Data Selection Methods for Semi-Supervised Learning in Automatic Speech Recognition

Chanho Park

Supervisor: Prof Thomas Hain

Scool of Computer Science
University of Sheffield

This thesis is submitted for the degree of
Doctor of Philosophy

Speech and Hearing Research Group January 2025

To my loving wife,
Youngkyung Kim,
who has always supported me.

To my daughter,
Dain Park,
who brings joy to my life.

And to myself,
for starting this journey in 2020.

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains fewer than 55,000 words including 29 figures, 75 tables, 82 equations, bibliography, and footnotes.

Chanho Park
January 2025

Acknowledgements

Thanks to my supervisor:
Prof Thomas Hain made me walk and talk in this field.

Thanks to my co-authors (in publication order):
Dr Rehan Ahmad, Dr Mingjie Chen, MSc Hyunsik Kang, and MSc Chengsong Lu.

Thanks to my colleagues (No specific order):
Dr Mauro Nicolao, for hiring me; Dr Md Asif Jalal, whose desk I used; Dr Rosanna Milner, for the always-enjoyable lunch breaks at the office; Dr Jose Antonio Lopez Saenz, who helped me join this research group; Dr Anna Ollerenshaw, who answered all my questions during my first year; Elaf Islam, with whom I enjoyed working; Olga Iakovenko, my ICASSP 2022 travel companion; Dr Protima Nomo Sudro, with whom I had memorable walks; Dr Muhammad Umar Farooq, my PhD cohort; Dr Shreyas Ramoji, who shared the office during my PhD; and next-generation researchers: Amit Meghanani, Xinying Wei, and Wenjie Peng. Apologies to anyone I may have forgotten—my poor memory!

Thanks to my closest family:
Youngkyung and Dain, for being with me. You have both always stood by my side. I love you.

Lastly, but not least, thanks to all my family back home and friends in South Korea:
I am doing well, and I hope to see you soon.

Abstract

For an automatic speech recognition (ASR) system to perform well on new target data, it must be trained on data from the same domain as the target. Semi-supervised learning, which integrates both manually transcribed and untranscribed data, can be beneficial in this context. However, manually transcribed data are often unavailable. This challenge can be addressed by generating a training dataset from a data pool whenever new target data are introduced.

This thesis introduces a novel approach to semi-supervised learning for ASR, focusing on selecting highly accurate ASR system-generated transcripts, referred to as ASR transcripts, and domain-relevant data from a multi-domain untranscribed data pool. To optimise ASR performance on target data, ensuring the accuracy of ASR transcripts is crucial. To this end, advanced word error rate (WER) estimation techniques are developed, including a fast WER estimation model, Fe-WER, and a multi-target regression model, MTR-ER4. These methods significantly improve both the speed and accuracy of WER estimation, making them effective for real-time applications.

Additionally, unsupervised domain similarity measurements, including acoustic and linguistic domain similarities, are proposed to enhance ASR performance by selecting data that closely match the target domain. This approach mitigates negative transfer and improves overall model robustness. To further address challenges in WER estimation, a system-independent WER estimator, SIWE, is designed to eliminate dependency on specific ASR systems, using data augmentation to simulate realistic ASR errors. For short utterances, where WER can produce high quantisation noise, an alternative error rate estimation model, Fe-CER, is explored.

By employing the methods proposed in this thesis, training data can be effectively selected from a multi-domain untranscribed data pool, leading to improved ASR performance. The findings of this research demonstrate the potential for leveraging large amounts of untranscribed data from multiple sources to enhance ASR systems.

Table of Contents

List of Figures

List of Tables

Nomenclature

Roman Symbols

$\mathbf{x}$	vector
D	data set
$d(\cdot)$	edit distance
$P(\cdot)$	probability distribution

Other Symbols

$\mathbf{M}$	matrix
$\mathcal{D}$	domain
$\mathcal{X}$	space

ADS	Acoustic Domain Similarity
AMI	Augmented Multi-party Interaction
AM	Acoustic Modelling
ASR	Automatic Speech Recognition
BERT	Bidirectional Encoder Representations from Transformers
BLEU	BiLingual Evaluation Understudy
BPE	Byte Pair Encoded
BiLSTM	Bidirectional LSTM
CER	Character Error Rate
CH5	CHiME-5
CNN	Convolutional Neural Network
CTC	Connectionist Temporal Classification
DER	Discrete unit Error Rate
DML	Deep Metric Learning
DNN	Deep Neural Network
DP	Data Pool
E2E	End-to-End
EC	Error Count
EDR	Edit Distance Ratio
ELU	Exponential Linear Unit
ER	Error Rate
FS	Fisher
GLM	Generalized Linear Model
GMM	Gaussian Mixture Model

GPT	Generative Pre-Training
GPU	Graphics Processing Unit
HMM	Hidden Markov Model
HuBERT	Hidden unit Bidirectional Encoder Representations from Transformers
HuBERT	Hidden-Unit BERT
IHM	Individual Head Microphones
KL	Kullback–Leibler
LDR	Levenshtein Distance Ratio
LDS	Linguistic Domain Similarity
LD	Levenshtein Distance
LM	Language Model
LSP	LibriSpeech
LSTM	Long Short-Term Memory
MLP	Multi-Layer Perceptron
MSE	Mean Squared Error
MTR	Multi-Target Regression
MT	Machine Translation
NCE	Noise-Contrastive Estimation
NLI	Natural Language Inference
NLP	Natural Language Processing
nRMSE	Normalised Root Mean Square Error
PCC	Pearson Correlation Coefficient
PER	Phone Error Rate
QE	Quality Estimation

RMSE	Root Mean Square Error
RNN-T	Recurrent Neural Network Transducer
RNN	Recurrent Neural Network
ReLU	Rectified Linear Unit
RoBERTa	Robustly optimized BERT pre-training approach
SIWE	System-Independent Word Error Rate Estimation
SLU	Spoken Language Understanding
SRCC	Spearman's Rank Correlation Coefficient
SSLR	Self-Supervised Learning Representation
STS	Semantic Textual Similarity
SWB	Switchboard
TDNN	Time-Delay Neural Network
TD	Tedtalks
TER	Token Error Rate
TL3	Ted-Lium 3
TL3	Ted-lium 3
WCS	Weighted Confidence Score
WER	Word Error Rate
WS0	WSJCAM0
WSJ	Wall Street Journal

Chapter 1

Introduction

Automatic Speech Recognition (ASR) is the field that concerns the conversion of speech into written words by a machine. The technologies for ASR have been developed over several decades, driven by the desire for office automation and human-machine interfaces via voice. Early ASR systems were built to recognise phonetic elements of speech, such as the digit recogniser invented by Bell Laboratories (Davis et al., 1952). With the rapid development of statistical models in the 1980s, large vocabulary, over 1000 words, speech recognition systems were implemented (Lee et al., 1990). By the late 1980s, neural networks began to be applied to ASR systems, as demonstrated by works like (Bourlard and Morgan, 1989). The introduction of hybrid models that combined statistical methods with neural networks further boosted the accuracy of ASR (Hinton et al., 2012). In the mid 2010s, End-to-End (E2E) ASR systems were reported to surpass traditional hybrid systems (Watanabe et al., 2017) and even achieve human-level performance (Amodei et al., 2016).

The development of these technologies, especially data-driven approaches to ASR, has increased the demand for larger amounts of manually transcribed data for training. However, annotation by human experts is known to be both costly and time-consuming. For example, Chapelle et al. (2006) reported that it took 400 hours to transcribe one hour of speech at the phonetic level for the Switchboard (Godfrey et al., 1992). Similarly, Gorisch and Schmidt (2024) reported that two annotators worked for 64 hours on nine hours of recordings at the segment level, averaging a factor of 7.1 (working time/audio time). Moreover, they compared the annotation time from scratch with the correction time for ASR transcripts but found no meaningful difference between them.

To address the scarcity of manual transcripts, researchers have proposed methods to exploit untranscribed data for ASR training. In this context, untranscribed data refer to speech recordings, hereafter referred to as utterances, without any accompanying transcripts. If the data are transcribed, whether automatically or manually, they consist of pairs of utterances and their corresponding transcripts. One of the early approaches utilising untranscribed data for ASR was a method to transcribe the data using an ASR system for training (Zavaliagkos et al., 1998b). The authors assumed that the amount of training data was a dominant factor in determining ASR performance. To mitigate the lack of training data, they added sufficiently accurate data to the training pool. For this purpose, they trained an ASR system using only three hours of manually transcribed utterances and then transcribed the remaining untranscribed data with the ASR system.

They argued that the performance of an ASR system trained on the ASR system-generated transcripts, hereafter referred to as ASR transcripts, could match that of a model trained on human-transcribed speech if the quantity of ASR transcripts was large enough. Inspired by their study, Kemp and Waibel (1998, 1999) used 30 minutes of manual transcripts and up to 50 hours of untranscribed data to train an ASR system. The amount of untranscribed data was further increased to several hundred hours (Lamel et al., 2001, 2002) and to several thousands of hours (Huang et al., 2013b; Long et al., 2019).

