Data-Diverse Redundant Processing for Noise-Robust Automatic Speech Recognition

---

A Thesis

Presented to

the faculty of the School of Engineering and Applied Science

University of Virginia

---

in partial fulfillment
of the requirements for the degree

Master of Science

by

Mustafa K. Hotaki

August 2020

APPROVAL SHEET

This Thesis
is submitted in partial fulfillment of the requirements
for the degree of
Master of Science

Author Signature: Mustafa Hotaki

This Thesis has been read and approved by the examining committee:

Advisor: Ronald D. Williams

Advisor: Homa Alemzadeh

Committee Member: Joanne B. Dugan

Committee Member: John A. Stankovic

Committee Member: _____

Committee Member: _____

Accepted for the School of Engineering and Applied Science:

Craig H. Benson, School of Engineering and Applied Science

August 2020

Data-Diverse Redundant Processing for Noise-Robust Automatic Speech Recognition

Mustafa K. Hotaki

Department of Electrical and Computer Engineering
University of Virginia

This thesis is submitted for the degree of
Master of Science
in
Computer Engineering

August 2020

I would like to dedicate this thesis to my loving parents.

Acknowledgements

The author is indebted to Professor Homa Alemzadeh and Professor Ronald D. Williams, who supervised this thesis, for their guidance, mentorship, and helpful criticism. I would also like to express my gratitude to my undergraduate advisor, Professor Joanne B. Dugan, for her invaluable support of all students in the Computer Engineering program at the University of Virginia. Finally, I would like to thank Professor John A. Stankovic on my defense committee for his time. I acknowledge that this work was supported by the award 60NANB17D162 from the United States Department of Commerce, National Institute of Standards and Technology (NIST).

Abstract

Robustness to acoustic noise remains a challenge in automatic speech recognition (ASR). In this work, we take a fault-tolerance approach to noise-robustness by applying data diversity [1] to the *input* speech signal of an ASR system. Motivated by the observation that ASR systems are sensitive to perturbations in the input under noisy conditions, our proposed framework, termed *data-diverse redundant processing*, creates a diverse set of *variants* of the input speech signal by applying label-preserving transformations such as time warping and speed modulation. Treating a given ASR system as a black box, we process the variants to generate a list of transcripts, termed *hypotheses*. Our experiments show that we are able to generate diverse hypotheses in noisy environments quantified by the average of pair-wise word error rate (WER) values, known as the Cross-WER [76]. We show error-correcting *potential* or complementarity of these hypotheses using the notion of an *oracle combination* or a "best possible combination" of them guided by the ground truth or reference transcripts, for which we provide an algorithm. Our results show potential for consistent reductions (in an ideal sense) in WER for noisy speech. We implement a modified version of the ROVER algorithm [23] for combining multiple hypotheses into a single hypothesis. We evaluate our framework on clean and realistic noisy speech from the CHiME3 dataset for the Google Cloud Speech to Text, IBM Watson Speech to Text, and Microsoft Azure Speech to Text systems. Our results maintain the original performances on clean data but achieve reductions of 2.31%, 3.88%, and 3.5% over baseline WERs by a simple majority voting mechanism using as few as five transformations. We further show empirical lower bounds on WER on generated confusion networks (CNs) promising even greater reductions in WER of 8.6%, 8.36%, and 6.51% for the Google, IBM, and Microsoft systems respectively. We point to existing work on more sophisticated mechanisms such as confusion network re-scoring using language understanding models to get WER values that more closely resemble these lower bounds. We conclude that data diversity is a viable orthogonal method for noise-robustness, but its efficacy is limited by the underlying ASR and its use is only encouraged when computational overhead of redundant processing is not a concern.

Keywords— Automatic speech recognition, noise-robustness, data diversity, hypothesis combination, ROVER

Publications

S. Preum, S. Shu, M. Hotaki, R. Williams, J. Stankovic, H. Alemzadeh, *"CognitiveEMS: A Cognitive Assistant System for Emergency Medical Services."* 7th IEEE Workshop on Medical Cyber-Physical Systems, 2018.

Table of contents

List of figures

List of tables

Nomenclature

General Notation

a	A scalar is represented by a lower-case, non-bold, and italic letter
$\boldsymbol{a}$	A vector, sequence, or set (or multiset [1]) is represented by a lower-case and bold letter
A	A variable of arbitrary/unknown dimensions
$\mathbf{A}$	A sequence of non-scalar elements, a matrix, or tensor
$\lvert\boldsymbol{a}\rvert$	Number of elements in a set or sequence $\boldsymbol{a}$
$\arg\max f(x)$	x such that $f(x)$ is maximized
$\arg\min_x f(x)$	x such that $f(x)$ is minimized
$\text{Unique}(\boldsymbol{a})$	Removes repeating elements from a sequence
$\mathbb{1}_{\text{(condition)}}$	Indicator function on some condition

Automatic Speech Recognition

x	An audio sample
$\boldsymbol{x}$	A sequence of speech samples corresponding to an utterance
w	A word
$\boldsymbol{w}$	A sequence of words
$\mathcal{A}$	Acoustic Domain
$\mathcal{V}$	A vocabulary
$\mathcal{W}$	A language
$\mathcal{F}(.)$	ASR function
$\boldsymbol{w}^{\text{Ref}}$	Ground truth reference transcript
$\hat{\boldsymbol{w}}$	Hypothesis transcript, output of an ASR
$\hat{\boldsymbol{w}}_{\text{MAP}}$	The maximum a posteriori sequence
$\hat{\boldsymbol{w}}_{\text{MBR}}$	The Minimum Bayes' Risk sequence
$\mathcal{L}(\boldsymbol{w}, \boldsymbol{w}')$	An arbitrary loss/distance between two word sequences
$\mathcal{L}_{\text{MAP}}(\boldsymbol{w}, \boldsymbol{w}')$	The MAP or one-zero loss function
$\mathcal{L}_{\text{Word}}(\boldsymbol{w}, \boldsymbol{w}')$	The Levenshtein distance between two word sequences
$\text{Levenshtein}_{\boldsymbol{a},\boldsymbol{b}}(i, j)$	The Levenshtein *function*, used to define the Levenshtein *distance*
$\text{Levenshtein-Align}(\boldsymbol{w}, \boldsymbol{w}')$	An alignment corresponding to a Levenshtein distance

[1] A multiset is a modification of the concept of a set that allows for repetitions of elements.

$\boldsymbol{w}^*$ An aligned sequence of words
N_{Ins} Number of insertions from a Levenshtein distance
N_{Del} Number of deletions from a Levenshtein distance
N_{Sub} Number of substitutions from a Levenshtein distance
N_{Ref} Number of words in a reference sequence $\boldsymbol{w}^{\text{Ref}}$
$\mathbf{X}$ Speech Utterances
$\mathbf{W}^{\text{Ref}}$ Ground truth reference transcripts
$\hat{\mathbf{W}}$ Hypothesis transcripts
$\text{WER}(\mathbf{W}^{\text{Ref}}; \hat{\mathbf{W}})$ Word Error Rate for multiple utterance
$\text{WER}(\boldsymbol{w}^{\text{Ref}}; \hat{\boldsymbol{w}})$ Word Error Rate for a single utterance
$\mathcal{C}_i$ The ith confusion set in a confusion network
$\mathcal{Q}$ A sequence of confusion sets representing a confusion network
$\mathcal{G}$ A Directed Acyclic Graph representing a hypothesis space
$\mathbf{O}$ A vector of acoustic observation vectors

Data Diversity for Automatic Speech Recognition

N The number of transformations
$\mathcal{T}_i(\boldsymbol{x})$ The ith transformation applied to a speech signal $\boldsymbol{x}$
$\boldsymbol{x}^{\mathcal{T}_i}$ The transformed speech signal (a variant) found by applying $\mathcal{T}_i(\boldsymbol{x})$
$\mathcal{X}$ Variants of a speech signal $\boldsymbol{x}$ after transformations are applied to it
$\boldsymbol{w}^{\mathcal{T}_i}$ A hypothesis sequence found by an ASR by processing a transformed speech signal $\boldsymbol{x}^{\mathcal{T}_i}$
$\mathcal{H}$ A sequence of hypotheses, where a hypothesis is a word sequence
$\mathcal{D}(\mathcal{H})$ A measure of diversity among hypotheses $\mathcal{H}$
$\text{Cross-WER}(\mathcal{H})$ Cross-WER among hypotheses $\mathcal{H}$ is measure of diversity
$\Phi(\mathcal{H})$ The oracle combination of hypotheses $\mathcal{H}$
$\boldsymbol{w}^{\Phi}$ The oracle combination of hypotheses $\mathcal{H}$
$\text{Filter}(\boldsymbol{a}; \boldsymbol{b})$ Removes elements from a sequence $\boldsymbol{a}$ that belong to the set $\boldsymbol{b}$
$\boldsymbol{w}^{\text{Base}}$ A sequence representing a baseline transcription, which we want to improve upon
$\Delta^{\Phi}_{\boldsymbol{w}^{\text{Base}}}(\mathcal{H})$ The improvement achieved by the oracle combination Φ over a baseline

Combination of Hypothesis

$\Psi(\mathcal{H})$ A combination of the hypotheses $\mathcal{H}$
$\boldsymbol{w}^{\Psi}$ A sequence found by combining hypotheses
$\Delta^{\Psi}_{\boldsymbol{w}^{\text{Base}}}(\mathcal{H})$ The improvement achieved by the combination Ψ over a baseline
$\boldsymbol{w}^{\text{Path}}$ A sequence of words corresponding to a path through a graph $\mathcal{G}$
$\text{MSA}(.)$ A multiple sequence alignment
$\mathcal{H}^*$ A list of aligned sequences, an alignment table
$\mathcal{J}_i$ A column in the alignment table $\mathcal{H}^*$
$\text{ROVER}(\mathcal{H})$ The ROVER combination of hypotheses $\mathcal{H}$
$\mathcal{Q}^{\text{ROVER}}$ A sequence of confusion sets generated by ROVER

$\mathcal{C}^{\text{ROVER}}$ A confusion set in $\mathcal{Q}^{\text{ROVER}}$

$\boldsymbol{w}^{\text{ROVER}}$ A sequence of words found by ROVER voting

Count(w) The number of times w appears in a column in an alignment table

Confidence(w) A confidence score associated with a word in a confusion set $\mathcal{C}^{\text{ROVER}}$

w_i^{ROVER} The ith word in a sequence found by ROVER voting

$\boldsymbol{w}^{\text{Best}}$ A sequence of words representing a path through the graph $\mathcal{G}$ that has the best WER

$\boldsymbol{w}^{\text{Worst}}$ A sequence of words representing a path through the graph $\mathcal{G}$ that has the worst WER

$\boldsymbol{w}^{\text{Random}}$ A sequence of words found by stochastic selection of a path from the graph $\mathcal{G}$

Probability and Distributions

$P(.)$ Probability Mass Function

$p(x)$ Probability Density Function

$p(x, y)$ Joint Probability Density Function

$p(x \mid y)$ Conditional probability density of x given y

Acronyms / Abbreviations

AM Acoustic Model

API Application Programming Interface

ASR Automatic Speech Recognition

BERT Bidirectional Encoder Representations from Transformers

CN Confusion Network

CNC Confusion Network Combination

DAG Directed Acyclic Graph

DNN Deep Neural Network

DTW Dynamic Time Warping

EMS Emergency Medical Services

EMT Emergency Medical Technician

GMM Gaussian Mixture Model

HMM Hidden Markov Model

LM Language Model

LPT Label-Preserving Transformation

MAP Maximum a Posteriori

MBR Minimum Bayes' Risk

MFCC Mel-Frequency Cepstral Coefficients

MT Minimum Mean Square Error

MSA Multiple Sequence Alignment

MT Machine Translation

NIST National Institute of Standards and Technology

NLP Natural Language Processing

PMF Probability Mass Function

REST Representational State Transfer
SDK Representational State Transfer
RNN Recurrent Neural Network
ROVER Recognizer Output Voting Error Reduction
SDK Software Development Kit
SNR Signal-to-noise Ratio
STFT Short-Time Fourier Transform
STT Speech-to-Text
TTS Text-to-speech
WER Word Error Rate
WSJ Wall Street Journal
WTN Word Transition Network

Chapter 1

Introduction

Speech is a natural method of communication between humans. The field of automatic speech recognition (ASR) has been an intensive area of research for many decades aimed at extending this intuitive mode of communication to between humans and machines. In this introductory chapter, we motivate the need for accurate automatic speech recognition in noisy environments and introduce our approach, termed *data-diverse redundant processing*, to improving the noise-robustness of these systems. We conclude the chapter with an overview of the the structure of this thesis.

1.1 Motivation

Automatic speech recognition (ASR) or Speech-to-Text (STT) is an essential human-computer interaction interface. A simple black box model (Figure 1.1) of an ASR system takes an acoustic waveform as an input and produces a sequence of words as an output.

Fig. 1.1 Black Box Model of an Automatic Speech Recognition System

ASR has been an active area of research for over five decades. An early system developed at the Bell Telephone Laboratories [15] used circuits to compare frequency content of speech to recognize spoken digits. Another early system used Dynamic Time Warping (DTW) [64] for recognition of spoken words. The introduction of the Hidden Markov Model (HMM) [6] framework for ASR allowed for speech recognition to be treated as a statistical pattern recognition problem with separate models to capture acoustic and linguistic aspects of speech. The replacement of Gaussian Mixture Model (GMM) acoustic models and n-gram language models by deep learning approaches [39, 55] has contributed to major progress in the recent

years. Some so-called "end-to-end" architectures using recurrent neural networks (RNNs) [32, 31, 34] have also been explored and shown great promise. Today, voice assistant systems such as Amazon Alexa, Apple Siri, Google Home, and Microsoft Cortana have become ubiquitous. Additionally, leading cloud computing platforms such Amazon Web Services, Google Cloud Platform, IBM Watson, and Microsoft Azure all provide state-of-the-art models as a service to be used in custom user applications. Despite the great ubiquity and advancements, however, the accuracy of ASR transcriptions degrades under realistic noisy conditions. Such deterioration in performance may be less tolerable in some applications such as safety-critical systems. An example application of ASR technology in such a safety-critical system is a voice-enabled *cognitive assistant system* for emergency medical services (EMS) in [63]. This device is expected to be used in environments with many acoustic noises such as those from traffic, people talking in the background, and radio chatter. Speech-to-Text and Text-to-Speech (TTS) are the main forms of interaction between an emergency medical technician (EMT) and a computer system in this device. It is crucial that ASR is accurate in transcribing speech.

1.2 Approach

In this work, we take a fault-tolerance approach to the noise-robustness problem in automatic speech recognition. Unlike previous work that use robust front-ends or enhance the underlying ASR models, we exploit data diversity [1] by algorithmically modifying the *input* speech signal of an ASR in many different ways. This is motivated by the empirical observation that ASR systems are sensitive to *input* perturbations under noisy conditions i.e., their output may change when the input is even slightly changed. Our proposed framework, termed *data-diverse redundant processing*, creates a diverse set of *variants* of the input speech signal by applying *label-preserving transformations* (LPTs), which add spectral and temporal variations to the signal without modifying its speech content i.e., such that an expert human transcriber's annotation of the speech signal does not change with the transformations. The set of data-diverse variants are all processed by an ASR system with the hope that they will generate *diverse* and *complementary* transcripts. We measure ASR *output* diversity using the average of pairwise WER values among the transcripts, also known as Cross-WER [76]. We demonstrate complementarity of hypotheses coming from different transformations using the notion of an *oracle combination* or "best possible combination" of them. This notion is somewhat ambiguous. We provide an algorithmic definition that iteratively aligns hypothesis transcripts to the reference transcript and builds a combination transcript by only keeping the correctly recognized words. We can show complementarity of hypotheses if the oracle combination of them leads to a hypothesis with fewer errors.

Once multiple complementary transcripts are generated by processing the diverse variants, a (non-oracle) *hypothesis combination* scheme is applied to their transcripts to generate a combined transcript. A simple approach to this problem is the Recognizer Output Voting Error Reduction (ROVER) [23] which uses multiple sequence alignment (MSA) to align the hypotheses together. It then generates a graphical representation of a *hypothesis space* from the alignment where words are represented as edges in a directed acyclic graph (DAG) known as a Confusion Network (CN) [1]. Finally, a "voting" mechanism selects a single hypothesis (a path through the confusion network) as the output of the system. For our work, we implemented a modified version of ROVER and used a simple majority voting scheme for finding paths through generated confusion networks. Figure 1.2 shows the data-diverse redundant processing framework investigated in this thesis[2]. This approach is orthogonal to existing approaches to noise-robustness, which generally either aim to enhance the input speech signals or the underlying ASR models. Our work is different, in that it exploits the sensitivity of ASR systems to perturbations under noisy conditions.

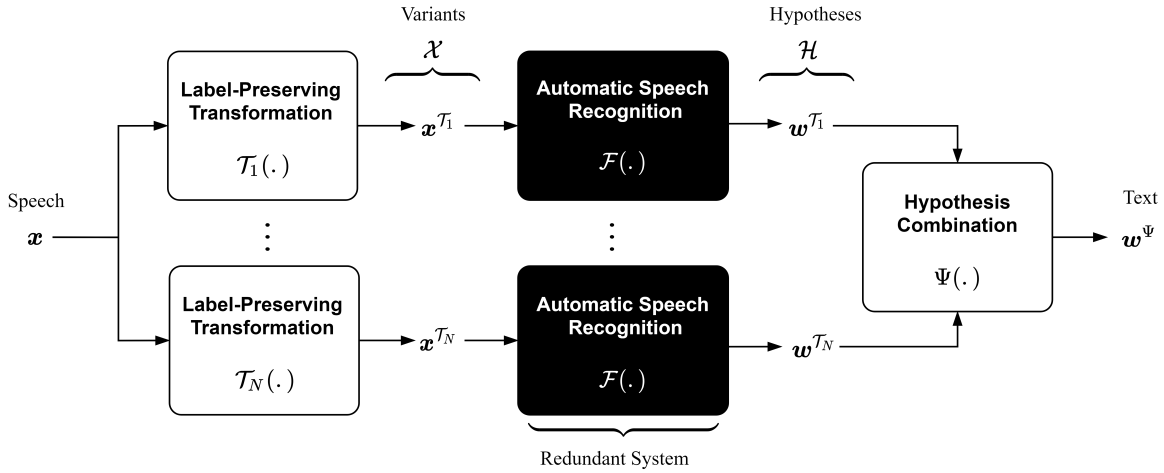


Fig. 1.2 Data-Diverse Redundant Processing Framework

We evaluate our framework in terms of word error rate (WER) using the Google Cloud Speech to Text[3], IBM Watson Speech to Text[4], and Microsoft Azure Speech to Text[5] on the `Evaluation` set of CHiME3 [7] dataset. These systems and the CHiME3 dataset are discussed in more detail in Appendix B. Our experimental results show that we could reduce WER by modest amounts of **2.31%**, **3.88%**, and **3.5%** using as few as *five* transformations. We also show empirical lower bounds on WER on best-performing paths on confusion networks generated by our ROVER-like combination scheme suggesting that even greater gains are

[1] ROVER calls this graph a Word Transition Network (WTN)
[2] The notation will be explored in later chapters.
[3] https://cloud.google.com/speech-to-text
[4] https://www.ibm.com/cloud/watson-speech-to-text
[5] https://azure.microsoft.com/en-us/services/cognitive-services/speech-to-text/

possible. These lower bounds correspond to upper bounds on *reduction* in WER of **8.6%**, **8.36%**, and **6.51%** for the Google, IBM, and Microsoft systems respectively. In agreement with prior work [38, 65], we claim that simple majority voting is incapable of exploiting full potential for improvements. We point to existing work on more sophisticated mechanisms as an avenue for future work to get WER values that more closely resemble these lower bounds. Our results indicate that data diversity is a viable orthogonal method to noise-robustness in ASR but its efficacy is limited by the underlying ASR and the combination method. We also note that the use of this approach is only encouraged when the cost associated with the redundant processing of speech is not a concern[6].

1.3 Thesis Organization

This thesis contains 6 chapters and is structured as follows:

- Chapter 2 covers the fundamentals of black box automatic speech recognition and noise-robustness. It also establishes a baseline for clean and noisy data.
- Chapter 3 presents our approach and results for using data diversity to generate diverse and complementary transcripts.
- Chapter 4 presents our approach and results for combining transcripts for reduced word error rates.
- Chapter 5 evaluates our framework on unseen data.
- Chapter 6 summarizes our work and presents high-level conclusions. It also suggests future directions in research.

[6]This is not an unrealistic use case, as it is possible that we may have unlimited access to just a single ASR.

Chapter 2

Automatic Speech Recognition

This chapter provides the necessary technical preliminaries and formulations for automatic speech recognition (ASR). We first define a black box model for ASR and provide the classic decision-theoretic formulations of the ASR task. Although not necessary in a black box setting, we provide high-level descriptions of traditional and end-to-end ASR architectures. We introduce the word error rate (WER) evaluation metric and considerations for computing it accurately. We then introduce graphical models for ASR outputs. Finally, we explore existing work on the problem of noise-robustness in ASR. We conclude the chapter by empirically quantifying the performance deterioration of multiple commercial ASR systems on noisy data from the `Development` set of the CHiME3 dataset.

2.1 Black Box Model for Automatic Speech Recognition

Automatic speech recognition (ASR) is a sequence-to-sequence task, where the input is an acoustic signal containing human speech, and the output is text. The input is often a digital representation of an analog pressure wave sampled at a given sampling rate and a given bit depth. The input acoustic waveform $\boldsymbol{x}$ could be represented as a time-series sequence of length T:

$$\boldsymbol{x} = x_1, x_2, \ldots, x_T \tag{2.1}$$

Text is often represented as a sequence of words of length L:

$$\boldsymbol{w} = w_1, w_2, \ldots, w_L \tag{2.2}$$

where the sequence of words $\boldsymbol{w}$ is an element of the *language set* $\mathcal{W}$, and a word w is an element of a *vocabulary set* $\mathcal{V}$[1]. An ASR system $\mathcal{F} : \mathcal{A} \mapsto \mathcal{W}$ maps elements from the

[1]We are not considering out-of-vocabulary (OOV) words here for simplicity

acoustic domain $\mathcal{A}$ to the language domain $\mathcal{W}$ i.e., the function $\mathcal{F}$ takes $\boldsymbol{x}$ as input and returns a hypothesized word sequence $\hat{\boldsymbol{w}}$:

$$\hat{\boldsymbol{w}} = \mathcal{F}(\boldsymbol{x}) = \hat{w}_1, \hat{w}_2, \ldots, \hat{w}_{\hat{L}}$$

We can also define a *reference* transcript $\boldsymbol{w}^{\text{Ref}}$ containing the actual words that were uttered in the speech signal $\boldsymbol{x}$. These are only available to us during development and evaluation of a system.

$$\boldsymbol{w}^{\text{Ref}} = \mathcal{R}(\boldsymbol{x}) = w_1^{\text{Ref}}, w_2^{\text{Ref}}, \ldots, w_{L^{\text{Ref}}}^{\text{Ref}} \tag{2.3}$$

The reference transcripts are usually manually annotated by expert human transcribers[2], which we denote as a function $\mathcal{R} : \mathcal{A} \mapsto \mathcal{W}$. Informally, a machine learning approach to ASR tries to learn the $\mathcal{F}$ as an estimate of $\mathcal{R}$. We provide the classic decision-theoretic formulation of ASR in § 2.2.

