{nd} editions are described in Section 4.2, Section 4.3 describes the ASR baseline for the 1^{st} edition, Section 4.4.3 will introduce our developed systems.

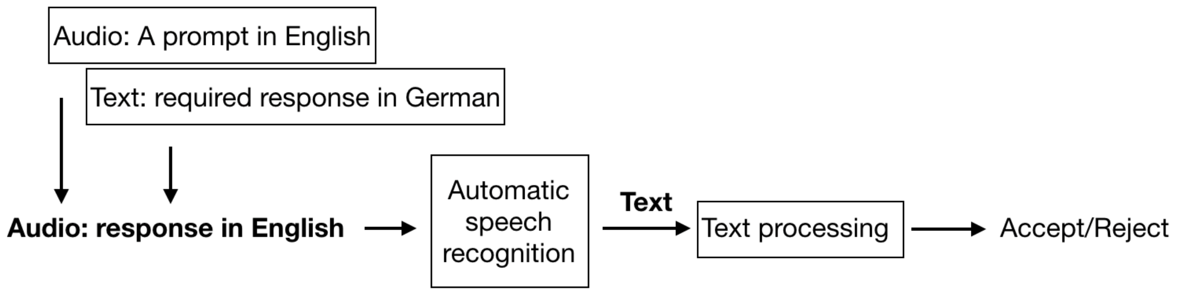

Audio: A prompt in English
Text: required response in German
Audio: response in English
Automatic speech recognition
Text
Text processing
Accept/Reject

Figure 4.1 An overview of the system.

4.1 Introduction

In the Speech and Language Technology in Education(SLaTE) CALL Shared Task, groups competed on a spoken prompt-response task. Data was collected, through an online CALL course, from Swiss German teenagers in their second and third years of English learning. The course runs on CALL-SLT (Rayner et al. 2010), a spoken CALL platform developed at Geneva University since 2009. The teenagers were asked to give response to a prompt. Each prompt contains a multimedia file in the L2 (English) and a written text instruction in the L1 (German). For example, the system plays a question in English spoken by a English native speaker (e.g. “How many nights would you like to stay at our hotel?”) and simultaneously displays the text of the required answer in German. The student are asked to give the answer in English (e.g. “I want a room for 3 nights.”) The system is to accept grammatically and linguistically correct responses and reject those incorrect either in grammer or in meaning. So the student is able to practise their oral communication skills in an interactive dialogue.

The system includes two part as shown in Figure 4.1: speech processing and text processing. During speech processing, a ASR system is applied to transcribe the student’s response, and then the recognised output is accepted if it is in the response grammar. A version of the existing CALL-SLT response grammar is provided as baseline, which contains 565 prompts and 43,862 possible responses in total.

The DNN-HMM ASR system introduced in this chapter was used as the baseline ASR system for the speech processing task. The challenges in building ASR for this task include: the in-domain data is limited, the speech is from non-native teenagers and is spontaneous, the data is collected in an real, uncontrolled environment. The details of the ASR system will be present in this section.

4.2 Spoken CALL Shared Task Data

4.2.1 Spoken CALL data for 1^{st} edition

The Shared Task data was collected in 15 school classes at 7 different schools in Germanophone Switzerland during a series of experiments in 2014 and early 2015 [10]. Audio files were manually transcribed by native German speakers fluent in English. Also, each prompt response were judged by three native English speakers according to its linguistic and gramatic correctness. The training corpus and the testing corpus contained 5,222 utterances (4.8 hours) and 996 utterances (0.89 hours), respectively. The numbers of different genders are balanced and the average age is around 14 in both training and testing corpus. During development, 90% of the training corpus was used to train the ASR and 10% of the corpus to evaluate the ASR performance, denoted as ST1_train and ST1_eval, respectively.

4.2.2 Spoken CALL data for 2^{nd} edition

For the 2^nd edition in 2018, a new subset of the corpus consisting of 6698 utterances (5.99 hours) was annotated to serve as additional training data. This is denoted as ST2_train. There is no overlap of individual students between the 1^{st} task and 2^{nd} task. The ST2_train was splited into 10 sub-sets. The best system from the 2017 submissions were evaluated on these 10 sub-sets, and the evaluation set during development was composed of the ST1_test and the two sub-sets with the best and the worst WER. The evaluation set was denoted as

ST12_eval. The rest of the ST2_train and the training corpus from the 1^{st} edition were used as in-domain training data. Subsequently a corpus consisting of 1000 utterance (0.91 hours) was released at test data, denoted as ST2_test.

4.3 Baseline DNN-HMM ASR for 1^{st} edition

4.3.1 Front-end speech processing

Front-end speech processing for ASR is as introduced in Section 2.2. The sampling rate of shared task data is 8kHz,thus the out-of-domian data corpus was down-sampled to 8kHZ. 25ms Hamming windows with 10ms shift was applied to the signal. For the GMM-HMM system, speech signal frames were represented using 13-dimensional MFCCs (Section 2.2.1) with delta and delta-delta coefficients appended, forming 39-dimensional feature vectors.

4.3.2 Language model and dictionary

A basic bigram language model (Section 2.5.5) was trained on the transcriptions of Shared Task training data for decoding. The BEEP dictionary was used to provide phonetic transcription of words with a phone set of 44 phones. This dictionary contains the phonemic transcriptions of approximately 250,000 English words (including words with multiple pronunciations, and words with non-letter symbols).

4.3.3 GMM-HMM monophone system

A GMM-HMM monophone system (Section 2.5.2) was initially trained with the 39-dimensional MFCC features vectors. The features were aligned to the phone-level transcription generated based on the BEEP pronunciation dictionary. In Kaldi, suffixes such as '_B', '_E', '_I' and '_S' are added to each of the phones to indicate begin, end, internal and singleton position in the word. Thus, there are 176 non-silence phones in the monophone system. The alignment generated from the monophone system was applied to train a triphone GMM-HMM (Section 2.5.2).

4.3.4 GMM-HMM triphone system

The triphone GMM-HMM system provides the triphone decision tree and the state level time alignment.

4.3.5 DNN-HMM system

For the DNN-HMM (Section 2.5.2) system, the input to the neural network is a 195-dimensional feature vector obtained by splicing together 15 frames (7 frames on each side), each frame is represented using 13-dimensional MFCCs. The hidden layers of the DNN are unsupervised pre-trained as a stack of RBMs (Section 2.4.2). The output layer is initialized randomly. Different network configurations were explored, with hidden layers from 4 to 6 and 1024,2048 or 4096 neurons in each layer. The network with 4 hidden layers and 1024 neurons in each layer works the best. The size of the softmax output layer is 1516, which is given by the decision tree from GMM-HMM system.

During DNN training, training data was split into 90% and 10% for cross-validation. The 10% of data was used as validation data and the DNN parameters were computed from the 90% data.

4.3.6 Experiments results

Three baseline deep neural network (DNN)-hidden Markov model (HMM) hybrid systems were trained on the training part of WSJCAM0 (WSJCAM0_TR) and ST1_train. Base_DNN1, Base_DNN2 and Base_DNN3 were trained on 1.WSJCAM0_TR, 2. WSJCAM0_TR plus ST1_train, and 3. ST1_train, respectively.

The results of recognition experiments on the ST1_test (Table 4.1) show that Base_DNN1 and Base_DNN2 with WSJCAM0 performed worse than Base_DNN3. This may be because the majority of the training set was from WSJCAM0, and the adult read speech from WSJ-CAM0 is not like the speech from the task. Thus the recognition performance on Spoken CALL test data was not good.

To make the system perform better on the shared task data, I further trained the network from the Base_DNN2 system with ST1_train to obtain DNN4. This involved keeping the DNN structure unchanged, randomising the weights in the last layer and then retraining the entire DNN using only the ST1_train data. With 13.92% word error rate, DNN4 outperforms the other three baseline systems. The DNN4 DNN-HMM system was provided as the baseline ASR model for the 2017 1st edition challenge.

Table 4.1 *Performance of the Spoken CALL Shared Task baseline ASR systems for 1st edition on ST1_test, WER(%)*

system	Training data	WER
Base_DNN1	WSJCAM0	72.12
Base_DNN2	WSJCAM0 + ST1_train	19.62
Base_DNN3	ST1_train	16.61
DNN4	ST1_train	13.92

4.4 Development of ASR for the 2^{nd} edition CALL Shared Task

4.4.1 DNN-HMM baseline trained on fMLLR transfromed features

For the 2^{nd} challenge in 2018, the AMI corpus (Section 3.4.2) was utilized to develop the baseline ASR, as it contains conversational speech of native and non-native speakers in English. This section also explores augmenting the ST training data ST12_train with 20% of AMI-IHM data and with 50% of AMI-IHM ata. The training process of the baseline DNN-HMM is the same as introduced in Section 5.3.3. The method for DNN adaptation applied in the 1^{st} edition is used here as well. The baseline systems were further trained with fMLLR (Section 2.5.7) transformed ST12_train data only. Results in Table 4.2 show that the use of 50% AMI-IHM provided better recognition performance and as such this was used in all of the developed ASR systems. Given the fact that more AMI-IHM data does not significantly improve the performance of recognition on shared task data, i.e. only 0.04% WER reduction, and it takes more computational cost, experiment with 100% AMI-IHM was not conducted.

Table 4.2 *Evaluation results (WER%) of Spoken CALL Shared Task DNN-HMM baseline ASR systems for 2^{nd} edition, when using different amount of AMI-ihm data*

Model	Data_Aug	Feature	ST12_eval
DNN-HMM	IHM20	fMLLR	12.68
	IHM50	fMLLR	12.64

4.4.2 Long Short-Term Memory

The LSTM networks (Section 4.3) were trained based on the alignments obtained from the current best DNN-HMM system. 13-dimensional MFCCs and 40-dimensional fMLLR features were compared, both with context of 5 frames (i.e. ± 2 frames). The LSTM network has 1024 memory cells, a hidden layer with 1024 units, a recurrent projection layer with 256 units and a non recurrent projection layer with 256 units.

Results, presented in Table 4.3, show that LSTMs trained on 20% of AMI-ihm perform better than 50% of AMI-ihm on ST12_eval, and also outperform the DNN-HMM baseline. When comparing the results of the two set of features, inconsistent performance improvements on fMLLR features can be seen for different models. When using smaller amount of AMI-IHM, fMLLR transformed feature performed better than MFCC feature, which is to our expectation. While using 50% AMI-IHM, MFCC feature shows slightly better WER than fMLLR transformed feature.

Table 4.3 *Evaluation results (WER%) of Spoken CALL Shared Task LSTM ASR systems for 2^{nd} edition, when using different amount of AMI-ihm data and features.*

Model	Data_Aug	Feature	ST12_eval
LSTM	IHM20	MFCC	12.79
	IHM20	fMLLR	12.11
	IHM50	MFCC	12.82
	IHM50	fMLLR	13.11

4.4.3 Sequence discriminative training

Systems introduced so far were trained to model the label posterior probability based on the cross-entropy criterion, which treats each frame independently. However, speech recognition is a sequence classification problem, sequence discriminative training criterion (Section2.5.4) may have better performance. The state level minimum Bayes risk (sMBR) were applied. For each iteration, the alignments and word lattices for ST12_train data were generated using the cross-entropy trained DNN.

Table 4.4 *Evaluation results (WER%) of Spoken CALL Shared Task sMBR ASR systems for* 2^{nd} *edition, when training with different iterations and learning rate*

Data_Aug	learning rate	nnet1	nnet2	nnet3	nnet4
IHM20	$1e^{-5}$	12.4	12.56	**12.28**	12.4
	$1e^{-6}$	12.89	12.82	12.74	12.7
	$1.25e^{-7}$	12.78	12.75	12.86	12.93
IHM50	$1e^{-5}$	12.09	12.10	**12.00**	12.01

Results for 4 iterations of sMBR training with different learning rates are shown in Table 4.4. sMBR training with 3 iterations and a learning rate of $1e^{-5}$ achieved the best performance so far. The systems were over-fit to the ST12_train with more iterations. Comparing the results in Table 4.3 and Table 4.4, it can be observed that during the development stage the sMBR trained DNNs slightly outperformed the LSTMs. The best WER is 12.00% by the sequence training model using 50% of AMI-IHM.

4.5 Summary

ASR plays an important role in computer assisted language learning. The ASR models presented here achieved good performances, which contribute to the final judgement of whether to accept or reject the student's response. This case study can be seen as a successful application of ASR in education domain.

DNN adaptation for the baseline ASR in 1^{st} edition achieved a WER of 13.92%. Among the 20 submissions for the 1^{st} edition shared task , 11 of them used the transcription from the baseline ASR presented here (Baur et al. 2017). The team that won the challange developed their ASR system based on this baseline ASR and achieved a WER of 9.16% by training with the AMI, PF-STAR(German) (Batliner et al. 2005) and Shared Task corpora (Qian et al. 2017). Oh et al. (2017) developed their DNN-HMM ASR system by adapting an common-domain ASR system for Korean speakers using the Shared Task data and adding English pronunciation variants that are frequently mispronounced by the German people to pronunciation model. They have achieved a WER of 14.9%.

During the development of these ASR model for the 2^{nd} edition, the suitability of out-of-domain training data such as WSJCAM0 and AMI was explored. 50% AMI ihm data has shown to be more suitable for training models for the shared task data. LSTM were explored and achieved a slight WER reduction. Regarding DNN training, different training criteria were compared. sMBR trained DNNs achieved further WER reduction.

The experience from this case study was subsequently applied to building ASR system for MI. Since in these two applications, one common challenge is that the in-domain data are both limited.

Chapter Five

Automatic speech recognition for MI

5.1 Introduction

The assessment of clinicians' performance is based on the transcription of the clinicians' speech, so ASR is an essential part to automate the assessment. In this chapter, ASR technologies will be applied to MI. The experiments in this chapter will focus on the following issues:

1. There are several open-source datasets for training ASR systems, like WSJCAM0 (refer to Section 3.4.1) and AMI (refer to Section 3.4.2), which one, or combination is better for building ASR for recognition of MI data?

2. With a limited in-domain dataset and a large out-of-domain dataset, is it better to combine the two dataset during training, or to train a out-of-domain ASR and then conduct adaptative training?

3. How does the quality of alignments affect the ASR performance?

4. How does the fMLLR transforms affect the ASR performance?

5. What is the best training criterion for the application of ASR to a small dataset with poor recording quality like MI?

5.2 Dictionary and Language Model

Same as the ASR system for Spoken CALL shared task, the BEEP dictionay was used. There were 303 words in the MI corpus which were not present in the BEEP dictionary. Most of these words were medical terms related to diabetes. The phonetic transcription of these words were obtained manually using phonetic rules and the words were then added to the dictionary.

A trigram language model was trained using an online diabetes text corpus (refer to Section 8.2.2) as well as all the human transcripts of MI sessions excluding the testing data.

5.3 Acoustic modeling baseline

To explore the first issue, experiments following the standard Kaldi recipe with training data from WSJCAM0 and AMI-sdm were conducted and evaluated on MI testing data. This recipe was also used as the baseline system. Initially the waveform was down-sampled to 16 kHz and 25 ms Hamming windows with 10 ms spacing were applied to the signal. Speaker-level cepstral mean normalisation(CMN) was applied to the MFCC features at the beginning to reduce the effect of noise. The sequence for building the monophone and triphone GMM-HMM systems are the same as the ASR system for Spoken CALL shared task. The alignments generated from the triphone GMM-HMM system were used to train the following systems:

5.3.1 Triphone-LDA-MLLT

A 91-dimensional MFCC is comprised by splicing the 13-dimensional MFCC in time with a context of 7 (i.e., $\pm$ 3). Then the dimension of the MFCC feature vectors was reduced to 40 using linear discriminant analysis (LDA). The features were futher de-correlated with maximum likelihood linear transform (MLLT). Thus, the features can be more accurately

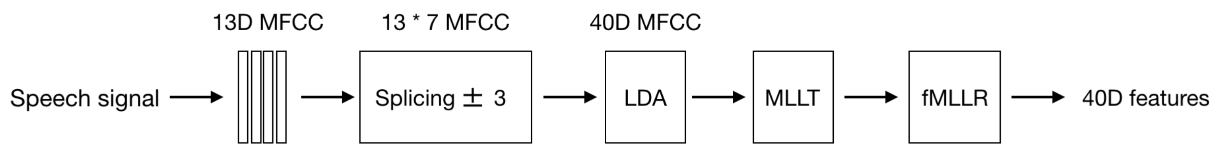


Figure 5.1 Generation of the baseline features.

modeled by diagonal-covariance Gaussians.