This method has been referred to as semi-supervised learning. The effectiveness of semi-supervised learning has been consistently demonstrated, even as ASR architectures have changed. It has been applied to ASR with Deep Neural Networks (DNNs) (Thomas et al., 2013; Zhang et al., 2014) and with transformers (Xu et al., 2021; Zhang et al., 2022, 2020b). Additionally, semi-supervised learning for ASR has often been integrated with iterative processes to improve model performance for domain adaptation (Yang et al., 2023) and robust speech recognition (Li et al., 2023). Recently, this iterative approach has also been referred to as self-training (Chen et al., 2020b; Georgescu et al., 2021; Kahn et al., 2020a) or iterative pseudo-labelling (Higuchi et al., 2021; Xu et al., 2020). For clarification, in this thesis, semi-supervised learning for ASR refers to a method that uses both manually and automatically transcribed data for training an ASR system, while iterative semi-supervised learning will be referred to as self-training.

One assumption of semi-supervised learning for ASR is the high accuracy of transcripts. Huang et al. (2013b) observed that the *quality* of transcription—though the quality is a subjective measurement, they quantified it using error rate at the phonetic level, referred to as Phone Error Rate (PER)—affected the performance gap between ASR systems with supervised and semi-supervised learning. For example, when the PER of ASR transcripts used for training was 6%, the performance gap between the ASR systems was 6%. The gap increased to 13% as the PER increased to 16%. This suggests that the accuracy of ASR transcripts plays a crucial role in determining the effectiveness of semi-supervised learning for ASR. In other words, a lower error rate in ASR transcripts is essential for achieving performance comparable to that of fully supervised models. In many studies (Chen et al., 2023, 2020b; Georgescu et al., 2021; Khurana et al., 2021; Li et al., 2023; Park et al., 2020; Xu et al., 2020), the error rate at the word level, referred to as Word Error Rate (WER), of ASR transcripts ranged from 1.88% to 27.76% in semi-supervised learning for ASR, although there were a few successful experiments where the initial WER of the ASR transcripts were relatively high, such as 64.1% (Kahn et al., 2020a) and 37.02% (Khurana et al., 2021).

To ensure the high accuracy of ASR transcripts in training data, they are necessarily selected using various methods. Zavaliagkos et al. (1998b) selected ASR transcripts based on word-level confidence estimation and thresholding. Since then, many data selection methods have been based on confidence estimation, such as confidence scores on word and phoneme occurrence (Thomas et al., 2013), frame-/word-/utterance-level confidence scores (Long et al., 2019), an ensemble of confidence scores with multiple systems (Huang et al., 2013b) and a distribution filter based on confidence scores (Li et al., 2023). For E2E systems, log probability has been used for data selection, such as length-normalised log likelihood (Kahn et al., 2020a) and a token-normalised score (Park et al., 2020). Alternatively, ASR transcripts have been compared with available text, such as closed-captions (Lamel et al., 2001) or other textual information (Georgescu et al., 2021).

Among these methods, confidence-based scoring has been popular due to the convenience of implementation and established baselines. However, Georgescu et al. (2021) reported that overconfident data did not improve the performance of an ASR system and argued that ASR transcripts with low confidence scores but correct transcripts were beneficial as supported by their results. Considering that confidence scores are derived from ASR decoding, these results indicate that an accuracy estimation method for an ASR system's output, hereafter referred to as ASR output, is necessary and should be independent of ASR decoding to avoid this overconfident scoring issue. For this purpose, Negri et al. (2014) proposed a Quality Estimation (QE) method for ASR—where, as previously mentioned, quality is interchangeable with accuracy and measured using WER. They categorised signal and textual features into two types based on their dependency on information obtained from ASR decoding. The first type is glass-box features, which are internal information from ASR, such as word-level confidence. The second type is black-box features, which are obtained externally to the ASR system, such as silences per second. Their results showed that black-box features could also be used to predict the WER of ASR output, though they did not surpass the performance achieved with glass-box features. This work inspired subsequent research to utilise black-box features for QE using WER, hereafter referred to as WER estimation, in ASR. Ali and Renals (2018) aimed to predict WER for ASR output's quality using an Multi-Layer Perceptron (MLP). The idea was further developed by adopting Convolutional Neural Networks (CNNs) in later research (Ali and Renals, 2020) and Self-Supervised Learning Representations (SSLRs) (Chowdhury and Ali, 2023).

Until Chowdhury and Ali (2023) proposed the E2E Multi-lingual WER estimator, referred to as e-WER3, the performance of WER estimation models using black-box features had not surpassed that of models using glass-box features. e-WER3 outperformed their previous models (Ali and Renals, 2018, 2020) by relying solely on black-box features for WER estimation tasks. However, the model utilised bi-directional context from frame-level representations, which required significant computational resources as the length of the audio increased. This might explain why it received little attention as a method for data selection in semi-supervised learning for ASR.

Despite this limited attention, WER estimation methods have several advantages over confidence-based methods. First, these models predict the accuracy of ASR transcripts, including deletion errors. Words deleted in an ASR transcript—or, in other words, not generated by the ASR system—cannot have confidence scores computed by the system. Second, WER is estimated for paired utterances and ASR transcripts without ASR decoding, which is particularly useful when data are collected from the internet. Third, because access to ASR decoding is no longer required, the performance of the WER model can be optimised independently of the specific ASR system. Therefore, in this thesis, WER estimation is investigated through Chapter 3, 4 and 7.

After estimating the WER of ASR transcripts, data are selected if their transcripts are estimated to have a low WER. Once transcripts with low WER are obtained, the performance of the ASR system is expected to improve as the amount of data increases through semi-supervised learning. However, a low WER of ASR transcripts is not enough for the performance improvement. What if the data selected for semi-supervised learning have different characteristics, such as recording conditions or speech content, from the target data? In literature, Doulaty et al. (2015) demonstrated performance degradation in an ASR system when there was a domain mismatch between training and test data, such as telephone speech and read speech. Although their experiment was conducted in a supervised manner, the same principle applies to semi-supervised learning (Chen et al., 2023; Georgescu et al., 2021). For example, Georgescu et al. (2021) reported that the domain mismatch between untranscribed data and target data was not helpful to improve the performance of an ASR system by semi-supervised learning with a confidence-based method. Therefore, the domain of the data selected for semi-supervised learning should match that of the target data.

The challenge in the domain mismatch issue lies in defining the domain of data, as it could be influenced by various factors like noise type, speaker characteristics, emotions or the topic of speech. Additionally, selecting data from the same domain as the target domain remains challenging. However, with these issues, the effectiveness of semi-supervised learning is limited to utterances within a homogeneous domain and the application of this learning method is also restricted. By allowing the use of untranscribed data from multiple domains, the potential for semi-supervised learning can be greatly expanded. Therefore, this thesis investigates a method selecting data that considers the domain of target data in semi-supervised learning for ASR through Chapter 5 and 6.

So far, semi-supervised learning for ASR and its major challenges have been introduced. Semi-supervised learning for ASR is a method for improving the performance of an ASR system by using ASR transcripts in the training process. One critical factor in improving ASR system performance through semi-supervised learning is the accuracy of ASR system-generated transcripts, referred to as ASR transcripts, used for training, as higher transcript accuracy directly enhances system performance. To mitigate the dependency on ASR decoding, WER estimation with black-box features will be investigated. Moreover, the performance gains from semi-supervised learning could be limited by domain mismatches when an untranscribed data pool consists of multiple domains. To avoid this phenomenon, data for semi-supervised learning must be carefully selected to ensure a domain match between the source and target data. Henceforth, data that closely matches a target domain will be referred to as domain-relevant data.

1.1 Research Questions

The research questions are:

- Is WER estimation without ASR decoding effective in semi-supervised learning for ASR?
- How can the similarity between an utterance and a set of target data be measured?
- Do Low-WER and Domain-Relevant Data Selection Methods complement each other in semi-supervised learning for ASR?

1.2 Contributions

The following lists the contributions of this thesis:

- **Transcript Information in Utterance Representations:** This work investigates whether transcript information is retained in a speech representation at the utterance level, referred to as an utterance representation hereafter. Experimental results demonstrate a strong correlation between the edit distance of transcripts and the distance between utterance representations after learned through deep metric learning. A novel multi-distance N-pair sampling technique is proposed, allowing stable convergence with N-pair loss by drawing samples from multiple distance groups. Additionally, an utterance-transcript matching method is introduced to generate paired training data from unpaired utterances and transcripts using bidirectional normalised temperature-scaled cross entropy loss (Chapter 3).