2.2 Decision-Theoretic Formulation of ASR

Automatic speech recognition can be treated as a discriminative classification task that uses the posterior probability $P(\boldsymbol{w} \mid \boldsymbol{x})$. One intuitive *decision rule* for ASR is the Maximum a Posteriori (MAP) [5]:

$$\hat{\boldsymbol{w}}_{\text{MAP}} = \arg\max_{\boldsymbol{w}} P(\boldsymbol{w} \mid \boldsymbol{x}) \tag{2.4}$$

Given an acoustic input $\boldsymbol{x}$, the MAP decision rule finds the word sequence $\boldsymbol{w}$ that maximizes the posterior probability $P(\boldsymbol{w} \mid \boldsymbol{x})$. The MAP decision rule can be rewritten as [77]:

$$\begin{aligned} \hat{\boldsymbol{w}}_{\text{MAP}} &= \arg\min_{\boldsymbol{w}'} \sum_{\boldsymbol{w} \neq \boldsymbol{w}'} P(\boldsymbol{w} \mid \boldsymbol{x}) \\ &= \arg\min_{\boldsymbol{w}'} \sum_{\boldsymbol{w}} \mathcal{L}_{\text{MAP}}(\boldsymbol{w}, \boldsymbol{w}') P(\boldsymbol{w} \mid \boldsymbol{x}) \end{aligned} \tag{2.5}$$

where

$$\mathcal{L}_{\text{MAP}}(\boldsymbol{w}, \boldsymbol{w}') = \begin{cases} 0 & \text{if } \boldsymbol{w} = \boldsymbol{w}' \\ 1 & \text{otherwise} \end{cases} \tag{2.6}$$

[2]It is possible that different human transcribers produce different transcriptions for the same speech. This is not addressed in our work.

We observe that MAP decision rule incurs equal cost for all misclassifications i.e., it minimizes the expected misclassifications. The MAP decision rule can be generalized by the Minimum Bayes' Risk (MBR) decision rule [56, 57, 69, 27]:

$$\hat{\boldsymbol{w}}_{\text{MBR}} = \arg\min_{\boldsymbol{w}'} \sum_{\boldsymbol{w}} \mathcal{L}(\boldsymbol{w}, \boldsymbol{w}') P(\boldsymbol{w} \mid \boldsymbol{x}) \tag{2.7}$$

where $\mathcal{L}$ is an arbitrary *loss function*. The sum in Equation 2.7 is an (conditional) expectation of the loss function $\mathcal{L}$. It is referred to as a (conditional) *risk* in Bayesian Decision Theory formalism [18]. To minimize error on the word level rather than sentence[3] level, the Levenshtein distance [50] or the (minimum) edit distance between two word sequences $\boldsymbol{w}$ and $\boldsymbol{w}'$ can be used:

$$\mathcal{L}_{\text{Word}}(\boldsymbol{w}, \boldsymbol{w}') = N_{\text{Ins}} + N_{\text{Del}} + N_{\text{Sub}} \tag{2.8}$$

where N_{Ins}, N_{Del}, and N_{Sub} refer to the number of insertions, deletions, and substitutions that make up the Levenshenstein distance. The Levenshtein distance is discussed further in Appendix A.1.

We observe that this treatment so far assumes that the posterior probability $P(\boldsymbol{w} \mid \boldsymbol{x})$ is known. However, this is not the case in practice. Fortunately, we can apply Bayes' Theorem:

$$P(\boldsymbol{w} \mid \boldsymbol{x}) = \frac{P(\boldsymbol{x} \mid \boldsymbol{w}) P(\boldsymbol{w})}{P(\boldsymbol{x})} \tag{2.9}$$

Since term $P(\boldsymbol{x})$ in the denominator does not change with different $\boldsymbol{w}'$ when minimizing over all possible $\boldsymbol{w}' \in \mathcal{W}$ in Equation 2.7, the MBR decision rule can be expressed as:

$$\hat{\boldsymbol{w}}_{\text{MBR}} = \arg\min_{\boldsymbol{w}'} \sum_{\boldsymbol{w}} \mathcal{L}(\boldsymbol{w}, \boldsymbol{w}') P(\boldsymbol{x} \mid \boldsymbol{w}) P(\boldsymbol{w}) \tag{2.10}$$

The term $P(\boldsymbol{x} \mid \boldsymbol{w})$ is known as the *likelihood* of the acoustic vector $\boldsymbol{x}$, and the term $P(\boldsymbol{w})$ is know as the *prior probability* of the word sequence $\boldsymbol{w}$. The challenge in designing statistical ASR systems has to do with computing these probabilities and searching over word sequences.

2.3 Architecture

This thesis is not concerned with the inner workings of automatic speech recognition systems. However, a high-level view of common architectures can be useful. Below, we very briefly describe a traditional pipeline, which depends on multiple separate components. We also

[3] We note that the term sentence is a semantic concept but is used to mean a sequence of words here.

briefly discuss more recent efforts to build end-to-end ASR systems, with the aim of replacing these components with a single system.

2.3.1 Traditional Architectures

Traditional ASR architectures depend on multiple components to perform speech recognition. First, the input speech signal $\boldsymbol{x}$ is processed by a *frontend* stage to generate feature vectors $\mathbf{O} = \mathbf{o}_1, \mathbf{o}_2, \ldots, \mathbf{o}_\tau$. These features are usually compact time-frequency representations of speech such as Mel Frequency Cepstral Coefficients (MFCC). Feature normalization techniques may also be applied in the frontend stage. These feature vectors are passed to a *decoder* module, which searches over sequences of words, making use of the language and acoustic models. This may be done using Viterbi approximation or using forward-backward operations over a lattice [77]. In a traditional pipeline, the acoustic model is trained to compute the likelihood $P(\mathbf{O} \mid \boldsymbol{w})$. Hidden Markov Models (HMMs) and Deep Neural Networks (DNN) have shown major success on this task. The language model is trained to compute the prior probability $P(\boldsymbol{w})$ of a particular word sequence. A common approximation is the n-gram language model, which under certain assumptions, can compute the conditional probability of a word given $n-1$ previous words. More recently, neural masked language models such as the Bidirectional Encoder Representation from Transformers (BERT) [16] have become popular. These models train networks to predict a probability distribution $P(w_i \mid \text{context})$, where context refers to both previous and future words. ASRs may also make use of a phonetic dictionary, which is a mapping between words and phonetic units such as phones. These systems are usually quite large and are hosted in the cloud or clusters of machines.

2.3.2 End-to-end Architectures

More recent approaches to ASR consider so-called "end-to-end" architectures [32, 34, 31]. These approaches replace traditional HMM-DNN pipelines with variants of recurrent neural networks (RNNs). This has largely been possible with the introduction of the Connectionist Temporal Classification (CTC) [30] which allows for RNNs to work with unsegmented speech data. Deep Speech [34] by Baidu is one architecture that has shown great improvements. Mozilla's DeepSpeech[4] is a popular open-source system based on this architecture. We note, however, that most of these systems are not truly end-to-end as of yet and rely to varying degrees on traditional components such as language models.

[4] https://github.com/mozilla/DeepSpeech

2.4 Performance Evaluation

The performance of an ASR is empirically evaluated on speech datasets. In § 2.4.1, we discuss considerations for selecting speech datasets. In § 2.4.2, we discuss the word error rate (WER), the standard metric for evaluating ASR systems.

2.4.1 Speech Datasets

A number of considerations such as speaker variability, vocabulary size, accents and dialects, context, speaking style, and background noise need to go into choosing a benchmark dataset. A benchmark dataset needs to be representative of the type of speech we wish to recognize with an ASR. In this work, we are interested in improving the performance of an ASR on noisy speech such that it more closely resembles performance on clean speech. The difficulty in this lies in creating a dataset with both clean and noisy subsets, where all other variables except for noise are consistent. Artificial corruption of audio such as corruption with additive and/or convolutional noise has been explored in prior work [40]. This approach is appealing because we only need to collect clean audio. Noise could be added systematically using different noise models or actual noisy audio samples. This also allows for more direct control over the signal-to-noise ratio (SNR). However, phenomena such as the *Lombard effect* [74], an observed tendency of people to change their speech in noisy environments, threaten the validity of such models. In this work, we make use of the 3rd CHiME Speech Separation and Recognition Challenge (CHiME3) [7] dataset. CHiME3 is designed around the Wall Street Journal (WSJ) [26] corpus and contains speech recorded in both noise-free and challenging noisy environments. A detailed description of of the subsets of dataset used in this thesis can be found in Appendix B.2.

2.4.2 Word Error Rate

The standard metric for assessing the performance of automatic speech recognition is the Word Error Rate (WER). We first define WER for a R speech utterances (i.e., data points) $\mathbf{X} = \boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_R$. Ground-truth labels or reference transcripts for these data points are the *references* $\mathbf{W}^{\text{Ref}} = \boldsymbol{w}_1^{\text{Ref}}, \mathbf{w}_2^{\text{Ref}}, ..., \boldsymbol{w}_R^{\text{Ref}}$. Feeding each $\boldsymbol{x}_i$ to an ASR, we get the outputs $\hat{\mathbf{W}} = \hat{\boldsymbol{w}}_1, \hat{\boldsymbol{w}}_2, ..., \hat{\boldsymbol{w}}_R$, which we will refer to as the *inferences* or *hypotheses*. The word error rate is then the sum of (minimum) edits distances or Levenshtein distances (see Appendix A.1 for more details) between corresponding members of $\mathbf{W}^{\text{Ref}}$ and $\hat{\mathbf{W}}$ divided by the total number of words in all members of $\mathbf{W}^{\text{Ref}}$. This is sometimes referred to as *micro-averaging* and is done in order to weigh all errors equally across data points:

$$\text{WER}(\mathbf{W}^{\text{Ref}}; \hat{\mathbf{W}}) = \frac{\sum_{i=1}^{R} \mathcal{L}_{\text{Word}}(\boldsymbol{w}_i^{\text{Ref}}, \hat{\boldsymbol{w}}_i)}{\sum_{j=1}^{R} |\boldsymbol{w}_j^{\text{Ref}}|} \tag{2.11}$$

If computing WER for a single data point (i.e., $R = 1$), the WER defintion of Equation 2.11 reduces to:

$$\text{WER}(\boldsymbol{w}^{\text{Ref}}; \hat{\boldsymbol{w}}) = \frac{N_{\text{Ins}} + N_{\text{Del}} + N_{\text{Sub}}}{N_{\text{Ref}}} \tag{2.12}$$

where

$$N_{\text{Ref}} = |\boldsymbol{w}^{\text{Ref}}| \tag{2.13}$$

The WER is often multiplied by 100 and reported as a percent. It should be noted that WER can exceed one hundred percent due to non-zero number of insertions N_{Ins}. The computation is efficiently done using dynamic programming (DP). The computation can be related to pair-wise sequence alignment, where the alignment satisfies the Levenshtein criterion i.e., minimizes the total number of insertions, deletions, and substitutions. The Wagner-Fischer algorithm [75] is the most common algorithm for this task. We employed a modified open-source implementation of the WER computation[5] in this thesis. The example below illustrates WER computation for a single data point.

Reference	the	cat	in	the	hat	sat	_	on	the	mat
Hypothesis	the	bat	in	_	hat	sat	down	on	the	mat
Operation	OK	SUB	OK	DEL	OK	OK	INS	OK	OK	OK

$$\text{WER} = \frac{1+1+1}{9} \times 100 \approx 33.33\%$$

2.4.3 Considerations for Word Error Rate Computation

We observe that although many efficient implementations of the Wagner-Fischer algorithm exist, care must be taken for accurate computation of the WER. Stylistic differences between reference and recognition transcripts can make this computation quite problematic. For this work, we implemented a text normalization module informed by manually reviewing the errors. Text normalization refers to the conversion of both the reference transcripts $\mathbf{W}^{\text{Ref}}$ and recognition transcripts $\hat{\mathbf{W}}$ into a standardized format for more accurate WER computation. The example below illustrates a problem when the reference transcription and the ASR system represent and format numbers differently:

[5] https://github.com/Franck-Dernoncourt/ASR_benchmark

Reference	employment	fell	forty	seven	thousand	after
Hypothesis	employment	fell	_	_	47000	after

We addressed a number of categories in our text normalization module. These included acronyms, abbreviations, ordinal and cardinal numbers, decimals, dates, currency, symbols, and alternate spellings. We adopted the Python `num2words` [6] and `words2num` [7] modules for conversions dealing with numbers. Some cases were more challenging than others to write simple conversion rules for such as:

Reference	one	quarter
Hypothesis	1/4	_

For such cases, if they occurred frequently in a dataset, we implemented heuristic normalization rules. It should be noted that our normalization does not address all possible issues. For this reason, the WER values reported in this thesis may be slightly off from actual WERs.

2.5 Graphical Models for ASR Outputs

ASR tools are typically quite configurable and could output more than just a single *one-best* sequence of words. The output is often represented as a directed acyclic graph (DAG) denoted as $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ with edges representing words. The edges could be associated with a wealth of metadata such as timing information and confidences from different models. A hypothesis is then simply a sequence of words along edges from a *start* node on the left to a *stop* node on the right. In our work, we only consider the one-best sequence as the ASR output, but the DAG model of *hypothesis spaces* is useful for the hypothesis combination problem of Chapter 4. Below, we discuss some common ASR output formats in ascending order of complexity.

One-Best Sequence: The simplest output is the *one-best sequence* which consists of a single hypothesis. This is typically the most confident sequence of words found by the ASR. Figure 2.1 shows the graphical representation of a one-best sequence:

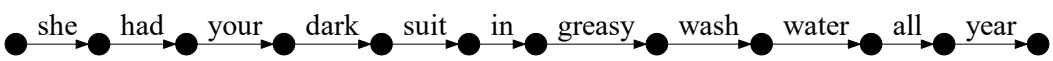


Fig. 2.1 Graphical Model of a One-Best ASR Result (Example)

N-Best List: ASR systems may provide the user with multiple possible hypotheses (usually ranked). These *N-best* outputs can also be graphically represented as shown in Figure 2.2:

[6] https://pypi.org/project/num2words/
[7] https://pypi.org/project/words2num/

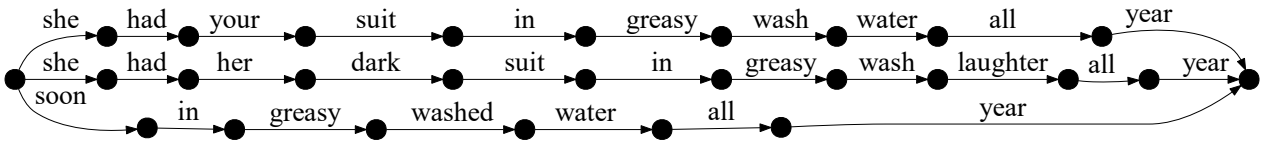


Fig. 2.2 Graphical Model of a N-Best ASR Result (Example)

Confusion Network: An alternate ASR output format is the Confusion Network (CN), informally called a "sausage." For every word, confusion networks allow for multiple alternative words. More formally, confusion networks are simple linear DAGs with the property that each path from the start node to the stop node goes through all the other nodes. The set of words represented by edges between two nodes is called a confusion set [54], or correspondence set [23], or cohort set [3]. Confusion networks are often generated by transforming a word lattice [54] (discussed next). They can also be generated by applying multiple sequence alignment (MSA) on multiple different hypotheses. This is the approach taken by the ROVER algorithm [23] where the resulting graph is called a Word Transition Network (WTN)[8]. They could also be generated using timing information rather than multiple sequence alignment based on a scoring function (See Appendix A.2 for more on MSA). Due to their simple structure, confusion networks can efficiently model and store *many* competing hypotheses. We only need to store a sequence of M confusion sets $\mathcal{Q}$. Figure 2.3 shows an example of a confusion network. The "_" symbol is a special character indicating a null or skip[9]. Null arcs allow for hypotheses with "lengths" less than M to be generated from a confusion network with M confusion sets.

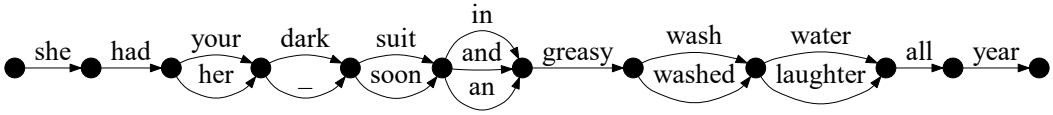


Fig. 2.3 A Confusion Network (Example)

As mentioned, we can store confusion networks as a sequence of M confusion sets.

$$\mathcal{Q} = \mathcal{C}_1, \mathcal{C}_2, ..., \mathcal{C}_M \tag{2.14}$$

This is illustrated below for the example in Figure 2.3:

$$\begin{bmatrix}\text{she}\end{bmatrix}\begin{bmatrix}\text{had}\end{bmatrix}\begin{bmatrix}\text{your}\\ \text{her}\end{bmatrix}\begin{bmatrix}\text{dark}\\ \text{\_}\end{bmatrix}\begin{bmatrix}\text{suit}\\ \text{soon}\end{bmatrix}\begin{bmatrix}\text{in}\\ \text{and}\\ \text{an}\end{bmatrix}\begin{bmatrix}\text{greasy}\end{bmatrix}\begin{bmatrix}\text{wash}\\ \text{washed}\end{bmatrix}\begin{bmatrix}\text{water}\\ \text{laughter}\end{bmatrix}\begin{bmatrix}\text{all}\end{bmatrix}\begin{bmatrix}\text{year}\end{bmatrix}$$

[8]The concept of Word Transition Network seems to be more general than confusion networks.
[9]Some literature denote null arcs with the labels "!NULL" or "<skip>" or "ϵ"

Word Lattice: Word lattices are another common format in ASR technology. They are more complex in terms of their structure than confusion networks and only allow for strict patterns to be generated. They also encode timing information more accurately than confusion networks. This, however, makes them inefficient to store in memory and more challenging to manipulate. Word lattices are generally internally held in an ASR and not always available in a black box setting. Our work does not deal with word lattices.

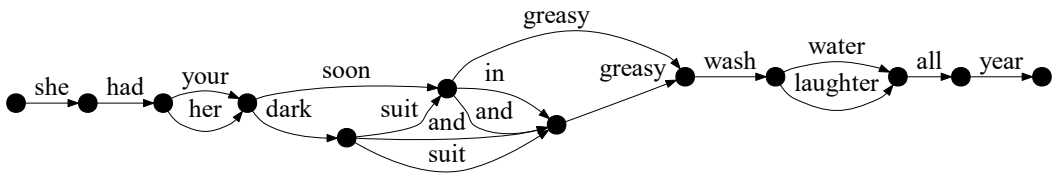


Fig. 2.4 A Word Lattice (Example)

2.6 Noise-Robustness

Speech may be corrupted by a wide variety of *noises*, such as background sounds by people speaking nearby, traffic, devices, and environmental variations. It is not possible to enumerate all possible noises or their complex and dynamic impact on speech and speech recognition. A *model* of the environment by Hansen [35] in Figure 2.5 captures a number of *sources* of noise and speech characteristics, including the speech produced by stress, task workload, speaker emotions, ambient noise, the Lombard effect (the tendency of people to increase their vocal effort in the presence of noise), microphone distortion, and transmission distortion. A major source of noise in this model is the ambient noise z_{env}. It has additive components but also affects the speaker.

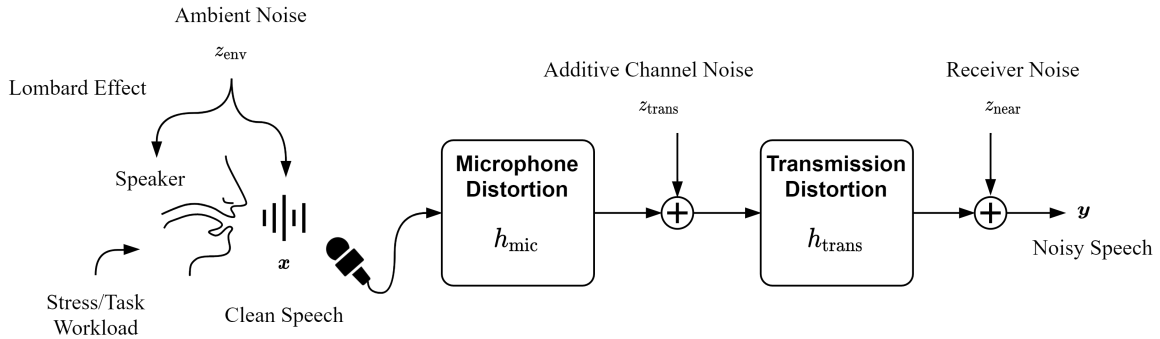


Fig. 2.5 A Model of the Sources of Noise

Now that we have considered some sources of noise, we consider the problem of automatic speech recognition in speech affected by such noises. It is a well-known problem that the

performance of automatic speech recognition deteriorates in the presence of noise. More precisely, the WER achieved for noisy speech is often much greater than the WER achieved for clean speech of the same type, as will be shown in the experiments of § 2.7. Noise-robustness in ASR (informally) refers to reducing the WER for noisy speech such that it is comparable to the WER achieved on clean speech. Any reduction in WER achieved in noisy categories, however, can be seen as a step towards noise-robustness. Noise-robustness has been a major area of research in the field of automatic speech recognition. A *very* large number of methods have been proposed, published, and implemented over the decades. Li et. al [51] provides a comprehensive overview and novel categorization of these methods. Although they use five different criteria for categorization, we note that there are two major approaches to noise-robustness. The first approach, termed *feature-based methods*, works on the *input* noisy speech. Some methods such as Perceptually based Linear Prediction (PLP) [36] are motivated by human hearing and work by extracting robust features from audio. Other techniques such as Wiener filtering [52] and spectral subtraction [9, 8] aim to reconstruct clean speech from noisy speech. The other approach, termed *model-based methods*, aims to train models that are inherently robust to noisy inputs. Recent successes in noise-robust ASR systems have largely been due to this approach. In particular, the replacement of older models with deep neural networks (DNNs) [14, 39] have shown great success. For example, a work by Seltzer and Wang [67] found that multi-condition training, noise-aware training, and dropout[10] training of DNNs significantly improved noise-robustness of ASRs. Our work is different from previous work in the sense that we are not *enhancing* the signal *or* the model. Instead, we are exploiting the observation that ASR systems are sensitive to perturbations in the input speech under noisy environments. This is done by redundantly processing diverse variants of speech. This aligns our work more closely with work in ensemble and multi-stream approaches to ASR [37]. Our framework, the intuition behind it, and empirical evidence supporting our claim are further explored in Chapter 3.