5.3.2 Speaker adaptative training

The fMLLR (Section 2.5.7) was applied to discover a linear transform for each speaker (or each session if the speaker information was unknown) to maximize the probability of the data given the speaker independent model. When the fMLLR transforms for training data and testing data are learned from the same speaker independent model, it can be seen as a way of speaker normalisation, since it removes the variations between different speakers.

For training data, the fMLLR transforms were learned in a supervised way. Because the speaker identity is not given in the MI data, it is not clear that whether the test speakers appear in the training data. Thus fMLLR transform learned from training data couldn't be used for testing speakers. As for testing data, transcripts are not available. An initial decoding pass without fMLLR is applied to get the text. The text was used to get initial fMLLR transformation for each testing speaker. Then the testing data was decoded with the fMLLR. New transcripts were used to re-estimated the fMLLR transform. The process was iterated until the final decoding.

The process of the generation of the fMLLR transformed features is shown in Figure 5.1.

5.3.3 DNN-HMM

The training procedure for DBN-DNN is described in Section 2.4.2. A DBN with 6 hidden layers and 2048 hidden units in each layer was trained on the 40 dimensional fMLLR transformed features and was used as the pre-training for DNN. A DNN was created by adding a softmax layer onto the DBN. The alignments provided by the SAT GMM-HMM system was used to fine-tune the parameters of the DNN by error back propogation. The training data was split into 90% and 10% for cross-validation. Early stopping was applied to prevent the network from over-fitting.

5.3.4 Experiments results

Table 5.1 shows the results of evaluating the above 5 acoustic models trained with different data corpora. From monophone to DNN-HMM, every acoustic model was trained based on the alignments generating from its preceding model. When evaluating the out-of-domain acoustic models, all of the acoustic models trained on AMI-sdm (B1) outperformed the models trained on AMI-ihm (B2). It is to our expectation, because the acoustic condition in AMI-sdm is close to the MI data. They are both far-field speech. The WERs from the evaluations of WSJCAMO trained models (B0) were much higher than AMI, this suggests that the read speech is not suitable for training ASR for conversational speech like MI. The best WER for now is from the DNN-HMM model trained on MI_train only. Thus the rest of the ASR experiments are to explore the best way to combine the large amount of out-of-domian data (e.g. AMI-sdm) and limited in-domian data (e.g. MI_data).

Table 5.1 *Evaluation results on MI_test, WER (%), for acoustic modeling with different training corpora.*

Model	Training data	monophone	tri1	tri-LDA-MLLT	SAT	DNN-HMM
B0	WSJCAM0	91.94	91.65	93.49	100.14	-
B1	AMI-sdm	85.37	75.10	72.97	64.62	55.38
B2	AMI-ihm	88.93	86.51	88.89	76.34	68.25
B3	MI_train	76.59	67.04	66.16	61.39	54.41

5.4 Quality of alignment

The performance of AMI-ihm ASR system evaluation on AMI-ihm testing data (28.30% WER) is better than AMI-sdm ASR system's evaluation on AMI-sdm testing data (60.86% WER). Apart from the challenging nature of the acoustic condition of AMI-sdm, the poor alignment is also a major factor to train a good acoustic model for ASR. Given the fact that the recording in AMI-sdm and AMI-ihm were made simultaneously, we use the alignment from AMI-ihm ASR system to train the model with AMI-sdm acoustic data.

As shown in Table 5.2, a 2.88% and a 4.49% absolute WER reduction were obtained compared to system with AMI-sdm alignment for GMM-HMM and DNN-HMM systems, respectively. Since model S1 achieved the current best WER for MI_test, we expected it can provide better alignment for MI_train. Hence the alignment MI# was obtained from model S1 and achieved a better WER in model S3. For the rest of the experiments in this paper, we will use MI# and AMI-ihm for the alignments of the MI_train and AMI-sdm dataset.

Table 5.2 *Evaluation results on MI_test, WER (%), with different alignments*

Model	Training data	Alignment	SAT GMM-HMM	DNN-HMM
S0	AMI-sdm	AMI-sdm	65.96	55.38
S1	AMI-sdm	AMI-ihm	63.08	50.89
S2	MI_train	MI	59.45	54.41
S3	MI_train	MI#	58.63	48.23

5.5 fMLLR adaptation

The datasize in the out-of-domain data corpus AMI-sdm is much larger than MI_train. Large datasize can help to train a good ASR system but the mismatch between training and testing data will degrade the performance. The models in Table 5.3, S4 to S7, used different ways of combining the two datasets as well as fMLLR adaptation to make the best use of the two datasets.

5.5.1 fMLLR adaptation to a AMI + MI trained model

As shown in figure 5.2, a generic SAT model was trained on a data corpus combined with AMI_sdm and MI_train. fMLLR transforms for each speaker were learned from this generic model. S4 and S6 denoted the models trained with features been adapted to this generic space.

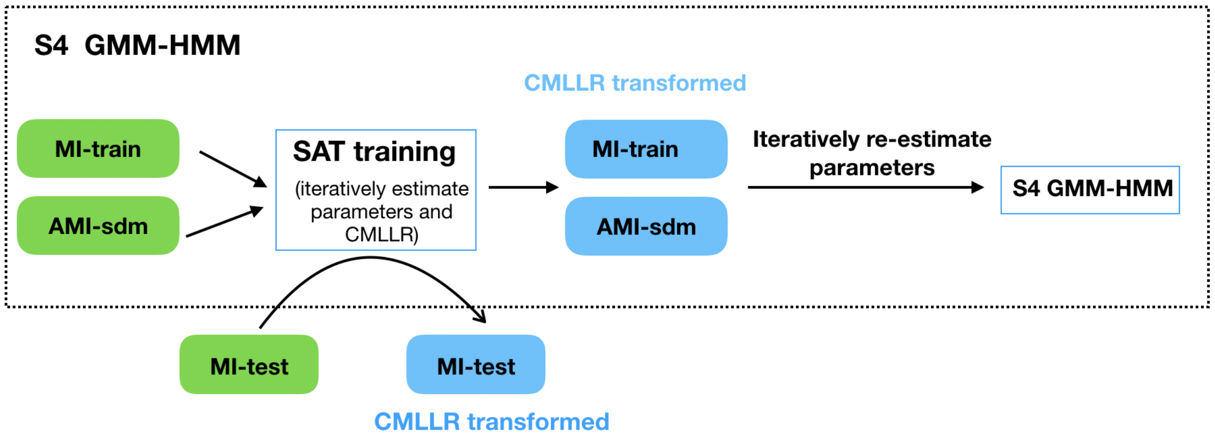


Figure 5.2 Training procedure for Model S4

5.5.2 fMLLR adaptation to a AMI trained model

In Figure 5.3, the SAT model that was used to learn fMLLR transforms was trained on AMI_sdm only, which resulted in the features were adapted to a more compact space. In model S5 and S7, the data were fMLLR transformed to this compact space.

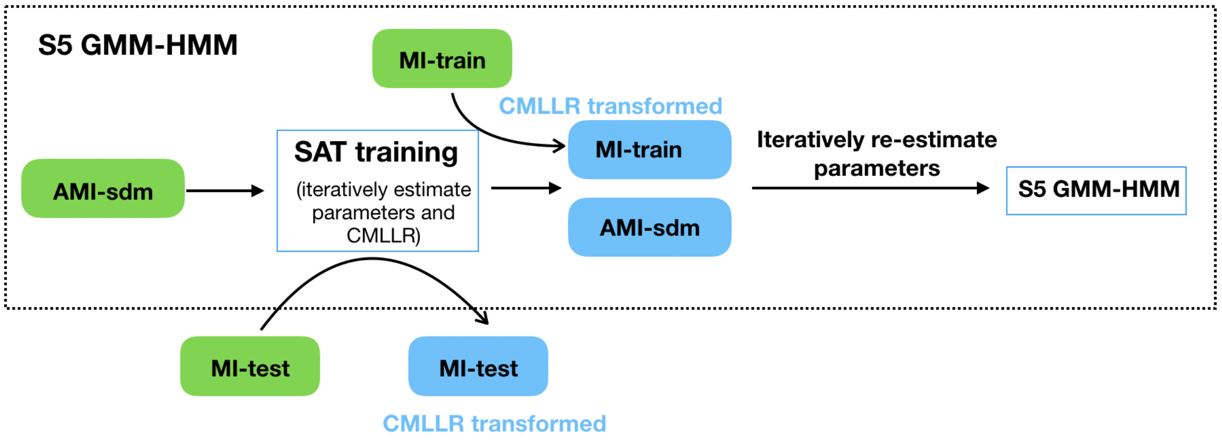


Figure 5.3 Training procedure for Model S5

5.5.3 Experiment results

The results of the experiments described above in Section 5.5.1 are summarised in Table 5.3. Systems in S5 and S7 followed the methodology described in Figure 5.3, and the only difference is that S7 only trained with transformed MI_train data. The AMI-sdm and MI_train data were adapted to the S1 GMM-HMM model, denoted as AMI-sdm$_{S1}$ and MI$_{S1}$, respectively. Systems in S4 and S6 were trained with the procedure described in Figure 5.2. The training data in S6 is adapted to the GMM-HMM model in S4, denoted as MI$_{S4}$.

It can be seen from the table that models in S5 outperformed S4 in terms of GMM-HMM as well as DNN-HMM significantly. When trained on MI_train data only, DNN-HMM model in S7 achieve better performance than S6. The WER reduction obtained by DNN-HMM from S4 to S5 is similar to the reduction from the DNN-HMM in S6 to S7, about $3\% \sim 4\%$. These suggest that fMLLR is more effective when adaptation is performed relative to the more compact AMI-trained model (S1) rather than a model that was constructed using a combination of AMI and MI data (S4). S5 achieved the best DNN-HMM performance so far.

When comparing the results of S6 and S7 in this table with the results of S3 in Table 5.2, DNN-HMM in S3 outperformed the DNN-HMM models in S6 and S7. The training data was adapted to a GMM-HMM model trained on MI_train only, which was more compact than AMI-trained model and model trained on combination. This is consistent with the conclusion above.

Table 5.3 *Results, WER (%), for MI training with cross-domain fMLLR adaptation.*

Model	Training data	GMM-HMM	DNN-HMM
S4	AMI-sdm+MI	59.16	52.97
S5	AMI-sdm$_{S1}$+MI$_{S1}$	56.68	49.00
S6	MI$_{S4}$	56.61	52.59
S7	MI$_{S1}$	56.84	49.16

5.6 DNN adaptation with Cross-Entropy

The systems in S1 were trained on AMI-sdm only and the systems in S5 were trained on a combination of AMI-sdm and MI_train. The size of AMI-sdm training data was much larger than the size of MI_train data. Thus the DNN-HMM model trained on these were dominated by AMI-sdm data. DNN adaptation was applied to improve the performance on recognition of MI testing data.

DNN adaptations on S1 and S5 were done by re-aligning the MI$_{S1}$ using S5 DNN-HMM, randomising the weights in the final layer of the baseline DNN and then re-training the whole DNN with cross-entropy (CE) criterion using only the MI$_{S1}$ data. Results are presented in Table 5.4. It can be seen that this resulted in considerable WER reduction. The DNN-adapted-CE models achieved 3.45% and 5.28% absolute WER reduction compared with the original DNN model S1 and S5, respectively.

Table 5.4 *Recognition performance, WER (%), on MI_test obtained by DNN adaptation with CE training.*

DNN model	DNN-adapt-CE	WER reduction
S1	45.61	3.45
S5	45.55	5.28

5.7 DNN adaptation with sequence descriminative training

With the alignment and lattice generated from the DNN models S1 or S5, I also conducted the same DNN adaptation experiments in section 5.6 with 1-4 iterations of sMBR sequence descriminative training (denoted as nnet1-4). The results obtained using these systems are shown in Table 5.5.

The DNN adaptation trained with one iteration outperformed the DNN adaptation with CE. The WER first decreased and then increased with more iterations due to over-fitting. The best performance 43.59% WER was obtained by S1 DNN adaptation with 3 iterations of sMBR training.

Table 5.5 *Recognition performance, WER (%), on MI_test obtained by DNN adaptation with sMBR training.*

DNN-model	nnet1	nnet2	nnet3	nnet4
S1	44.18	44.28	43.59	45.04
S5	44.21	43.94	43.72	43.84

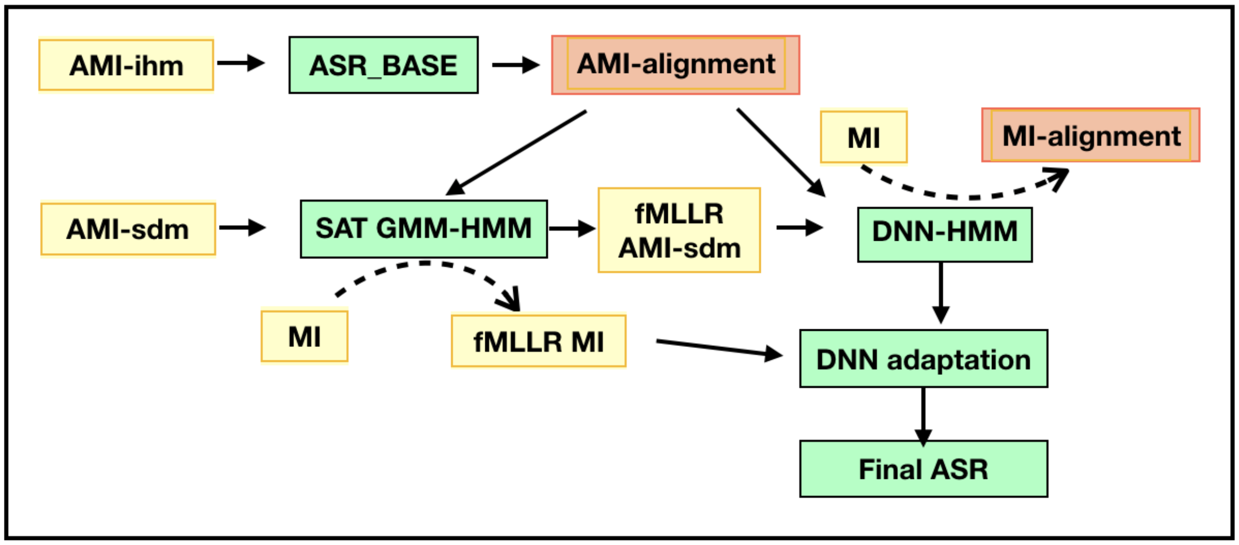


Figure 5.4 Diagram for the final ASR system.

5.8 Final ASR system

Figure 5.4 shows a diagram for our final ASR system that we used to get the transcription for the automatic assessment system. First, data from AMI-ihm were used to train a baseline ASR system and AMI-alignment were generated using this ASR system. With AMI-alignment and data from AMI-sdm, a SAT GMM-HMM system was trained. FMLLR transform for each speaker from MI and AMI-sdm were computed from this SAT system. The transformed data from MI and AMI-sdm were denoted as fMLLR-MI and fMLLR-AMI-sdm, respectively. MI-alignment was generated from a DNN-HMM ASR system trained only on fMLLR-AMI-sdm. The DNN adaptation was conducted using only fMLLR MI and the MI-alignment to get our final ASR system.