- **Fast Word Error Rate Estimation (Fe-WER):** This thesis introduces Fe-WER, a model for fast WER estimation using average pooling aggregation. Including inference time, it is compared against three baselines: deep metric learning for WER estimation, confidence-based WER estimation and e-WER3 Chowdhury and Ali (2023). Extensive experiments with 16 combinations of self-supervised learning representations demonstrate Fe-WER's effectiveness in both performance and inference time. Part of this work has been accepted for presentation at the 2025 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2025). Furthermore, the Multi-Target Regression for Four Error Rates (MTR-ER4) model is proposed to estimate not only WER but also detailed error rates, including substitution, insertion, and deletion (Chapter 4).
- **Unsupervised Domain Similarity Measurement for ASR:** This thesis defines and measures Acoustic Domain Similarity (ADS) and Linguistic Domain Similarity (LDS) for ASR. An experiment utilising ADS demonstrates that training data in the target domain can be selected from a multi-domain data pool in an unsupervised manner and this work has been published in ICASSP 2022. Additional experiments focus on hyper-parameter tuning for LDS (Chapter 5).
- **Low-WER and Domain-Relevant Data Selection Method:** A novel data selection method for semi-supervised learning in ASR is proposed to specifically address domain mismatch between untranscribed training data and target data. This method employs the MTR-ER4 model and ADS for data selection from a multi-domain data pool. Experimental results demonstrate the effectiveness of WER estimate from MTR-ER4 in selecting low-WER ASR transcripts and show that the ASR performance improves when domain-mismatched data are filtered using ADS. To the best of my knowledge, this is the first investigation of semi-supervised learning for ASR using WER estimation with black-box features (Chapter 6).

- **Robust Error Rate Estimation for Automatically Transcribed Data:** This thesis proposes an ASR System-Independent Word Error Rate Estimation (SIWE) method to reduce performance degradation when ASR transcripts are generated by different ASR system. A novel data generation method is proposed using acoustic similarity between phoneme sequences and linguistic probability of words. Additionally, a Character Error Rate (CER) estimation method for short utterances is proposed to reduce the performance degradation observed in WER estimation on such utterances. Normalised Root Mean Square Error (nRMSE) is suggested as an new evaluation metric to ensure fair comparison between PER, CER, WER and Token Error Rate (TER) estimation models. These works have been published in the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024) (Park et al., 2024a) and the 32nd European Signal Processing Conference (EUSIPCO 2024) (Park et al., 2024b) (Chapter 7).

While the thesis primarily focuses on using WER estimation in the context of semi-supervised learning and data selection, it is important to note that these WER estimation models have broader potential applications. Beyond selecting training data for ASR, they could be applied to ASR system evaluation, transcription quality control, post-editing ASR transcripts and other areas of speech processing. These broader applications, however, are not explored within the scope of this thesis and are left for future work.

1.3 Thesis Outline

The structure of this thesis is designed to address the research questions aimed at improving the performance of an ASR system through semi-supervised learning with data selection techniques. Chapter 2 introduces background information and related works. Chapter 3 explores the possibility of utilising the transcript information retained in speech representation to measure the accuracy of ASR transcripts by learning the edit distance between utterance and transcript representations. In Chapter 4, various WER estimation models are proposed to estimate the accuracy of ASR transcripts by deep metric learning, average pooling aggregation and multi-target regression. To address the challenge of selecting untranscribed data from the same domain as the target domain, a domain for ASR is defined acoustically and linguistically and measured in an unsupervised manner in Chapter 5. Chapter 6 integrates Low-WER and Domain-Relevant Data Selection Methods, exploring their complementary effects in a semi-supervised manner to enhance ASR performance. Chapter 7 explores methods for generalising the WER estimation model. Finally, the concluding chapter synthesises the findings from each of the previous chapters, and discusses the implications of the results for future ASR research and applications.

Chapter 2

Background and Related Works

This chapter explores the fundamental concepts and recent advances in semi-supervised learning for Automatic Speech Recognition (ASR), data selection methodologies, deep metric learning, representation learning, and Word Error Rate (WER) estimation. It begins by examining the various terminologies used in semi-supervised learning for ASR and introduces an algorithm for semi-supervised learning with automatic transcription. Through detailed discussions of techniques for representation learning at the utterance level and transcript accuracy estimation in ASR, this chapter lays the groundwork for understanding data selection methods aimed at enhancing ASR systems through semi-supervised learning.

2.1 Semi-Supervised Learning for Automatic Speech Recognition

2.1.1 Definition of Terms

Semi-supervised learning utilises both labelled and unlabelled data for training. Primitive ideas of this utilisation can be found under the term *self-taught* in some literature. For example, Scudder (1965) suggested a self-taught learning machine for an adaptive communication receiver.

> The gremlin receiver is a taught-learning machine since, after it makes a decision, a gremlin tells it what the correct decision was. The decision-directed receiver is a self-taught learning machine, using its own output instead of a gremlin's.

Here, the gremlin represents a supervised model, while the author's proposal was a model trained on the output of the supervised model, so-called pseudo-labels. Similarly, a mathematical model for pattern recognition without external aid was suggested (Fralick, 1967).

> It is possible to use this sequence to "learn" the values of the unknown parameters, even when the correct classification of any one (or all) of the members of the sequence is unknown. Machines which accomplish this task are said to "learn without a teacher." (When the correct classification of the learning sequence is the machine learns with a teacher.)

Fralick also emphasised two characteristics of the model: the optimality of the system depended on the accuracy of the model and its repetition led to an adaptive system. A widely known example of the early work on this idea is (Yarowsky, 1995) in the 1990s. Yarowsky (1995) proposed a system for disambiguating word senses in untagged corpora using an unsupervised algorithm. Contrary to the self-taught methods above, they not only used seed sets but also iteratively labelled data and trained classifiers. This iterative process of training has been more developed and categorised as self-training (Chapelle et al., 2006).

In the ASR field, Zavaliagkos et al. (1998a) observed that the performance of an acoustic model improved by an absolute 5% when the data size increased eightfold. This observation inspired Zavaliagkos et al. (1998b) to introduce a method using untranscribed training data to build an ASR system for a new language, where only a small amount of transcribed training data were available. They assumed three types of data for training: a text corpus, a few hours of speech and much more untranscribed data. An ASR model was built and evaluated on CALLHOME Spanish[1]. They used three hours of transcribed speech and the remaining 50 hours of speech in the corpus as untranscribed training data. The untranscribed speech was transcribed by the seed model trained on the three hours of reference transcripts. The data, pairs of utterances and transcripts, were filtered by a confidence score and three hours of data remained for training. Despite their errors, the results showed a performance gain from the ASR system-generated transcripts, referred to as ASR transcripts. This research was followed by (Kemp and Waibel, 1998, 1999). Using 30 minutes of initial training data, they found that a reliable measure of confidence was essential. Moreover, Ma and Schwartz (2008) investigated the performance gap between acoustic model training using untranscribed data and training with manually transcribed data, a concept referred to as WER recovery. On a conversational telephone speech corpus, the WER recovery was from 59.2% to 80.8% as the amount of untranscribed data increased, using one hour of manual transcripts. Building on this experiment, (Novotney et al., 2009) proposed an unsupervised language model training for ASR by combining n-gram counts with their confidence scores within a semi-supervised learning scenario.

Semi-supervised learning for ASR has been continuously investigated, often referred to as self-training (Kahn et al., 2020a), pseudo-labelling (Moritz et al., 2021), student

[1]https://catalog.ldc.upenn.edu/LDC96S35

training (Chen et al., 2023) or unsupervised learning (Mai and Carson-Berndsen, 2022). Among these, self-training and pseudo-labelling have been actively studied in ASR. First, self-training is a semi-supervised learning method focusing on the iterative process. This method has been widely adopted for low-resource language (Singh et al., 2023), domain adaptation (Khurana et al., 2021) and self-supervised learning representation (Hsu et al., 2021). It is sometimes applied to ASR training with large datasets, such as one million hours data (Krishnan Parthasarathi and Strom, 2019; Radford et al., 2023). Second, pseudo-labelling focuses more on how to utilise the pseudo-labels for learning. For example, Zhu et al. (2023) introduced a method providing binary tokens in the positions of incorrect pseudo-labels for semi-supervised learning.

In terms of utilising both manually transcribed and untranscribed data, there are more semi-supervised learning methods available beyond simply training an ASR system with ASR transcripts. For example, cycle-consistency training (Hori et al., 2019) employed a method to convert speech representation into a sequence of characters, then reconstruct speech representation from the character sequence. The cycle-consistency loss between speech representation and the reconstructed representation is used to update the ASR model's parameters. Another approach combines a self-supervised learning method for speech representation with supervised learning for ASR training, hereafter referred to as pre-training and fine-tuning, respectively. Self-supervised learning is a method that generates labels using information derived from the input itself. Although it is supervised learning from a methodological aspect, it can also be defined as unsupervised learning from a data utilisation aspect. This self-supervised learning method is more widely applied to learning representation in ASR (Mohamed et al., 2022) than training an ASR model. When self-supervised learning is combined with ASR training, a representation model is trained on untranscribed data, while an ASR system is trained on manually transcribed data. Therefore, the entire process of self-supervised learning and ASR training can be regarded as semi-supervised learning in a broad view, as the process involves both manually transcribed and untranscribed data; however, self-supervised learning itself is not considered as a semi-supervised learning method in this research.