2.7 Experiments

In this section, we wish to quantify the degradation of ASR performance in terms of WER in both clean and noisy environments. We report our experimental results on the `Development` subset of the CHiME3 [7] dataset using leading Cloud ASR systems, including Google Cloud Speech to Text, IBM Watson Speech to Text, and Microsoft Azure Speech to Text. The CHiME3 dataset is designed around the Wall Street Journal (WSJ) [26] corpus and contains speech recorded in both noise-free and challenging noisy environments. The ASR systems and the CHiME3 dataset are used consistently throughout this thesis and are discussed in more

[10] Originally devised to prevent overfitting.

detail in Appendix B. We note that the selected ASR systems are highly configurable and allow for fine-tuning of different aspects of ASR such as acoustic and language models. We use default options based on official code samples. Our results, however, should not be taken as a general assessment of the performance of these systems. We also re-emphasize that our WER values are somewhat rough due to limitations in our text normalization (see § 2.4.3).

2.7.1 Empirical Assessment of ASR in Noisy Environments

In this section, we establish a baseline performance for the Google, IBM, and Microsoft ASR systems in terms of WER. Figure 2.6 plots the WER values for the different systems grouped by the noise categories. We can see that all systems do well on clean (Booth) data with WER values of less than 5%. The IBM system narrowly leads the other systems on clean data. This reverses on noisy data, where the IBM system has the worst performance in terms of WER across all categories. The Microsoft system has the best performance in nearly all categories, except for the Bus category where the Google system performs slightly better.

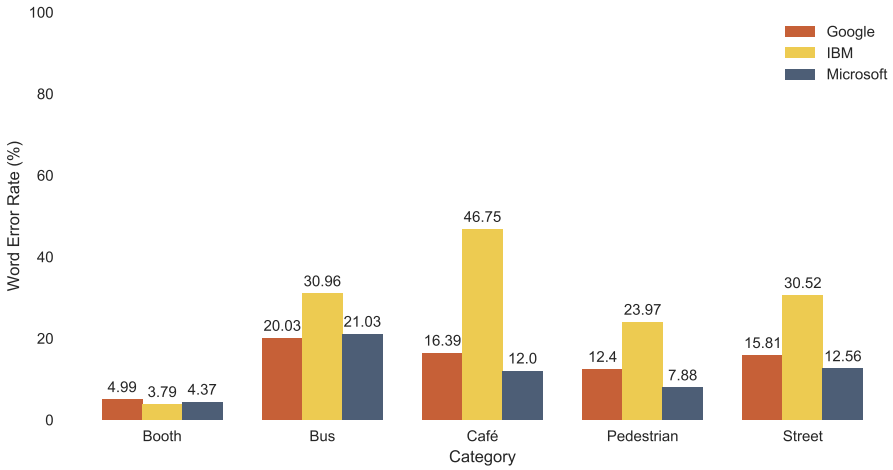


Fig. 2.6 Development Set WER (%) by the ASR Systems Grouped by Categories

These results show that there is a clear degradation in performance of ASRs when we move from clean speech to noisy speech. In the next chapters, we aim to reduce the WER values for the noisy categories such that they are closer to the WER for the Booth (Clean) category.

2.8 Summary and Conclusions

In this chapter, we explored the preliminaries of automatic speech recognition (ASR). We presented classic decision-theoretic formulation of the ASR task. Despite our black box treatment of ASR, we also presented a high-level architecture of a traditional ASR pipeline.

We introduced the word error rate (WER) evaluation metric and considerations for benchmark datasets. We discussed existing approaches to noise-robustness in ASR and briefly differentiated our approach. Finally, we empirically quantified performance degradation of ASR in terms of WER on leading cloud ASR systems using the `Development` subset of the CHiME3 dataset.

Chapter 3

Data Diversity for Automatic Speech Recognition

This chapter presents our approach to generating diverse transcriptions of speech by applying diverse transformations on audio containing speech and redundantly performing speech-to-text conversion. We first introduce data diversity as a fault-tolerance mechanism, as formulated by Ammann and Knight [1, 46]. We present some high-level intuition for applying data diversity in the context of automatic speech recognition and define the data-diverse redundant processing framework. We present metrics from literature to quantify diversity among hypotheses and introduce a visual representation of diversity. We introduce the notion of an oracle combination of hypotheses to quantify complementarity or error-correcting potential of the hypotheses. Finally, we present our experimental results on a number of label-preserving transformations with multiple commercial ASR systems on the `Development` set of the CHiME3 dataset.

3.1 Data Diversity as a Fault-Tolerance Mechanism

Data diversity, as formulated by Amman and Knight [1, 46], is a fault-tolerant strategy that takes advantage of the observation that software execution conditions are often sensitive to perturbations in the *input* data. It is possible for an application to produce very different results on very similar or even logically equivalent data. Data diversity takes advantage of such sensitivity by executing the same implementation of software on diverse data. Data diversity could be employed in a system similar to N-version programming [4] called N-copy programming [1]. As with N-version programming, it is hoped that the diversity in the execution conditions of software leads to diverse outputs that do not contain all of the the same errors i.e., are complementary.

The first step in N-copy programming is data *re-expression*, which refers to algorithmically modifying the input data[1]. This re-expression could be exact such as applying a random shuffle (i.e., permutation) of data in an array fed to a sorting algorithm. It could also be inexact or approximate such that it is still acceptable to the system. The re-expressed *variants* of an input are fed to identical copies of a system. In some cases, the output from the system may need to undergo *reverse re-expression*. The final stage in the system consists of aggregating the outputs from each system and finding consensus among them. Ammann [1] uses the term "voter" as a generic stand-in for this process. We note that data diversity may work for different reasons in data-driven systems such as automatic speech recognition than those of traditional logic-based software where errors are largely due to "bugs."

3.2 Data Diversity for Automatic Speech Recognition

We begin our investigation of data diversity for automatic speech recognition by noting that speech is a complex and dynamic phenomenon without clearly distinguishable parts. The temporal and spectral characteristics of speech are subject to a wide range of variations. These variations include dialects, accents, speaking styles, pronunciation, speech rates, and the emotional state of the speaker. Environmental noise, reverberation, and recording equipment result in additional variability, which make ASR a particularly challenging task. We observe that decision-theoretic approaches to ASR such as the maximum a posteriori decision rule in Equation 2.4 consider the *true* posterior distribution $P(\boldsymbol{w} \mid \boldsymbol{x})$. The factored form of the posterior still requires the *true* likelihood $P(\boldsymbol{x} \mid \boldsymbol{w})$ and *true* prior $P(\boldsymbol{w})$. In practice, however, we cannot compute these probabilities. In fact, it is usually difficult to characterize the true distribution of *any* data. In practice, proxy models whose parameters are learned from data are used. Mismatch between the model and input data is a major contributor to ASR errors under adverse environments. These could be related to Dietterich's *statistical* and *representational* reasons for why ensemble learning methods may work [17]. Dietterich argues that due to lack of training data and models being approximations to real processes that generate data, an ensemble of multiple systems may perform better than individual systems. Similarly, we argue that deliberate variations in the input could lead to some different parts of speech being labeled differently by an ASR system. We can also relate data diversity with Dietterich's *computational* reason for why ensemble methods works. Given that ASR is a search problem and the search space is too large, it is reasonable to assume that the search is done over a small subset of all possible hypotheses. Data diversity may introduce variations that would result in different subsets every time. We hypothesize that *ASR systems are more susceptible to such variations in the input when the input data is noisy*. Empirical evidence presented later in §

[1]Diverse data could also be captured naturally such as from different sensors

3.6.2 support this hypothesis. We wish to exploit this sensitivity by perturbing the input to generate diverse hypotheses from a single ASR system. It is then hoped that these diverse hypotheses are complementary i.e., they do not all contain the same mistakes.

3.3 Data-Diverse Redundant Processing Framework

To apply data diversity to an ASR system, we need to find re-expression algorithms that generate *variants* of the original input speech. In this work, we consider transformations that are, at least in principle, *label-preserving*. We use this term to emphasize that we are not concerned with how much a transformed signal differs from the original based on some distance metric for as long as the label is preserved. This is motivated by the observation that diversity in the input may not have a straight-forward relationship with diversity in the output. For example, small variations in the input could possibly generate as much (or perhaps more) variations in the output as large variations in the input. We define the set of variants $\mathcal{X}$ as the set of acoustic signals after N transformations have been applied to an acoustic signal $\boldsymbol{x}$:

$$\mathcal{X} = \boldsymbol{x}^{\mathcal{T}_1}, \boldsymbol{x}^{\mathcal{T}_2}, \ldots, \boldsymbol{x}^{\mathcal{T}_N} \quad \text{, where } \boldsymbol{x}^{\mathcal{T}_i} = \mathcal{T}_i(\boldsymbol{x}) \tag{3.1}$$

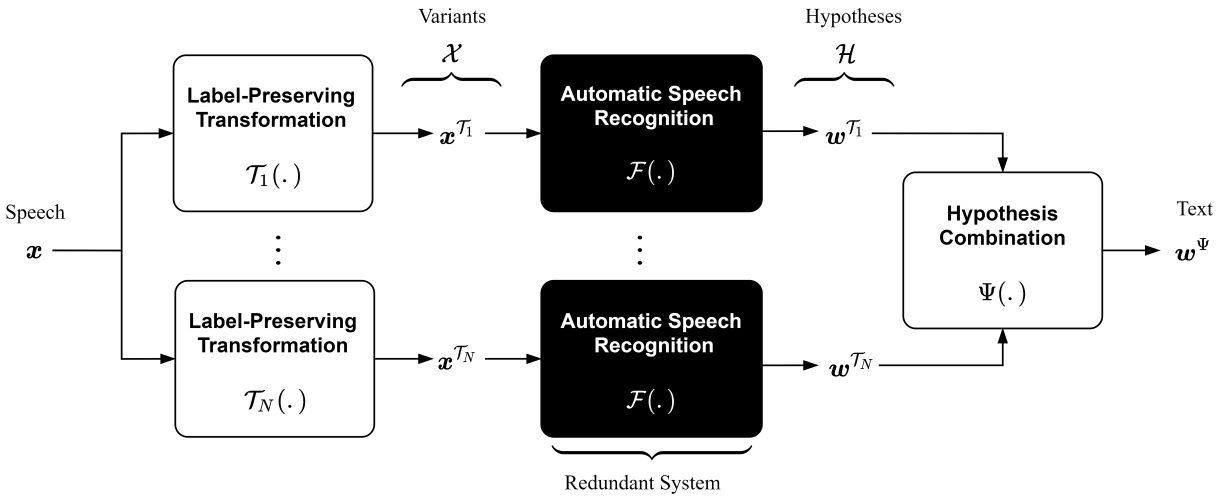
Feeding $\boldsymbol{x}^{\mathcal{T}_i}$ to an ASR, we get a hypothesis $\mathcal{F}(\boldsymbol{x}^{\mathcal{T}_i}) = \boldsymbol{w}^{\mathcal{T}_i}$. Grouping the hypotheses in a multiset of hypotheses $\mathcal{H}$:

$$\mathcal{H} = \boldsymbol{w}^{\mathcal{T}_1}, \boldsymbol{w}^{\mathcal{T}_2}, \ldots, \boldsymbol{w}^{\mathcal{T}_N} \tag{3.2}$$

The hypotheses could then be fed to a hypothesis combination module Ψ that combines them to generate a single hypothesis:

$$\boldsymbol{w}^{\Psi} = \Psi(\mathcal{H}) \tag{3.3}$$

A detailed view of our framework, termed *data-diverse redundant processing*, is shown in Figure 1.2 (repeated below for convenience). This chapter is concerned with generating the hypotheses $\mathcal{H}$ and quantifying *diversity* and *complementarity* between them. Chapter 4 explores combination schemes for the hypothesis combination module Ψ.



3.4 Quantifying Diversity among Hypotheses

Given a multiset of hypotheses corresponding to the same input speech, we wish to quantify the measure of diversity between them. Audhkhasi et. al [2] propose averaging of pairwise WER values as an intuitive measure of diversity. Formally, given that $\mathcal{H} = \boldsymbol{w}_1, \boldsymbol{w}_2, ..., \boldsymbol{w}_N$ is a list of N hypothesis sentences corresponding to speech signal $\boldsymbol{x}$, the diversity $\mathcal{D}$ among them can be defined as:

$$\mathcal{D}(\mathcal{H}) = \frac{2}{N(N-1)} \sum_{i=1}^{N} \sum_{j>i} \text{WER}(\boldsymbol{w}_i; \boldsymbol{w}_j) \tag{3.4}$$

Wong. et. al [76, 77] propose a very similar metric known as Cross-WER. Cross-WER also averages pairwise WER values, but it takes into account the fact that WER is not a symmetric function. It computes both $\text{WER}(\boldsymbol{w}_i; \boldsymbol{w}_j)$ and $\text{WER}(\boldsymbol{w}_j; \boldsymbol{w}_i)$.

$$\text{Cross-WER}(\mathcal{H}) = \frac{1}{N(N-1)} \sum_{i=1}^{N} \sum_{j\neq i} \text{WER}(\boldsymbol{w}_i; \boldsymbol{w}_j) \tag{3.5}$$

We observe that $\mathcal{D}(\mathcal{H})$ in Equation 3.4 is sensitive to the ordering of transcripts within the multiset of hypotheses $\mathcal{H}$[2]. For this reason, we make use of the Cross-WER metric rather than $\mathcal{D}$ to quantify diversity in our work. To get a sense of diversity at a more granular level, we propose constructing an $N \times N$ matrix of the pairwise normalized edit distances (i.e., WERs). This matrix, henceforth referred to as the *diversity matrix*, is computed as follows:

[2]The authors do not provide a formal definition of WER. It is likely that they use a different definition than the standard normalized edit distance definition. Later work by the same authors [3, 47] suggest that they define it as the Levenshtein distance between two word sequences, but this no longer a *rate*.

$$\begin{bmatrix} 0 & \text{WER}(\boldsymbol{w}_1;\boldsymbol{w}_2) & \dots & \text{WER}(\boldsymbol{w}_1;\boldsymbol{w}_N) \\ \text{WER}(\boldsymbol{w}_2;\boldsymbol{w}_1) & 0 & \dots & \text{WER}(\boldsymbol{w}_2;\boldsymbol{w}_N) \\ \vdots & \vdots & \ddots & \vdots \\ \text{WER}(\boldsymbol{w}_N;\boldsymbol{w}_1) & \text{WER}(\boldsymbol{w}_N;\boldsymbol{w}_2) & \dots & 0 \end{bmatrix} \tag{3.6}$$

We can gain insights by visualizing the diversity matrix as a heat map. We note that the diversity metric $\mathcal{D}$ in Equation 3.4 uses the elements above the major diagonal[3]. The Cross-WER metric in Equation 3.5 uses the numbers above and below the major diagonal. We note that Equations 3.4 and 3.5 are defined for a single data point. They could be extended to multiple data points by micro-averaging in the same fashion as computing WER for multiple data points (Equation 2.11).

3.4.1 Example

Consider the list $\mathcal{H}$ below consisting of multiple hypotheses (padded for visual enhancement) for the a speech signal $\boldsymbol{x}$, with the reference transcript: "the cat in the hat sat on the mat."

the	cat	in	the	hat	sat	on	the	mat
the	bat	in	the	hat	sat	on	the	cat
the	cat	_	_	_	_	_	_	_
the	cat	and	the	bat	sat	on	the	mat
the	bat	and	the	cat	sat	on	the	mat
that	cat	on	the	mat	sat	by	the	mat
the	cat	and	the	bat	sat	on	the	mat
sat	on	the	mat	_	_	_	_	_

Figure 3.1 shows an example of two diversity matrices corresponding to the same set of hypotheses above but with different ordering of transcripts within $\mathcal{H}$. We see that the value of $\mathcal{D}(\mathcal{H})$ is not consistent between them (i.e., 1.17 versus 0.85). This is because $\mathcal{D}(\mathcal{H})$ only sums the cells on top of the major diagonal in the matrix. A change in the ordering of $\mathcal{H}$ (i.e., a permutation) causes the cells to move around within the matrix. We see that Cross-WER is conistent between the permutations (i.e., 0.90 for both).

[3]Also called the principal diagonal, primary diagonal, leading diagonal, or major diagonal in linear algebra.

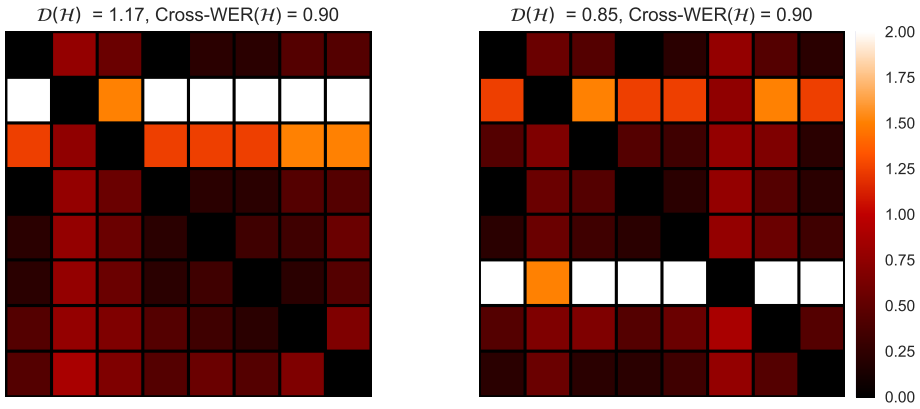


Fig. 3.1 Diversity Matrices of a Randomly Shuffled Set of Hypotheses $\mathcal{H}$

Treating $\mathcal{D}$ as a random variable, with the underlying randomness coming from random permutations of $\mathcal{H}$, a Monte Carlo simulation suggests that expected value of $\mathcal{D}$ is the Cross-WER[4]. Figure 3.2 shows the first 100 iterations of the Monte Carlo simulation for the example above. We can see that the cumulative average of the $\mathcal{D}(\mathcal{H})$ approaches the Cross-WER value, which is constant over the permutations.

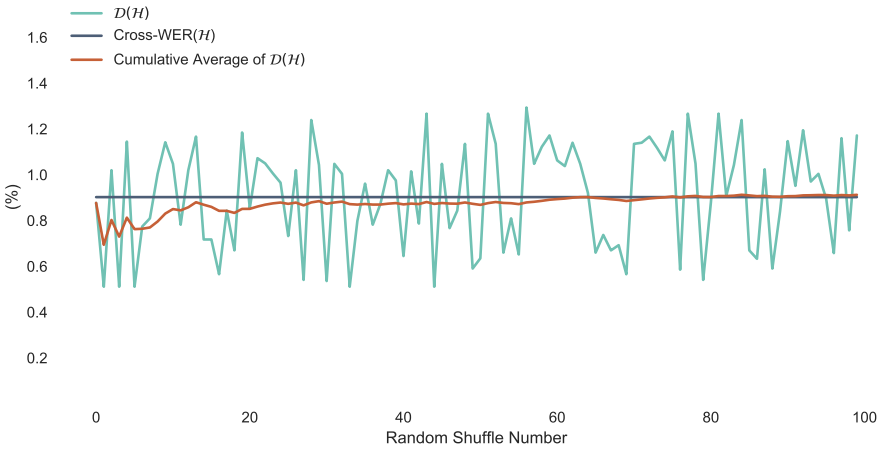


Fig. 3.2 Monte Carlo Simulation of $\mathcal{D}$ and Cross-WER for Permutations of the List of Hypotheses $\mathcal{H}$

[4]A general proof, however, is required to show this.

3.5 Quantifying Complementarity among Hypotheses

For diversity to be useful, the hypotheses need to be complementary (i.e., have mistakes in different locations). Diversity does not necessarily indicate that the hypotheses in a multiset $\mathcal{H}$ are complementary (e.g., hypotheses can be wrong in different ways). To show complementarity, we conceive of finding the "best possible combination" of a given number of hypotheses using an *oracle machine* with knowledge of the reference transcript. This *oracle combination*, denoted as $\Phi(\mathcal{H})$, could put together the hypotheses in a way that reduces WER. In practice, we cannot always compute such a combination since we only have access to the reference transcripts during development and evaluation. Showing that complementary transcripts are generated consistently for a large dataset during development could, however, provide justification for the utility of data diversity. We denote the oracle combination of a multiset of N hypotheses $\mathcal{H} = \boldsymbol{w}_1, \boldsymbol{w}_2, ..., \boldsymbol{w}_N$ as:

$$\boldsymbol{w}^{\Phi} = \Phi(\mathcal{H}) = \Phi(\boldsymbol{w}_1, \boldsymbol{w}_2, ..., \boldsymbol{w}_N) \tag{3.7}$$

This is shown in Figure 3.3, which shows multiple hypotheses as the input and a single hypothesis as the output. The dotted line indicates implicit knowledge of the reference transcript.

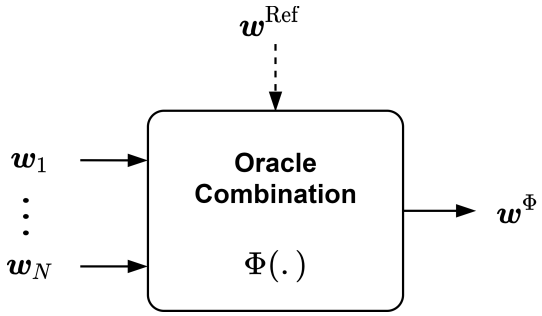


Fig. 3.3 Oracle Combination of Multiple Hypotheses

Once an oracle combination $\boldsymbol{w}^{\Phi}$ is found, the term in Equation 3.8 represents a kind of *lower-bound* on WER that could *potentially* be achieved by combining the hypotheses in $\mathcal{H}$. We can claim that the multiple hypotheses are complementary (in an ideal sense) if the oracle combination of them produces a lower WER than WERs achieved by the individual hypotheses.

$$\text{WER}(\boldsymbol{w}^{\text{Ref}}; \boldsymbol{w}^{\Phi}) \tag{3.8}$$

3.5.1 An Algorithmic Definition of Oracle Combination

We note that the notion of a "best possible combination" of multiple hypotheses is ambiguous. Algorithm 1 describes a simple scheme for this combination. This definition allows for removal

or filtering of all incorrect words[5] and preserves the original ordering of words within each hypothesis. We start an iterative process by initializing the sequence $\boldsymbol{w}^{\Phi}$ to be of the the same size as $\boldsymbol{w}^{\text{Ref}}$. The elements are initialized as null elements. We iterate over the hypotheses with i as the index of iteration. The Filter function removes all incorrect words (i.e., words that are not in the reference transcript) from the ith hypothesis $\boldsymbol{w}_i$ to yield a filtered version of the hypothesis $\boldsymbol{w}^{\text{Filtered}}$ (which changes every iteration). The filtered hypothesis $\boldsymbol{w}^{\text{Filtered}}$ and the reference transcript $\boldsymbol{w}^{\text{Ref}}$ are aligned using the Levenshtein criterion by the Wagner-Fischer algorithm[6] to yield aligned (and hence equally sized) sequences $\boldsymbol{w}^{\text{Filtered}*}$ and $\boldsymbol{w}^{\text{Ref}*}$. All matched words in the alignment are placed within $\boldsymbol{w}^{\Phi}$ in the same location as their location in the unaligned reference transcript $\boldsymbol{w}^{\text{Ref}}$[7]. If the combination sequence achieves a WER of zero, we stop. Otherwise, we continue iterating until we are out of hypotheses. Finally, if there are null elements in the combined sequence $\boldsymbol{w}^{\Phi}$, we remove those to yield the final sequence. We note that our notion of an oracle combination is is similar to Breslin's [11] notion of "IDEAL combination" in the context of confusion network combination (CNC).