5.9 Summary

When building ASR system for MI, the suitability of the use of out-of-domain datasets, WSJCAM0 and AMI, was explored. The model built on AMI-sdm outperformed the one built on AMI-ihm or WSJCAMO on recognition of the MI data. In the DNN-HMM hybrid system, the DNN was trained to estimate the label posterior probability based on a given data alignment. The quality of the alignment could considerably affect the quality of the trained models, resulting in a difference of the ASR performance. The AMI-ihm data were used in initial stages of the training process to provide alignments which were then used with the AMI-sdm data to train the acoustic models.

The experiments explored fMLLR transformation, which transformed the LDA-MLLT MI feature vectors to a speaker-independent model (e.g. SAT models trained with AMI data). When it was conducted on training and testing data, the features can be seen as normalized (speaker information was suppressed). This improved performance on the GMM-HMM system as well as the DNN-HMM system. The results suggested that fMLLR transformation towards a more compact model trained on AMI only was more effective when compared to a model that was trained on a combination of AMI and MI.

The experience gained from the Spoken CALL shared task has also been applied to DNN adaptation for MI, that is randomising the weights in the final layer of the baseline DNN and re-training the DNN with CE criterion using only the in-domain data. DNN adaptation with sMBR criterion has also been explored, and the best ASR system was obtained by DNN adaptation with 3 iterations of sMBR training. The best WER for recognition of MI_test is 43.59% so far, based on only 7 hours of in-domain training data. Compared to the ASR system built by Pan et al. (2020) for dementia detection, they have applied transfer learning with 7 hours 40 minutes in-domain data based on an TDNN model trained on 64 hours of conversational recordings between doctors and patients, and acheieved a WER of 32.3%.

Chapter Six

Speaker diarisation

6.1 Introduction

Since the assessment of a motivational interviews is based on the clinician's speech, a speaker diarisation system that can automatically split the clinician's speech from the patient's speech is an essential part towards the automatic assessment. The speaker boundaries obtained from the diarisation results can also help with ASR. As introduced in Section 5.5, the fMLLR adaptation is applied at speaker level.

A flowchart for the speaker diarisation system is shown in Figure 6.1. The speaker features extracted from the input audio were 39 dimensional MFCC features with delta and acceleration. First, the input audio is segmented into short segments with overlap. It is assumed that there is no speaker change in each temporal segment. Then CMN is applied to remove the possible convolutional noise, and the non-speech parts are removed using energy-based voice activity detection (VAD). An i-vector (a low dimensional representation) is extracted for each segment. Last, the extracted i-vectors are clustered into 2 clusters and further smoothed to produce the final diarisation results

These five issues are explored in these chapter:

1. For temporal segmentation, a longer segment length provides more information about speech, it may compromise the accuracy of finding the speaker change point. So what

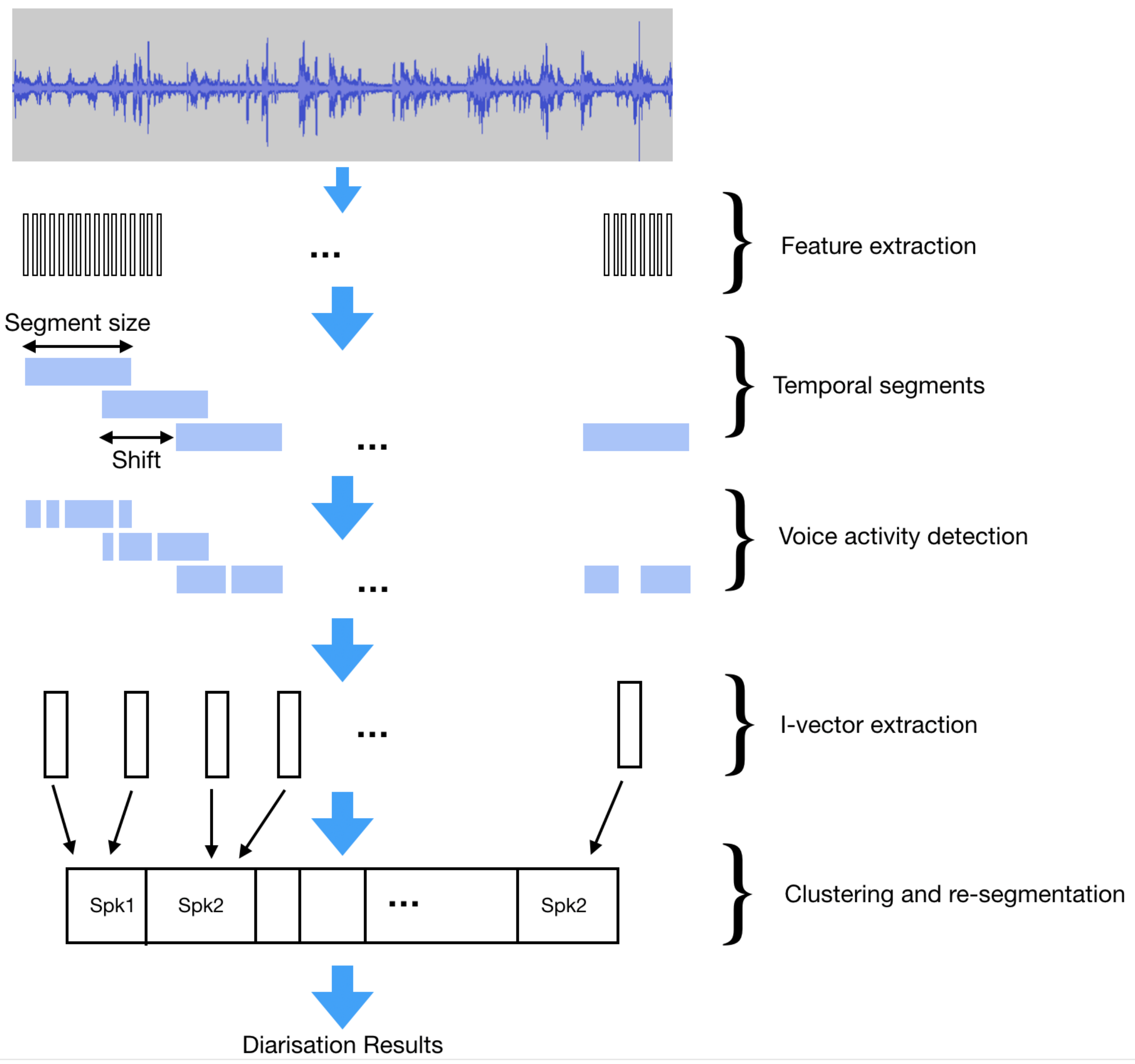


Figure 6.1 A flowchart for the i-vector based speaker diarisation.

is the most suitable length of segments and overlap?

2. What is the best configuration for i-vector extraction?

3. What is the best way for i-vector clustering?

4. Does the speaker boundaries obtained from the speaker diarisation system help with the ASR?

The rest of this chapter will be organised as follows: I-vector extraction method is introduced in section 2.6.4 and the evaluation criterion will be introduced in section 6.2. The experiments I have conducted to explore the best way to train UBM and to find the most suitable dimension for UBM and i-vectors are in section 6.6.2, section 6.6.4, respectively. Different ways for i-vector clustering are illustrated in section . Section 6.6.5 shows experiments to find the best configurations for temporal segmentation.

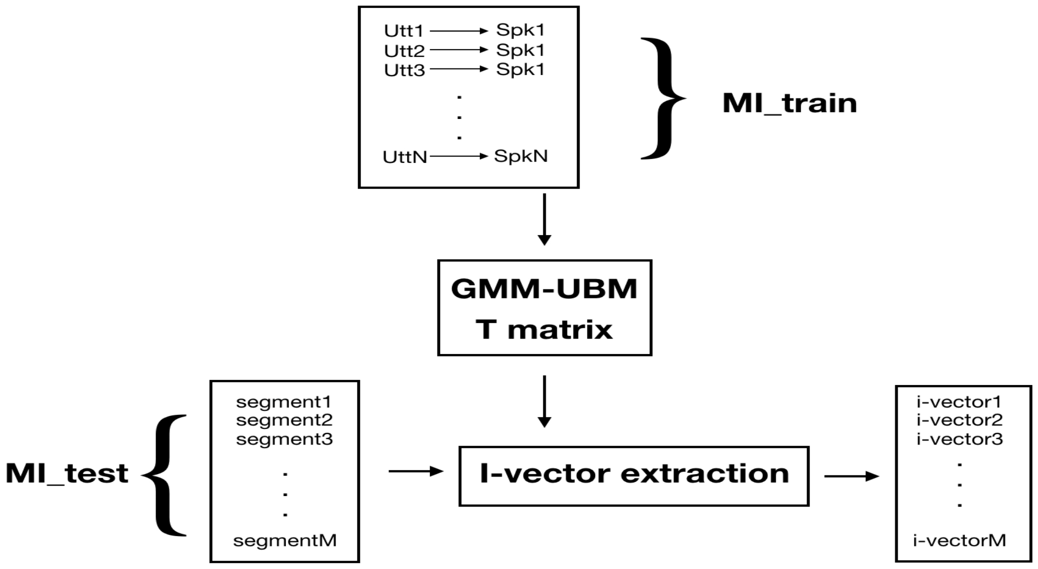


Figure 6.2 A diagram for i-vector extraction.

6.2 GMM i-vector system

A diagram for GMM i-vector system is shown in Figure 6.2. First, a speaker-independent GMM-UBM with 1024 mixture components was trained on speech data from every speaker in MI_train. As mentioned in section 2.2, the front-end analysis for speaker diarisation should preserve the speaker information as much as possible while suppressing the noise signal. Utterance level CMN was applied to remove constant component in the log spectral domain for each utterance. It removed noise that exits over a short period of time and it might also remove speaker information. Since the speaker information is constant during each utterance as well. Speaker level CMN was to subtract the constant component among all the utterances from each speaker in each session, and session level CMN was to apply CMN over each session. VAD was applied to remove the non-speech frames. Section 6.6.2 shows the experiments exploring with different level of CMN for GMM-UBM training.

For the training of T matrix, only session level CMN was applied. That is because T matrix is to map the super-vector of speaker representation to a low dimensional space (the i-vector space), variation between speakers should be preserved. The i-vector dimension is set to 64. In a GMM-UBM i-vector system, the UBM was used to provide posterior probabilities to train the subspace T matrix and to extract i-vectors. Two methods for T matrix training were explored: (1) utterances were treated as from independent speakers (an i-vector was extracted for every utterance), denoted as *utt-T* or; (2) actual speaker information was used to extract i-vectors, denoted as *spk-T*. The experiments demonstrate the effect of these two T matrix are in section 6.6.3.

For MI_test data, only session level CMN was applied. I-vectors for the temporal segments from a recording were then extracted using the trained GMM-UBM and T matrix.

6.3 DNN i-vector

In traditional i-vector system, each component of GMM-UBM is corresponding to a sub-phonetic class. But it is not guaranteed that the GMM-UBM trained in an un-supervised way can be phonetic-aware.

The DNN acoustic models built for ASR were introduced to provide frame posterior for senones (tied triphone phone states from the ASR decision tree) as shown in Figure 6.3. Speaker features can be different from ASR features in this pipeline. VAD was applied to speaker features. But for the inputs to the DNN, VAD will affect the correct context information. Thus the VAD results were used to remove the posteriors corresponding to non-speech frames. The posteriors were used to compute the sufficient statistics and these were all that was needed to train T matrix and to extract i-vectors.

The acoustic models trained on fMLLR features have shown to be effective for ASR, and to provide better posterior. However, fMLLR transformation is believed to remove speaker specific information. For experiments in section 6.6.6, speech recognition features were also used as speaker features to investigate whether the fMLLR transformed features still contain speaker information. I also conducted experiments comparing statistics from different ASR

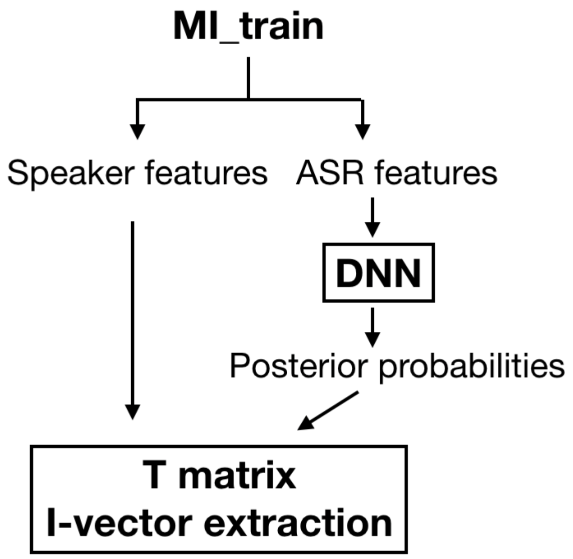


Figure 6.3 A diagram for incorporating ASR DNN acoustic model into i-vector extraction.

DNN acoustic models in section 6.6.6.

6.4 I-vector clustering and re-segmentation

For each interview session, i-vectors for the temporal segments were then clustered into two clusters. It was known that there were two speakers in a given MI session. Two unsupervised clustering technology methods were compared: hierarchical clustering and GMM based clustering.

6.4.1 Hierarchical clustering

In agglomerative hierarchical clustering, each data point was considered as an individual cluster in the beginning. Then for each iteration, similar clusters were merged until two clusters remained. The average, maximum and minimum of the distances between data points from the sets were computed to obtain the three types of similarities between the clusters. Cosine similarity was used as a measurement of distance between i-vectors.

6.4.2 GMM based clustering

A GMM (λ_s) with two components was trained on the i-vectors for each session in an unsupervised way. Each component corresponded to a speaker, but the identity of the speaker was unknown. The decision on who the k^{th} temporal segment of speech belonged to (S_k) was then based on the maximum a-posterior probability of the segment's i-vector $w(x_k)$ belonging to one of the GMM components.

$$S_k = \underset{i=0,1}{argmax}\ p(i|w(x_k)) = \underset{i=0,1}{argmax}\ \frac{w_i\mathcal{N}(w(x_k);\mu_i,\Sigma_i)}{\sum_{j=0}^{1} w_j\mathcal{N}(w(x_k);\mu_j,\Sigma_j)} \tag{6.1}$$

This resulted in a speaker sequence based on the assumption that who the current segment belonged to was independent of the previous segments.

6.4.3 HMM smoothing - Viterbi decoding

As shown in the histogram of the utterances' lengths in MI training data in section 3.3, the speaker turns are more likely to last more than 2 seconds. Which means the speakers are not changing very frequently. The resulting sequence of speaker identities could be smoothed, for instance, using simple median filtering. I took a more generic approach and modelled the speaker changes in a conversation with an two-state Markov model as shown in Figure 6.4. At each state, the value of the self-loop probability a is the probability of staying at the same speaker and it expressed a typical speaker-turn duration. Then $1-a$ is the probability of changing speaker. The a could be estimated using actual speakers' turn training data but in this thesis the self-loop probability was set from 0.1 to 0.9 with 0.1 interval and from 0.9 to 0.99 with a 0.025 interval. Best result for each experiment was presented.

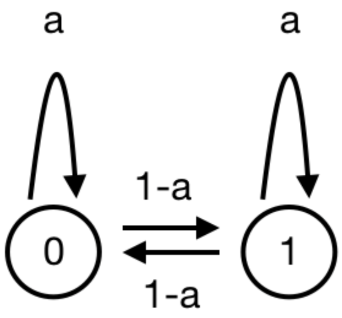


Figure 6.4 Two state Markov process.