Before delving into semi-supervised learning for ASR, it is necessary to clarify the terminologies used in literature. Terms as pseudo-labels and labels are used with varying definitions in ASR. For example, automatically generated transcriptions, tentative hypotheses (Zavaliagkos et al., 1998b), speech recogniser's transcriptions (Kemp and Waibel, 1998), approximate transcriptions, approximately labelled training data (Lamel et al., 2001), hypothesised transcriptions (Lamel et al., 2002), erroneous transcriptions (Huang et al., 2013b), imperfect transcriptions (Zhang et al., 2014), generated transcripts (Ulasik et al., 2020) are used in the place of pseudo-labels, while humanly transcribed data (Zavaliagkos et al., 1998b), manual transcriptions, manually generated transcriptions (Kemp and Waibel, 1998), manually transcribed training data (Lamel et al., 2001), exact transcriptions, manual transcripts, orthographic transcriptions, accurate speech transcriptions (Lamel et al., 2002), human transcription (Huang et al., 2013b), gold standard transcription (Ulasik et al., 2020), careful transcriptions, golden transcription (Long et al., 2019) were used in the place of labels. In this thesis, pseudo-labels and labels will be referred to as *ASR transcripts* and *reference transcripts*, respectively.

2.1.2 Automatic Speech Recognition Systems

ASR systems have made tremendous progress in recent years, with significant differences in many ways, including model architectures, input features and training objectives (Li, 2022). One type of ASR system is based on hybrid modelling (Povey et al., 2015; Swietojanski et al., 2013), consisting of multiple independent modules with a training objective, such as lattice-free maximum mutual information (LF-MMI) (Povey et al., 2016). End-to-end (E2E) ASR systems have gained much attention for their simplified architectures. For example, to make End-to-End (E2E) models robust to long contexts, Recurrent Neural Network Transducer (RNN-T) Graves (2012) has been adopted by He et al. (2019). Its encoder is jointly trained with prediction networks that rely on previous labels.

Additionally, Transformer-based models have been integrated into E2E systems, including wav2vec 2.0 (Baevski et al., 2020), Conformer (Gulati et al., 2020) and Whisper (Radford et al., 2023). Wav2vec 2.0 is pre-trained to learn contextualised representations. The model is optimised by minimising the Connectionist Temporal Classification (CTC) loss. The Conformer model, a hybrid of Transformer and Convolutional Neural Networks (CNNs) for feature extraction and sequence modelling. Finally, Whisper is an ASR system trained on large-scale transcripts sourced from the internet, designed for multiple tasks, such as speech recognition and language identification.

2.1.3 Semi-Supervised Learning with Automatically Generated Transcripts

The basic concept of semi-supervised learning is to train a model with automatically transcribed data under the assumption that there are few errors in the transcripts. To maintain the errors as few as possible, data are often filtered based on a measurement for data selection. The initial model, hereafter referred to as a seed model, trained using reference transcripts and used to generate automatically transcribed data. Confidence scores are then used to filter the data, with the selected data being added to the training dataset. This can be formalised as an algorithm as follows:

Algorithm 1 Algorithm of semi-supervised learning with automatically transcribed data.

```
data: Manually transcribed data $D_M$, Untranscribed data $D_U$
result: Acoustic Model $A_\theta$
Initialise $A_\theta$ by training it with $D_M$
repeat
    Transcribe $D_U$ using $A_\theta$ to generate $D_A$
    Select $D_S$ from $D_A$ using a data selection measurement
    $D_T = \{D_M \cup D_S\}$
    Train $A_\theta$ with $D_T$
until $A_\theta$ converges or the maximum number of iterations is reached
```

2.2 Data Selection for Automatic Speech Recognition

2.2.1 Data Selection for Semi-Supervised Learning

Semi-supervised learning for ASR has been explored in both in-domain (Chen et al., 2020b; Kahn et al., 2020a; Li et al., 2023; Park et al., 2020; Xu et al., 2020) and out-of-domain scenarios (Chen et al., 2023; Georgescu et al., 2021; Zhu et al., 2023). In the in-domain scenario, where the source and target data are from the same domain, the objective is to leverage ASR transcripts to achieve performance comparable to that of fully supervised models, particularly in low-resource settings. In the out-of-domain scenario, researchers aim to surpass the performance of fully supervised models by using additional data from different domains. Across these scenarios, various data selection methods have been employed to minimise errors in the ASR transcripts. These methods include confidence-based measurement (Kahn et al., 2020a), uncertainty estimation (Khurana et al., 2021), approximate transcription (Georgescu et al., 2021), language model filtering (Chen et al., 2023) and filtering score (Park et al., 2020).

2.2.1.1 Confidence Score for Word Spotting

In the 1990s, confidence score methods were based on scoring methods for word spotting (Jeanrenaud et al., 1993; Siu et al., 1997; Weintraub, 1995). As an example of confidence scoring, Jeanrenaud et al. (1993) employed posterior probability scoring, which was utterance-based and combined forward and backward scores, $\alpha(s,t)$ and $\beta(s,t)$, respectively, in the Baum-Welch algorithm (Baum and Petrie, 1966). The posterior probability $p(w,t)$ of being in state e_w at time t is given by:

$$p(w,t) = \frac{\alpha(e_w,t)\beta(e_w,t)}{\sum_{\forall s}\alpha(s,t)\beta(s,t)} \tag{2.1}$$

where s represents a state in the Hidden Markov Model (HMM) and e_w is the last state of word w. The posterior probability score (PPS) is the sum of the posterior probability during the time that the word is spoken:

$$\text{PPS}(w,\tau) = \sum_{t=\tau-d/2}^{\tau+d/2} p(w,t) \tag{2.2}$$

where d represents the approximate duration of the word and τ is the approximate time of ending of the word. Weintraub (1995) used log-likelihood ratio scoring to compute the likelihood of a keyword w in the i-th hypothesis from the n-best list:

$$S_{nb}(w) = \frac{\sum_{i \text{ if } w \in nb_i} p(nb_i)}{\sum_i p(nb_i)} \tag{2.3}$$

where $p(nb_i)$ is the likelihood of the i-th hypothesis from the n-best list and i if $w \in nb_i$ is the indexes of hypotheses including w in n-best hypotheses. Siu et al. (1997) proposed a Generalized Linear Model (GLM) to estimate confidence in the words generated by an ASR system. They exploited four types of features: N-best scoring, language modelling information, acoustic information and context features. They evaluated the effectiveness of features using normalised mutual information, which measured the relative change in uncertainty in estimating correctly recognised words. They demonstrated that the correlation coefficient between predicted error and recognition error was 0.87 on Switchboard (SWB) (Godfrey et al., 1992) when all features were used. The GLM was adopted by Zavaliagkos et al. (1998b) for semi-supervised learning in ASR. The process was as follows:

1. Build an initial model using the transcribed data available
2. Transcribe all the untranscribed data using the model
3. Estimate a confidence score at the word level
4. Discard words below a threshold of the confidence score
5. Add viable transcripts to the training data set
6. Re-train the initial model

An example of confidence scoring at the word level and thresholding for word selection is described in Table 2.1.

Table 2.1 An example of selection procedure using a confidence score from (Zavaliagkos et al., 1998b). The threshold was 0.8 and only words w2, w3 and w5 were kept.

hypothesis	SIL	w1	w2	w3	w4	SIL	w5	w6	SIL
start frame	0	0	21	42	57	63	69	81	101
confidence		.15	.83	.91	.67		.9	.3	

2.2.1.2 Confidence Score with Connectionist Temporal Classification Segmentation

CTC was proposed to label unsegmented sequence data (Graves et al., 2006). In this section, confidence scoring at the word level with forced alignment using CTC segmentation is described.

2.2.1.2.1 Connectionist Temporal Classification Let $X = \{x_1, x_2, \dots, x_T\}$ be the input sequence and let $Y = \{y_1, y_2, \dots, y_J\}$ be the label sequence. A label for a blank ε, which does not belong to any other label, is added to the set $L = \{l_1, l_2, \dots, l_I\}$. Therefore, $L' = L \cup \{\varepsilon\}$. With ε, the outputs are segmented without collapsing the labels. For example, if the output is $sspee\varepsilon e\varepsilon\varepsilon ccch$, it becomes $spe\varepsilon e\varepsilon ch$ with the removal of repetition. With the removal of ε, it becomes *speech*.

Let $A_{X,Y}$ be the set of all possible alignments with the output. The CTC conditional probability is defined as follows:

$$p(Y|X) = \sum_{A \in A_{X,Y}} \prod_{t=1}^{T} p_t(a_t|X) \tag{2.4}$$

where $A = \{a_1, a_2, ..., a_t\}$, p_t is the probability of the current state a_t at time step t. For training, the loss function is defined as follows:

$$\sum_{(X,Y) \in D} -\log p(Y|X) \tag{2.5}$$

For inference, Y is selected to maximise the conditional CTC probability:

$$Y = \arg\max_{Y} p(Y|X) \tag{2.6}$$

Let α be the score of the merged alignments. For example, $\alpha_{j,t}$ is the CTC score of the subsequence $Y_{1:j}$ at time step t. When $y_j = \varepsilon$ or y_{j-1} is ε and occurs between repeated labels:

$$\alpha_{j,t} = (\alpha_{j-1,t-1} + \alpha_{j,t-1})p_t(y_j|X) \tag{2.7}$$

In the case where $y_{j-1} = \varepsilon$ and occurs between unique characters:

$$\alpha_{j,t} = (\alpha_{j-2,t-1} + \alpha_{j-1,t-1} + \alpha_{j,t-1})p_t(y_j|X) \tag{2.8}$$

2.2.1.2.2 Forced Alignment with Visualisation This section describes how to compute a confidence score at the word level with a visualised example. The plotting functions from PyTorch[2] are used for visualisation. The audio file used in this example was sampled from a Wall Street Journal (WSJ) training dataset and its reference transcript for scoring after pre-processed was:

HIS|SUCCESSOR|HAS|NOT|BEEN|NAMED.