Algorithm 1 Oracle Combination $\Phi(\mathcal{H})$

Let $\boldsymbol{w}^{\Phi}$ be a sequence of size $|\boldsymbol{w}^{\text{Ref}}|$ with all NULL elements
for $i = 1, ..., |\mathcal{H}|$ **do**
    $\boldsymbol{w}^{\text{Filtered}} \leftarrow \text{Filter}(\boldsymbol{w}_i; \{w \mid w \in \boldsymbol{w}_i \wedge w \notin \boldsymbol{w}^{\text{Ref}}\})$
    $\boldsymbol{w}^{\text{Ref}*}, \boldsymbol{w}^{\text{Filtered}*} \leftarrow \text{Levenshtein-Align}(\boldsymbol{w}^{\text{Ref}}, \boldsymbol{w}^{\text{Filtered}})$
    location $\leftarrow 1$
    for $k = 1, ..., |\boldsymbol{w}^{\text{Ref}*}|$ **do**
        if $\boldsymbol{w}^{\text{Ref}*}[k] == \boldsymbol{w}^{\text{Filtered}*}[k]$ **then**
            $\boldsymbol{w}^{\Phi}[\text{location}] \leftarrow \boldsymbol{w}^{\text{Filtered}*}[k]$
        end if
        if $\boldsymbol{w}^{\text{Ref}*}[k] != \text{NULL}$ **then**
            location $\leftarrow$ location + 1
        end if
    end for
    if $\text{WER}(\boldsymbol{w}^{\text{Ref}}, \boldsymbol{w}^{\Phi}) == 0$ **then**
        Break
    end if
end for
$\boldsymbol{w}^{\Phi} \leftarrow \text{Filter}(\boldsymbol{w}^{\Phi}; \{\text{NULL}\})$

[5] Words that are in the hypothesis but not the reference. Not to be confused by insertions, deletions, and substitutions.

[6] The Needleman-Wunsch algorithm [58] may provide better results without the need for filtering of incorrect words since it maximizes the match between two sequences. We used the Wagner-Fischer algorithm [75] because it was readily available from our implementation of the WER computation.

[7] Removing incorrect words from $\boldsymbol{w}_i$ generally leads to $\boldsymbol{w}^{\text{Ref}}$ and $\boldsymbol{w}^{\text{Ref}*}$ being the same since most insertions will be removed. However, this is not always the case. For example, incorrect repetition of a correct word in the hypothesis can make $\boldsymbol{w}^{\text{Ref}*}$ longer than $\boldsymbol{w}^{\text{Ref}}$.

3.5.2 Oracle Combination Improvement

We note that the oracle combination WER in Equation 3.8 is not a relative value. To show *improvement* of an oracle combination of a list of N hypotheses $\mathcal{H} = \boldsymbol{w}_1, \boldsymbol{w}_2, ..., \boldsymbol{w}_N$ over some baseline transcript $\boldsymbol{w}^{\text{Base}}$, we can define the *oracle combination improvement* $\Delta^{\Phi}_{\boldsymbol{w}^{\text{Base}}}(\mathcal{H})$ as the difference between the WER of the baseline transcript and the WER achieved by the oracle:

$$\Delta^{\Phi}_{\boldsymbol{w}^{\text{Base}}}(\mathcal{H}) = \text{WER}(\boldsymbol{w}^{\text{Ref}}; \boldsymbol{w}^{\text{Base}}) - \text{WER}(\boldsymbol{w}^{\text{Ref}}; \Phi(\mathcal{H})) \tag{3.9}$$

It follows that the oracle improvement is non-negative if the baseline transcript is included in the set of hypotheses since the combination transcript will be at least as good as the best individual hypothesis:

$$\Delta^{\Phi}_{\boldsymbol{w}^{\text{Base}}}(\mathcal{H}) \geq 0 \quad \text{, if } \boldsymbol{w}^{\text{Base}} \in \mathcal{H} \tag{3.10}$$

The function $\Delta^{\Phi}_{\boldsymbol{w}^{\text{Base}}}$ is valid for any non-empty subset of $\mathcal{H}$. This could be useful in seeing how different transcriptions complement each other. We can arrange *pairwise* improvements in a matrix, henceforth referred to as the *oracle combination improvement matrix*. As with the diversity matrix, this matrix may be visualized as heat map. In the context of data-diverse redundant processing framework, this matrix tells us how transcripts from different transformations complement other transcripts.

$$\begin{bmatrix} \Delta^{\Phi}_{\boldsymbol{w}_1}(\boldsymbol{w}_1, \boldsymbol{w}_1) & \Delta^{\Phi}_{\boldsymbol{w}_1}(\boldsymbol{w}_1, \boldsymbol{w}_2) & \dots & \Delta^{\Phi}_{\boldsymbol{w}_1}(\boldsymbol{w}_1, \boldsymbol{w}_N) \\ \Delta^{\Phi}_{\boldsymbol{w}_2}(\boldsymbol{w}_2, \boldsymbol{w}_1) & \Delta^{\Phi}_{\boldsymbol{w}_2}(\boldsymbol{w}_2, \boldsymbol{w}_2) & \dots & \Delta^{\Phi}_{\boldsymbol{w}_2}(\boldsymbol{w}_2, \boldsymbol{w}_N) \\ \vdots & \vdots & \ddots & \vdots \\ \Delta^{\Phi}_{\boldsymbol{w}_N}(\boldsymbol{w}_N, \boldsymbol{w}_1) & \Delta^{\Phi}_{\boldsymbol{w}_N}(\boldsymbol{w}_N, \boldsymbol{w}_2) & \dots & \Delta^{\Phi}_{\boldsymbol{w}_N}(\boldsymbol{w}_N, \boldsymbol{w}_N) \end{bmatrix} \tag{3.11}$$

Finally, we note that improvement metric in Equation 3.9 is defined for transcriptions of a single data point $\boldsymbol{x}$. It could be extended to multiple data points by taking the difference of WER values for multiple baseline transcripts and multiple oracle combination transcripts.

3.5.3 Seemingly Counter-intuitive Results

Intuitively, we do not expect a transcript to complement itself (or two identical hypotheses to be complementary), but it turns out that a lower WER can sometimes be achieved if we follow Algorithm 1. This is partially because the algorithm removes (i.e., filters) all incorrect words from the hypothesis before constructing the combination sequence $\boldsymbol{w}^{\Phi}$. The improvement may

come from the removal of any insertions in the hypothesis. A more interesting case of such *self-improvement* is when even the removed insertions do not account for the full improvement. We attribute this to the counter-intuitive results obtained by the Levenshtein distance (see Appendix A.1 for details) criterion for aligning sequences. The Levenshtein distance is the minimum number of insertion, deletion, and substitution operations required to transform one sequence into another sequence. The Wagner-Fischer algorithm [75] computes this by finding an alignment that satisfies this requirement. Examining the alignments reveals that they can be counter-intuitive at times. Consider the example below, where a Levenshtein alignment is found between a reference and a hypothesis transcript:

Reference	the	black	cat	in	the	hat
Hypothesis	cat	that	was	in	the	hat
Operation	SUB	SUB	SUB	OK	OK	OK

It may seem peculiar that the words "cat" in the reference and hypothesis sequences were not aligned. This Levenshtein alignment considers the word "cat" to have been "substituted" by the word "was." However, this is not the intuitive understanding of a substitution. A more "sensible" alignment may be:

Reference	the	black	cat	_	_	in	the	hat
Hypothesis	_	_	cat	that	was	in	the	hat
Operation	DEL	DEL	OK	INS	INS	OK	OK	OK

However, we can see that only the first alignment minimizes the number of operations (i.e., three substitutions) to transform one sequence into the other. The second alignment requires four operations (i.e., two deletions and two insertions). This illustrates that minimizing the edit distance does not necessarily generate "sensible" alignments[8]. The algorithm may consider correctly recognized words as "errors" in order to achieve the smallest possible distance. If the list of hypotheses consists of this single hypothesis i.e., $\mathcal{H} = \boldsymbol{w}$ in the above example, the oracle combination algorithm $\Phi(\boldsymbol{w})$ returns the following:

Reference	the	black	cat	in	the	hat
Oracle Combination	_	_	cat	in	the	hat
Operation	DEL	DEL	OK	OK	OK	OK

Computing the WER for the "combination," we get a value of $2/6 = 0.3\bar{3}$, which is better than the baseline WER of $3/6 = 0.5$. This improvement will have seemed odd without examining the alignment. However, it could be argued that this expected behaviour from an

[8]This also illustrates the difference between the minimum distance criterion (computed with the Wagner-Fischer algorithm [75]) and the maximum match criterion (computed using the Needleman-Wunsch algorithm [58]) for aligning sequences. These are sometime confused. Our filtering of incorrect words, however, means that the combination sequence $\boldsymbol{w}^{\Phi}$ will be the same in most cases regardless of which algorithm is used.

oracle machine. We do claim that such *self-improvements* are small in practice in the conext of our work, as will be shown empirically in the experiments section of this chapter.

3.6 Experiments

In this section, we present experimental results on the `Development` set of CHiME3 dataset using the Google Cloud Speech to text, IBM Watson Speech to Text, and Microsoft Azure Speech to Text systems. The configuration and setup is the same as the previous chapter (further details can be found in Appendix B). In § 3.6.1, we discuss the transformations considered followed by WER results for each. In § 3.6.2, we present our results quantifying diversity among transcripts generated by these transformations. In § 3.6.3, we presents results concerning the complementarity of the transcripts generated by the transformations.

3.6.1 Label-Preserving Transformations

Given that the goals of this work is to show the feasibility of generating diverse outputs using label-preserving transformations rather than finding the best transformations, our choice of transformations was somewhat arbitrary. Future work, however, can find transformations automatically, perhaps using statistical optimization methods. For now, we consider six general classes of transformations that modify the speech signal in both time and amplitude axes. Table 3.1 lists 20 transformations considered, their labels, and their parameters.

The **Identity** transformation refers to using the original unaltered audio. This was done because many ASR systems (e.g., Google[9]) do not recommend modifying the audio signals. Including them also ensures that the oracle combination improvement $\Delta^{\Phi}_{\boldsymbol{w}^{\text{Base}}}$ will be non-negative, since the transcriptions of unaltered speech are the baselines against which we compare the combination of hypotheses.

The **Amplitude Normalization** transformation refers to scaling the speech signal such that it fits within the +1/-1 rails. We emphasize that this is uniformly applied to the entire signal and should not be confused with automatic gain control.

The **Additive White Gaussian Noise** transformation refers to injecting randomness at high signal-to-noise ratio (SNR). This was achieved using the `awgn` function of MATLAB with `signalpower` specified as `measured` to determine the appropriate noise level based on a specified SNR.

The **Speed Modulation (Time Warping)** class of transformations refers to changing the speed of signal by stretching or compressing the audio along the time axis. This was achieved by the `resample` function in MATLAB, which also "applies an FIR Antialiasing Lowpass

[9] https://cloud.google.com/speech-to-text/docs/best-practices

Class	Label	Parameters / Description
Identity	None	Baseline
Amplitude Normalization	Normalized	+/- 1
Additive White Gaussian Noise	Gaussian	SNR = 30 dB
Speed Modulation (Time Warping)	Slow	Speed Multiplier = 10/11
	Slower	Speed Multiplier = 10/12
	Slowest	Speed Multiplier = 10/13
	Fast	Speed Multiplier = 11/10
	Faster	Speed Multiplier = 12/10
	Fastest	Speed Multiplier = 13/10
Speed Modulation (Phase Vocoder)	Slow (Vocoder)	Speed Multiplier = 10/11
	Slower (Vocoder)	Speed Multiplier = 10/12
	Slowest (Vocoder)	Speed Multiplier = 10/13
	Fast (Vocoder)	Speed Multiplier = 11/10
	Faster (Vocoder)	Speed Multiplier = 12/10
	Fastest (Vocoder)	Speed Multiplier = 13/10
Speech Enhancement /Denoising Algotihms	Berouti-79	Spectral Subtraction [8]
	Boll-79	Spectral Subtraction [9]
	Ephraim-Malah-84	MMSE [19]
	Ephraim-Malah-85	Log MMSE [20]
	Gustafsson-01	Spectral Subtraction [33]

Table 3.1 Details on Instances of the Label-Preserving Transformation Classes

Filter... and compensates for the delay introduced by the filter"[10]. Particular levels of this speed modulation transformations were determined by speed multipliers, which we selected to deviate slightly from the original speed.

The **Speed Modulation (Phase Vocoder)** class refers to changing the speed of the audio while maintaining short-term spectral characteristic of the signal. This was done with a phase vocoder [24]. We used an open-source implementation[11]. Particular levels of modulation for this transformation were also determined by speed multipliers, which were selected to be the same as those of the time warping class of transformations.

The **Speech Enhancement/Denoising Algorithms** class refers to transformations that actively seek to enhance the speech signal. We considered some early techniques such as spectral subtraction [8, 9, 33] and minimum mean square error (MMSE) estimators [19, 20]. We used MATLAB implementations of these by Loizou [53]. Since these algorithms generally resulted in volume reduction, we normalized the outputs to within +/- 1. We did not expect these transformations to do very well on their own since they tended to generate anomalous speech signals.

[10] https://www.mathworks.com/help/signal/ref/resample.html
[11] http://www.ee.columbia.edu/~dpwe/resources/matlab/pvoc/

Figure 3.4 lists the WER for all transformations on full `Development` set of the CHiME3 dataset (across all five categories). We see that transformations with minimal changes to the signals such as the identity transformation, normalizing, and adding Gaussian noise at very high SNR do very well across the ASRs. This is consistent with guidelines for these systems recommending not to to modify the audio. As for both types of speed multipliers, slowing the audio down seems to do better than speeding the audio up. Speech enhancement algorithms generally do not do well, but they are inconsistent across systems. For example, the transformation labeled "Ephraim-Malah-85" does fairly well with the Microsoft system but does not perform as well with the Google and IBM systems.

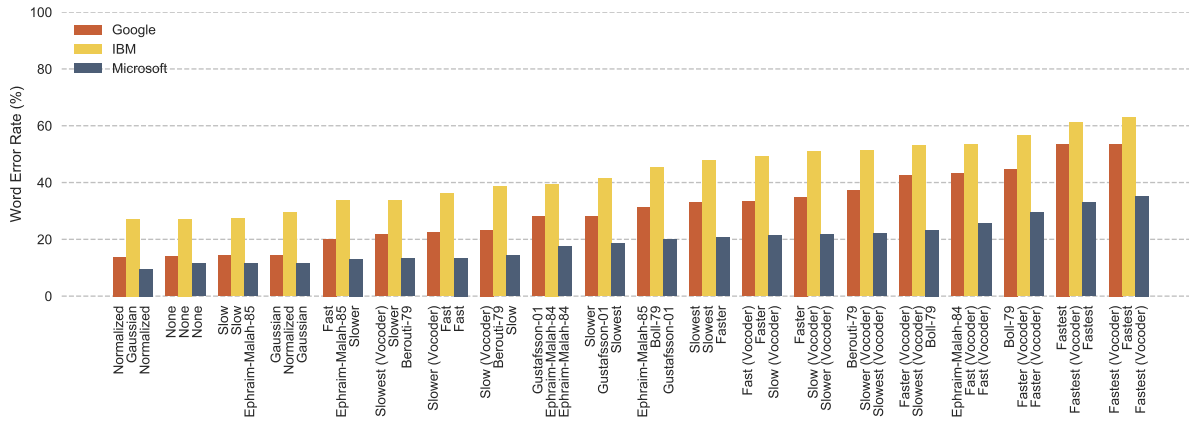


Fig. 3.4 Sorted WER Results for Different Transformations over all Noise Profiles

Figure 3.5 plots sorted WER values broken down by the noise categories. We can see that most transformations do consistently well for clean (Booth) audio. However, the performance is not consistent across transformations for noisy audio. Some transformation do much more poorly than others. We also observe that transformations do not necessarily perform consistently across noise profiles and different ASRs.

Fig. 3.5 Sorted WER Results for Different Transformations

3.6.2 Diversity among Hypothesis

Figure 3.6 plots the diversity matrices for the clean and noisy categories. The matrices are computed based on Equation 3.6, but for multiple utterances[12]. The matrices are interpreted as heat maps with each cell in a matrix representing the WER (Equation 2.11) between the row transformation transcripts (considered the references) and the column transformation transcripts (considered the hypotheses). Brightness is on the scale of [0, 2]. The major diagonal consists of all zeros since all transcripts fully agree with themselves. We also note that the matrices are somewhat symmetric along the major diagonal. This is because the cells at (i, j) and (j, i) within a matrix only differ by a normalization factor. The diversity matrices for clean data are fairly dark (smaller Cross-WERs). This indicates that we were not able to generate much diversity with clean speech. The plots for noisy data are much brighter for all ASRs, indicating that more diversity was generated in these categories. We observe that the Google and IBM systems are more susceptible to perturbations in the input than the Microsoft system.

[12]The computation consists of micro-averaging, which is equivalent to treating the multiple utterances as one long utterance.

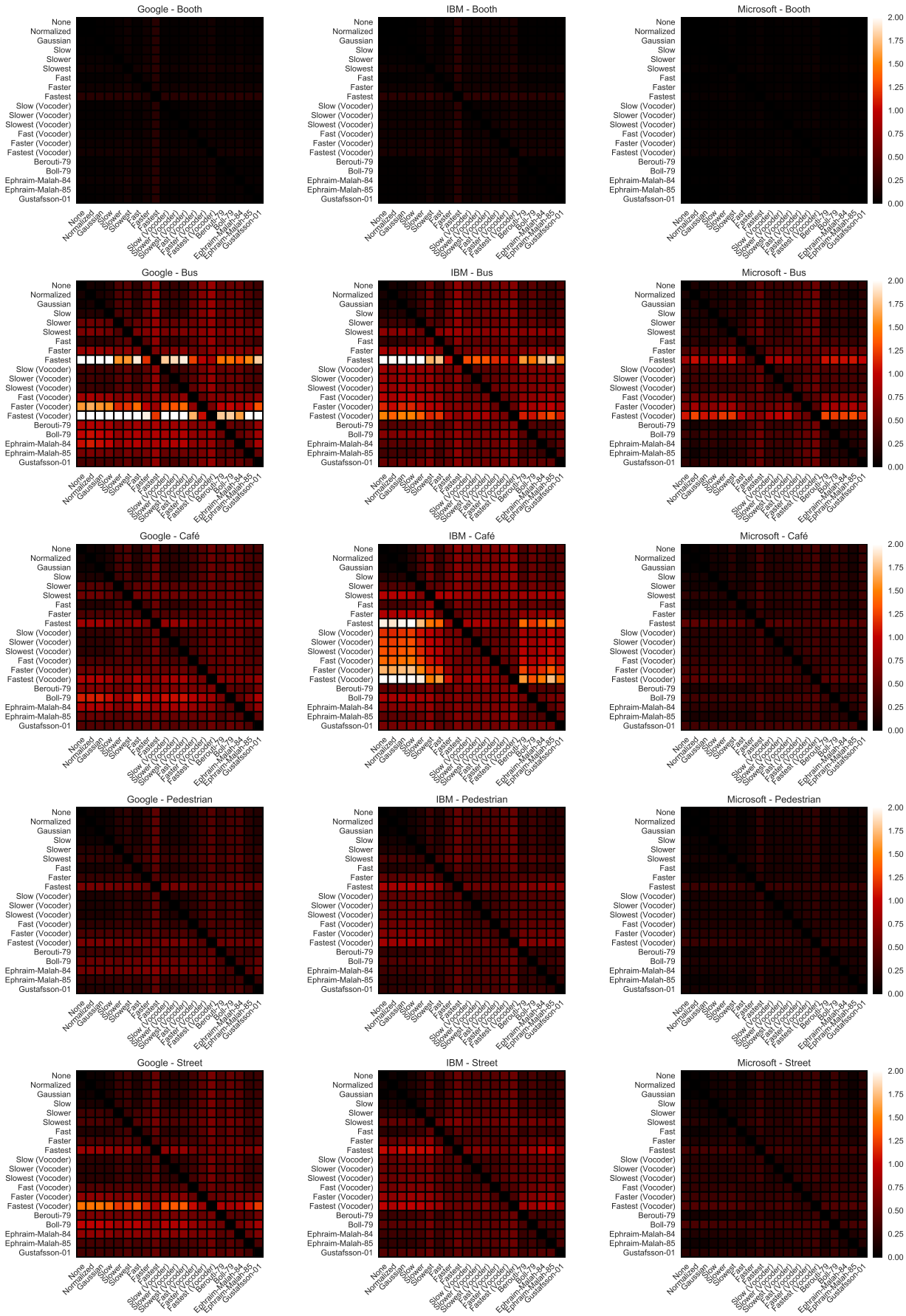


Fig. 3.6 Diversity Matrix Heat Maps for the `Development` set of the CHiME3 dataset for all ASRs and Noise Categories

Figure 3.7 aggregates and reports the Cross-WER values (%) for all five categories. Cross-WER can be thought of as the average brightness of the heat maps (excluding the diagonal) in the Figure 3.6. Interpreting Cross-WER as a measure of diversity, we can again see that we are able to induce diversity in the output of the ASR by applying diversity in the input. We observe that more diversity can be achieved in noisy categories than clean categories. We also see that Google and IBM tools are much more susceptible to diversity than Microsoft.