The most probable speaker change sequence (the diarisation results) can be found using the Viterbi algorithm. The probability of current segment k belongs to speaker j and having been through the most probable sequence of $k-1$ preceding speakers is denoted as $\hat{\alpha}_j(k)$, and $\hat{\alpha}_j(k)$ is computed as below:

$$\hat{\alpha}_j(k) = \underset{i=0,1}{max}(\hat{\alpha}_i(k-1)a_{ij})p(j|w(x_k)) \quad \text{for} \quad 1 < k \leq T \tag{6.2}$$

Let $\hat{\pi}_j(k)$ denote the most probable state at $k-1$ for being state j at k

$$\hat{\pi}_j(k) = \underset{i=0,1}{argmax}(\hat{\alpha}_i(k-1)a_{ij}p(j|w(x_k)) \tag{6.3}$$

Defining $\hat{\alpha}_F(T)$ as the probability of all the segments being aligned to the most probable speaker sequence $S_1, ..., S_T$, the value is given by:

$$\hat{\alpha}_F(T) = \underset{i=0,1}{max}(\hat{\alpha}_i(T)) \tag{6.4}$$

Thus the optimal speaker sequence S_k is obtained by tracing back through $\hat{\pi}$ starting at state F ar time T.

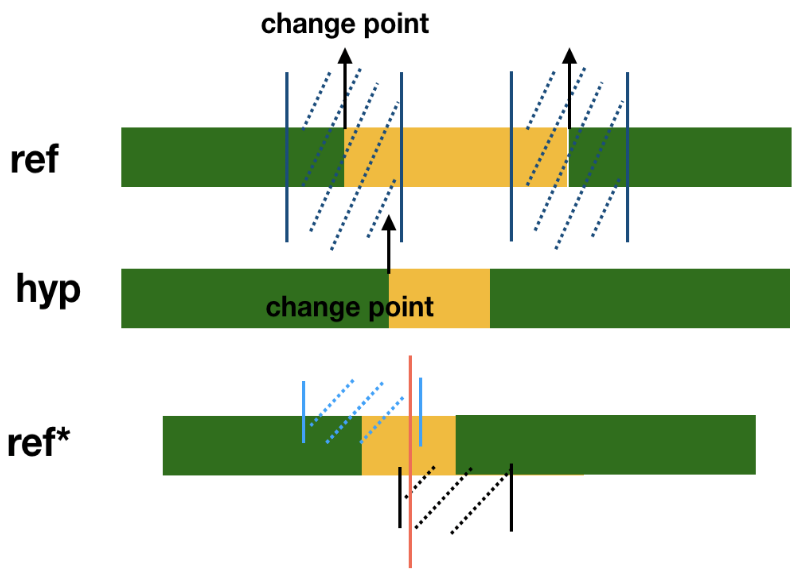


Figure 6.5 Diagram for the final ASR system.

6.5 Evaluation criterion

As shown in Figure 6.5, the hypothesis change point is considered to be correct if it is within ±1.5 seconds of the true reference change point. If two true change points are closer than 1.5 seconds, the middle point is taken to be the decision boundary. The hypothesis change point is given by the start time of each clustering segments. To deal with the long overlap between temporal segments (for example, 3-second segment with 1.5-second shift), the hypothesis change point is delayed by 0.75 seconds (the middle point of the overlap part).

The speaker diarisation systems were evaluated using the precision (P), recall (R) and the F-measure as below:

$$P = \frac{N_c}{N_{hyp}} \tag{6.5}$$

$$R = \frac{N_c}{N_{ref}} \tag{6.6}$$

$$F = \frac{2.0 * P * R}{P + R} \tag{6.7}$$

where N_c, N_{hyp} and N_{ref} denote the total number of the correct detections, the detected change points from diarisation results (hypothesis) and the actual change points in the reference, respectively. Evaluations of the diarisation system were performed on the MI_test data which contained 315 speaker changes.

6.6 Experiments

6.6.1 Clustering and re-segmentation

Hierarchical clustering

Table 6.1 shows the performance of speaker diarisation when applying Hierarchical clustering as described in section 6.4.1 to group the i-vectors into 2 sets. Three different ways of computing similarities between clusters were found to have the same effect on i-vector clustering.

In Hierarchical clustering, the segments ordering was not taken in to account and there was no prior information on speaker-turn duration. So GMM based clustering and smoothing were applied. It achieved a better performance than Hierarchical clustering and was used as the default method of i-vector clustering in this thesis.

Table 6.1 *Evaluation of the speaker diarisation systems on MI_test data, with hierarchical clustering*

distance between clusters	P	R	F
average	0.838	0.686	0.754
minimum	0.838	0.686	0.754
maximum	0.838	0.686	0.754

GMM clustering with smoothing

As introduced in section 6.4.2, the self-loop probability a was set experimentally. It was set from 0.1 to 0.9 with 0.1 interval and from 0.9 to 0.99 with a 0.025 interval. Figure 6.6 shows

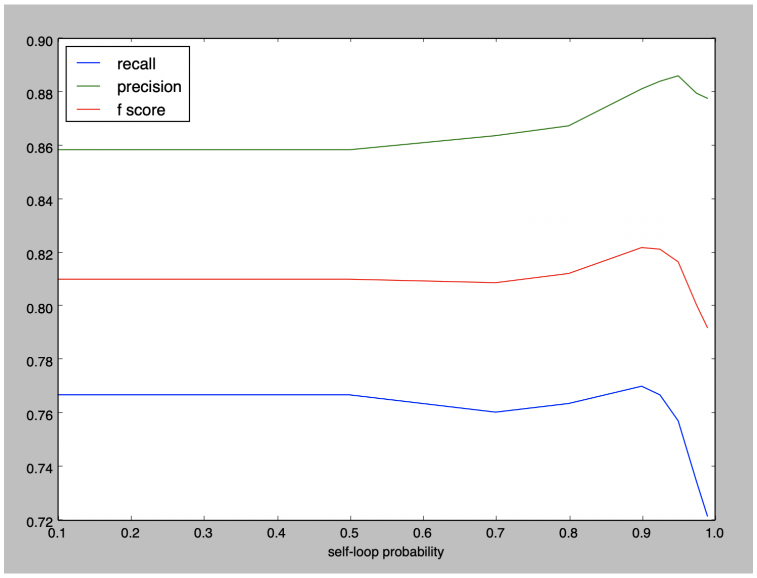


Figure 6.6 recall precision and f score with different self-loop probabilities.

how the speaker diarisation precision, recall and f score change with the self-loop probability. It can be seen that GMM based clustering is better than Hierarchical clustering. For the rest of this thesis, the GMM based method is used as the default seting for clustering and resegmentation. For each experiments, the self-loop probability that gives the highest F score was kept and the corresponding results were presented.

6.6.2 CMN for UBM training

As introduced in section 6.2, different level of CMN might have an effect of UBM training. The results in Table 6.2 are to our expectation, the utterance level CMN for UBM performs best when comparied to speaker and session level. This suggests that the convolutional noise might have changed within the session, thus the speaker- and session- level CMN could not remove it. Utterance level CMN works best for training a speaker- and channel- independent UBM.

For T matrix training, only session level CMN was applied. To summarize, we used utterance level CMN for UBM training and session level CMN for T martix training for the rest experiments in this thesis.

Table 6.2 *Results of the i-vector speaker diarisation systems on the MI_ test data, with different CMN level for UBM and T matrix training. Dimension for UBM and i-vector are 512 and 64.*

Level of CMN		Evaluation measure		
for UBM	for T matrix	P	R	F
utterance	session	0.879	0.754	0.812
speaker	session	0.841	0.751	0.793
session	session	0.844	0.751	0.795

6.6.3 T matrix training

Table 6.3 shows the results of training T matrix with utterance level and speaker level. The i-vector model with utterance level T matrix outperformed the speaker level T matrix. It suggests that utterance level T matrix is better at capturing speaker variance between short segments, although the data for each i-vector is more sufficient during speaker level T matrix training. The utterance level T matrix is used as the default setting for the rest of the experiments in this thesis.

Table 6.3 *Evaluation of the speaker diarisation systems on MI_ test data, with different T matrix training. The dimensions of UBM and i-vector are set to be 1024 and 64.*

level of T matrix	P	R	F
utterance	0.879	0.754	0.812
speaker level	0.848	0.741	0.791

6.6.4 I-vector Configurations

To find the best configuration for the i-vector based speaker diarisation system, table 6.4 shows results of experiments with different numbers of components for UBM and various dimensions for i-vector representation. It shows that UBM with 1024 mixture components and 64 dimensional i-vector perform the best on F-measure. Specifically, 64 dimensional i-vector always outperform the smaller dimension and larger dimension with any UBMs. This suggests that although the state-of-art speaker diarisation systems are using 200 dimensional i-vector or higher, 64 is a suitable dimension to contain enough speaker information for the i-vector extraction given our data size limitation and short length of segments.

Table 6.4 *Results of the i-vector speaker diarisation systems on the MI_ test data, when every utterance from a given speaker was regarded as different speakers for T matrix training. Utterance level CMN was applied for UBM training and session level CMN was applied for T matrix training. Temporal segmentation with 2-second segment and 1.5-second shift.*

Number of components in UBM	Dimension of ivector	Evaluation measure P	R	F
256	32	0.841	0.754	0.795
256	64	0.882	0.748	0.809
256	128	0.832	0.770	0.800
512	32	0.856	0.728	0.787
512	64	0.879	0.754	0.812
512	128	0.886	0.728	0.799
1024	32	0.849	0.748	0.795
1024	**64**	**0.881**	**0.770**	**0.822**
1024	128	0.825	0.764	0.793
2048	32	0.834	0.731	0.779
2048	64	0.832	0.738	0.782
2048	128	0.851	0.722	0.781

Table 6.5 *Results of the i-vector speaker diarisation systems on the MI_ test data, when every utterance from a given speaker was regarded as different speakers for T matrix training. Utterance level CMN was applied for UBM training and session level CMN was applied for T matrix training. Temporal segmentation with 3-second segment and 2.5-second shift*

Number of components	Dimension of	Evaluation measure		
in UBM	ivector	P	R	F
256	32	0.758	0.508	0.608
256	64	0.808	0.544	0.650
256	128	0.746	0.495	0.595
512	32	0.787	0.528	0.632
512	64	0.789	0.534	0.637
512	128	0.798	0.511	0.623
1024	32	0.799	0.528	0.635
1024	64	0.788	0.540	0.641
1024	128	0.749	0.502	0.601
2048	32	0.779	0.515	0.620
2048	64	0.781	0.495	0.606
2048	128	0.759	0.489	0.594

6.6.5 Temporal segmentation

For the first issue at the beginning of this chapter, the effects of using different values of the segment size and shift for temporal segmentation were explored.

First, experiments using temporal segmentation with 3-second segment length and 2.5-second shift are shown in Table 6.5. Thus the overlap between the neighboring segments is the same as the systems in Table 6.4. The results were worse in comparison to the systems with 2-second segment length and 1.5-second segment shift. Specifically, the recall decreased significantly. This was within our expectation, because the longer segment shift would have missed the correct speaker change points.

Results in Table 6.6 showed the experiments where the segments were 3-second long but with 1.5-second shift (the same segment shift as in Tabel 6.4). Thus the total number of temporal segments is the same with the experiment in Table 6.4. The precisions are comparable with the precision from the experiments with 2-second temporal segmentation, but the recalls are worse. This suggested that longer overlap between the neighboring temporal segments made it harder for the system to distinguish if they are from two speakers.

Table 6.6 *Results of the i-vector speaker diarisation systems on the MI_ test data, when every utterance from a given speaker was regarded as different speakers for T matrix training. Utterance level CMN was applied for UBM training and session level CMN was applied for T matrix training. Temporal segmentation with 3-second segment and 1.5-second shift.*

Number of components	Dimension of	Evaluation measure		
in UBM	ivector	P	R	F
256	32	0.841	0.615	0.710
256	64	0.868	0.660	0.750
256	128	0.865	0.641	0.736
512	32	0.852	0.634	0.727
512	64	0.848	0.615	0.713
512	128	0.901	0.651	0.756
1024	32	0.858	0.647	0.738
1024	64	0.874	0.654	0.748
1024	128	0.876	0.663	0.755
2048	32	0.848	0.615	0.713
2048	64	0.862	0.631	0.729
2048	128	0.844	0.612	0.709

6.6.6 DNN-ivector

Experiments in Table 6.7 compared the speaker diarisation systems with different speaker features and posteriors from DNN trained with different training criterion. The number of triphone senones in ASR system is 3981. The dimension of i-vector is set to 64. The MFCC speaker features outperformed the fMLLR transformed features. Interestingly, fMLLR transformed features worked fine in speaker diarisation, which suggested that the fMLLR features still contain speaker information.

But the DNN-ivector did not achieve better performance when comparing to GMM-ivector with same dimension. This might because the senone set was determined by a decision tree to cover all the vocabulary in AMI and MI_train during ASR training. However, the MI sessions might contain only a small part of this senone set. The high dimensional supervector for MI data tends to be sparse.

Table 6.7 *Evaluation of the speaker diarisation systems on MI_test data, when statistics for i-vector extraction were collected from ASR DNN acoustic models.*

speaker feature	ASR acoustic model	P	R	F
MFCC	cross-entropy DNN	0.844	0.699	0.765
MFCC	sMBR DNN	0.814	0.734	0.772
fMLLR	cross-entropy DNN	0.801	0.702	0.748
fMLLR	sMBR DNN	0.824	0.696	0.754

6.6.7 Evaluation of ASR combined with speaker diarisation information

The normalisation and adaptation of the feature vectors for ASR were conducted on speaker level. The speaker information were obtained manually. I explored whether incorporating the speaker diarisation information can automate this process. Specifically, we use the speaker boundaries from the best diarisation results in performing fMLLR adaptation. If no speaker information is present, fMLLR adaptation is applied at session level. Then I can compare the benefit of using the automatic speaker information and manual speaker information. Evaluations of this system, in addition to the use of cleaned MI_test data, which manually excludes segments of noise and overlapped speech, were also performed on 'raw' uncleaned MI_test data. Results are presented in Table 6.8. It can be seen that a 2.37% and a 1.72 % absolute WER reduction is achieved by using fMLLR informed by the automatic speaker diarisation on cleaned and uncleaned 'raw' MI_test data, respectively. However, using manual speaker change information can achieve a 4.34% and a 3.17% absolute WER reduction when comparing to applying single fMLLR for each session.

Table 6.8 *Recognition performance, WER (%), of the S6 DNN-adapt (see Table 5.4) when speaker diarisation information is employed for fMLLR adaptation.*

Speaker diarisation information used	Excluding segments of noise and overlapped speech	
	yes	no
none (single fMLLR)	48.06	56.09
automatic (2 fMLLR)	46.74	54.37
manual (2 fMLLR)	43.72	52.92

6.7 Summary

In this chapter, an automatic speaker diarisation system based on i-vector is introduced. It is an essential component of the automatic assessment system. For extracting i-vectors to represent the characteristics of the speakers, the best way to train UBM and T matrix has been explored, Utterance level CMN for UBM training have shown to achieve best performance when comparing to speaker and session level CMN. The 64 dimensional i-vectors always performed the best when comparing to the smaller dimension and larger dimension with different UBM. The 64 dimensional i-vector extracted from UBM with 1024 components was the best configuration so far. ALso longer lengths and steps of the window for temporal segmentation were explored, and a window size of 2s with 1.5s shifting was the most suitable configuration. DNN i-vector was also explored but did not bring any promising results. Experiment with ASR suggested that speaker identities extract from the automatic speaker diarisation perform better than ASR without any speaker information.

It still remains one problem that the speaker diarisation can only separate the two speakers from the recording. Which speaker is the clinician is unknown.