The first step is to generate the label probability of each frame. The labels are the tokens used to train the acoustic model. Here is an example of the set of tokens used for training an acoustic model: {'|', 'E', 'T', 'O', 'A', 'H', 'I', 'N', 'S', 'U', 'R', 'L', 'D', 'Y', 'W', 'M', 'G', 'C', 'F', 'B', 'K', 'P', 'V', 'J', 'X', 'Q', 'Z'} where '|' is a token for a word boundary. In addition to the token set, a blank token ε is added for the CTC algorithm. The probability p_t, defined in the previous section, is referred to as the emission probability for the labels at time t and is represented as a matrix $\mathbf{M}_e$ containing its log values.

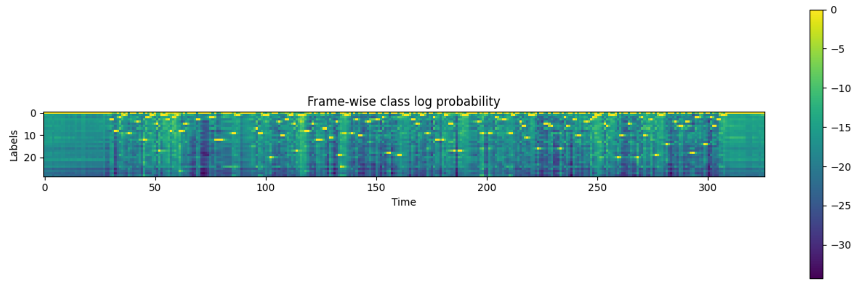


Figure 2.1 Frame-wise class log probabilities for token emissions in a CTC-based ASR model. The heatmap visualises the emission probabilities of different labels over time, where brighter colours indicate higher probabilities. The blank token is included for alignment in the CTC algorithm.

[2]https://pytorch.org/audio/0.13.1/tutorials/forced_alignment_tutorial.html

From the emission probability, $\alpha_{j,t}$, referred to as the probability of a token sequence occurred at each time frame t, is represented using another matrix $\mathbf{M}_\tau$ containing its log values. Let y_j be the j-th label. To generate the probability of time step $t+1$, the emission at time step $t+1$ and the probability of the current time step t are considered. The first path to $t+1$ is to stay at the same label y_j:

$$\mathbf{M}_\tau[t+1, j+1] = \mathbf{M}_\tau[t, j+1] + \mathbf{M}_e[t, \text{index}(\varepsilon)] \tag{2.9}$$

where index$(\cdot)$ is the function that outputs the index of the input label. The second path is to transition to the next label:

$$\mathbf{M}_\tau[t+1, j+1] = \mathbf{M}_\tau[t, j] + \mathbf{M}_e[t, \text{index}(y_{j+1})] \tag{2.10}$$

Therefore, $\mathbf{M}_\tau[t+1, j+1]$ is defined as follows:

$$\mathbf{M}_\tau[t+1, j+1] = \max(\mathbf{M}_\tau[t, j+1] + \mathbf{M}_e[t, \text{index}(\varepsilon)], \mathbf{M}_\tau[t, j] + \mathbf{M}_e[t, \text{index}(y_{j+1})]) \tag{2.11}$$

After generating $\mathbf{M}_\tau$, the most likely path is found by backtracking from the last label. The label changes when $\mathbf{M}_\tau[t-1, j-1] + \mathbf{M}_e[t-1, \text{index}(y_j)]$ is greater than $\mathbf{M}_\tau[t-1, j] + \mathbf{M}_e[t-1, \text{index}(\varepsilon)]$. When the label changes at time t, $(t-1, j-1)$ is added to the path list.

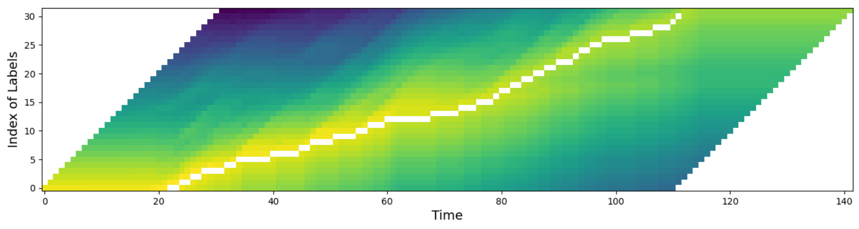


Figure 2.2 Example of a most likely path computed by backtracking from the emission probability matrix. The heatmap represents the log probabilities of token emissions over time, while the white path illustrates the selected sequence of labels.

Then, the probability is averaged along the path at the token level:

$$p(j) = \frac{1}{|T_{y_j}|} \sum_{t \in T_{y_j}} \mathbf{M}_\tau[t, j] \tag{2.12}$$

where T_{y_j} is the set of time steps belong to y_j. The path labelled with the probability is illustrated in Figure 2.3.

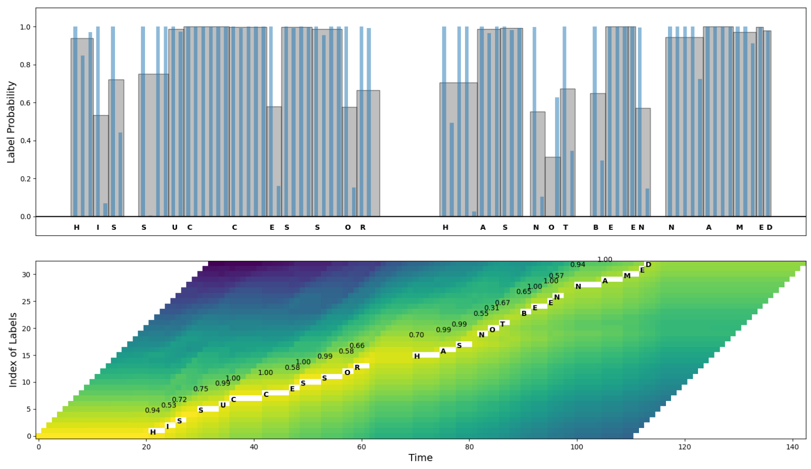


Figure 2.3 Illustration of a path and probability for each label. The top plot shows the label probabilities over time, while the bottom plot visualises the most likely sequence of labels computed through backtracking.

Finally, the audio is segmented using the word boundary token. The confidence score at the word level is computed by averaging the probability of the labels for each word:

$$\text{Confidence Score}_{w_i} = \frac{1}{|L_{w_i}|} \sum_{j \in L_{w_i}} p(y_j) \tag{2.13}$$

where L is a set of indexes of the labels belonging to the word w_i. The results of the forced alignment are shown in Figure 2.4.

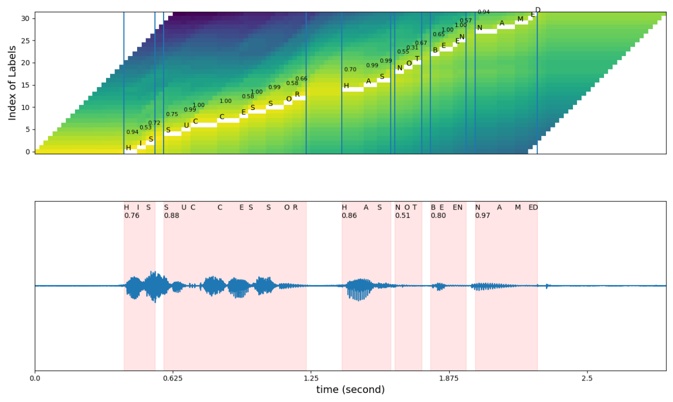


Figure 2.4 Example of forced alignment. The top plot shows the alignment path with label probabilities, while the bottom plot represents the segmentation of the audio waveform based on the computed alignment.

2.2.2 Data Selection for Acoustic Domain Match

Data Selection methods for matching acoustic domains have been investigated in tasks such as speaker diarisation and speech processing. In speaker diarisation, cosine similarity between speaker embeddings has been used to measure the similarity of utterances for domain identification (Kumar et al., 2022), such as in audiobooks and broadcast interviews. In speech processing tasks, Wu et al. (2021) proposed a method leveraging the similarity between different spoken languages for cross-lingual transfer. They employed CNNs to extract features from spoken utterances for language classification. In ASR, to reduce domain mismatch between source and target data, confidence score-based methods have been proposed (Afshan et al., 2021; Zhang et al., 2014). Moreover, Žmolíková et al. (2016) proposed a summary vector to represent the noise condition of an utterance. The cosine distance between the summary vectors was used to measure the domain similarity between the utterances and to select data with respect to similarity of acoustic conditions for ASR training. Furthermore, Gody and Harwath (2023) proposed a data selection method for fine-tuning by selecting data based on pre-training loss and the perplexity of byte pair encoded clustered units.