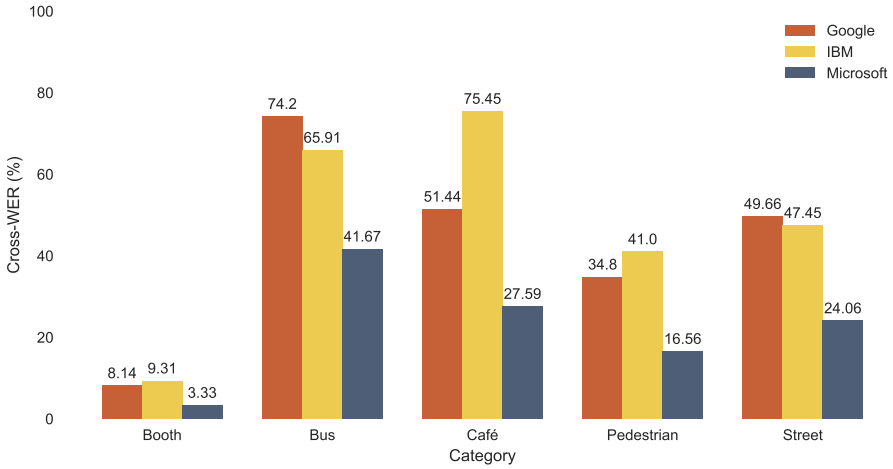


Fig. 3.7 Cross-WER (%) Values for ASRs and Noise Categories Using all 20 Transformations on the `Development` set of the CHiME3 dataset

3.6.3 Complementarity of Hypothesis

Figure 3.8 plots pairwise oracle improvement matrices as heat maps between all transcripts of all transformations for the different ASRs grouped by noise categories. The matrices are computed according to Equation 3.11 but for multiple utterances[13]. Each cell in a matrix represents the oracle improvement by combining row transformation transcripts and the column transformation transcripts over the row transformation transcripts. Brightness is on the scale of [0, 0.5]. The major diagonals represents the seemingly counter-intuitive *self-improvements* (See § 3.5.3). We hypothesized that these improvements will be minimal. This is supported by the dark major diagonals here, indicating that improvements due to removing extra words and alignment issues are minimal.

[13]The computation consists of micro-averaging, which is equivalent to treating the multiple utterances as one long utterance.

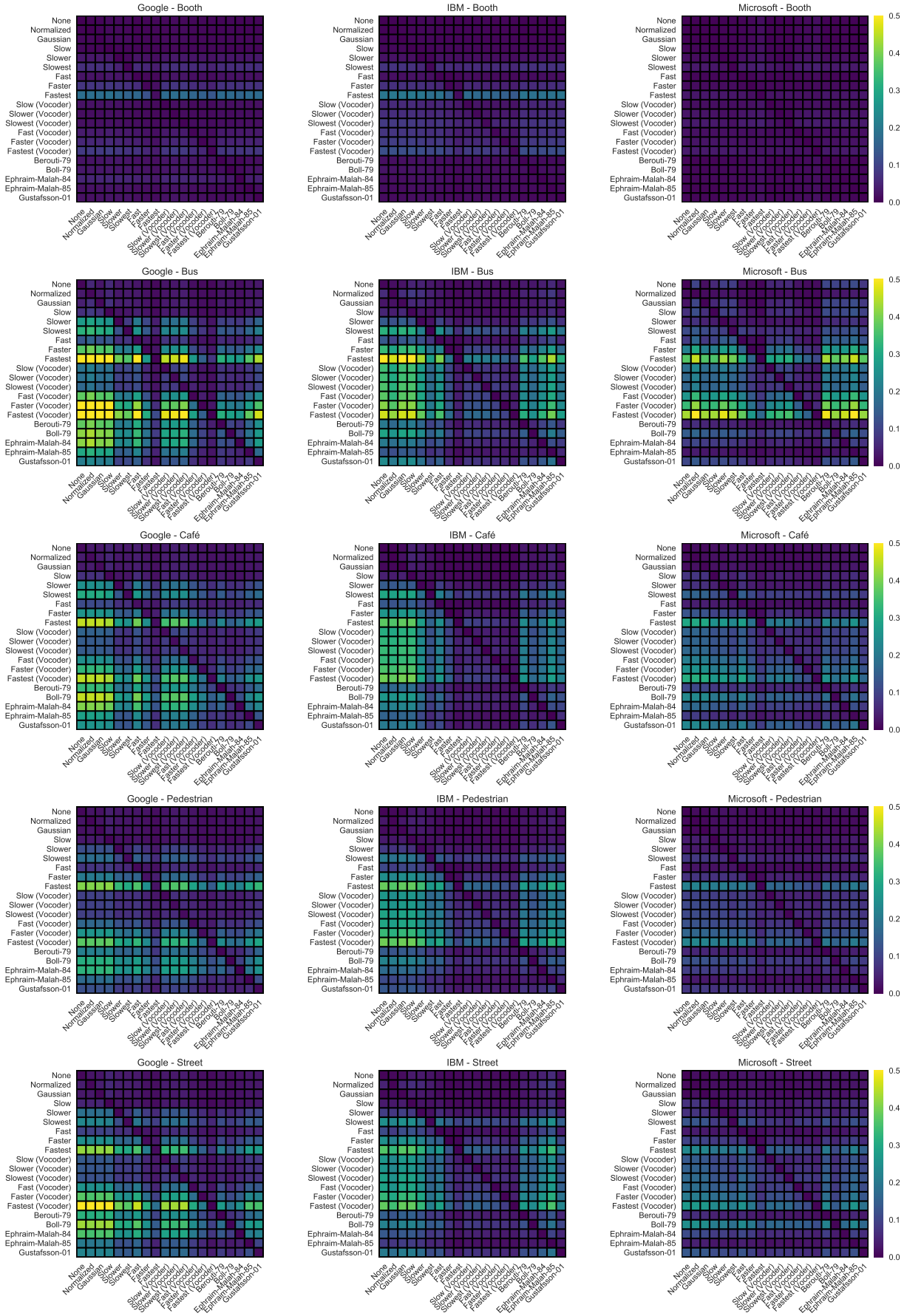


Fig. 3.8 Pairwise Oracle Improvements Between Transformations

Figure 3.9 shows the oracle combination improvements over the baseline performance for the ASRs in all categories using all 20 transformations. We can see that very little ideal improvements exists in the Booth (clean) category. Greater improvements exist in noisy categories.

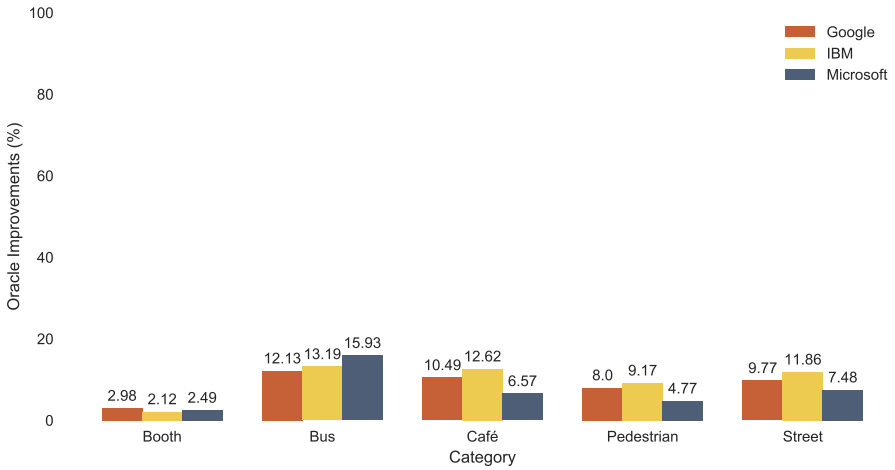


Fig. 3.9 Oracle Combination Improvements (Ideal Reduction in WER) using all Transformations

Figure 3.9 reported oracle combination improvements (ideal reductions in WER) using all 20 transformations considered in the experiments. However, we do not expect all transformations to equally contribute to these reductions. We can show how ideal combination WER changes as we use different subsets of the transformations. If most of the improvements come from just a few transformations, we can limit our use to those. Figure 3.10 shows *incremental* oracle combination WER values as we start with one transformation and add more transformations based on the ranking of transformations in Figure 3.4. This ordering is motivated by the fact that we want to use fewer transformations and want individual transformations to have small error rates on their own. The dashed lines represent the baseline WER values. We see that oracle combination WER improves as we add more transformations. We also see that the oracle combination WER saturates after a certain point, and that most improvements can come from the best few transformations.

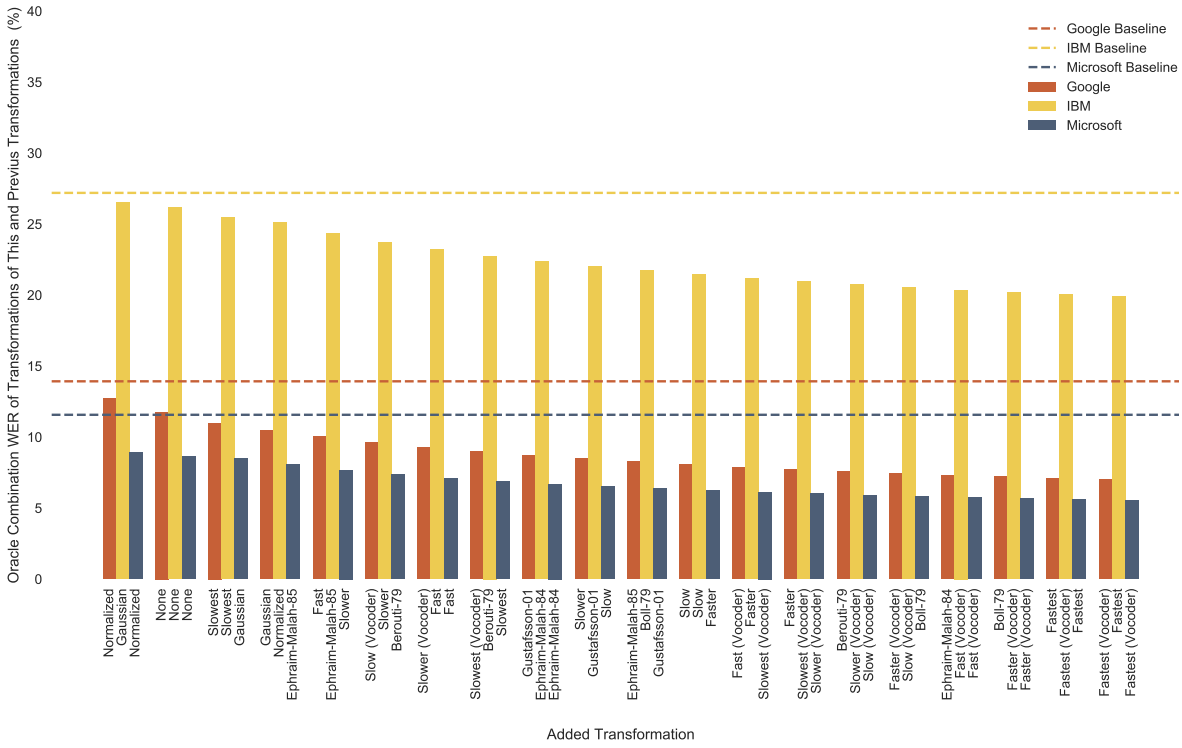


Fig. 3.10 Incremental Oracle Combination WER (%) as we start with a single transformation and continue adding transformations for the full `Development` set of the CHiME3 dataset.

3.7 Related Work

In this work, we use the formulation of data diversity as a fault-tolerance mechanism by Ammann and Knight [1, 46] who make the observation that software systems are often sensitive to perturbations in the input data. It is possible for software systems to produce different results on equivalent input data. Their proposed N-copy programming system is a direct analogue of the concept of N-version programming introduced by Avizienis et. al [4]. Ammann terms diversity inherent to design of systems as *design diversity*. Some works [13, 60] have made explicit use of this formulation of data diversity where an N-copy programming-like system is employed and referred to as a *multi-variant* system. A related concept is the notion of *ensemble methods* [17] in machine learning. Ensemble methods could be related to both data and design diversity. Some techniques such as boosting [25] combine weak learners into a strong learner. Other methods such as bagging (short for bootstrap aggregating) [10] generate diverse systems by training on different random subsets of the training set.

An early prominent work exploiting diversity in automatic speech recognition is by Fiscus [23] who showed that lower word error rates can be achieved by combining transcriptions from different ASR systems. The Recognizer Output Voting Error Reduction (ROVER) algorithm

developed as a part of this work has been the basis for many hypothesis-level ensemble methods in automatic speech recognition. Other approaches have considered combination of systems at the frame level rather than the hypothesis level [11, 77]. Some ensemble-like approaches are called *multi-stream* methods [37]. Examples of this include systems where different systems are trained to work with different frequency bands [70, 61].

A work that very closely resembles our work is by Kumar et. al [47] who show that diverse hypotheses can be generated by applying different denoising algorithms to the input speech. Their intuition is similar to that of the work in [49], which claims denoising and speech enhancement algorithms enhance some parts but deteriorate other parts of speech. Although our experimental work considers some speech enhancement algorithms as label-preserving transformations, we make the claim that diversity in speech input in general can be used to generate diverse outputs. We make use of metrics defined in previous work for quantifying diversity [2, 76]. Our approach to measuring complementarity of multiple transcripts via an oracle is similar to Breslin's [11] notion of IDEAL combination in the context of confusion network combination (CNC). An even earlier work by Burget [12] introduces the notion of Dependent Word Error Rate (DWER) as another measure of complementarity.

3.8 Summary and Conclusions

In this chapter, we explored injecting diversity in the input of an ASR system as a means of inducing diversity in the output. We generated variants of speech by applying label-preserving transformations such as time-warping and speed modulation. We processed these variants to generate a list of hypotheses. Our experimental results using multiple commercial ASR systems on the CHiME3 `Development` set showed that we were able to generate diverse hypotheses, and that greater diversity was achieved under noisy conditions. We quantified overall diversity among hypotheses generated by different transformations with the Cross-WER metric [76]. We introduced diversity matrices as a visual representation of diversity. We introduced the notion of an oracle combination of hypotheses and provided an algorithm for computing it. We used this oracle combination to show complementarity or error-correcting potential of hypotheses. We visualized complementarity of a pair-wise transformations using the oracle improvement matrices as heatmaps. Our experimental results showed that hypotheses generated by redundantly processing speech-to-text on diverse variants of speech have great potential for reduction in WER in noisy environments.

Chapter 4

Hypothesis Combination

This chapter investigates the problem of combining multiple hypotheses (i.e., transcriptions of speech). In Chapter 3, we showed that we are able to generate diverse and complementary hypotheses. In this chapter, our goal is to combine these hypotheses into a single hypothesis with the aim of reducing errors. We discuss the well-known ROVER algorithm and explore some factors that may contribute to poor combinations with it. We propose some (minor) modifications aimed at making ROVER perform better when a simple majority voting scheme is utilized. The goal of this chapter is not to find an optimal combination scheme, but rather to demonstrate modest improvements with the data-diverse redundant processing framework as a noise-robustness method[1].

4.1 The Hypothesis Combination Problem

The problem of combining hypotheses from different sources has application in many fields including automatic speech recognition (ASR) and machine translation (MT). Given that $\mathcal{H} = \boldsymbol{w}_1, \boldsymbol{w}_2, ..., \boldsymbol{w}_N$ is a multiset of N hypotheses corresponding to a speech signal $\boldsymbol{x}$, the goal of hypothesis combination is to combine the hypotheses using words from any or all individual hypotheses to generate a single hypothesis $\boldsymbol{w}^{\Psi}$ in a way that word error rate (WER) will be reduced. This can be thought of as an approximation of the oracle combination $\Phi(\mathcal{H})$ or the "best possible combination" of hypotheses discussed in 3.5.1. Figure 4.1 illustrates this system. We note that unlike the oracle combination in Figure 3.3, we no longer have access to the reference transcript $\boldsymbol{w}^{\text{Ref}}$.

[1] Although the "ideal" performance of the oracle combination in Algorithm 1 makes a good case for the use of data diversity as a noise-robustness approach and treats the actual combination problem as an "externality", the results may be unconvincing, as they require recourse to reference transcripts for combination.

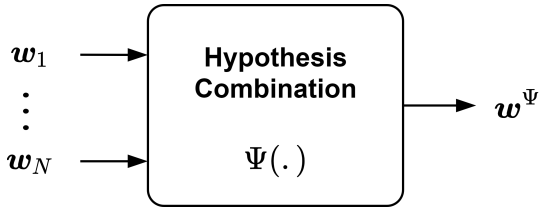


Fig. 4.1 Hypothesis Combination of Multiple Hypotheses

As with the oracle combination $\Phi(\mathcal{H})$, we can define an improvement metric for an arbitrary combination scheme $\Psi(\mathcal{H})$:

$$\Delta^{\Psi}_{\boldsymbol{w}^{\text{Base}}}(\mathcal{H}) = \text{WER}(\boldsymbol{w}^{\text{Ref}}; \boldsymbol{w}^{\text{Base}}) - \text{WER}(\boldsymbol{w}^{\text{Ref}}; \Psi(\mathcal{H})) \tag{4.1}$$

Unlike the oracle improvement $\Delta^{\Phi}_{\boldsymbol{w}^{\text{Base}}}$, however, we are not guaranteed a non-negative improvement $\Delta^{\Psi}_{\boldsymbol{w}^{\text{Base}}} \geq 0$ if $\boldsymbol{w}^{\text{Base}} \in \mathcal{H}$. This means that a poor combination scheme could be more detrimental than helpful. An intuitive approach to combination consists of first finding areas of agreement and disagreement between the hypotheses and finding schemes for selecting one of many options in areas of disagreement. The Recognizer Output Voting Error Reduction (ROVER) algorithm [23] does exactly that and is introduce in § 4.2. We note that ROVER has been extensively studied by researchers and is the basis of many "hypothesis-level" combination methods. This chapter considers a ROVER-like approach to hypothesis combination to show improvements in ASR performance by exploiting data diversity. We review work related to ROVER in § 4.6.

4.2 ROVER

The Recognizer Output Voting Error Reduction (ROVER) algorithm [23] was devised at NIST and is a simple approach to combining multiple hypotheses. ROVER is broken down into multiple stages, as shown in Figure 4.2.

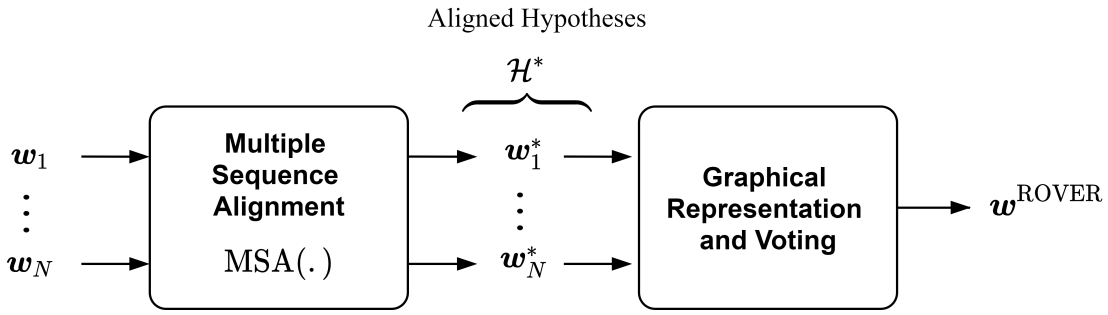


Fig. 4.2 ROVER System Architecture

Given that $\mathcal{H} = \boldsymbol{w}_1, \boldsymbol{w}_2, ..., \boldsymbol{w}_N$ is a list of hypotheses to be combined, ROVER first applies multiple sequence alignment (MSA) to the hypotheses to align them to each other. This consists of strategically inserting "_" tokens (called gaps, nulls, or skips) into the sequences such that they (i.e,, the sequences) will all have the same length and alike words in different sequences are aligned. The MSA problem has widely been studied in the field of bioinformatics, where biological sequences such as DNA are often aligned. We provide a brief formal description of this problem in Appendix A.2. The aligned sequences $\mathcal{H}^* = \boldsymbol{w}_1^*, \boldsymbol{w}_2^*, ..., \boldsymbol{w}_N^*$ consist of N word sequences of M words (including gaps) each. $\mathcal{H}^*$ is termed the *alignment table* in this thesis since the elements can be arranged in an $N \times M$ grid:

$$\begin{array}{rcccc} \boldsymbol{w}_1^* = & w_{11}^* & w_{12}^* & \dots & w_{1M}^* \\ \boldsymbol{w}_2^* = & w_{21}^* & w_{22}^* & \dots & w_{2M}^* \\ & \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{w}_N^* = & \underbrace{w_{N1}^*}_{\mathcal{J}_1} & \underbrace{w_{N2}^*}_{\mathcal{J}_2} & \dots & \underbrace{w_{NM}^*}_{\mathcal{J}_M} \end{array} \tag{4.2}$$

ROVER's original formulation in [23] uses an approximate MSA method using iterative applications of pairwise alignments. Figure 4.3 shows an example of an alignment table of multiple hypotheses corresponding to there reference transcript: "the cat in the hat sat on the mat." Gaps (i.e., "_" tokens) have been inserted in particular places in the sequences such that alike words from different sequences line up and all sequences have the same length.

the	cat	and	the	hat	_	on	_	mat
the	bat	in	that	hat	sat	in	the	mat
the	cat	end	_	hat	_	on	the	mat
the	cat	in	the	at	sat	on	the	mat
the	cat	in	_	at	sat	on	_	mat

Fig. 4.3 Multiple Sequence Alignment (MSA) Table

ROVER generates a Word Transition Network (WTN) from the alignment table $\mathcal{H}^*$ by representing unique elements of each column in the table (i.e., words in corresponding locations across hypotheses) and representing them as edges between two nodes in a directed acyclic graph. Figure 4.4 shows the WTN for the example in the alignment table above. We note that this graph is a confusion network (defined in § 2.5).