Chapter Seven

Topic modeling

7.1 Introduction

The MITI assessment criteria include a criterion called "Direction" (refer to Section 3.2.1). This is a global scale that measures the degree to which clinicians maintain focusing on a specific target behavior and direct the patient to the target. Thus keeping a track of whether the conversation is on topic (in our case whether the two participates are talking about diabetes) helps to obtain a global impression of the ability of the clinician to delivery the interview. Two most widely used technologies for topic modelling, LSA and LDA, have been introduced in Section 2.9.2 and Section 2.9.3.

This chapter is to explore using LDA to track the topic during the interview. First, true transcriptions from the 128 AMI sessions and 28 MI sessions were stemmed and each session was characterised as a vector of word counts. The word count vectors are collected into a document-word matrix. Then the matrix was used to train a LDA model. By fitting each session into the LDA model, a topic distribution can be obtained, which was used as the topic vector for each session. A MI global topic vector is obtained by taking the average of the topic vectors for the 28 MI sessions.

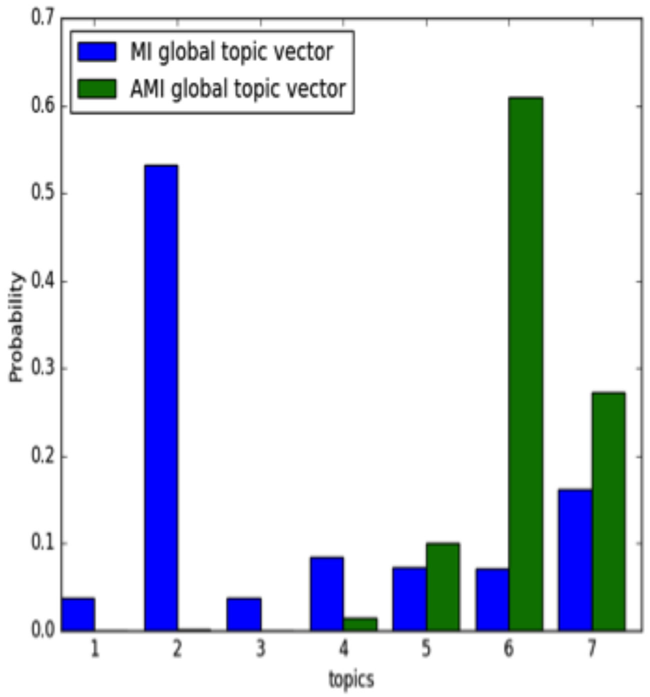
Three conversations from the MI_test data were used for evaluation, each between 10-20 minutes' long. Each session was split into 15-seconds segments, the true transcription as well as ASR output for each segment are avaliable. In total, there were 150 segments for evaluation, out of which 88 were and 62 were not related to the topic of 'diabetes'. A segment topic vector can be obtained by fitting the segment into the trained LDA model. Because the "on topic" and "off topic" labels were not available for the training data, the topic detection couldn't be achieved with a supervised classification method. Two LDA based methods were applied to assess how likely a given segment is related to 'diabetes': (1) the proportion of the topic "diabetes" in the segment topic vector and (2) the cosine similarity between the segment topic vector and the global MI topic vector. The topic detection was evaluated on true transcriptions as well as ASR outputs, in order to denmonstrate how ASR WER can affect LDA topic modelling.

7.2 Global topic vector

During LDA model training, 5,7 and 9 topics were explored. The results presented here are obtained using 7 topics. Each topic was represented as a multinomial distribution over words, and the N most frequent words are presented here, some of them can show an intuitive interpretation of each topic. For example, the topic#2 can be interpreted as related to 'diabetes' as it contains with a high probability words like 'blood', 'sugar', 'insulin', which are commonly used when people talk about 'diabetes'.

A topic distribution for each session can be obtained by fitting it into the LDA model. The porportion of the topics were used as the topic vector for each session. The MI global topic vector is obtained by taking the average of the MI topic vectors, and the AMI global topic vector is obtained in the same way. Figure 7.1 shows the topic proportions in the MI and AMI global topic vector, and the top most frequent words in the four most frequent topics.

It can be seen that MI global topic vector has a high proportion on topic#2. The topic distribution in MI is very different from AMI. The AMI global topic vector has a low proportion on topic#2 and high on topic#6.



Topic#2	Topic#6	Topic#5	Topic#7
umm	button	uh	uh
eat	remot	mean	right
sai	don	don	ve
got	control	word	mean
blood	make	sai	gonna
did	maybe	hmm	ll
sugar	ye	probabl	sai
dai	design	right	sort
mean	colour	meet	got
ye	function	differ	don
morn	ll	segment	need
…	right	mayb	caus
insulin	peopl	make	actual
right	screen	data	kind
meal	…	sort	Peopl

Figure 7.1 Results of LDA-based topic modelling. The global distribution of topics in MI and AMI documents (left) and the most frequent words in the four most frequent topics (right).

7.3 Diabetes topic detection

First, the proportion of topic#2 in the segment topic vector was used as a measurement to assess how likely a given segment is related to 'diabetes'. If it was below a threshold, the segment was rejected. The DET curves in Figure 7.2 shows the plots of false reject rate VS false accept rate with different threholds, blue curve was evaluation using true transcription and green curve using ASR outputs. It suggests that the detection accuracy using ASR outputs is worse than using true transcriptions.

Rather than just using the single value of the topic vector, the cosine similarity between the segment topic vector and the global MI topic vector was used to measure how likely a given segment is related to 'diabetes'. The red and yellow curves in Figure 7.2 shows the performance of this method. With true transcriptions, using just the portortion of topic#2 is slightly better than using the cosine similarity to the global MI topic vector. The difference between two methods is not as big as the difference between the use of true transcriptions and ASR outputs.

The best equal error rate (EER) is 0.276 when using the true transcription. The ASR ouputs also provide useful information for topic detection with a EER 0.353.

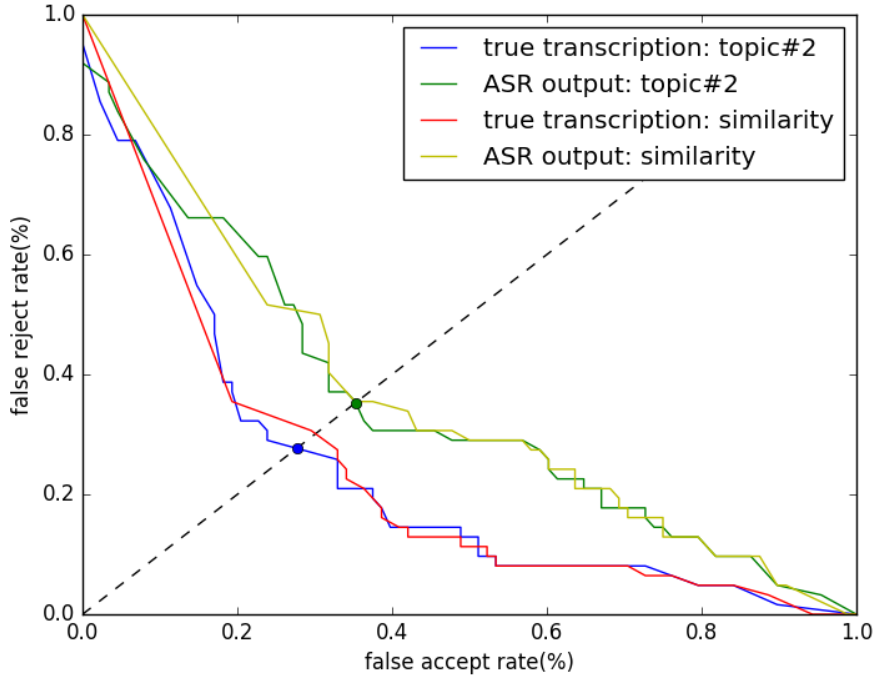


Figure 7.2 DET curve for automatic detection of 'diabetes' topic using the true transcription of MI_test data and ASR output.

7.4 Summary

This chapter explored using LDA for topic modeling and topic tracking during the interview. By fitting a corpus into a LDA model, each document can be represented as a distribution of topics and each topic is represented as a distribution of words. A topic that contains high probability words like 'blood', 'sugar', 'insulin' can be interpreted as related to 'diabetes'.

In terms of topic tracking during interview, segment topic vector can be obtained by fitting the given segment into the trained LDA model. The proportion of 'diabete' topic as well as the cosine similarity between the segement vector and the global MI topic vector were used as the measurements to assess how likely the segment is related to 'diabetes'. The best equal error rate (EER) is 0.276 when using the true transcription. With a WER of 43.72%, the ASR ouputs also provide useful information for topic detection with a EER 0.353.

Chapter Eight

Behaviour code classification

8.1 Introduction

The MITI criteria provides a set of guidelines for effective motivational interview as well as different criteria to assess the clinicians' adherence to these guidelines. In terms of locally behaviour assessment, the counts of behavior codes also shows the clinicians' ability to use the MI skills. For example, it is better to use more open questions rather than closed questions and more MIAs than MINAs for a positive clinical outcome.

The MITI criteria apply mainly to the clinician conducting the interview. Therefore, in order to automate the application of these criteria, the clinician and patient turns in the interview must be separated and the speech must be transcribed. Therefore a speaker diarisation system was built to seperate the two speakers in the interview in Chapter 6 and an ASR system for MI was built to transcribe the speech into text in Section 5.

In this chapter, a subset of the MITI criteria were considered and only a behaviour code classification system is built to classify the clinicians' speech in text into different behavior codes. Not all of the clinician's speech will be assigned a behaviour code. As introduced in Section 3.2.2, there are six types of behaviour codes. The classifiers are trained and evaluated only on the coded utterances.

First, utterances from clinician's speech were represented using TF-IDF document vector or sequence of Word2Vec word vector (refer to Section 2.7). Then, SVM or KNN classifiers with different distance metrics were applied to identify whether a paragraph vector or sequence of word vectors corresponds to a MITI behavioural code and if so, which code it corresponds to.

As an initial study, only the following binary classification tasks were investigated:

1. Question vs. Non-Question behaviour
2. Open Question vs. Closed Question
3. Motivational Interview Adherence vs. Motivational Interview Non-Adherence behavior

denoted as Q/NQ, OQ/CQ and MIA/MINA, respectively.

Intuitively, for the Q/NQ distinction, the information on word ordering might be more important. For example, the occurrence of words such as "who", "what", "which" or "why" at the start of an utterance typically indicates a question ("why did you do that") whereas the occurence of the same words at other positions in the utterance does not ("Nobody knows why you do that").

On the other hand, the semantic meaning of the words might bring more benefit for MIA/MINA classification. Thus for these three different tasks, alternative vector representation and classification are going to be explored in this chapter.

The rest of this chapter is organised as follows: text data with behaviour codes and an online diabete text dataset are described in Section 8.2.1; classification evaluation criterion is introduced in Section 8.5; three methods of training vector representation are presented in Section 8.3 and classification methods are introduced in Section 8.4. Finally, Section 8.6 presents the results of the classification experiments.

8.2 Text data

8.2.1 Data with MITI code

For each of the 8 full sessions (refer to Section 3.3), the two psychologists have identified and categorised the clinician's utterances into six behaviour codes according to MITI guidelines: Giving Information, Closed Question, Open Question, Reflection, MI Adherent, MI Non-Adherent. Table 8.1 shows the total counts of each behaviour code category from the 8 full coded sessions. Spirit and empathy are also measured from a five-point scale to characterise the entire conversation, but are not used for this initial study.

Table 8.1 *MITI behavior code counts from 8 full session.*

bahavior code	counts
Giving information	127
Closed question	174
Open question	62
Reflection	98
MI Adherent	53
MI Non-Adherent	66

8.2.2 Online diabetes data

Because most diabetes technical terms will not occur in a generic word list and the word frequency in a clinical conversation is different from the word frequence in a generic dataset (i.e. a Google News dataset), an in-domain large dataset is needed to train a Word2Vec model as well as TF-IDF document vectors. The set of downloaded texts from "Diabetes UK" website (*Diabetes UK* n.d.) is referred to as "Diabetes_online". This dataset contains approximately 22526 sentences, 274,000 words and has a vocabulary of 7073.

8.3 Text processing

8.3.1 TF-IDF

The inverse document frequency (IDF) (refer to Section 2.7.2) for each word was computed from the Diabetes_online dataset, each sentence in Diabetes_online was treated as a separate document for IDF calculation. The IDF statistics were then used to calculate TF-IDF (refer to Section 2.7.2) vectors for the actual MI sentences. Each utterance from the 8 transcribed MI sessions was represented by a 7073-dimensional TF-IDF document vector. As the utterances are short, the TF-IDF vector representation can be very sparse, and it only concerns the occurence of the words.

8.3.2 In-domian word2vec

The Diabetes_online dataset was also used to train an in-domian word2vec (refer to Section 2.7.3) model using both the CBOW and skip-gram methods, and in both cases the context was ± 2. In the CBOW model, the task of the neural network is to predict the current word from the two preceding and following words. In the Skip-gram model, the task is to predict the neighboring words given the current word.

The stop words have shown positive effects in the classification tasks because they can indicate the structure of the utterances. Then the stop words were kept to train the in-domain word2vec models.

Different dimensions of the word2vec embeddings (100, 200 and 300) were explored, and a dimension of 300 works best. For the training method, skip-gram word2vec model works better than CBOW word2vec model. The resulting word2vec model was denoted as MI_w2v.

8.3.3 Generic word2vec

The Google's pre-trained Word2Vec model ("Google Code Archive - Long-term storage for Google Code Project Hosting." n.d.), denoted as Google_w2v, was used as a generic word2vec. It is trained with around 100 billion words from a Google News dataset with a vocabulary of 3 million words. The dimension of the Google_w2v is 300. There are not many medical terms, especially diabetes' terminology. The stop words were absent in this model.

8.4 Classification method

Scikit-learn (Pedregosa et al. 2011) was used for the inplementation of TF-IDF vectors, SVM and KNN, and gensim (Řehůřek and Sojka 2010) was used for training Word2Vec models.

SVM

With TF-IDF vectors, SVM (refer to Sectiion 2.8.2) with different kernel functions (i.e. sigmoid, RBF and linear kernel functions) were applied to the three classfication tasks. The tuning parameters are regularisation parameter C , γ. Each combination is evaluated using 5 cross validations, and the best combination of C and γ is selected by a grid search with exponentially growing sequences of C and γ (for example, $C = 2^{-2}, 2^{-1}, 2^{0}, 2^{1}$; $\gamma = 2^{-4}, 2^{-3}, ..., 2^{1}$).

KNN

For KNN (refer to Section 2.8.3), we explored the number of neighbors (K), considering all the odd number between 1 and 20 for Q/NQ task. For the CQ/OQ and MIA/MINA where the number of samples is limited, K was restricted to a range of 1 to 5.

For each sample in testing data, its distances to every sequence in the training set were computed and the K neighbors with the closest distances were chosen. The testing sample was then classified by the majority vote.

The distances were first obtained from the Eculidian distances between TF-IDF vectors. When using word2vec, each utterance is represented as a sequence of word vectors, the distance metric used in the KNN classifier must be able to accommodate the fact that the number of word vectors in an utterance varies. Two different distance functions were considered, WMD (Section 2.8.4) and DTW (Section 2.8.5). In both cases the distance between two word vectors is Eculidian distance. The generic word2vec model Google_w2v as well as the in-domian word2vec model MI_w2v were used in the KNN experiments. We were particularly interested in whether the fact that DTW preserves word order is useful in classification, in particular for distinguishing between questiuons and non-questions.

Each K choice was evaluated using cross validation, and the parameter with best binary accuracy and class accuracies was picked.