2.2.3 Data Selection for Linguistic Domain Match

Lu et al. (2022) proposed a method to identify linguistically similar speech data for the target domain, thereby reducing the need for extensive pre-training data. This method employed quantisation techniques to transform continuous speech data into discrete tokens, selecting data matched to the target domain based on their similarities using contrastive data selection. Additionally, linguistic characteristics and domain-specific features have been captured using K-nearest neighbour selection component to further identify target domain data (Lagos and Calapodescu, 2024).

2.2.4 Submodular Function Maximisation

Selecting data from a data pool is a problem of finding discrete sets of feasible solutions. This problem can be solved by adopting a submodular function (Krause and Golovin, 2014) for the set:

$$f : 2^V \to \mathbb{R} \tag{2.14}$$

where V is a finite set, 2^V is the set of all subsets of V and $f(\varnothing) = 0$. A function f is submodular if $f_A(e) \geq f_B(e)$ for all $A \subseteq B \subseteq V$ and $e \in V \backslash B$ where $f_A(e) = f(A \cap \{e\}) - f(A)$.

If the function is concave, the optimal solution S is a set that maximises the value of the submodular function:

$$\max\{f(S) : S \subseteq V\} \tag{2.15}$$

To solve this problem, a greedy solution can be used. However, it is NP-hard and requires non-deterministic polynomial time. One possible approach to avoid this complexity is for the function to be monotonic.

$$f(A) \leq f(B), \text{ if } A \subseteq B \subseteq V \tag{2.16}$$

With monotonic functions, it is clear that f is maximised at V. The optimal solution can then be found by considering a constraint k for data selection:

$$\underset{|S| \leq k}{\arg\max}\{f(S)\} \tag{2.17}$$

In other words, each element of the set can be selected in order, based on the value of the function.

2.3 Deep Metric Learning

As deep learning has attracted considerable interest, metric learning using Deep Neural Networks (DNNs) has also been proposed for non-linear data, such as images (Kaya and Bilge, 2019). Deep metric learning is a method that maximises the similarity between samples in the same class while minimising it between samples in different classes. Samples in the same class are referred to as anchor and positive samples, while those in different classes are referred to as negative samples. The three key components of deep metric learning are the structure of the networks, the metric loss function and the sampling technique.

Regarding network structure, shared networks, such as the Siamese network (Bromley et al., 1993) and Triplet network (Hoffer and Ailon, 2015), have been adopted for uni-modal tasks (Benajiba et al., 2019), while separate networks have been used for cross-modal tasks (Liong et al., 2017; Mei et al., 2022; Xu et al., 2019). In the early stages, either a positive or a negative sample was used. To overcome the difficulty of model convergence, Triplet loss (Schroff et al., 2015) and N-pair loss functions (Sohn, 2016) were proposed for faster convergence (see Section 2.3.5). Finally, the sampling strategy plays an important role in maximising model performance and training speed. One strategy is hard negative mining. Hard negative samples are closer to the anchor than the positive samples. However, it is challenging to select hard negative samples for each anchor based on the distance metric during training. To mitigate the difficulty of hard negative mining, semihard negatives—those that are farther from the anchor than the positive but still relatively close—have been selected as proposed (Schroff et al., 2015). Additionally, N-pair loss with N negatives from multiple classes can be used for fast convergence (Sohn, 2016).

2.3.1 Multi-layer Perceptrons

An Multi-Layer Perceptron (MLP) is a model consisting of multiple layers, including more than one hidden layer in addition to input and output layers (LeCun et al., 2015). An MLP model is illustrated in Figure 2.5.

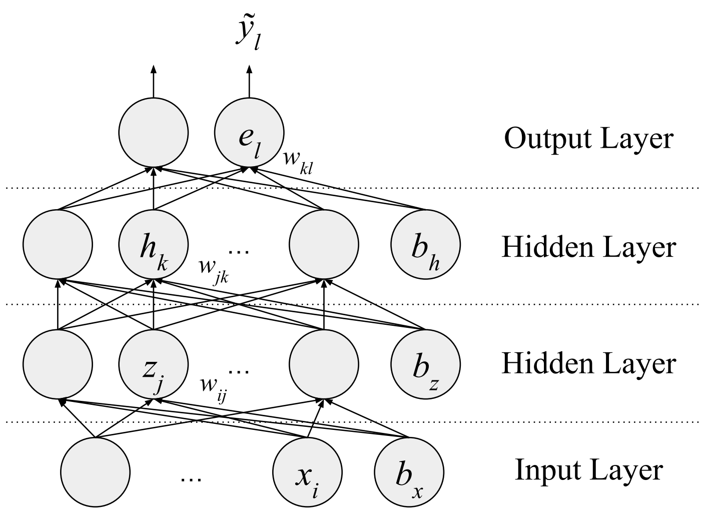


Figure 2.5 Multi-Layer Perceptron

2.3.1.1 Input, Hidden and Output Layers

The input layer consists of nodes that contain the input data. The weighted sum of these data is forwarded to the next layer. This output is activated by a non-linear function, referred to as an activation function. As the outputs are transformed along the hierarchically higher layers, they are known to represent information at a more abstract level.

2.3.1.2 Forward Pass

Let X, Z, H, and E be the node sets of the input, first hidden, second hidden, and output layers, respectively. Let x_i, z_j, h_k, and e_l be a node of each layer, and w_{ij}, w_{jk}, and w_{kl} be the weights between the layers. Let b_x, b_z, and b_h be the biases for the next layers and $f_x(\cdot)$, $f_z(\cdot)$, $f_h(\cdot)$, and $f_e(\cdot)$ be activation functions for each layer (see Figure 2.5). The forward pass through each layer is defined as follows:

$$\begin{aligned}
z_j &= \sum_{x_i \in X} w_{ij} x_i + b_x \\
y_j^z &= f_z(z_j) \\
h_k &= \sum_{z_j \in Z} w_{jk} y_j^z + b_z \\
y_k^h &= f_h(h_k) \\
e_l &= \sum_{h_k \in H} w_{kl} y_k^h + b_h \\
y_l^e &= f_e(e_l).
\end{aligned} \tag{2.18}$$

2.3.1.3 Back-propagation

Let y_l be the label corresponding to y_l^e and let the loss function $\mathcal{L}(y_l^e, y_l)$ be $(y_l^e - y_l)^2$. The back-propagation through each layer is defined as follows:

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial y_l^e} &= 2(y_l^e - y_l) \\
\frac{\partial \mathcal{L}}{\partial e_l} &= \frac{\partial \mathcal{L}}{\partial y_l^e} \frac{\partial y_l^e}{\partial e_l} \\
\frac{\partial \mathcal{L}}{\partial y_k^h} &= \sum_{e_l \in \tilde{Y}} w_{kl} \frac{\partial \mathcal{L}}{\partial e_l} \\
\frac{\partial \mathcal{L}}{\partial h_k} &= \frac{\partial \mathcal{L}}{\partial y_k^h} \frac{\partial y_k^h}{\partial h_k} \\
\frac{\partial \mathcal{L}}{\partial y_j^z} &= \sum_{h_k \in H} w_{jk} \frac{\partial \mathcal{L}}{\partial h_k} \\
\frac{\partial \mathcal{L}}{\partial z_j} &= \frac{\partial \mathcal{L}}{\partial y_j^z} \frac{\partial y_j^z}{\partial z_j}.
\end{aligned} \tag{2.19}$$

2.3.1.4 Activation Function

An activation function is a function applied to the output of nodes in neural networks. Sigmoid is used for binary classification, while softmax is used for multi-class classification.