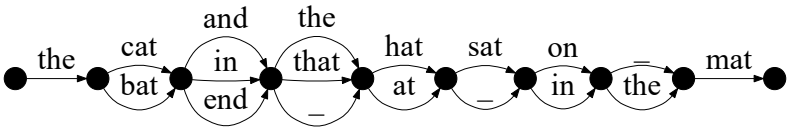


Fig. 4.4 ROVER Word Transition Network (WTN) or Confusion Network (CN)

We refer to the multiset of words in the ith column of the alignment table as a *correspondence* or *confusion list* $\mathcal{J}_i = w^*_{1i}, w^*_{2i}, ..., w^*_{Ni}$. The set of *unique* elements in the ith column is the *correspondence* or *confusion set* $\mathcal{C}_i^{\text{ROVER}} = \text{Unique}(\mathcal{J}_i)$. The generated confusion network consists of M confusion sets denoted as $\mathcal{Q}^{\text{ROVER}} = \mathcal{C}_1^{\text{ROVER}}, \mathcal{C}_2^{\text{ROVER}}, ..., \mathcal{C}_M^{\text{ROVER}}$. The ith confusion set consists of K_i words: $\mathcal{C}_i^{\text{ROVER}} = w_1, w_2, ...w_{K_i}$. ROVER finds a single path through the confusion network represented by a sequence of words:

$$\boldsymbol{w}^{\text{ROVER}} = \text{ROVER}(\mathcal{H}) = w_1^{\text{ROVER}}, w_2^{\text{ROVER}}, ..., w_M^{\text{ROVER}} \tag{4.3}$$

Each word within a confusion set $\mathcal{C}_i^{\text{ROVER}}$ is scored based on a convex combination[2] of the word's relative frequency of occurrence and a confidence score. The ROVER result $\boldsymbol{w}^{\text{ROVER}}$ is then found by selecting the word with the highest score (breaking ties arbitrarily) from each confusion set:

$$w_i^{\text{ROVER}} = \underset{w \in \mathcal{C}_i^{\text{ROVER}}}{\arg\max} \left[\alpha \frac{\text{Count}(w)}{N} + (1 - \alpha)\, \text{Confidence}(w) \right] \tag{4.4}$$

The term $\text{Count}(w)$ is the number of times the word w appears in the ith column of the alignment table (i.e., in the the confusion list $\mathcal{J}_i$). The term N is the number of hypotheses to be combined. The term $\text{Confidence}(w)$ returns a confidence score for the word within the confusion set. The confidence score returned by the $\text{Confidence}(w)$ function could be average, maximum, or some other combination of confidences from the ASR for a word w. It could alternatively have been assigned before the alignment or once the confusion network is generated. The convex combination parameter α could be tuned on a development set[3]. Once the ROVER sequence $\boldsymbol{w}^{\text{ROVER}}$ is found, we can define the ROVER improvement metric:

$$\Delta^{\text{ROVER}}_{\boldsymbol{w}^{\text{Base}}}(\mathcal{H}) = \text{WER}(\boldsymbol{w}^{\text{Ref}}; \boldsymbol{w}^{\text{Base}}) - \text{WER}(\boldsymbol{w}^{\text{Ref}}; \text{ROVER}(\mathcal{H})) \tag{4.5}$$

In this work, we make use of a simple implementation of the ROVER algorithm around an open-source multiple sequence alignment library in Python[4]. Our work did not consider confidences i.e., $\alpha = 1$, which is referred to as *majority voting*, as captured by Equation 4.6. The factor $1/N$ is dropped since it does not change the maximization.

$$w_i^{\text{ROVER}} = \underset{w \in \mathcal{C}_i^{\text{ROVER}}}{\arg\max}\, \text{Count}(w) \tag{4.6}$$

[2]Linear combination of points where coefficients are non-negative and add up to 1.
[3]Experiments by Fiscus [23], however, did not show that this tuning could generalize to unseen data.
[4]https://github.com/Franck-Dernoncourt/ASR_benchmark

Figure 4.5 shows the confusion network of Figure 4.4 where the paths with the highest frequencies are emphasized. We note that there are multiple possibilities and one is arbitrarily selected by Equation 4.6.

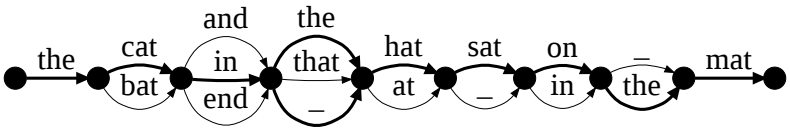


Fig. 4.5 ROVER Majority Voting

4.3 Best and Worst Confusion Network Paths (Oracles)

The sequence $\boldsymbol{w}^{\text{ROVER}}$ describes a path through the ROVER confusion network represented by the sequence of confusion sets $\mathcal{Q}^{\text{ROVER}}$. However, due to the simplicity of the "voting" method in Equation 4.4, we will not always make good choices i.e., we may select a path in the confusion network that may not be optimal in terms of WER. This has been observed in prior work [38], who propose that more sophisticated methods than majority voting should be used for finding paths. In fact, it is a distinct possibility that ROVER$(\mathcal{H})$ will yield a higher WER than the individual hypotheses to be combined i.e., $\text{WER}(\boldsymbol{w}^{\text{Ref}}; \boldsymbol{w}^{\text{ROVER}}) > \text{WER}(\boldsymbol{w}^{\text{Ref}}; \boldsymbol{w})\ \forall \boldsymbol{w} \in \mathcal{H}$. In this section, we propose computing *oracle* bounds on WER achieved from the confusion network $\mathcal{Q}^{\text{ROVER}}$. These bounds will be assessments of the confusion networks themselves rather than the "voting" scheme of Equation 4.4 or Equation 4.6. We first define an arbitrary path through the confusion network as a sequence of words (including skip or null elements), where the ith word is selected from the ith confusion set:

$$\boldsymbol{w}^{\text{Path}} = w_1^{\text{Path}}, w_2^{\text{Path}}, ..., w_M^{\text{Path}} \quad \text{, where } w_i^{\text{Path}} \in \mathcal{C}_i^{\text{ROVER}} \tag{4.7}$$

Next, we consider the case of when we pick a best performing path from the confusion network $\mathcal{Q}^{\text{ROVER}}$. We can define a best path $\boldsymbol{w}^{\text{Best}}$ as a path through the confusion network that minimizes the WER, as expressed by Equation 4.8. We note that this path is not necessarily unique.

$$\boldsymbol{w}^{\text{Best}} = \underset{\boldsymbol{w}^{\text{Path}}}{\arg\min}\, \text{WER}(\boldsymbol{w}^{\text{Ref}}; \boldsymbol{w}^{\text{Path}}) \tag{4.8}$$

Figure 4.6 shows the confusion network of Figure 4.4 with the best path highlighted, which contains no errors. For this particular example, the best path is unique.

The idea of selecting the best path from a confusion network is similar to the idea of oracle combination $\Phi(\mathcal{H})$ that we introduced in § 3.5.1. However, the confusion network generated by ROVER is more constrictive because it imposes a more rigid structure on the combination. For

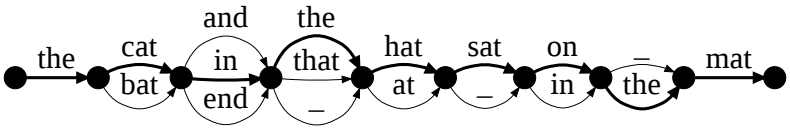


Fig. 4.6 Best Path in the Confusion Network

example, if we are computing the best path and a confusion set $\mathcal{C}_i^{\text{ROVER}}$ within the confusion network contains all incorrect words, we are still forced to select a word $w_i \in \mathcal{C}_i^{\text{ROVER}}$ from that confusion set. This is in contrast to our definition of $\Phi(\mathcal{H})$, where we can simply filter out incorrect words. In general, we expect the oracle combination to be better than the ROVER confusion network best path[5]:

$$\text{WER}(\boldsymbol{w}^{\text{Ref}}; \boldsymbol{w}^{\text{Best}}) \geq \text{WER}(\boldsymbol{w}^{\text{Ref}}; \boldsymbol{w}^{\Phi}) \tag{4.9}$$

We can also define a worst path $\boldsymbol{w}^{\text{Worst}}$ as a path through the confusion network $\mathcal{Q}^{\text{ROVER}}$ that maximizes WER (Equation 4.10). ROVER is *guaranteed* to be helpful if the WER achieved by the worst path is less than that of a baseline. However, this is generally hard to achieve in practice. This worst performing path may also not be unique.

$$\boldsymbol{w}^{\text{Worst}} = \arg\max_{\boldsymbol{w}^{\text{Path}}} \text{WER}(\boldsymbol{w}^{\text{Ref}}; \boldsymbol{w}^{\text{Path}}) \tag{4.10}$$

Figure 4.7 shows the confusion network of Figure 4.4 with the worst paths highlighted. We note that there are many options here and one is arbitrarily picked by Equation 4.10.

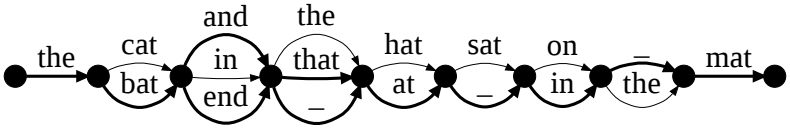


Fig. 4.7 Worst Path in the Confusion Network

Implementation Detail: As with finding the oracle combination $\boldsymbol{w}^{\Phi}$, we need access to the reference transcript $\boldsymbol{w}^{\text{Ref}}$ to find $\boldsymbol{w}^{\text{Best}}$ and $\boldsymbol{w}^{\text{Worst}}$. A naive implementation of the oracles will check all possible paths in the graph and keep track of the best and worst performing paths. However, the number of paths is generally too large and checking every path is infeasible. The number of paths in a confusion network is the product of the cardinality of all confusion sets, which increases very rapidly with the number of confusion sets M and confusion sets with cardinalities greater than unity:

[5]This is usually the case but does not mathematically follow from our algorithmic definition of Φ. It may be possible to conceive of cases where the confusion network contains a better sequence than the oracle combination, This is because the our definition of the oracle combination in Algorithm 1 imposes some structure on the combination and is not the best possible *arrangement* of words from the hypotheses.

$$\text{Number of Paths}(\mathcal{Q}^{\text{ROVER}}) = \prod_{i=1}^{M} |\mathcal{C}_i^{\text{ROVER}}| \tag{4.11}$$

We implement the oracles by pruning the confusion networks first. For example, when computing the best path $\boldsymbol{w}^{\text{Best}}$, we can remove all edges corresponding to incorrect words except for null arcs and making sure at least a single edge is kept in each confusion set. This significantly reduces the size of the graph and allows for brute-force checking of fewer paths. Figure 4.8 illustrates the pruning of the confusion network of Figure 4.4 to optimize finding the best path. The edges with dotted lines are no longer considered a part of the confusion network. We can see that we need to check fewer paths than the original network.

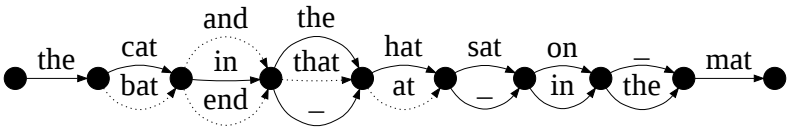


Fig. 4.8 Pruned Confusion Network in Order to Find the Best Path

4.4 Tuning ROVER for Better Performance

As we will see in the experiments in § 4.5, results obtained from ROVER show great *potential* in improvement in terms of a best oracle WER. However, they also show great potential in making things worse in terms of the worst oracle WER. We will also see that a simple majority voting could not exploit the potential for improvements in many cases. In this section, we describe tuning ROVER along a number of dimensions with the goal of improving the majority voting performance.

Pre-Alignment Filter: We observe that too much diversity among the hypotheses in $\mathcal{H}$ may cause a number of issues in hypothesis combination via ROVER. For example, if we use transcripts from poor-preforming transformations, we will see a lot of diversity, but this will likely be due to many incorrect words. This will result in poor confusion networks in terms of best, worst, and majority voting results. We propose not to include certain transcripts if we believe they will do more harm than good. This can be done by simply dropping certain transformations that perform poorly in general[6].

Improving Alignments: The goal of multiple sequence alignment (MSA) in the context of ROVER hypothesis combination is to find areas of agreements and disagreements over time among the hypotheses. We observe that it is possible for standard MSA to align tokens that

[6]It can also be done dynamically if we can predict which hypotheses will not contribute to the combination.

did not originally overlap in time. Similarly, it may not align tokens that did overlap in time. Consider the example below with just two sequences:

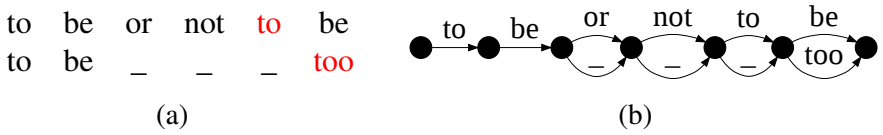


Fig. 4.9 Example of an Alignment Issue

This alignment maximizes the match for sequences if the scoring function penalizes all inequalities similarly. However, it may be more sensible to align the word "to" with the word "too" since they are homophones:

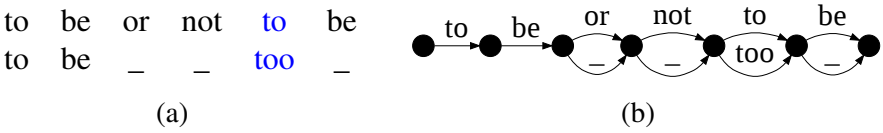


Fig. 4.10 Improved Alignment

Although very sophisticated improvements can be applied to MSA, we supply the MSA function with a custom scoring function that considers certain tokens to be equal for the alignment purpose. These include words that are homophones (found using the CMU phonetic dictionary[7]), words that share a linguistic root (determined by lemmatization and stemming functions of the Python Natural Language Toolkit[8]), and commonly mistaken words (determined by empirical study of the errors). The hope is that this would generate better alignments.

Post-Alignment Filter: Once an MSA table is generated but before a confusion network is built, we can apply some pruning to remove possibility of problematic sequences. A particular case of interest to us are null arcs (denoted by "_") in confusion networks. Null arcs are important components of confusion networks, as they allow for us to skip selecting words from a confusion set. However, they could significantly reduce performance by outvoting correct words. Consider the example below with just three hypotheses for the reference sentence: "I do not like green eggs and ham".

I	do	not	like	green	eggs	and	_
_	_	_	_	green	eggs	_	_
_	_	_	_	_	_	and	ham

The confusion network generated by this example allows for a path of all nulls, as shown in Figure 4.11. The majority voting result only returns the phrase: "green eggs and".

[7] http://www.speech.cs.cmu.edu/cgi-bin/cmudict
[8] https://www.nltk.org/

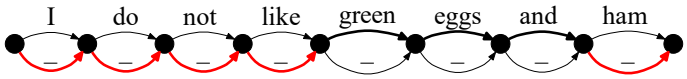


Fig. 4.11 Example of a Confusion Network where an All-Null Path is Possible

We make the observation that the number of confusion sets M in a confusion network is roughly the same as the number of words actually spoken[9]. This observation has been made before [62]. With this, we claim that the best path (represented by a sequence) through a confusion network should have roughly M non-null elements i.e., only very few or no skips will exist in the best path. We claim that ROVER can perform better if some null arcs are pruned before voting. We propose not to add null arcs that in the alignment table are leading or trailing. Some potential improvement will inevitably be sacrificed, but we assert this will make the worst case and majority voting results much better overall. For the example above, the generated confusion network after pruning leading and trailing nulls is shown in Figure 4.12, which no longer requires voting:

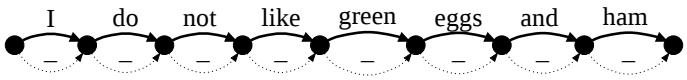


Fig. 4.12 Example of a ROVER Confusion Network after Post-Alignment Filtering

4.5 Experiments

In this section, we present our experimental results for combining hypotheses using ROVER and our modifications (excluding pre-alignment filter) to ROVER (referred to as ROVER+ for convenience). We performed experiments on the `Development` set of the CHiME3 dataset using the Google Cloud Speech to Text, IBM Watson Speech to Text, and Microsoft Azure Speech to Text. The configuration and setup is the same as the previous chapters (Appendix B).

4.5.1 Incremental Combination of Hypotheses

To find which transformations to drop (i.e., to define the pre-alignment filter), we considered how the combination WER changed as we started with a single transformation and added more transformations. The order of transformations we selected was based on the ascending order of WERs, as reported in Figure 3.4. This was motivated by the fact that we want to use fewer transformations and want individual transformations to have small error rates on their own.

Figures 4.13 shows our results for the Google, IBM, and Microsoft systems on the entire `Development` set of the CHiME3 dataset. The *Baseline* (dotted blue line) curves refer to the

[9]Or at least, the number of words actually spoken is not much smaller than the number of confusion sets M

baseline performance in terms of WER of the corresponding ASR on the entire `Development` set of the CHIME3 dataset before we consider any label-preserving transformations. We see that the Microsoft system has the best overall performance. The Google system narrowly follows. The IBM system has the worst overall performance. The *Oracle Combination* curves indicate incremental oracle combination of the transformations. Each dot on this curve represents WER of the oracle combination (Algorithm 1) of all transcripts of the transformations up to that point. This is a repetition of the numbers reported in Figure 3.10. We expect these curves to be monotonically decreasing, as adding more transformations can only make WER better.

Figure 4.13 also reports ROVER and ROVER+ best paths, worst paths, random paths[10], and majority voting paths. We make some general observation across all ASRs. We observe that as more transformations are added, WER for ROVER and ROVER+ best paths decrease and WER for ROVER and ROVER+ worst paths increase. This is because we are adding both correct and incorrect edges to confusion networks. This supports our claim that hypothesis combination has potential to both improve but also to deteriorate performance. The best path curves for both ROVER and ROVER+ follow the oracle combination curve but are higher generally. This is because these WERs are found by picking paths from a confusion network which imposes a rigid structure on the possible combination of hypotheses. Similar to the oracle combination curves, we do see that the ROVER and ROVER+ best path curves stagnates at some point. They actually get slightly worse when adding more transformation at some points. We attribute this to the fact that as more transformations are added, multiple sequence alignment (MSA) becomes a harder problem which could lead to sub-optimal confusion networks. The oracle combination curve does not suffer from this issue since it does not depend on alignment of hypotheses to *each other* but rather aligns the hypotheses to the reference transcript i.e., it can only get better by adding more transformations. We note that ROVER+ in general has a better worst path and slightly worse best path compared to ROVER. This can be attributed to our pruning of the confusion networks, which inevitably removes some optimal paths. In terms of majority voting, we observe that it is a fairly poor method for finding optimal paths through the confusion networks. In fact, it actually makes the performance worse if too many hypotheses are combined. This can also be attributed to poor confusion networks (e.g., votes not getting aggregated due to misalignments or too many incorrect words). We do, however, observe that as hypothesized, ROVER+ performs better in terms of majority voting than ROVER. We highlight the best possible non-oracle combination with a red square. These are reductions of 1.29% with 4 transformations for the Google system, 1.73% with 6 transformations for the IBM system, and 3.07% with 9 transformations for the Microsoft system. These are all achieved by ROVER+.

[10]By selecting edges with equal probability from each confusion set. Empirically, they tend to be fairly stable for large data and can be intuitive (admittedly informal) "average" performance assessments of confusion networks.

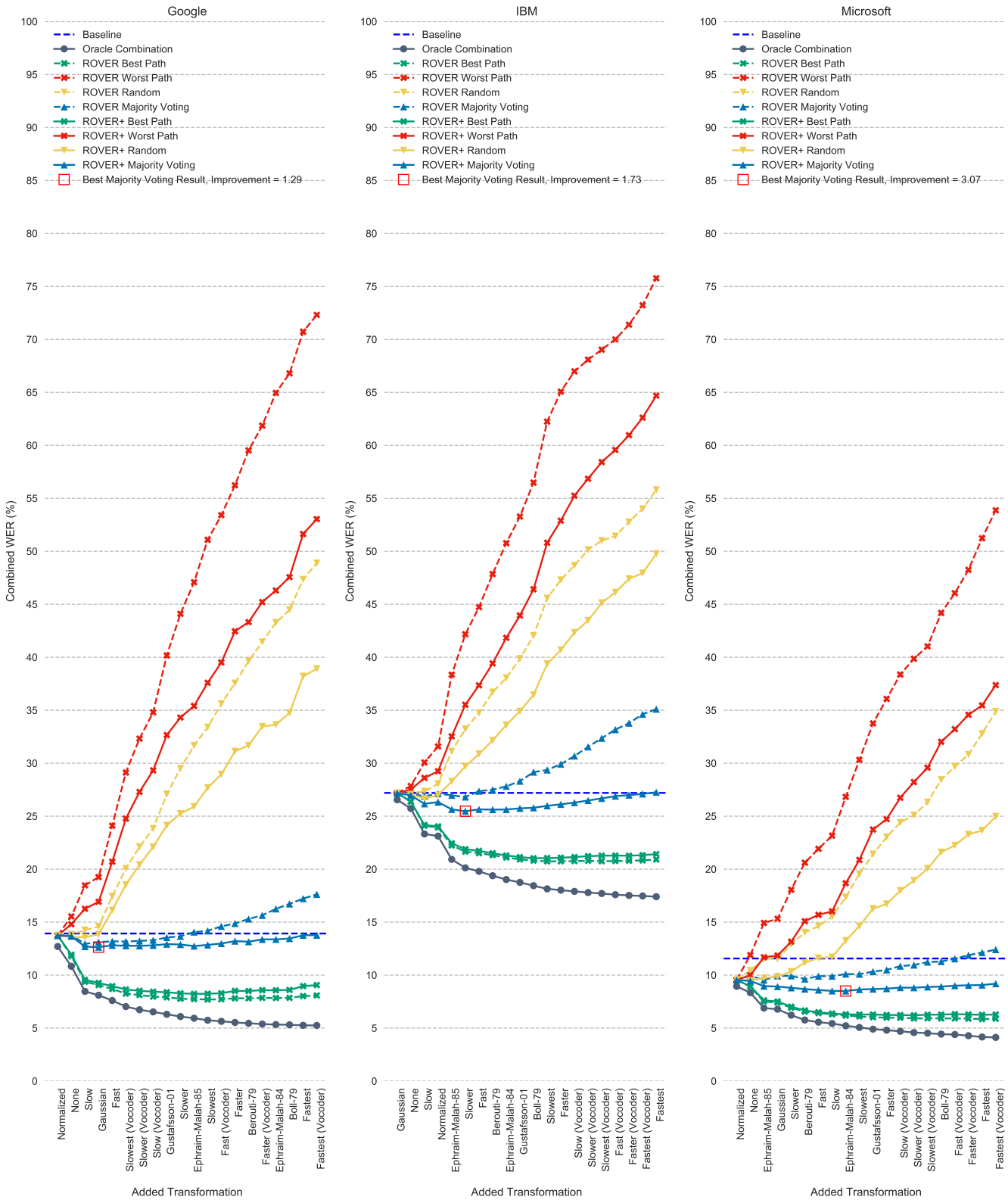
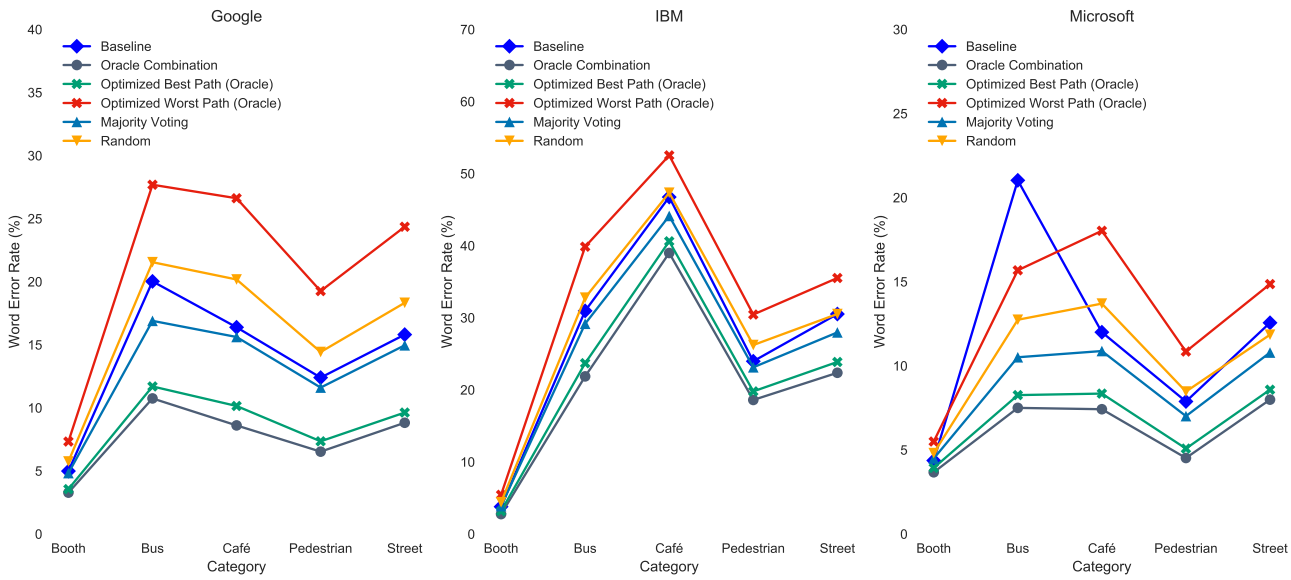


Fig. 4.13 Combination WER (%) for ROVER and ROVER+ on the full (across all categories) CHiME3 Development Dataset. The horizontal axis represents a transformation added to be combined with previous transformations. The vertical axis represents a combined WER of all transformations up to that point in the inverse image axis.