8.5 Evaluation criterion

As shown in Table 8.1, the data is very imbalanced in some of the classification task, thus binary accuracy (B_acc) cannot precisely measure the performance. Instead, class accuracy was used to measure the proportion of the correct identifications for each class. For example, class accuracy for Non-Question is denoted as NQ_acc and class accuracy for Question is denoted as Q_acc. The three accuracies are computed as below:

$$\begin{aligned} B\_acc &= \frac{T_Q + T_{NQ}}{Q + NQ} \\ NQ\_acc &= \frac{T_{NQ}}{NQ} \\ Q\_acc &= \frac{T_Q}{Q} \end{aligned}$$

where T_Q, T_{NQ}, Q, and NQ denote number of Questions that are correctly classified as Question, number of Non-Questions that are correctly classified as Non-Question, total number of Questions and total number of Non-Questions, respectively. If NQ is much bigger than

Q, and all the testing sequences were justified as NQ, then B_acc can still be high. Thus the B_acc can not precisely indicate the performance of the classifier, but Q_acc can do.

All accuracies are averaged over 5 cross validation tests on the classifiers using 80% of the coded utterances for training and 20% for testing. The data was split with same distribution of classes, or as close as possible.

8.6 Experiments

8.6.1 Question vs Non-Question

Results for Q/NQ task are shown in table 8.2. When using TF-IDF as vector representation, classification using SVM with sigmoid and linear kernel functions performed better than RBF kernel function. And KNN with Euclidian distance outperformed SVM.

When exploring KNN with different distance metrics and word embedding, DTW performed better than WMD with both in-domian word2vec and generic word2vec. These results were expected, because the information on word wordering matters when classifying question from non-question.

The best classification method for now is KNN with DTW distance metric. The binary accuracy, Non-Question class accuracy and Question class accuracy are 0.78, 0.89 and 0.61, respectively.

Table 8.2 *Class accuracies and Binary accuracies of the Question/Non-Question (236/342) classification tasks on manual transcriptions of MI_8.*

Method	B_acc	NQ_acc	Q_acc
TF-IDF + SVM-sigmoid	0.73	0.84	0.58
TF-IDF + SVM-RBF	0.72	0.86	0.52
TF-IDF + SVM-linear	0.73	0.83	0.59
TF-IDF + Euc-KNN	0.74	0.83	0.6
MI_w2v + WMD-KNN	0.72	0.82	0.58
MI_w2v + DTW-KNN	**0.78**	**0.89**	**0.61**
Google_w2v + WMD-KNN	0.71	0.69	0.73
Google_w2v + DTW-KNN	0.78	0.89	0.61

8.6.2 Closed Question vs. Open Question

As shown in Table 8.3, utterances represented as TF-IDF and classified with SVM using sigmoid kernel function performed the best for Closed/Open Question. The binary accuracy, Closed Question accuracy and Open Question accuracy are 0.80, 0.77 and 0.83, respectively. It suggests that the word order is more important for Q/NQ and less important for CQ/OQ.

When classifying with KNN, using DTW as the distance metrics outperformed WMD both with MI_w2v embeddings and with Google_w2v embeddings. This is as expected because DTW preserves word ordering information, however WMD does not. With the same distance metric, MI_w2v embeddings outperformed Google_w2v embedding significantly. That might because of the absence of the stop words and the terminology that occurs in motivational interviews with diabetes patients in Google_w2v.

Table 8.3 *Class accuracies and Binary accuracies of the Closed Question / Open Question classification tasks on manual transcriptions of MI_8.*

Method	B_acc	CQ_acc	OQ_acc
TF-IDF + SVM-sigmoid	**0.80**	**0.77**	**0.83**
TF-IDF + SVM-RBF	0.77	0.78	0.75
TF-IDF + SVM-linear	0.77	0.77	0.77
TF-IDF + Euc-KNN	0.59	0.54	0.7
MI_w2v + WMD-KNN	0.68	0.60	0.77
MI_w2v + DTW-KNN	**0.75**	**0.67**	**0.83**
Google_w2v + WMD-KNN	0.63	0.67	0.55
Google_w2v + DTW-KNN	0.73	0.79	0.60

Table 8.4 *Class accuracies and Binary accuracies of the MIA/MINA (66/51) classification tasks on manual transcriptions of MI_8.*

Method	B_acc	MIA_acc	MINA_acc
TF-IDF + SVM-sigmoid	0.74	0.71	0.77
TF-IDF + SVM-RBF	0.73	0.57	0.86
TF-IDF + SVM-linear	0.76	0.75	0.77
TF-IDF + Euc-KNN	**0.78**	**0.76**	**0.79**
MI_w2v + WMD-KNN	**0.72**	**0.72**	**0.71**
MI_w2v + DTW-KNN	0.62	0.77	0.5
google_w2v + WMD-KNN	0.64	0.64	0.64
google_w2v + DTW-KNN	0.71	0.77	0.67

8.6.3 Motivational Interview Adherence vs Motivational Interview Non-Adherence

The results of the experiments to distinguish between MIA and MINA utterances are show in Table 8.4. The best performance is obtained using a KNN classifier with TF-IDF document vectors.

When comparing WMD and DTW using MI_w2v word vectors, WMD performed better. This is as expected, for semantic meaning is important in MIA/MINA task.

Google_w2v performed worse than MI_w2v with WMD, and outperformed MI_w2v when using DTW as distance metirc for KNN. The absence of the stops words and MI terminology can affect the term frequency and thus the calculation of WMD between two utterances, so WMD with Google_w2v couldn't perform well when comparing to Google_w2v based DTW.

8.7 Summary

This chapter explored three ways of representing an utterance spoken by a clinician in a MI interview, including TF-IDF, MI_w2v and Google_w2v. SVM and KNN with different distance metrics were investigated for three binary classification tasks.

For CQ/OQ and MIA/MINA tasks, the TF-IDF document vectors based methods performed better than word2vec word vectors based methods. This suggests that for futher work, a neural network based document vector representation doc2vec might improve the performance. In terms of word2vec, the results suggest that DTW based KNN outperforms WMD based KNN when the information on word ordering is important. When the semantic meaning matters, WMD performs better than DTW if term frequency can be correctly obtained.

This initial study was only conducted on a small text dataset. If more utterances with behavior codes are available in the future, it is possible to build a classifier to automatically code the utterance from clinicians. Given the results from the three tasks, for the full automatic assessment, KNN with DTW distance metric classification on MI_w2v embedding was applied for Q/NQ task, SVM with sigmoid kernel function classification on TF-IDF was applied for CQ/OQ task and KNN with Eculidian metric classification on TF-IDF was applied for MIA/MINA.

There are also some research efforts towards the automation of behaviour code identification for MI assessment. Pérez-Rosas et al. (2017) proposed to use linguistic based features as inputs to classify reflection Vs other MITI code and Questions VS other MITI codes for each utterance using support vector machine (SVM) classifier, with a F score of 0.84 and 0.82, respectively. Gibson et al. (2015) have found that local behaviour code vectors carries important information for predicting empathy. Their proposed system first mapped linguistic based input vectors to a target that represents the MISC code for each turn. Then in Gibson et al. (2017), they employed long short term memory (LSTM) networks with an attention mechanism to predict these behaviour codes. Gibson et al. (2019) then proposed a multi-label multi-task deep learning approach for behavioral coding. Their work suggested that if more training data coded with MITI was available for this project, a better behaviour code classification can be built with deep learning approaches.

Chapter Nine

Automatic assessment system

9.1 Introduction

Chapter 8 presents the results of automatic behaviour code classifcation using manual speaker diarization and manual transcrition of the clinician's speech. Unfortunately the manual effort required for diarization and transcription accounts for most of the effort required for clinicians' behaviour code classification. Therefore to be trully useful automatic behaviour code classification must use the results of automatic speech recognition and diarization.

This chapter integrates the best speaker diarisation system from Chapter 6, final ASR system from Chapter 5 and behaviour code classification system from Chapter 8 for a full automatic assessment system that can analyse the clinicians' use of behaviour codes, and hence conduct a fully automatic assessment of the clinician's performance in the MI.

In this chapter, only Q/NQ behaviour codes classification was investigated. The best Q/NQ classification system was KNN with DTW distance metric based on MI_w2v embedding. In Chapter 8, the clinicians' coded utterancs in the 8 full session were splited into 80% and 20% for cross-validation training and testing. In this chapter, these coded utterances from 8 full sessions were used as training data for behaviour code classification system, and the full automatic assessment system was evaluated on MI_test. There was no overlap between these two datasets.

The performance of the full automatic assessment system can be affected by any of the three sub-systems. The effect of the errors from ASR will be investigated in a semi-automatic assessment system in Section 9.2, where manual speaker change information is used. Then, the effect of the automatic speaker diarisation system will be analysed in Section 9.3 with a full automatic assessment system that incorporates the three sub-system.

9.2 Semi-automatic assessment system

First, sessions in MI_test were manully segmented into per-speaker utterances and mannlly transcribed. KNN with MI_w2v based DTW distance metric was applied to classify the clinicians' utterances into Q and NQ, denoted by the M-M-Assess. The results are shown in the first row of Table 9.1. By replacing the manual transcriptions with the automatic transcriptions (denoted by 'A') from ASR, a semi-automatic system was obtained, and denoted as M-A-Assess system. The results are shown in the second row of Table 9.1. As expected, the accuracy for detecting questions decreased. A large increase in transcription error rate (from 0% to 43.59%) due to the move from manual to ASR transcriptions leads to a relatively modest decrease in behaviour code classifcation performance.

Table 9.1 *Detailed results of the behavior code classification on ASR outputs*

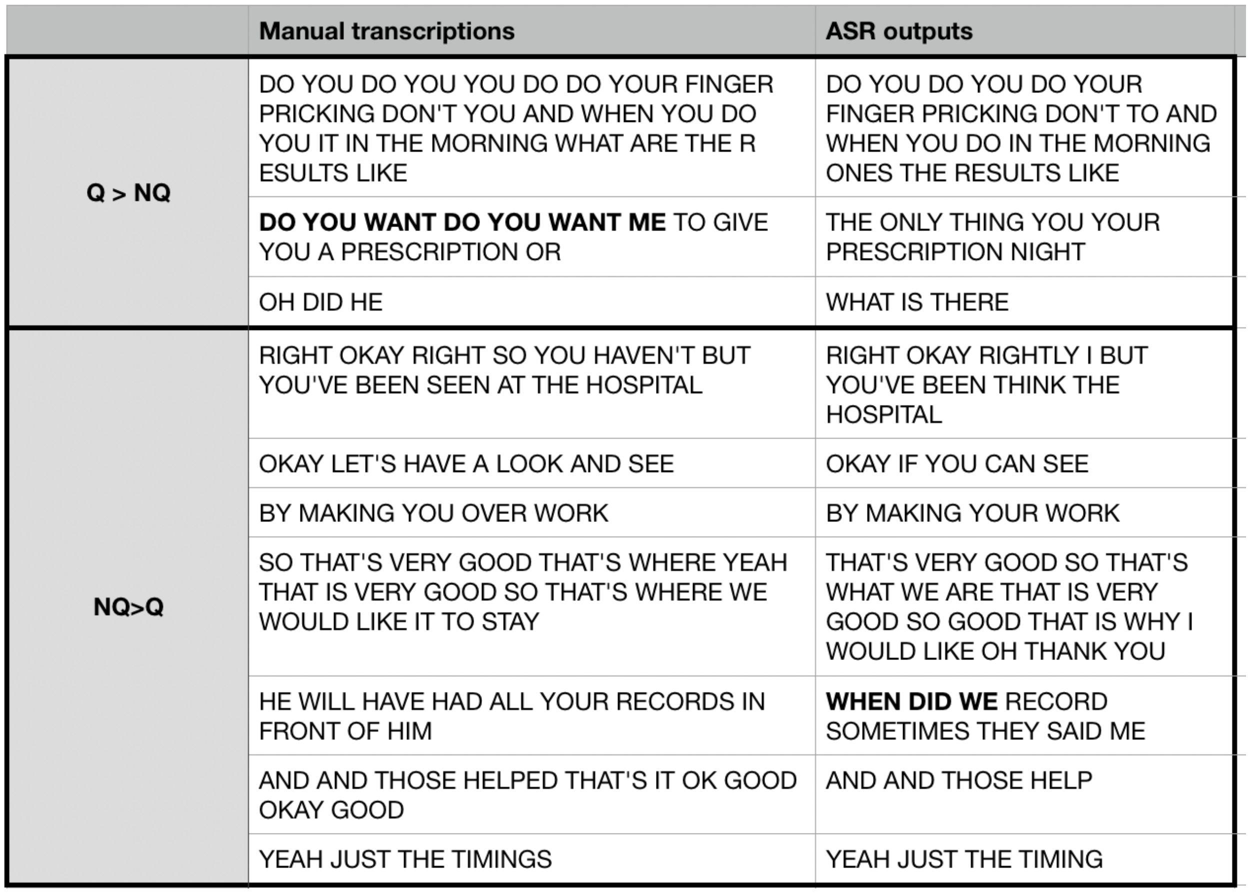
System	SD	ASR	NQ/Q		
			B_acc	NQ_acc	Q_acc
M-M-Assess	M	M	0.78	0.84	0.60
M-A-Assess	M	A	0.79	0.88	0.55

Among the 33 false detections from this semi-auto assessment system, 10 were new errors compared to M-M-Assess. These new errors include two scenarios. The scenario where the M-M-Assess system correctly identified Question, but M-A-Assess system made the wrong decision is denoted by 'Q→NQ', and 'NQ→Q' vice versa. These new errors are as shown in

Figure 9.1. From the Figure it is evident that only two of the behaviour code classification errors are caused by ASR errors. These are shown in bold and are the second 'Q→NQ' error and the fifth 'NQ→Q' error.

However, for example, for the first 'Q→NQ' error the ASR output is still a Question but is incorrectly classified by the behaviour code classification system. Conversely, for the first 'NQ→Q' error the ASR output is still not a Question but is incorrectly classified as a Question by the behaviour code classification system. The behaviour code classification made the most number of errors (31/33) in this semi-auto system.

The errors in Figure 9.1 suggested that although ASR made wrong transcriptions, a better behaviour code classifier could have good tolerance on these ASR errors. Thus for the future work, the improvement on the behaviour code classifications would benefit the assessment system significantly.



	Manual transcriptions	ASR outputs
Q > NQ	DO YOU DO YOU YOU DO DO YOUR FINGER PRICKING DON'T YOU AND WHEN YOU DO YOU IT IN THE MORNING WHAT ARE THE R ESULTS LIKE	DO YOU DO YOU DO YOUR FINGER PRICKING DON'T TO AND WHEN YOU DO IN THE MORNING ONES THE RESULTS LIKE
	DO YOU WANT DO YOU WANT ME TO GIVE YOU A PRESCRIPTION OR	THE ONLY THING YOU YOUR PRESCRIPTION NIGHT
	OH DID HE	WHAT IS THERE
NQ>Q	RIGHT OKAY RIGHT SO YOU HAVEN'T BUT YOU'VE BEEN SEEN AT THE HOSPITAL	RIGHT OKAY RIGHTLY I BUT YOU'VE BEEN THINK THE HOSPITAL
	OKAY LET'S HAVE A LOOK AND SEE	OKAY IF YOU CAN SEE
	BY MAKING YOU OVER WORK	BY MAKING YOUR WORK
	SO THAT'S VERY GOOD THAT'S WHERE YEAH THAT IS VERY GOOD SO THAT'S WHERE WE WOULD LIKE IT TO STAY	THAT'S VERY GOOD SO THAT'S WHAT WE ARE THAT IS VERY GOOD SO GOOD THAT IS WHY I WOULD LIKE OH THANK YOU
	HE WILL HAVE HAD ALL YOUR RECORDS IN FRONT OF HIM	**WHEN DID WE** RECORD SOMETIMES THEY SAID ME
	AND AND THOSE HELPED THAT'S IT OK GOOD OKAY GOOD	AND AND THOSE HELP
	YEAH JUST THE TIMINGS	YEAH JUST THE TIMING

Figure 9.1 Errors analysis made by the M-A-Assess system.