2.3.1.4.1 Rectified Linear Unit Rectified Linear Unit (ReLU) is a ramp function that outputs the input values if it is non-negative, denoted as φ. The concept of this function was introduced (Householder, 1941) and later used by Fukushima (1975). Furthermore, Glorot et al. (2011) demonstrated that ReLU improved the effectiveness of supervised training in DNNs.

$$\varphi(x) = \begin{cases} x & (x \geq 0) \\ 0 & (x < 0). \end{cases} \tag{2.20}$$

2.3.1.4.2 Exponential Linear Unit Exponential Linear Unit (ELU) is proposed for faster and more precise learning in DNNs by allowing negative values to push mean unit activations closer to zero with lower computational complexity (Clevert et al., 2016).

$$\phi(x) = \begin{cases} x & (x > 0) \\ \alpha(\exp(x) - 1) & (x \leq 0). \end{cases} \tag{2.21}$$

2.3.1.4.3 Sigmoid Rumelhart et al. (1986) proposed the sigmoid function as a unit for learning representations through back-propagation. It is a logistic function that maps a real value into a range between 0 and 1, denoted as σ.

$$\sigma(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{1+e^x}. \tag{2.22}$$

2.3.1.4.4 Softmax A softmax function maps a set of k real values into a set of normalised exponential values, summing up to 1, for multi-input generalisation of logistic non-linearity (Bridle, 1990). This function is used as probability distribution of outputs. Formally, let softmax : $\mathbb{R}^k \rightarrow (0,1)^k$, where $k > 0$. Softmax is defined as follows:

$$\text{softmax}(\mathbf{x})_i = \frac{\exp(x_i)}{\sum_{j=1}^{k} \exp(x_j)} \tag{2.23}$$

where $\mathbf{x} = [x_1, x_2, \ldots, x_k]$, $\exp(x) = e^x$ and e is an exponential constant. Although the input can be negative, positive or zero, the output of the function is a value between 0 and 1 and the sum of all the elements in the output vector is 1. This is not a real probability of the output, but it is widely used for classification. For example, the class label can be predicted using the index of the maximum softmax value in the vector as follows:

$$\tilde{y} = \arg\max_i \text{softmax}(\mathbf{x})_i. \tag{2.24}$$

2.3.2 Distance Metrics

When the representations are in vector form, the distance between two representations is typically measured by cosine distance (Mithun et al., 2018) or p-norm distance (Chung et al., 2020, 2021b). Cosine distance is negatively proportional to cosine similarity, which measures the cosine of the angle between two vectors. On the other hand, p-norm distance is the p-norm of the difference between the vectors. 1-norm and 2-norm represent Manhattan and Euclidean distances, respectively. The distance metrics used in this thesis are summarised in Table 2.2. Let $\mathbf{a}$ and $\mathbf{b}$ be vectors and let a_i and b_i be the components of the vectors, respectively.

Table 2.2 Distance Metrics.

cosine distance	$d_{cosine}(\mathbf{a},\mathbf{b}) = 1 - \frac{\mathbf{a}\cdot\mathbf{b}}{\lVert\mathbf{a}\rVert\lVert\mathbf{b}\rVert} = 1 - \frac{\sum_{i=1}^{N} a_i b_i}{\sqrt{\sum_{i=1}^{N} a_i^2}\sqrt{\sum_{i=1}^{N} b_i^2}}$
1-norm distance (Manhattan or L1 Norm)	$d_{1-norm}(\mathbf{a},\mathbf{b}) = \lVert\mathbf{a}-\mathbf{b}\rVert_1 = \sum_{i=1}^{N} \lvert a_i - b_i \rvert$
2-norm distance (Euclidean or L2 Norm)	$d_{2-norm}(\mathbf{a},\mathbf{b}) = \lVert\mathbf{a}-\mathbf{b}\rVert_2 = \sum_{i=1}^{N} (a_i - b_i)^2$

2.3.3 Contrastive Loss Function

A contrastive loss function is used to learn a distance metric from data. It was proposed for face verification by minimising a discriminative loss function for pairs from the same person and maximising it for pairs from different persons (Chopra et al., 2005).

A pair from the same class consists of an anchor and a positive sample, while a pair from different classes consists of an anchor and a negative sample. Let $f_\theta(\cdot) : \mathcal{X} \to \mathbb{R}^d$ be a function that takes x and generates a representation vector. Let x_i and y_i be an input and its class label. The loss is then defined as follows:

$$\begin{aligned} \mathcal{L}_{\text{contrastive}}(x_i, x_j, m; f_\theta) =& (1 - \delta(y_i, y_j)) d_{\text{metric}}(f(x_i), f(x_j)) \\ &+ \delta(y_i, y_j) \max(0, m - d_{\text{metric}}(f(x_i), f(x_j))) \end{aligned} \tag{2.25}$$

where d_{metric} is a distance metric and $\delta(y_i, y_j)$ is an indicator function that is 0 if $y_i = y_j$ and 1 otherwise.

2.3.4 Triplet Loss Function

In representation learning, contrastive loss minimises the distances between representations within the same category while maximising the distances between representations from

(a) Positive Sample (b) Negative Sample

Figure 2.6 Contrastive Loss Function.

different categories. Let x be a sample and x^+ be another sample from the same class as e. The distance between e and x^+ is minimised. Conversely, if x^- is a sample from a different class, the distance between e and x^- is maximised. Building upon contrastive loss, triplet loss incorporates both positive and negative samples in the function. Specifically, the distance between e and x^+ should be shorter than the distance between e and x^-, with a margin m added to the distance between e and x^+. Thus, triplet loss $\mathcal{L}_{\text{triplet}}(x,x^+,x^-,m)$ is defined as follows:

$$\mathcal{L}_{\text{triplet}}(x,x^+,x^-,m;f_\theta) = \max\Big(0, d_{\text{metric}}(f(x),f(x^+)) - d_{\text{metric}}(f(x),f(x^-)) + m\Big) \tag{2.26}$$

where $d_{\text{metric}}(\cdot,\cdot)$ is a distance metric between representations.

2.3.5 N-pair Loss Function

However, contrastive and triplet losses use only two or three samples at a time, which can lead to slow convergence. To address this issue, N-pair loss was proposed by Sohn (2016), who utilised more negative samples from multiple classes. Instead of maximising the distance from a single negative sample, the sum of distances from multiple negative samples is used. For instance, if the data are divided into N classes, then the sum of triplet losses on a single positive sample and $N-1$ negative samples is expressed as:

$$\mathcal{L}_{\text{N-pair}}(x,x^+,\{x_i^-\}_{i=1}^{N-1};f_\theta) = \log\Big(1 + \sum_{i=1}^{N-1} \exp\big(d_{\text{metric}}(f(x),f(x^+)) - d_{\text{metric}}(f(x),f(x_i^-))\big)\Big). \tag{2.27}$$

2.3.6 Normalised Temperature-Scaled Cross Entropy Loss Function

Chen et al. (2020a) introduced a method to learn non-linear transformations between the representations for image classification. They randomly drew N samples from a batch B and trained a model with $2(N-1)$ pairs from augmented samples derived from the batch. They argued that contrastive learning benefits from larger batch sizes and more training steps compared to supervised learning.

$$\mathcal{L}_{NT-Xent}(B,\tau;f_\theta) = -\frac{1}{|B|}\sum_{x_i \in B} \log \frac{\exp(\mathrm{sim}(f(x_i),f(x_i^+))/\tau)}{\sum_{x_{i,j}^- \in \tilde{B}} \exp(\mathrm{sim}(f(x_i),f(x_{i,j}^-))/\tau)} \tag{2.28}$$

where x_i is the anchor sample, x_i^+ and $x_{i,j}^-$ are its corresponding positive and j-th negative samples, respectively. These negative samples are from the augmented dataset $\tilde{B} = \{\mathrm{aug}_1(b), \mathrm{aug}_2(b) | b \in B\}$ and τ is a temperature for scaling.

2.4 Representation Learning

2.4.1 Self-Supervised Learning Representation

Self-supervised learning is a method that leverages information inherent in the input data as labels for training. This concept has drawn significant attention since it was applied to a masked language model (Devlin et al., 2019) with self-attention, which showed impressive performance on Natural Language Processing (NLP) tasks. Simultaneously, this idea was implemented with contrastive predictive coding (Oord et al., 2018) for learning speech representation. Representation learning in a self-supervised manner has provided a way to utilise untranscribed data for ASR and related areas.

2.4.1.1 Masked Language Model

Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2019) is a language representation model trained using a masked language model objective. This objective involves predicting the masked parts of input tokens that have been randomly selected. After pre-training the model with this objective, it can be fine-tuned for specific tasks, such as sentence understanding. Additionally, a special token [*CLS*] is inserted at the beginning of the input and aggregated with other representations. This special token can be used for classification tasks, as it contains information about the entire context. This token is provided by the models sharing the architecture used for BERT, such as RoBERTa (Radford et al., 2023).

2.4.1.2 Noise-contrastive Representation Learning

A contrastive loss function maximises the similarity between data representations within a category while minimising it across different categories. When this method incorporates noise, it is referred to as Noise-Contrastive Estimation (NCE) (Gutmann and Hyvärinen, 2012). The objective of noise-contrastive loss is to learn to classify between observed data and artificially generated noise. A representation learning method using the noise-contrastive loss function was proposed (Oord et al., 2018; Schneider et al., 2019).