4.5.2 Results with Five Transformations

We observed *diminishing returns* with the incremental combinations in Figure 4.13. In particular, we saw that most of the *potential* in improvements (in terms of oracle combination or ROVER or ROVER+ best oracles) could come from the best few transformations. We also found that our modifications in § 4.4 to ROVER made majority voting better, and that the performance of the majority voting either stagnated or became worse after some point. In this section, we report our results broken down by categories on the development set of the CHiME3 dataset. We selected the top *five* transformations for each ASR (i.e., this is the pre-alignment filter). This was chosen as a trade-off between potential in improvements, actual improvements with majority voting, the cost of using fewer transformations, and the need to compare between ASRs (which requires N to be the same between them). Figure 4.14 reports WER (%) values achieved using five transformations for all five categories of the development subset of the CHiME3 dataset and the ASRs.



(a) (b) (c)

Fig. 4.14 ROVER+ WER (%) Broken Down by Categories of the CHiME3 Development Dataset using the Best Five Transformations and Majority Voting

Figure 4.15 plots the improvements using ROVER+. We can see that the clean performance has not experienced much change. However, we gain improvements in noisy categories for all systems. The best improvement is in the Bus category for the Microsoft system at 10.52%. This is a little inconsistent with the other improvements, which are generally lower. Upon investigating, we found that many of the Microsoft transcriptions for the baseline were empty for this category. This indicates that the effect of noise on these system may have sudden jumps e.g., the system may not return anything at all under some noisy conditions. We speculate

that this is likely an intentional feature of these ASR systems where they choose not return anything at all, if they consider the input too noisy. This is further evidence that ASR systems are sensitive to perturbations in the input when the input is noisy, and that data diversity can exploit it.

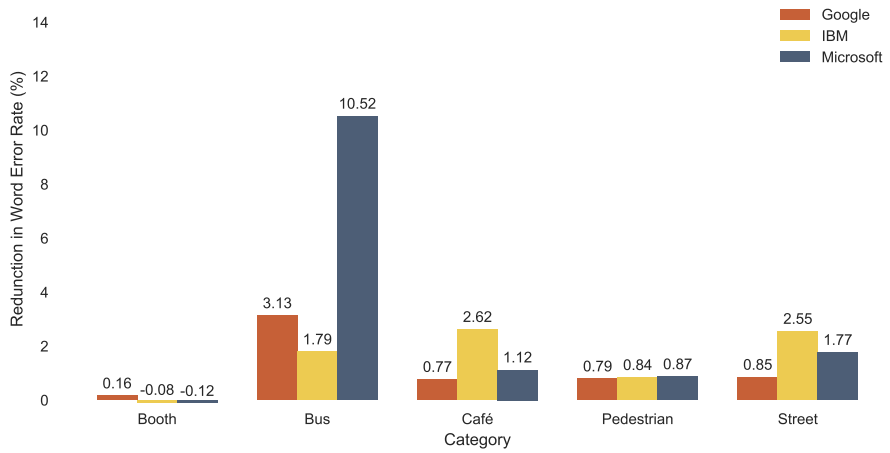


Fig. 4.15 ROVER+ Reductions in WER (%) using Five Transformations and Majority Voting on the CHiME3 Development Dataset

To summarize our results, we can aggregate all the noisy categories together and compute an overall WER. Table 4.1 summarizes our results on the CHiME3 `Development` set. We see that modest improvements of 1.42%, 1.99 %, and 3.5% are found by the ROVER+ for the Google, IBM, and Microsoft systems respectively. We also show that even greater gains of 6.45%, 6.06%, and 5.8% are possible. For consistency, we only report the ROVER+ numbers. Best possible improvements with ROVER are even greater, since the confusion networks are not pruned.

	Google	IBM	Microsoft
Baseline	16.16	33.05	13.37
ROVER+ Best Oracle	9.71	26.99	7.57
ROVER+ Worst Oracle	24.48	39.59	14.86
ROVER+ Majority Voting	14.75	31.06	9.87
ROVER+ Best Oracle Improvement	**6.45**	**6.06**	**5.8**
ROVER+ Majority Voting Improvement	**1.42**	**1.99**	**3.5**

Table 4.1 WER (%) and Improvements (%) on Full Noisy Subset (Bus + Café + Pedestrian + Street) of the CHiME3 Development Dataset using the Best Five Transformations for Each ASR

To achieve WERs that are closer to the above lower bounds, however, we need more sophisticated mechanisms (some of them briefly discussed next in § 4.6) than simple majority

voting for finding paths from the confusion network $\mathcal{Q}^{\text{ROVER}}$. We will leave this as future work. We note that although some improvements are achieved, we are still very far from solving the problem of noise-robustness. Even the best possible improvements for noisy date do not yield word error rates comparable to word rates achieved for clean data.

4.6 Related Work

The Recognizer Output Voting Error Reduction [23] introduced in this chapter is the basis for many hypothesis combination algorithms. Many generalizations and improvements have been made to it over the years. Mangu et. al [54] formalize the notion of confusion networks and present algorithms for generating them from lattices. Evermann and Woodland [22] introduce a generalization of ROVER known as the confusion network combination (CNC) and demonstrated small improvements over ROVER. Schwenk et. al [66] study some of the peculiarities of the ROVER algorithm such as the impact of order of pairwise alignments in an approximate multiple sequence alignment algorithm on the combined word error rate. They also consider the case of breaking ties during voting, normalization of outputs, and incorporating language modelling in the selection process. Stolke et. al introduces the N-best ROVER [68] as another generalization of ROVER and a special case of CNC. Another generalization of ROVER is called e-ROVER [48, 28] and was shown to provide slight improvements. Xue et. al [78] very interestingly modifies the concept of confusion networks for when a single word is split into two words during recognition. Sankar et. al [65] propose a Bayesian decision-theoretic approach known as BAYCOM that shows significant improvements over standard voting schemes. Zhang et. al [79] proposes a neural network-based insertion detection and word scoring scheme for selecting hypotheses from the ROVER confusion networks. They also make the observation that frequency of occurrence and confidence scores are not enough for optimal selection of hypotheses. Hillard et. al [38] present iROVER that uses a classifier to make better decisions than voting schemes. Audkhasi et. al [3] presents a theoretical basis for ASR diversity and WER achieved by ROVER. Hoffmeister et. al [41] introduces the minimum fWER combination scheme by replacing the multiple sequence alignment part of ROVER with a time frame-wise word error cost. Hoffmeister et. al [42] extends the idea of iROVER [38] to confusion network combination [22]. We wish to incorporate some of these techniques in the data-diverse redundant processing framework for better improvements in the future.

4.7 Summary and Conclusions

In this chapter, we explored the problem of combining multiple hypotheses into a single hypothesis. We considered the well-known ROVER algorithm and proposed some minor

modifications to improve its results. We further proposed to find oracle (having knowledge of the reference) paths from confusion networks to show the range of possible WERs. Our experiment on diverse transcriptions of the `Development` set of the CHiME3 dataset show that modest actual improvements and *potential* for even greater improvements in terms WER are possible.

Chapter 5

Evaluation

In Chapters 3 and 4, we developed techniques, gained insights, and tuned parameters to show actual and potential improvements on the `Development` set of the CHiME3 dataset with the proposed data-diverse redundant processing framework (Figure 1.2). We showed that it is possible to generate diverse and complementary transcripts by applying diversity in the inputs. We showed that modest improvements can be found using a modified version of the ROVER algorithm with a simple majority voting method. We also showed empirical bounds on WER from generated confusion networks promising even greater gains. In this Chapter, we evaluate the system developed in previous chapters on the unseen `Evaluation` set of the CHiME3 dataset.

5.1 Experiments

In this section, we report our experimental results on the `Evaluation` subset of the CHiME3 dataset with the Google Cloud Speech to Text, IBM Watson Speech to Text, and Microsoft Azure Speech to text systems. The configuration and setup is the same as the previous chapters. Further details on the configuration and the dataset can be found in Appendix B). In § 5.1.1, we establish baseline WER values for the different categories of the dataset. In § 5.1.2, we apply data-diverse redundant processing using the configuration found in Chapter 4 i.e., modified ROVER (labeled ROVER+ here) with top five transformations for each ASR.

5.1.1 Baseline Performance

In this section, we establish baseline performances for the Google, IBM, and Microsoft systems in terms of WER. The bar plot in Figure 5.1 reports WER (%) values achieved for all five categories of the `Evaluation` subset of the CHiME3 dataset and the ASRs.

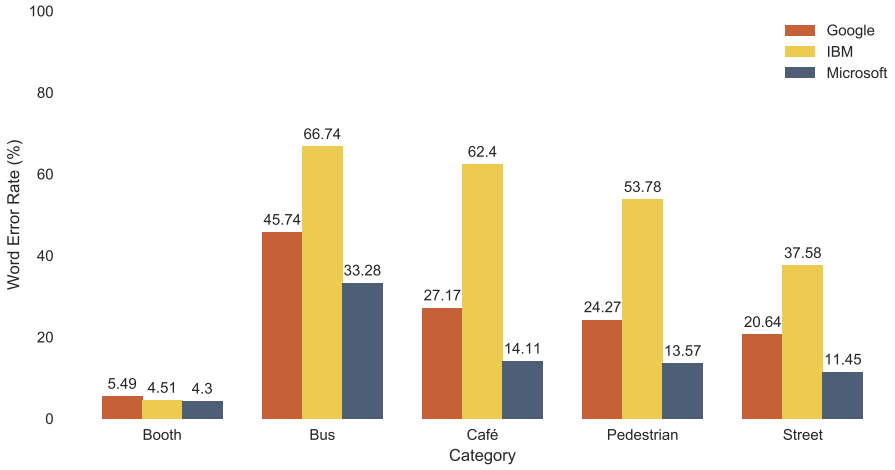
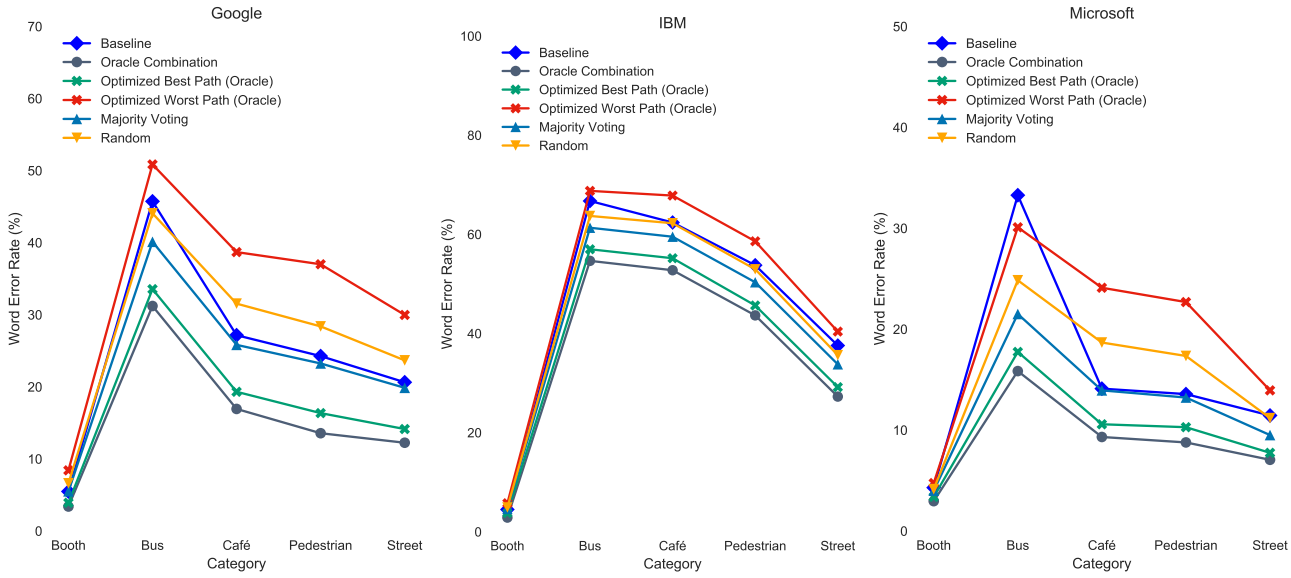


Fig. 5.1 Comparison of ASR Transcriptions in Terms of WER on the Evaluation Set

The results are similar to the results we found for the development set, as reported in Figure 2.6. All ASRs do well on clean speech and poorly on noisy speech. The degradation is the worst with the IBM system. The best results are found with the Microsoft system. The results are generally worse than the results achieved for the `Development` set of the CHiME3 dataset, as reported in Figure 2.6. We attribute this mismatch to the fact that the `Development` and `Evaluation` sets of the CHiME3 dataset do not have speakers in common.

5.1.2 Data-Diverse Redundant Processing Results

To apply data-diverse redundant processing framework for the `Evaluation` set, we used the same setup as that for the `Development` set in Chapter 4. Figure 5.2 reports WER (%) values achieved using five transformations for all five categories of the `Evaluation` subset of the CHiME3 dataset and the ASR, Here, we make similar observations as the ones we made the `Development` set. We are able to achieve modest improvements with a simple majority voting scheme. However, this does not seem to exploit the full potential for improvements.



(a) (b) (c)

Fig. 5.2 ROVER+ WER (%) Broken Down by Categories of the CHiME3 Evaluation Dataset using the Best Five Transformations and Majority Voting

Figure 5.3 plots the improvements (reductions in WER) using ROVER+. Just like the results for the `Development` set in the previous chapter (Figure 4.15), we can see that we have not gained any improvement on clean data. We were, however, able to achieve improvements in noisy conditions.

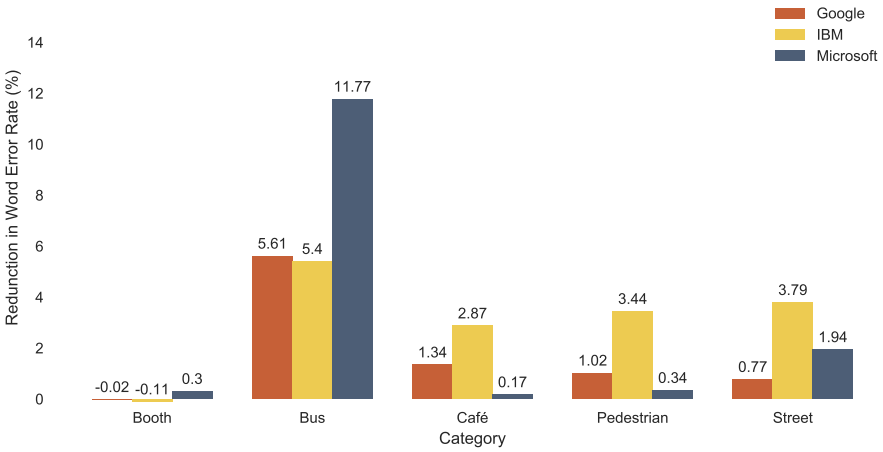


Fig. 5.3 ROVER+ Reductions in WER (%) using Five Transformations and Majority Voting on the CHiME3 Evaluation Dataset

Table 5.1 summarizes our results by aggregating all of the noisy categories and computes an overall WER. As before, we can see that modest improvements of 2.31 %, 3.88%, and 3.5% are achieved with ROVER+ for the Google, IBM, and Microsoft systems respectively. We also

found best possible improvements of 8.6%, 8.36%, and 6.51%. These results are better but consistent with the results found for the `Development` dataset.

	Google	IBM	Microsoft
Baseline	29.45	55.12	18.10
ROVER+ Best	20.85	46.76	11.59
ROVER+ Worst	39.14	58.90	22.71
ROVER+ Majority Voting	27.13	51.24	14.60
ROVER+ Best Oracle Improvement	**8.6**	**8.36**	**6.51**
ROVER+ Majority Voting Improvement	**2.31**	**3.88**	**3.5**

Table 5.1 WER (%) and Improvements (%) on Full Noisy Subset (Bus + Café + Pedestrian + Street) of the CHiME3 Evaluation Dataset using the Best Five Transformations for Each ASR

5.2 Summary and Conclusions

In this chapter, we presented our results on the `Evaluation` set of the CHiME3 dataset using the data-diverse redundant processing technique developed in earlier chapters. Our results show that our framework retains the baseline performance for clean data and shows modest but consistent improvement in noisy conditions. As before, our results also indicate that a simple voting mechanism is insufficient in exploiting the full potential to reduce WER.

Chapter 6

Discussion

In this thesis, we investigated the problem of generating complementary transcripts from a single ASR system using data diversity and combining them for reduced WER in noisy environments. In this concluding chapter, we will review the thesis and suggest possible future directions in research.

6.1 Review of Work and Conclusions

In Chapter 2, we introduced a black box model of automatic speech recognition, preliminaries, and the problem of noise-robustness in automatic speech recognition. We performed experiments on clean and noisy speech from the `Development` set of the CHiME3 dataset with the following leading cloud speech-to-text systems: Google Cloud Speech to Text, IBM Watson Speech to Text, and Microsoft Azure Speech to Text. Our results showed that all ASR systems show some degradation in performance on noisy speech in terms of WER. This degradation, however, was not consistent across ASR systems, with some systems showing more inherent robustness than others. The degradation was also not consistent across categories for different ASR systems indicating that some systems are more robust to some types of noise. We defined noise-robustness as the objective of reducing WER for noisy speech such that it is closer to the WER achieved for clean speech.

In Chapter 3, we introduced data diversity as fault tolerance mechanism, as formulated by Ammann and Knight [1]. We asserted that ASR performance deterioration can be attributed to a number of factors including the mismatch between a trained model and the testing data. We asserted that the ASR systems are more susceptible to perturbations in the input under noisy conditions due to the uncertainty they introduce to the system. We introduced the data-diverse redundant processing framework for generating diverse and complementary hypotheses and combining them. We introduced the Cross-WER metric from the work in [76] to measure diversity among hypotheses. We introduced the notion of an oracle combination of hypotheses

to show complementarity or error-correcting potential among hypotheses. Our experiments on the `Development` set of the CHiME3 dataset showed that we are indeed able to generate diverse and complementary hypotheses. Our results showed a consistent divergence in terms of Cross-WER between clean and noisy speech. We also showed that the results tend to be somewhat complementary and that an "ideal" or oracle combination of them could lead to significant decrease in WER for noisy speech and modest improvements for clean speech. We also showed that only a few transformations could account for most of the improvements.

In Chapter 4, we introduced the problem of combining multiple hypotheses. We introduced the ROVER algorithm [23] for combining multiple hypotheses. We defined the notion of best and worst paths in a confusion network as means of computing *empirical bounds* on WER. We discussed that these bounds are assessments of the confusion networks themselves rather than the ROVER scoring method based on frequency of occurrence and a confidence score. We discussed some modifications to ROVER that we expected will improve majority voting performance. Our experimental results on the `Development` set of the CHiME3 dataset showed that we were able to gain modest improvements with five transformations. The empirical bounds on the confusion networks promised even greater gains, which we asserted could be achieved by more sophisticated "voting" schemes such as the use of language models. The same experiments were repeated for the `Evaluation` set of the CHiME3 dataset in Chapter 5, which gave similar results as the `Development` set. Given that the results showed modest improvements in terms of WER and did not approach WER achieved for clean data for all systems, we conclude that data-diverse redundant processing is a viable orthogonal method for noise-robustness but its efficacy is limited by the underlying ASR and combination method. Finally, we note that the use of this approach is only encouraged when the cost of repeatedly performing speech-to-text conversion is not a concern.

6.2 Future Work

Based on the results obtained in this work, future work is suggested as follows:

- Given our black box treatment of ASR systems, we did not do any theoretical analysis of diversity. In future, we hope to explore theoretical basis of data diversity on common ASR architectures and components such as HMM-DNN acoustic models.
- In this work, the choice of transformations (e.g., time warping or amplitude normalization) was somewhat arbitrary. We hope to better optimize this in the future. Applying methods such as reinforcement learning [45, 71], genetic algorithms, and generative networks [29] for sequence-to-sequence transformations may be one avenue of exploration.
- The results for the diversity matrix (introduced in § 3.6.2) confirms our claim that ASR systems are more sensitive to perturbations in the presence of noise. We hypothesize that

these matrices could be used for performance monitoring and quality estimation [59] in automatic speech recognition i.e., we can determine if an ASR system is doing poorly without recourse to reference transcripts or confidences from the systems by simply seeing how much transcripts generated by label-preserving transformations disagree.

- In this work, we did not do any cost analysis of redundantly processing speech by ASR systems. In the real world, any additional processing will cost CPU cycles, memory, and dollars. Future work should consider such things. We concede that our approach has limited applicability and narrow use case. In practice, the cost of redundantly executing a particular ASR many times may be more than using another noise-robustness technique that provides better improvements.
- In this work, we did not look beyond WER for characterizing noise-robustness. It may be useful to empirically study the errors that were corrected.
- In this work, we did not make use of any metadata from ASR tools such as confidences, timing information, or even alternative transcriptions (N-best lists and confusion networks). These information could be invaluable in generating hypothesis spaces and finding consensus among transcriptions.
- During development, we observed that many errors in ASR are not errors where a single word is misinterpreted as another word. It is very common for a single word to be recognized as multiple words or multiple words to be recognized as a single word. The ROVER algorithm and confusion networks in general are unequipped to handle such cases. Figure 6.1a shows a how a confusion network is unable to model the fact that the word "horizon" is confused with the phrase "the rise in."