9.3 Full automatic assessment

Next, the manual speaker change information was replaced by the outputs from the automatic speaker diarisation system. The sessions from the MI_test were segmented into utterances according to the automatic speaker change information. There was only one speaker in each utterance. The ASR system was applied to automatically transcribe the utterances. The automatic speaker change information was also used to train the speaker level fMLLR transform for the testing data. Since from the current speaker diarisation system, the speakers' identities were unknown, the clinician's utterances were manually decided based on the ASR transcription. Q/NQ behaviour code classification was applied to classify the clinician's utterances into Questions and Non-Questions. This full automatic assessment system was denoted as A-A-Assess system. The result of the full automatic assessment system is a sequence of decisions including Questions, Qon-Questions and Patients, denoted as Q, NQ and P, respectively.

Compared with the manual system, for the automatic system there is an additional error type due to incorrect diarization. As shown in Figure 9.2, the reference for each session was from a manual sequence of Q, NQ and P.

Assessment Error Rate (AER)

The errors from speaker diarisation system (for example, missing speaker changes or false alarm speaker changes) make the total number of clinicians' utterances different from the reference. Therefore, a new metric for the performance of this fully automatic system was introduced: Assessment Error Rate (AER).

AER is obtained by first aligning the sequence of the decisions from A-A-Assess with the sequence of the correct decisions. The alignment used Levenshtein distance as the distance metric. The Levenshtein distance is a string metric for measuring the difference between two sequences. The Levenshtein distance between two words is the minimum number of single-

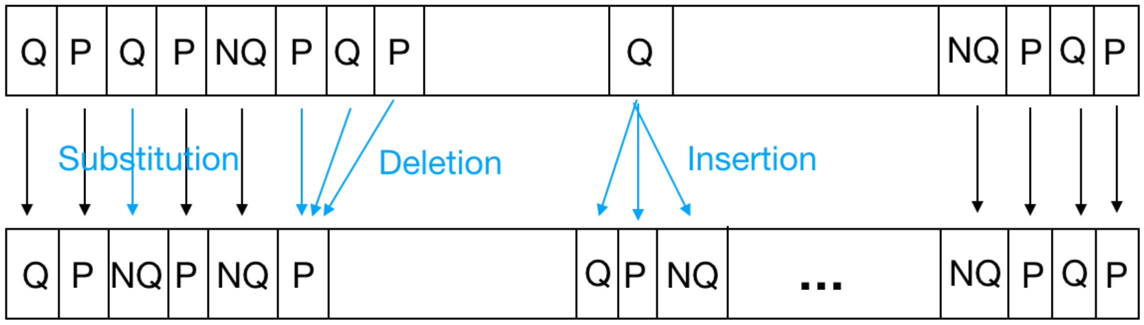


Figure 9.2 Alignment between the true decision sequence (top) and the A-A-Ass decision sequence (bottom), and three sorts of errors: substitution, deletion and insertion.

character edits (insertions, deletions or substitutions) required to change one word into the other. For the alignment of the decisions, Levenshtein distance is the minimum number of the three categories of errors as shown in Figure 9.2: Substitution (S), Deletion (D) and Insertion (I).

AER is computed as below:

$$AER = \frac{S + D + I}{N} \tag{9.1}$$

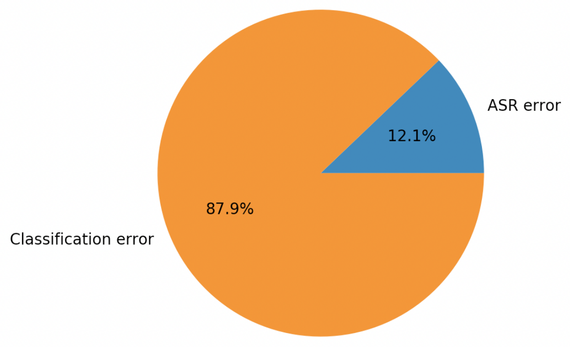
where N is the total number of decisions in the reference.

A-A-Assess system

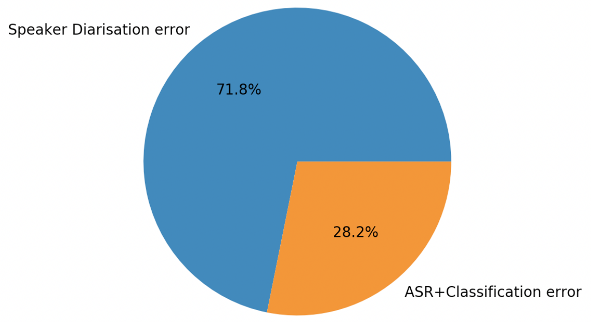
An AER of 22.54% for the A-A-Assess system was obatined. Among the total 71 errors, there are 3 insertion errors, 48 deletion errors and 20 substitution errors. So insertion and deletion errors caused by speaker diarisation account for 51 of the 71 errors.

Figure 9.3 shows the error distributions in systems with different levels of automation. In the case of M-A-Assess system that uses manual speaker information and automatic transcription, classification made the most of the errors (87.9%). In the case of A-A-Assess system that incoporates the speaker diarisation system and ASR system, speaker diarisation system made most of the errors (71.8%).

The error analysis suggests that although the current ASR only achieves a overall WER of 43.59%, it is still possible to use the automatic transcriptions from it to build a full assessment system. For future work, more investigations are need on the speaker diarisation system and then behaviour code classification.



(a)



(b)

Figure 9.3 Error distribution in system with different level of automation. (a) Error distribution in NQ/Q classification system with manual speaker diarisation and automatic transcription: classification made the most of the errors. (b) Error distribution in NQ/Q classification with fully automatic assessment system: speaker diarisation made the most of the errors.

9.4 Summary

In this chapter, a semi-automatic assessment system and a full automatic assessment system for Q/NQ behaviour code classification were built. The semi-automatic assessment integrated the ASR and Q/NQ behaviour code classification system. From the analysis of the source of the errors, it suggested that the current ASR system with WER of 43.59% was capable of automate the transcription for the automatic assessment process. A better classification method is needed for the future work. The full automatic assessment integrated the automatic speaker diarisation, ASR and behaviour code classification system. The performance is evaluated based on the alignment of the sequence of the decisions: Question (Q), NON-Question (NQ) and Patients (P). From the analysis of the miss-aligned errors (i.e. Substitution, Deletion and Insertion), it can be seen that the speaker diarisation system made most of the errors.

Chapter Ten

Conclusion

10.1 Overview

This thesis has demonstrated how speech and language techniques can be applied to the automatic assessment of motivational interview with diabetes patients. This can be transferred to the automatic assessment of any other clinical conversations if data is available. The automatic assessment can help provide feedback for a clinician conducting the MI session on the clinician's adherence to MITI guideline so that the clinician can obtian better MI skills. Conventional assessment requires two pyschologists manually counting the use of behaviour codes and giving scores for the whole session based on transcription or listening to the recording, which are costly. It will also help automatically measure the patients' willingness to behaviour change towards self-management if that score for each session is available.

There has been some progress on partially automatically assessing some of the MITI components from the language, but these research projects still rely on manual segmentation or transcription. This thesis explored a fully automated process. It explored an automatic speaker diarisation system to separate the clinician's and patient's speech. Then an automatic speech recognition system was built to transcribe the conversation. Topic analysis technologies were applied to track the conversation to decide whether it is always on topic (related to diabetes). The use of bahaviour code is an important scale when assessing the clinicians' MI skills. Behaviour code classification was applied to classify clinicians' utterances. For example, question or non-question, open or closed question and MI aderence or non-adherence. With a semi-automatic assessment that integrated an ASR system and behaviour code classification, the impact of the ASR error on behaviour code classification was explored. Finally a full automatic assessment was built using the output from the speaker diarisation system to seperate the two speakers in a session.

10.2 Contribution

10.2.1 ASR system

ASR systems were built for two different applications. Chapter 4 described a DNN-HMM ASR system provided as the baseline for the 2017 1st edition of Spoken CALL shared task. Among nine different groups who took part in this competition, four used this baseline ASR for the pre-recognised transcriptions. And our developed system won the first place in the competition. For the ASR system in the 2nd edition, the use of various amount of AMI-ihm data was explored and fMLLR transformation and achieved a significant WER reduction. LSTM based ASR systems were applied and two sets of features including MFCC features and fMLLR transformed features were investigated. sMBR training cirteria was utilised to train the DNN, which outperformed the cross-entropy trained DNN.

As a part of the automatic assessment system, an ASR system was built to transcribe the recordings of Motivational interviews into text. This is presented in Chapter 5. This is a challenging problem due to the conversational nature of speech, the usage of medical terminology and recordings performed using a distant microphone of unknown quality in real-world environment. Several approaches to training a DNN-HMM ASR system using the AMI speech corpus plus a limited amount of MI speech corpus were explored, from the following aspects:

- Training data: the open-source dataset AMI meeting corpus is much better for training an MI ASR system rather than WSJCAM0.

- Alignments: ASR system performance depends critically on the quality of alignment. Better quality alignments were obtained by using the AMI individual headset microphone recordings. And these alignments together with the single distance microphone acoustic data, has shown to be more appropriate for MI GMM-HMM acoustic model training.

- Cross-domain feature-space maximum likelihood linear transform (fMLLR) adaptation: fMLLR was applied to discover a linear transform for each speaker to maximise the probability of the data given the speaker independent model. FMLLR transforming of the feature vectors to different generalised model was explored. The results suggested that fMLLR is more effective when adaptation is performed relative to the more compact AMI-trained model rather than a model that was constructed using a combination of AMI and MI data.

- DNN adaptation: In addition to conventional DNN training, adapting the DNN specifically to MI data was explored. This involved keeping the DNN structure unchanged, randomising weights in the last layer and then retraining the entire DNN using only the MI data.

- Training criteria: cross-entropy as well as sequence discriminative training as the optimisation criteria for DNN were explored. The sequence discriminative training outperformed the cross-entropy training.

10.2.2 Speaker diarisation system

Speaker diarisation is a process of determining who is speaking at a given time. The speaker diarisation system is based on i-vectors, which provide a low-dimensional vector representation of a signal segment. Experience in automatic speaker recognition demonstrates that i-vectors contain information relevant to separating the speech of different subjects. These i-vectors were clustered using a GMM with 2 components, as we know that there are two speakers in a given MI recording. The decision about each segment of speech belonging to a speaker was then based on the maximum-a-posterior probability of the segment's i-vector belonging to one of the GMM components. The decision was then smoothed by a 2-state Markov model. I explored training GMM-UBM with different numbers of components and extracting various dimensions of i-vector. 64 dimensional i-vector extracted from GMM-UBM with 1024 components was the best configuration.

Apart from the GMM-i-vector, DNN-i-vectors were also evaluated. The DNN trained for ASR was applied to obtain posterior probabilies for senones and for the sufficient statistics of T matrix training and i-vector extraction. But the DNN-ivector did not achieve better performance than GMM-ivector.

For DNN-ivector extraction, MFCC features and fMLLR transformed features were compared. My first intuition was that fMLLR would normalise the speaker difference within the sessions, and thus would not be able to perform good speaker diarisation. However, from my experiments, it turned out that when there were significant differences across different sessions, applying fMLLR to the entire session could normalise this kind of differences while keeping the speakers variance.

Experiments also suggested that the automatic speaker information from the best speaker diarisation can be used for speaker level fMLLR transformation on testing data. The evaluation with automatic speaker information outperformed the evaluation on testing data with no speaker change information with 1.32% WER reduction on cleaned data and 1.72% WER reduction on uncleaned data.

10.2.3 Topic modelling

With the transcriptions of the interview, topic modelling techniques were applied to detect whether parts of the conversation were on the topic of ‘diabetes’ or not. Latent Dirichlet Allocation (LDA) were employed, which is an unsupervised technique to identify latent topic in a set of documents. From the word distribution of each topic, one of the topics (topic#2) is very likely to be related to ‘diabetes’. With the inferred LDA topic vector for each 15-seconds segment, the weight of the topic#2 or the cosine similarity between the segment topic vector and the global MI topic vector were applied to assess how likely the given segment was related to ‘diabetes’. Using the weight of topic#2 with a equal error rate (EER) of 0.276 was found to perform better than the cosine similary for detecting topic. Results from the experiments with ASR outputs were consistent.

10.2.4 Behaviour code classification

Categorising the clinicians’ utterance into behaviour codes according to MITI guidelines is an important step in assessing the clinicians’ performance. The state-of-art technologies have been explored to classify the clinicians’ speech into Question/Non-Question, Open Question/Closed Question, and MI adherence/MI Non-adherence. Two classification methods were used: Support Vector Machine (SVM) and K-Nearest Neighbor (KNN). Since KNN is an algorithm based on the distance between samples, the distance metric was computed in three ways: Euclidean distance (ED) based on TF-IDF document vectors, and the use of Word Mover Distance (WMD) and Dynamic Programming (DP) to measure distance between two sequences of word2vec word embeddings.

For different classification tasks, when word ordering matters (for example, Question/Non-Question and Open/Closed Question classifications), DTW performs better than WMD.

When semantic meaning matters, WMD performs better than DTW.

For Close/Open Question task, TF-IDF based SVM performed the best. It suggests that word order is more important for Question/Non-Question than for Close/Open Question.

10.2.5 Automatic assessment of MI

With the combination of ASR and speaker diarisation systems, automatic transcription of the clinicians' speech can be obtained. The effect of ASR error and speaker diarisation error were explored by comparing the results with manual transcription and speaker information. The behaviour code classifications in Chapter 8 were conducted on manual speaker information and manual transcription. A semi-automatic assessment system on Question/Non-Question was obtained by only replacing the manual transcription with the automatic transcription from ASR system. For the 31 out of the 33 new errors, ASR make incorrect transcription but can still be manully seen as a Question or Non-Question. This suggeted that in this semi-automatic assessment system the classification system made the most of the errors.

For a full automatic assessment system where the outputs from the automatic speaker diarisation system were used as the speaker change boundaries, I have introduced a new evaluation criterion, i.e. Assessment Error Rate (AER). I have found that the insertion and deletion errors made most of the errors, especially deletion errors. These two types of errors are very likely to be caused by the speaker diarisation system.

10.3 Future work

The research has been consistently hampered by a shortage of fully transcribed real MI data. Given more data to train the ASR system it is likely that the ASR error rate could be further improved. However, the improvement is likely to be limited by the style of speech and quality of the recording. Although the style of speech cannot be changed, improvements in ASR accuracy could be obtained by using better quality recording equipment. Ideally the clinician could wear a head or lapel mounted microphone and this would aid ASR accuracy and diarization. The shortage of data also meant that there were few examples of each MITI behaviour code and this severely restricted the extent to which behaviour code classification could be explored. Therefore the most important item of future work would be to obtain a bigger corpus of properly transcribed and assessed MI recordings. ASR technology is constantly improving because of its commercial value. Therefore, periodically the most recent developments in ASR should be evaluated on MI data. For example, time-delayed DNN (TDNN) (Peddinti, Povey, and Khudanpur 2015b) could be applied for better performance. Performance could also be improved by the use of a customised language model for MI - it is unlikely that general language models developed for commercial ASR systems will contain the correct vocabulary of syntactic structures for MI targeted at a particular medical application such as diabetes.

In the case of speaker diarisation system, x-vector could be explored since x-vector (Snyder et al. 2018) is the state of the art technique for speaker representation.

The error analysis in Chapter 9 suggests that the behaviour code classification makes the most errors in the semi-automatic assessment system. If more utterances labelled with behaviour codes are available, a better behaviour code classification system could be obtained and also deep learning approach could be applied. With more coded MI session, the assessment of global measurement like *sprit* and *empathy* could be automated using language processing techniques.