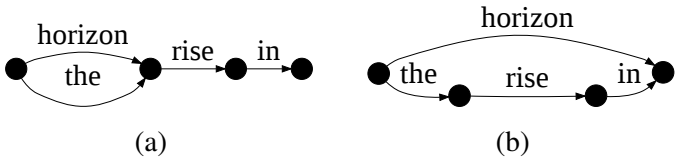


Fig. 6.1 Modified Confusion Network

The work in [78] introduces modified confusion network topology that allows for words and phrases to compete. Their work modifies the algorithm by Mangu, Brill, and Stolcke [54] that converts a word lattice into a confusion network. A more recent work by Jeon et. al [44] also introduces such a topology, which they term *heterogeneous word confusion network*. We propose that ROVER can similarly be modified to generate such modified confusion networks. Figure 6.1b shows how such a modified confusion network can model the problem of phrase confusions.

- In this work, we explored the notions of best and worst path oracles in the context of confusion networks. These oracles can be useful in bounding the possible WER achieved

by a confusion network. However, better metrics may be needed to characterize the many other possible WERs that can be achieved from a confusion network. Inspired by Monte Carlo methods, we wish to build methods around the notion of random paths through confusion networks to describe *distribution* of possible WERs.

- We were only able to achieve a fraction of the potential improvements in WER with majority voting. This observation has been made before, and a number of more powerful approaches have been explored [38, 65, 66]. In the future, we hope to explore and apply the many techniques from prior work using language modelling and machine learning to ROVER[1]. We could also look into re-scoring confusion networks with recent advances in natural language processing (NLP) such as masked language modelling (MLM) using the Bidirectional Encoder Representation from Transformers (BERT) [16].

[1]ROVER itself could possibly be replaced by an end-to-end hypothesis combination network.

References

[1] Ammann, P. E. and Knight, J. C. (1988). Data diversity: An approach to software fault tolerance. *Ieee transactions on computers*, 37(4):418–425. vi, 2, 17, 18, 36, 57

[2] Audhkhasi, K., Zavou, A. M., Georgiou, P. G., and Narayanan, S. (2013). Empirical link between hypothesis diversity and fusion performance in an ensemble of automatic speech recognition systems. In *INTERSPEECH*, pages 3082–3086. 20, 37

[3] Audhkhasi, K., Zavou, A. M., Georgiou, P. G., and Narayanan, S. S. (2014). Theoretical analysis of diversity in an ensemble of automatic speech recognition systems. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(3):711–7‘126. 12, 20, 51

[4] Avizienis, A. (1977). On the implementation of n-version programming for software fault tolerance during execution. *Proc. COMPSAC, 1977*, pages 149–155. 17, 36

[5] Bahl, L. R., Jelinek, F., and Mercer, R. L. (1983). A maximum likelihood approach to continuous speech recognition. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (2):179–190. 6

[6] Baker, J. (1975). The dragon system–an overview. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 23(1):24–29. 1

[7] Barker, J., Marxer, R., Vincent, E., and Watanabe, S. (2015). The third ‘chime’ speech separation and recognition challenge: Dataset, task and baselines. *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 504–511. 3, 9, 14, 70

[8] Berouti, M., Schwartz, R., and Makhoul, J. (1979). Enhancement of speech corrupted by acoustic noise. In *ICASSP’79. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 4, pages 208–211. IEEE. 14, 28

[9] Boll, S. (1979). Suppression of acoustic noise in speech using spectral subtraction. *IEEE Transactions on acoustics, speech, and signal processing*, 27(2):113–120. 14, 28

[10] Breiman, L. (1996). Bagging predictors. *Machine learning*, 24(2):123–140. 36

[11] Breslin, C. (2008). *Generation and combination of complementary systems for automatic speech recognition*. PhD thesis, University of Cambridge. 24, 37

[12] Burget, L. (2004). Measurement of complementarity of recognition systems. In *International Conference on Text, Speech and Dialogue*, pages 283–290. Springer. 37

[13] Cox, B., Evans, D., Filipi, A., Rowanhill, J., Hu, W., Davidson, J., Knight, J., Nguyen-Tuong, A., and Hiser, J. (2006). N-variant systems: A secretless framework for security through diversity. In *USENIX Security Symposium*, pages 105–120. 36

[14] Dahl, G. E., Yu, D., Deng, L., and Acero, A. (2011). Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Transactions on audio, speech, and language processing*, 20(1):30–42. 14

[15] Davis, K. H., Biddulph, R., and Balashek, S. (1952). Automatic recognition of spoken digits. *The Journal of the Acoustical Society of America*, 24(6):637–642. 1

[16] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*. 8, 60

[17] Dietterich, T. G. (2000). Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer. 18, 36

[18] Duda, R. O., Hart, P. E., and Stork, D. G. (2012). *Pattern classification*. John Wiley & Sons. 7

[19] Ephraim, Y. and Malah, D. (1984). Speech enhancement using a minimum-mean square error short-time spectral amplitude estimator. *IEEE Transactions on acoustics, speech, and signal processing*, 32(6):1109–1121. 28

[20] Ephraim, Y. and Malah, D. (1985). Speech enhancement using a minimum mean-square error log-spectral amplitude estimator. *IEEE transactions on acoustics, speech, and signal processing*, 33(2):443–445. 28

[21] Erickson, J. (2019). *Algorithms*. Independently Published. 67

[22] Evermann, G. and Woodland, P. (2000). Posterior probability decoding, confidence estimation and system combination. In *Proc. Speech Transcription Workshop*, volume 27, pages 78–81. Baltimore. 51

[23] Fiscus, J. G. (1997). A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (rover). In *1997 IEEE Workshop on Automatic Speech Recognition and Understanding Proceedings*, pages 347–354. IEEE. vi, 3, 12, 36, 39, 40, 41, 51, 58

[24] Flanagan, J. L. and Golden, R. (1966). Phase vocoder. *Bell System Technical Journal*, 45(9):1493–1509. 28

[25] Freund, Y. and Schapire, R. E. (1995). A desicion-theoretic generalization of on-line learning and an application to boosting. In *European conference on computational learning theory*, pages 23–37. Springer. 36

[26] Garofalo, J., Graff, D., Paul, D., and Pallett, D. (2007). Csr-i (wsj0) complete. *Linguistic Data Consortium, Philadelphia*. 9, 14, 70

[27] Goel, V., Byrne, W., and Khudanpur, S. (1998). Lvcsr rescoring with modified loss functions: A decision theoretic perspective. In *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'98 (Cat. No. 98CH36181)*, volume 1, pages 425–428. IEEE. 7

[28] Goel, V., Kumar, S., and Byrne, W. (2004). Segmental minimum bayes-risk decoding for automatic speech recognition. *IEEE transactions on Speech and Audio Processing*, 12(3):234–249. 51

[29] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680. 58

[30] Graves, A., Fernández, S., Gomez, F., and Schmidhuber, J. (2006). Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376. 8

[31] Graves, A. and Jaitly, N. (2014). Towards end-to-end speech recognition with recurrent neural networks. In *International conference on machine learning*, pages 1764–1772. 2, 8

[32] Graves, A., Mohamed, A.-r., and Hinton, G. (2013). Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. IEEE. 2, 8

[33] Gustafsson, H., Nordholm, S. E., and Claesson, I. (2001). Spectral subtraction using reduced delay convolution and adaptive averaging. *IEEE Transactions on Speech and Audio Processing*, 9(8):799–807. 28

[34] Hannun, A., Case, C., Casper, J., Catanzaro, B., Diamos, G., Elsen, E., Prenger, R., Satheesh, S., Sengupta, S., Coates, A., et al. (2014). Deep speech: Scaling up end-to-end speech recognition. *arXiv preprint arXiv:1412.5567*. 2, 8

[35] Hansen, J. H. (1996). Analysis and compensation of speech under stress and noise for environmental robustness in speech recognition. *Speech communication*, 20(1-2):151–173. 13

[36] Hermansky, H. (1990). Perceptual linear predictive (plp) analysis of speech. *the Journal of the Acoustical Society of America*, 87(4):1738–1752. 14

[37] Hermansky, H. (2013). Multistream recognition of speech: Dealing with unknown unknowns. *Proceedings of the IEEE*, 101(5):1076–1088. 14, 37

[38] Hillard, D., Hoffmeister, B., Ostendorf, M., Schlüter, R., and Ney, H. (2007). i rover: improving system combination with classification. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers*, pages 65–68. Association for Computational Linguistics. 4, 42, 51, 60

[39] Hinton, G., Deng, L., Yu, D., Dahl, G., Mohamed, A.-r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Kingsbury, B., et al. (2012). Deep neural networks for acoustic modeling in speech recognition. *IEEE Signal processing magazine*, 29. 1, 14

[40] Hirsch, H.-G. and Pearce, D. (2000). The aurora experimental framework for the performance evaluation of speech recognition systems under noisy conditions. In *ASR2000-Automatic Speech Recognition: Challenges for the new Millenium ISCA Tutorial and Research Workshop (ITRW)*. 9

[41] Hoffmeister, B., Klein, T., Schlüter, R., and Ney, H. (2006). Frame based system combination and a comparison with weighted rover and cnc. In *Ninth International Conference on Spoken Language Processing*. 51

[42] Hoffmeister, B., Schlüter, R., and Ney, H. (2008). icnc and irover: The limits of improving system combination with classification? In *Ninth Annual Conference of the International Speech Communication Association*. 51

[43] Huson, D. (2003). Algorithms in bioinformatics i. 68

[44] Jeon, W., Jordan, M., and Krishnamoorthy, M. (2019). On modeling asr word confidence. *arXiv preprint arXiv:1907.09636*. 59

[45] Keneshloo, Y., Shi, T., Ramakrishnan, N., and Reddy, C. K. (2019). Deep reinforcement learning for sequence-to-sequence models. *IEEE Transactions on Neural Networks and Learning Systems*. 58

[46] Knight, J. (2012). *Fundamentals of Dependable Computing for Software Engineers*. Chapman & Hall/CRC, 1st edition. 17, 36

[47] Kumar, N., Van Segbroeck, M., Audhkhasi, K., Drotár, P., and Narayanan, S. S. (2014). Fusion of diverse denoising systems for robust automatic speech recognition. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5557–5561. IEEE. 20, 37

[48] Kumar, S. and Byrne, W. (2002). Risk based lattice cutting for segmental minimum bayes-risk decoding. In *Seventh International Conference on Spoken Language Processing*. 51

[49] Le Roux, J., Watanabe, S., and Hershey, J. R. (2013). Ensemble learning for speech enhancement. In *2013 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 1–4. IEEE. 37

[50] Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710. 7, 67

[51] Li, J., Deng, L., Gong, Y., and Haeb-Umbach, R. (2014). An overview of noise-robust automatic speech recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(4):745–777. 14

[52] Lim, J. S. and Oppenheim, A. V. (1979). Enhancement and bandwidth compression of noisy speech. *Proceedings of the IEEE*, 67(12):1586–1604. 14

[53] Loizou, P. C. (2013). *Speech enhancement: theory and practice*. CRC press. 28

[54] Mangu, L., Brill, E., and Stolcke, A. (1999). Finding consensus among words: Lattice-based word error minimization. In *Sixth European Conference on Speech Communication and Technology*. 12, 51, 59

[55] Mikolov, T., Karafiát, M., Burget, L., Černockỳ, J., and Khudanpur, S. (2010). Recurrent neural network based language model. In *Eleventh annual conference of the international speech communication association*. 1

[56] Nadas, A. (1983). A decision theorectic formulation of a training problem in speech recognition and a comparison of training by unconditional versus conditional maximum likelihood. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 31(4):814–817. 7

[57] Nadas, A. (1985). Optimal solution of a training problem in speech recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 33(1):326–329. 7

[58] Needleman, S. B. and Wunsch, C. D. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3):443–453. 24, 26, 68

[59] Negri, M., Turchi, M., de Souza, J. G., and Falavigna, D. (2014). Quality estimation for automatic speech recognition. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1813–1823. 59

[60] Nguyen-Tuong, A., Evans, D., Knight, J. C., Cox, B., and Davidson, J. W. (2008). Security through redundant data diversity. In *2008 IEEE International Conference on Dependable Systems and Networks With FTCS and DCC (DSN)*, pages 187–196. IEEE. 36

[61] Okawa, S., Bocchieri, E., and Potamianos, A. (1998). Multi-band speech recognition in noisy environments. In *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'98 (Cat. No. 98CH36181)*, volume 2, pages 641–644. IEEE. 37

[62] Pan, Y.-C., Chang, H.-l., and Lee, L.-s. (2007). Analytical comparison between position specific posterior lattices and confusion networks based on words and subword units for spoken document indexing. In *2007 IEEE Workshop on Automatic Speech Recognition & Understanding (ASRU)*, pages 677–682. IEEE. 46

[63] Preum, S., Shu, S., Hotaki, M., Williams, R., Stankovic, J., and Alemzadeh, H. (2019). Cognitiveems: a cognitive assistant system for emergency medical services. *ACM SIGBED Review*, 16(2):51–60. 2

[64] Sakoe, H. and Chiba, S. (1978). Dynamic programming algorithm optimization for spoken word recognition. *IEEE transactions on acoustics, speech, and signal processing*, 26(1):43–49. 1

[65] Sankar, A. (2005). Bayesian model combination (baycom) for improved recognition. In *Proceedings.(ICASSP'05). IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005.*, volume 1, pages I–845. IEEE. 4, 51, 60

[66] Schwenk, H. and Gauvain, J.-L. (2000). Combining multiple speech recognizers using voting and language model information. In *Sixth International Conference on Spoken Language Processing*. 51, 60

[67] Seltzer, M. L., Yu, D., and Wang, Y. (2013). An investigation of deep neural networks for noise robust speech recognition. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 7398–7402. IEEE. 14

[68] Stolcke, A., Bratt, H., Butzberger, J., Franco, H., Gadde, V. R., Plauché, M., Richey, C., Shriberg, E., Sönmez, K., Weng, F., et al. (2000). The sri march 2000 hub-5 conversational speech transcription system. In *Proc. NIST Speech Transcription Workshop*. 51

[69] Stolcke, A., Konig, Y., and Weintraub, M. (1997). Explicit word error minimization in n-best list rescoring. In *Fifth European Conference on Speech Communication and Technology*. 7

[70] Tibrewala, S. and Hermansky, H. (1997). Sub-band based recognition of noisy speech. In *1997 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 2, pages 1255–1258. IEEE. 37

[71] Tjandra, A., Sakti, S., and Nakamura, S. (2018). Sequence-to-sequence asr optimization via reinforcement learning. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5829–5833. IEEE. 58

[72] Ulam, S. M. (1972). Some ideas and prospects in biomathematics. *Annual review of biophysics and bioengineering*, 1(1):277–292. 67

[73] Vintsyuk, T. K. (1968). Speech discrimination by dynamic programming. *Cybernetics and Systems Analysis*, 4(1):52–57. 67

[74] Vlaj, D. and Kačič, Z. (2011). The influence of lombard effect on speech recognition. *Speech Technologies*, page 151. 9

[75] Wagner, R. A. and Fischer, M. J. (1974). The string-to-string correction problem. *Journal of the ACM (JACM)*, 21(1):168–173. 10, 24, 26, 67

[76] Wong, J. H. and Gales, M. J. (2017). Multi-task ensembles with teacher-student training. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 84–90. IEEE. vi, 2, 20, 37, 57

[77] Wong, J. H. M. (2019). *Ensemble generation and compression for speech recognition*. PhD thesis, University of Cambridge. 6, 8, 20, 37

[78] Xue, J. and Zhao, Y. (2005). Improved confusion network algorithm and shortest path search from word lattice. In *Proceedings.(ICASSP'05). IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005.*, volume 1, pages I–853. IEEE. 51, 59

[79] Zhang, R. and Rudnicky, A. (2006). Investigations of issues for using multiple acoustic models to improve continuous speech recognition. 51

Appendix A

Additional Preliminaries

A.1 The Levenshtein Distance

The (minimum) edit distance, commonly known as the *Levenshtein distance*, is the minimum number of insertions, deletions, and substitutions required to transform one sequence into another another sequence. It was proposed by Vladimir Levenshtein [50] in the context of binary codes. T. K. Vintsyuk [73] also proposed such a distance metric in the context of speech recognition. It was also proposed by Stanislaw Ulam [72] in 1972 in the context of biological sequences. The Levenshtein distance between two sequences $\boldsymbol{a}$ and $\boldsymbol{b}$ with sizes $|\boldsymbol{a}|$ and $|\boldsymbol{b}|$ is:

$$\text{Levenshtein}_{\boldsymbol{a},\boldsymbol{b}}(|\boldsymbol{a}|, |\boldsymbol{b}|) \tag{A.1}$$

where the function $\text{Levenshtein}_{\boldsymbol{a},\boldsymbol{b}}(i, j)$ is defined as recursively as [21]:

$$\text{Levenshtein}_{\boldsymbol{a},\boldsymbol{b}}(i,j) = \begin{cases} \max(i,j) & \text{if } \min(i,j) = 0, \\ \min \begin{cases} \text{Levenshtein}_{\boldsymbol{a},\boldsymbol{b}}(i-1,j) + 1 \\ \text{Levenshtein}_{\boldsymbol{a},\boldsymbol{b}}(i,j-1) + 1 \\ \text{Levenshtein}_{\boldsymbol{a},\boldsymbol{b}}(i-1,j-1) + \mathbb{1}_{(\boldsymbol{a}[i] \neq \boldsymbol{b}[j])} \end{cases} \end{cases} \tag{A.2}$$

where the term $\mathbb{1}$ is the indicator function. The recursive implementation of Equation A.2 is not computationally optimal [1]. Typically, a dynamic programming (DP) solution is employed instead. The algorithm has a history of multiple invention but is often credited to Wagner and Fischer [75].

[1] The precise time complexity is not obvious.

A.2 Multiple Sequence Alignment

This section provides a brief formal introduction of multiple sequence alignment (MSA). Given a set of N sequences $\mathbf{S}$:

$$\mathbf{S} = \begin{cases} \boldsymbol{s}_1 = s_{11}, s_{12}, ..., s_{1l_1} \\ \boldsymbol{s}_2 = s_{21}, s_{22}, ..., s_{2l_2} \\ \vdots \\ \boldsymbol{s}_N = s_{N1}, s_{N2}, ..., s_{Nl_N} \end{cases} \tag{A.3}$$

A *multiple sequence alignment* of $\mathbf{S}$ is obtained by inserting *gaps* (denoted by "_") into the original sequences such that the resulting sequences all have the same length and no column consists of gaps only:

$$\mathbf{S}^* = \begin{cases} \boldsymbol{s}_1^* = s_{11}^*, s_{12}^*, ..., s_{1L}^* \\ \boldsymbol{s}_2^* = s_{21}^*, s_{22}^*, ..., s_{2L}^* \\ \vdots \\ \boldsymbol{s}_N^* = s_{N1}^*, s_{N2}^*, ..., s_{NL}^* \end{cases} \tag{A.4}$$

We see that L is at least the length of the largest sequence:

$$L \geq max(l_1, l_2, ...l_N) \tag{A.5}$$

Although an MSA is any arbitrary transformation that satisfies the above requirements, our goal typically is to find an optimal solution based on a scoring function. Different scoring functions can be defined. A discussion of these functions is beyond the scope of this thesis but can be found in [43]. Optimal MSA could be implemented by extending the pair-wise Needleman-Wunsch alignment algorithm [58] to hyper-dimensional space. This algorithm has a time complexity of $\mathrm{O}(mn)$ where m and n are the lengths of the two sequences being aligned. The computational complexity of the extended algorithm in hyper-dimensional space is $\mathrm{O}(\prod_{i=1}^{N} l_i)$, where l_i is the length of the ith sequence to be aligned. This calls for a cumbersome implementation and an unacceptable time complexity. Most applications use an approximate solution. Many approximate algorithms are freely and commercially available. We used a simple open-source method[2].

[2] https://github.com/Franck-Dernoncourt/ASR_benchmark

Appendix B

ASR Systems and Dataset

B.1 ASR Systems

We used multiple state-of-the-art leading cloud ASR systems for our development and evaluation. Below, we provide a short description of these systems and the recognition configurations used in this thesis.

Google Cloud Speech-to-Text

The Google Cloud Speech-to-Text[1] tool is a leader in cloud-based speech-to-text powered by Google's machine learning technologies. Google provides a REST API and client libraries for different programming languages. We used the Python client library and obtained transcriptions using the *synchronous* mode of the API. We used the default `en-US` model and only collected the most confident transcriptions. This setup is consistent for the entirety of this thesis.

IBM Watson Speech-to-Text

The IBM Watson Speech to Text[2] is also a cloud solution that uses deep-learning algorithms to apply knowledge about grammar, language structure, and audio/voice signal composition to create customizable speech recognition. Our recognition setup was similar to our setup with the Google Cloud Speech-to-text. We used the Python Client library and obtained transcriptions using the *synchronous* recognition mode. We used the default `en-US_BroadbandModel` mode and collected the one-best transcriptions. This setup is consistent for the entirety of this thesis.

[1] https://cloud.google.com/speech-to-text
[2] https://www.ibm.com/cloud/watson-speech-to-text

Microsoft Azure Speech-to-Text

The Microsoft Azure Speech to Text[3] is also leading cloud automatic speech recognition tool that enables real-time transcription of audio streams into text. It has replaced Microsoft's Bing Speech tool. We used the Python Speech Software Development Kit (SDK) and the obtained transcriptions using the `recognize_once` mode of recognition. This setup is consistent for the entirety of this thesis.

B.2 The CHiME3 Dataset

In this work, we make use of the CHiME3 dataset [7]. The 3rd CHiME Speech Separation and Recognition Challenge is designed around the Wall Street Journal (WSJ) [26] corpus and contains speech recorded in both noise-free and challenging noisy environments using a 6-channel tablet-based microphone array. In this thesis, we make use of the `Development` and `Evaluation` subsets of the dataset. We selected Channel 1 of the recordings.

Subset	Category	Data Points	Speakers
Development Isolated Channel 1	Booth	410	2 Male, 2 Female
	Bus (Real)	410	2 Male, 2 Female
	Café (Real)	410	2 Male, 2 Female
	Pedestrian (Real)	410	2 Male, 2 Female
	Street (Real)	410	2 Male, 2 Female

Table B.1 Summary of Development Subset of CHiME3 Dataset

Subset	Category	Data Points	Speakers
Evaluation Isolated Channel 1	Booth	330	2 Male, 2 Female
	Bus (Real)	330	2 Male, 2 Female
	Café (Real)	330	2 Male, 2 Female
	Pedestrian (Real)	330	2 Male, 2 Female
	Street (Real)	330	2 Male, 2 Female

Table B.2 Summary of Evaluation Subset of CHiME3 Dataset

[3] https://azure.microsoft.com/en-us/services/cognitive-services/speech-to-text/