References

Upsher, Rebecca et al. (2020). “Psychological interventions to improve glycemic control in adults with type 2 diabetes: a systematic review and meta-analysis”. In: *BMJ Open Diabetes Research and Care* 8.1.

Hadjiconstantinou, Michelle et al. (2020). “Using Intervention Mapping to Develop a Digital Self-Management Program for People With Type 2 Diabetes: Tutorial on MyDESMOND”. In: *J Med Internet Res* 22.5.

Chew, Boon How et al. (Sept. 2017). “Psychological interventions for diabetes-related distress in adults with type 2 diabetes mellitus”. In: *Cochrane Database of Systematic Reviews*.

Huffman, Jeff et al. (Apr. 2015). “Positive Psychological Interventions for Patients with Type 2 Diabetes: Rationale, Theoretical Model, and Intervention Development”. In: *Journal of Diabetes Research*, pp. 1–18.

Alvarado-Martel, Dácil et al. (2020). “Motivational Interviewing and Self-Care in Type 1 Diabetes: A Randomized Controlled Clinical Trial Study Protocol”. In: *Frontiers in Endocrinology* 11, p. 948.

Soderlund, Patricia Davern (2018). “Effectiveness of motivational interviewing for improving physical activity self-management for adults with type 2 diabetes: A review”. In: *Chronic Illness* 14.1, pp. 54–68.

Katie, Ridge et al. (2012). “Themes elicited during motivational interviewing to improve glycaemic control in adults with Type 1 diabetes mellitus”. In: *Diabetic Medicine* 29.1, pp. 148–152.

Gaume, Jacques et al. (2009). “Counselor skill influences outcomes of brief motivational interventions”. In: *Journal of Substance Abuse Treatment* 37.2, pp. 151 –159.

McCambridge, Jim et al. (2011). “Fidelity to Motivational Interviewing and subsequent cannabis cessation among adolescents”. In: *Addictive Behaviors* 36.7, pp. 749 –754.

Woodin, Erica, Alina Sotskova, and K. Daniel O’Leary (Nov. 2011). “Do Motivational Interviewing Behaviors Predict Reductions in Partner Aggression for Men and Women?” In: *Behaviour research and therapy* 50, pp. 79–84.

Moyers, T. et al. (Jan. 2016). “The Motivational Interviewing Treatment Integrity Code (MITI 4): Rationale, preliminary reliability and validity”. In: *Journal of Substance Abuse Treatment*.

Magill, Nicholas et al. (2018). “Assessing treatment fidelity and contamination in a cluster randomised controlled trial of motivational interviewing and cognitive behavioural therapy skills in type 2 diabetes”. In: *BMC Family Practice* 19.1, p. 60.

Xiao, Bo et al. (2015). “Analyzing Speech Rate Entrainment and Its Relation to Therapist Empathy in Drug Addiction Counseling”. In: *Proc. INTERSPEECH*. Dresden, Germany.

Gibson, J. et al. (2016). “A deep learning approach to modeling empathy in addiction counseling”. In: *Proc. INTERSPEECH*.

Chakravarthula, Sandeep Nallan et al. (2019). “Predicting Behavior in Cancer-Afflicted Patient and Spouse Interactions using Speech and Language”. In: *Proc. INTERSPEECH*.

Hinton, G. et al. (2012). “Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups”. In: *IEEE Signal Processing Magazine* 29.6, pp. 82–97.

Peddinti, Vijayaditya, Daniel Povey, and Sanjeev Khudanpur (2015a). “A time delay neural network architecture for efficient modeling of long temporal contexts”. In: *INTERSPEECH*.

Pan, Yilin et al. (2020). “Improving Detection of Alzheimer’s Disease Using Automatic Speech Recognition to Identify High-Quality Segments for More Robust Feature Extraction”. In: *INTERSPEECH*.

Moore, R. J. (2015). “Automated Transcription and Conversation Analysis”. In: *Research on Language and Social Interaction* 48.3, pp. 253–270.

Mirheidari, B. et al. (2016). “Diagnosing People with Dementia Using Automatic Conversation Analysis”. In: *Proc. INTERSPEECH*.

Mirheidari, B. et al. (2017). “An Avatar-Based System for Identifying Individuals Likely to Develop Dementia Analysis”. In: *Proc. INTERSPEECH*.

Miller, W. R. et al. (2003). “Manual for the motivational interviewing skill code (MISC)”. In: *Center on Alcoholism, Substance Abuse and Addictions, University of New Mexico*.

Pérez-Rosas, Verónica et al. (2017). “Predicting Counselor Behaviors in Motivational Interviewing Encounters”. In: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Valencia, Spain.

Gibson, James et al. (2015). “Predicting therapist empathy in motivational interviews using language features inspired by psycholinguistic norms”. In: *Proc. INTERSPEECH*.

Gibson, James et al. (2017). “Attention Networks for Modeling Behaviors in Addiction Counseling”. In: *Proc. INTERSPEECH*.

Gibson, James et al. (2019). “Multi-label Multi-task Deep Learning for Behavioral Coding”. In: *IEEE Transactions on Affective Computing*.

Tavabi, L et al. (2021). “Analysis of Behavior Classification in Motivational Interviewing”. In: *Proc. Association for Computational Linguistics*.

Ghahremani, Pegah et al. (2017). “Investigation of transfer learning for ASR using LF-MMI trained neural networks”. In: *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*.

Panayotov, Vassil et al. (2015). “Librispeech: An ASR corpus based on public domain audio books”. In: *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Takashima, Ryoichi, Tetsuya Takiguchi, and Yasuo Ariki (2020). “Two-Step Acoustic Model Adaptation for Dysarthric Speech Recognition”. In: *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Fakhan, Enver and E. Arisoy (2020). “Domain Adaptation Approaches for Acoustic Modeling”. In: *28th Signal Processing and Communications Applications Conference (SIU)*, pp. 1–4.

Srinivasamurthy, A. et al. (2017). “Semi-supervised Learning with Semantic Knowledge Extraction for Improved Speech Recognition in Air Traffic Control”. In: *Proc. INTERSPEECH*.

Qian, M. et al. (2017). “The University of Birmingham 2017 SLaTE Call Shared task system”. In: *Proc. of Speech and Language Technology in Education (SLaTE) workshop,* Sweden.

Yao, K. et al. (2012). “Adaptation of context-dependent deep neural networks for automatic speech recognition”. In: *IEEE Spoken Language Technology Workshop (SLT)*.

Woodland, P. C. et al. (2015). “Cambridge University transcription systems for the multi-genre broadcast challenge”. In: *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*.

Dempster, A. P., N. M. Laird, and D. B. Rubin (1977). “Maximum likelihood from incomplete data via the EM algorithm”. In: *Journal of the Royal Statistical Society: Series B* 39, pp. 1–38.

Rumelhart, David E., Geoffrey E. Hinton, and Ronald J. Williams (1986). “Learning Representations by Back-propagating Errors”. In: *Nature* 323.6088, pp. 533–536.

Hinton, Geoffrey, Simon Osindero, and Yee-Whye Teh (2006). “A fast learning algorithm for deep belief nets”. In: *Neural Computation* 18.7, pp. 1527–1554.

Williams, Ronald J. and David Zipser (1989). "A Learning Algorithm for Continually Running Fully Recurrent Neural Networks". In: *Neural Computation* 1, pp. 270–280.

Bengio, Y., P. Simard, and P. Frasconi (1994). "Learning Long-Term Dependencies with Gradient Descent is Difficult". In: *IEEE Transactions on Neural Networks* 5.2, pp. 157–166.

Hochreiter, Sepp and Jürgen Schmidhuber (Nov. 1997). "Long Short-Term Memory". In: *Neural Computation* 9.8, pp. 1735–1780.

Gers, Felix A. and Jürgen Schmidhuber (2000). "Recurrent nets that time and count". In: *Proc. International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium.*

Sak, Hasim, Andrew W. Senior, and Françoise Beaufays (2014). "Long short-term memory recurrent neural network architectures for large scale acoustic modeling". In: *Proc. INTERSPEECH.*

Graves, Alex et al. (Jan. 2006). "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks". In: *Proceedings of the 23rd International Conference on Machine Learning*, pp. 369–376.

Cho, Kyunghyun et al. (Oct. 2014). "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation". In: *Proc. Empirical Methods in Natural Language Processing (EMNLP).* Doha, Qatar: Association for Computational Linguistics.

Chorowski, Jan et al. (2015). "Attention-Based Models for Speech Recognition". In: *Proc. Neural Information Processing Systems - Volume 1.* NIPS'15. Montreal, Canada: MIT Press, 577–585.

Chai, Li et al. (2021). "Acoustic Modeling for Multi-Array Conversational Speech Recognition in the Chime-6 Challenge". In: *IEEE Spoken Language Technology Workshop (SLT).*

Young, S. J., J. J. Odell, and P. C. Woodland (1994). "Tree-Based State Tying for High Accuracy Acoustic Modelling". In: *Proceedings of the Workshop on Human Language Technology.* HLT '94. Plainsboro, NJ: Association for Computational Linguistics, 307–312.

Russell, M. and R. Moore (1985). "Explicit modelling of state occupancy in Hidden Markov Models for automatic speech recognition". In: *IEEE International Conferenceon Acoustics, Speech and Signal Processing (ICASSP).*

Oura, K., Y. Nankaku, and K. Tokuda (2006). "Hidden Semi-Markov Model Based Speech Recognition System using Weighted Finite-State Transducer". In: *Proc. IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP).*

Wei, Xizi, Melvyn Hunt, and Adrian Skilling (2019). "Neural Network-Based Modeling of Phonetic Durations". In: *Proc. INTERSPEECH.*

Povey, Daniel (Jan. 2003). “Discriminative Training for Large Vocabulary Speech Recognition”. In.

Ghoshal, Arnab and Daniel Povey (2013). “Sequence discriminative training of deep neural networks”. In: *Proc. INTERSPEECH*.

Gales, M.J.F. and P.C. Woodland (1996). “Mean and variance adaptation within the MLLR framework”. In: *Computer Speech & Language* 10.4, pp. 249–264.

Reynolds, D.A. and Richard Rose (Feb. 1995). “Robust text-independent speaker identification using Gaussian Mixture speaker models”. In: *Speech and Audio Processing, IEEE Transactions on* 3, pp. 72 –83.

Reynolds, Douglas A. (1997). “Comparison of background normalization methods for text-independent speaker verification”. In: *Proc. EUROSPEECH*.

Kenny, P., G. Boulianne, and P. Dumouchel (2005). “Eigenvoice modeling with sparse training data”. In: *IEEE Transactions on Speech and Audio Processing* 13.3, pp. 345–354.

Shum, Stephen H. et al. (2013). “Unsupervised Methods for Speaker Diarization: An Integrated and Iterative Approach”. In: *IEEE Transactions on Audio, Speech, and Language Processing* 21.

Sell, Gregory and Daniel Garcia-Romero (2014). “Speaker diarization with plda i-vector scoring and unsupervised calibration”. In: *IEEE Spoken Language Technology Workshop (SLT)*.

Guo, Jinxi et al. (Oct. 2018). “Deep neural network based i-vector mapping for speaker verification using short utterances”. In: *Speech Communication* 105.

Snyder, David et al. (2017). “Deep Neural Network Embeddings for Text-Independent Speaker Verification”. In: *Proc. INTERSPEECH*.

Sell, Gregory et al. (2018). “Diarization is Hard: Some Experiences and Lessons Learned for the JHU Team in the Inaugural DIHARD Challenge”. In: *INTERSPEECH*.

Fiscus, Jonathan G. et al. (2006). “The Rich Transcription 2006 Spring Meeting Recognition Evaluation”. In: *MLMI*.

Campbell, William M. et al. (2006). “SVM Based Speaker Verification using a GMM Supervector Kernel and NAP Variability Compensation”. In: *Proc. IEEE International Conferene on Acoustics Speech and Signal Processing Proceedings (ICASSP)*.

Dehak, N. et al. (2011). “Front-End Factor Analysis for Speaker Verification”. In: *IEEE Transactions on Audio, Speech, and Language Processing* 19.4, pp. 788–798.

Lei, Yun et al. (2014). "A novel scheme for speaker recognition using a phonetically-aware deep neural network". In: *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1695–1699.

Salton, Gerard and Christopher Buckley (1988). "Term-weighting approaches in automatic text retrieval". In: *INFORMATION PROCESSING AND MANAGEMENT*.

Mikolov, Tomas et al. (Oct. 2013a). "Distributed Representations of Words and Phrases and their Compositionality". In: *Advances in Neural Information Processing Systems* 26.

Mikolov, Tomas et al. (2013b). "Efficient estimation of word representations in vector space". In: *Proc. ICLR*.

Le, Quoc and Tomas Mikolov (2014). "Distributed Representations of Sentences and Documents". In: *Proceedings of the 31st International Conference on International Conference on Machine Learning*. Beijing, China.

Cortes, Corinna and Vladimir Vapnik (Sept. 1995). "Support-Vector Networks". In: *Mach. Learn.* 20.3, 273–297.

Kusner, Matt J. et al. (2015). "From Word Embeddings to Document Distances". In: *Proc. International Conference on International Conference on Machine Learning*.

Nasir, Md et al. (2019). "Modeling Interpersonal Linguistic Coordination in Conversations using Word Mover's Distance". In: *Proc. INTERSPEECH*.

Porter, M. F. (1997). "An Algorithm for Suffix Stripping". In: *Readings in Information Retrieval*, 313–316.

Dumais, S.T. (Jan. 2004). "Latent Semantic Analysis". In: *Annual Review of Information Science and Technology* 38, pp. 188–230.

Blei, David M., Andrew Y. Ng, and Michael I. Jordan (2003). "Latent Dirichlet Allocation". In: *J. Mach. Learn. Res.* 3, 993–1022.

Robinson, A. et al. (1995). "WSJCAM0: A British English Speech Corpus For Large Vocabulary Continuous Speech Recognition". In: *Proc. IEEE-ICASSP,* Detroit, MI.

Mccowan, I. et al. (2005). "The AMI Meeting Corpus". In: *Proc. International Conference on Methods and Techniques in Behavioral Research*.

Rayner, Manny et al. (2010). *A Multilingual CALL Game Based on Speech Translation*.

Baur, Claudia et al. (2017). "Overview of the 2017 Spoken CALL Shared Task". In: *7th ISCA International Workshop on Speech and Language Technology in Education, SLaTE*.

Batliner, A et al. (2005). "The PF_STAR children's speech corpus". In: *European Conference on Speech Communication and Technology*.

Oh, Yoo Rhee et al. (2017). "Deep-Learning Based Automatic Spontaneous Speech Assessment in a Data-Driven Approach for the 2017 SLaTE CALL Shared Challenge". In: *7th ISCA International Workshop on Speech and Language Technology in Education, SLaTE.*

Diabetes UK (n.d.). URL: https://www.diabetes.org.uk/.

"Google Code Archive - Long-term storage for Google Code Project Hosting." (n.d.). In: *Google* (). URL: https://code.google.com/archive/p/word2vec/.

Pedregosa, F. et al. (2011). "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12, pp. 2825–2830.

Řehůřek, Radim and Petr Sojka (May 2010). "Software Framework for Topic Modelling with Large Corpora". English. In: *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks.* Valletta, Malta: ELRA, pp. 45–50.

Peddinti, Vijayaditya, Daniel Povey, and Sanjeev Khudanpur (2015b). "A time delay neural network architecture for efficient modeling of long temporal contexts". In: *Proc. INTERSPEECH.*

Snyder, David et al. (2018). "X-Vectors: Robust DNN Embeddings for Speaker Recognition". In: *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).*