Let $f_\theta(\cdot) : \mathcal{X} \to \mathbb{R}^{d_x}$ be a function that maps an observation $x \in \mathcal{X}$ to a latent representation $\mathbf{z}_t \in \mathbb{R}^{d_x}$. Let $g_\omega(\cdot) : \mathcal{Z} \to \mathbb{R}^{d_c}$ be a function that maps a latent representation $\mathbf{z} \in \mathcal{Z}$ to a context representation $\mathbf{c}_t \in \mathbb{R}^{d_c}$. First, an observation x_t at time t is embedded by CNNs: $\mathbf{z}_t = f_\theta(x_t)$. Next, an embedding $\mathbf{z}_t$ is contextualised using an autoregressive model: $\mathbf{c}_t = g_\omega(\{\mathbf{z}_i\}_{i=1}^{t})$ to exploit high-level latent information across different parts. Instead of predicting x_{t+k} directly, the latent and context embeddings $\mathbf{z}_{t+k}$ and $\mathbf{c}_t$ are used to model a density ratio that preserves the mutual information between x_{t+k} and $\mathbf{c}_t$, where k is the number of future steps. The function $h_{\mathbf{W}_k}$ represents a density ratio for step k, where $\mathbf{W}_k \in \mathbb{R}^{d_x \times d_c}$ is a weight matrix for linear transformation. The density ratio function $h_{\mathbf{W}_k}$ is defined as follows:

$$h_{\mathbf{W}_k}(x_{t+k}, \mathbf{c}_t; \theta, \omega) = \exp(\mathbf{z}_{t+k}^{\top} \mathbf{W}_k \mathbf{c}_t). \tag{2.29}$$

InfoNCE was proposed as an objective function (Oord et al., 2018). Both the encoder and autoregressive model are trained to jointly optimise the InfoNCE loss:

$$\mathcal{L}(\{x_i\}_{i=1}^{t}, x_{t+k}, N_{t,k}; \mathbf{W}_k, \theta, \omega) = -\mathbb{E}_{N_{t,k}}\left[\log \frac{h_{\mathbf{W}_k}(x_{t+k}, \mathbf{c}_t; \theta, \omega)}{\sum_{x_j \in N_{t,k}} h_{\mathbf{W}_k}(x_j, \mathbf{c}_t; \theta, \omega)}\right] \tag{2.30}$$

where $X_{t,k} = \{x_1, x_2, \ldots, x_N\}$ is a set of samples for $(x_{t+k}, \mathbf{c}_t)$ and N is the number of random samples, including both positive and negative samples. To estimate the mutual information, a simple log-bilinear model was used (Oord et al., 2018) (see Equation (2.29)), whereas the wav2vec model (Schneider et al., 2019) applied a sigmoid function to $h_{\mathbf{W}_k}$.

2.4.1.3 Self-Supervised Learning Representation for Text

Building upon BERT, Radford et al. (2023) enhanced BERT with an improved training recipe, referred to as RoBERTa. This model was further extended to deal with multiple languages, referred to as XLM-R (Conneau et al., 2020), which was trained on one hundred languages and two terabytes of filtered CommonCrawl data (Wenzek et al., 2020). While the BERT models adopted masked prediction, Generative Pre-Training (GPT) models were based on the product of conditional probabilities (Radford, 2018; Radford et al., 2019).

$$L_\theta(U) = \sum_i \log P(u_i | u_{i-k}, \ldots, u_{i-1}; \theta) \tag{2.31}$$

where k is the context size, U is a sequence of tokens $u_1, u_2, \ldots, u_n$ and P is a conditional probability model using neural networks with their parameters θ. The initial model's performance was further improved by a large amount of data, such as millions of webpages and more parameters, such as 1.5 billion parameters. Lastly, DeBERTa (He et al., 2021b) adopted a disentangled attention mechanism and enhanced mask decoder. Based on DeBERTa, DeBERTa-V3 (He et al., 2021a) also applies the pre-training method of replaced token detection (Clark et al., 2020). The information summary of these Self-Supervised Learning Representation (SSLR) models is organised in Table 2.3.

Table 2.3 Information summary of SSLR models for text.

Model	**Size**	**# Param.**	**Pre-training Data**	**Perf. (Acc.)**	
				MNLI-m	SST-2
XLM-R[1]	Large	560M	CommonCrawl[5] in 100 language	89.1	95.0
RoBERTa[2]	Large	355M	BookCorpus[6], English WIKIPEDIA[7]	84.3	92.5
DeBERTa-V3[3]	Large	283M	BookCorpus, English WIKIPEDIA	91.8	96.9
GPT-2[4]	Medium	355M	WebText[8]	-	-

2.4.1.4 Self-Supervised Learning Representation for Speech

For ASR, approaches similar to BERT, such as Hidden-Unit BERT (HuBERT), have been proposed as a self-supervised representation learning model for speech recognition. The model employed K-means clustering to produce consistent annotations, referred to as pseudo labels, over frames. The model was trained to predict the pseudo labels of the masked inputs. After iterating between pseudo-labelling and masked prediction, each layer was used to extract features for speech recognition. Based on HuBERT, WavLM (Chen et al., 2022)

[1]https://github.com/facebookresearch/fairseq/blob/main/examples/xlmr/README.md
[2]https://github.com/facebookresearch/fairseq/blob/main/examples/roberta/README.md
[3]https://github.com/microsoft/DeBERTa
[4]https://github.com/openai/gpt-2
[5]https://commoncrawl.org
[6]https://yknzhu.wixsite.com/mbweb
[7]https://en.wikipedia.org/wiki/Main_Page
[8]https://github.com/openai/gpt-2-output-dataset

employed both spoken content and speaker identity modelling. Additionally, the training dataset was scaled up to 94k hours from 64k hours.

Wav2vec (Schneider et al., 2019) was built using InfoNCE (see Equation (2.30)). Wav2vec 2.0 (Baevski et al., 2020) extended the wav2vec architecture by adopting masked prediction for speech input in the latent space, in addition to the contrastive task. This model demonstrated strong performance on the ASR task achieved 4.8% and 8.2% in WER on clean and noisy audiobook speech data, respectively, with ten minutes of transcribed data and 53k hours of untranscribed data. It was further extended to deal with multiple languages, resulting in the multilingual model XLM-R (Conneau et al., 2020).

Lastly, data2vec (Baevski et al., 2022) was proposed as a uniform framework for images, speech and text. Instead of predicting modality-specific targets such as words, tokens or sound unit, the model predicted the contextualised latent representations of the input. The information summary of these SSLR models is organised in Table 2.4.

Table 2.4 Information summary of SSLR models for speech.

Model	**Size**	**# Param.**	**Untranscribed Data**	**WER(%) on LSP**	
				test-clean	test-other
HuBERT[10]	Large	316M	Libri-Light[14] 60k hr	1.9	3.5
WavLM[11]	Large	317M	Libri-Light 60k hr, GigaSpeech[15] 10k hr, VoxPopuli[16] 24k hr	2.1	4.0
XLSR-53[12]	Large	315M	MLS[17], CommonVoice[18], BABEL[19]	-	-
data2vec[13]	Large	313M	Libri-Light 960hr	-	4.6

To address the challenges of working with continuous speech data, discrete representation models have been proposed. For example, w2v-BERT (Chung et al., 2021a) learns discrete features by combining masked language modelling from BERT with contrastive learning from wav2vec 2.0. Another model, VQ-VAE (van den Oord et al., 2017), learns discrete tokens

[9]https://github.com/facebookresearch/fairseq/blob/main/examples/wav2vec/xlsr/README.md
[10]https://github.com/facebookresearch/fairseq/blob/main/examples/hubert/README.md
[11]https://github.com/arxyzan/data2vec-pytorch
[12]https://github.com/microsoft/unilm/tree/master/wavlm
[13]https://www.openslr.org/94
[14]https://commonvoice.mozilla.org/en/datasets
[15]https://catalog.ldc.upenn.edu/byyear
[16]https://github.com/facebookresearch/libri-light
[17]https://github.com/SpeechColab/GigaSpeech
[18]https://github.com/facebookresearch/voxpopuli

in an unsupervised setting by combining variational autoencoders with vector quantisation to encode data into a discrete latent space. Textless-lib (Kharitonov et al., 2022) offers a practical implementation of a method for obtaining discrete tokens. Its speech-to-unit encoder samples a spectral representation of the audio file at a lower frequency using a pre-trained HuBERT model, then applies K-means clustering to output a discrete token corresponding to the closest entry in the codebook. Unlike w2v-BERT and VQ-VAE, which use CNNs in their architectures, Textless-lib bypasses pre-training tasks and vector quantisation, providing a simpler and more direct approach to obtaining discrete representations from speech data.

2.4.2 Sequence-level Representation

This section explores fixed-length representations for speech and text, including i-vectors and x-vectors for speaker recognition, as well as sentence-level embeddings like SBERT for NLP tasks.

First, the i-vector (Dehak et al., 2011), which stands for “identity vector”, is a fixed-length representation of an utterance used for speaker recognition. It is the reduced-dimensional representation of a supervector, which is a stack of vectors and matrices of Gaussian mixture model parameters. A speaker-dependent supervector M is decomposed as follows:

$$\mathbf{M} = m + \mathbf{T}w$$

where m is a speaker-independent supervector, obtained from the universal background model of multiple speakers. T is a low-rank total variability matrix, and w is a vector of total factors, which are standard normal hidden variables. In the i-vector approach, T captures both speaker and session variabilities simultaneously. It is trained using the expectation–maximisation algorithm, and then w is used as an i-vector after post-processing, such as channel compensation.

Second, the x-vector (Snyder et al., 2018) is another fixed-length representation of an utterance used for speaker recognition. It was designed to capture the long-term characteristics of an utterance to discriminate between speakers from variable-length speech. The model for the x-vector consists of three components: Time-Delay Neural Networks (TDNNs) (Peddinti et al., 2015), statistics pooling and DNNs. First, the TDNNs extract a temporal context around the frame at time t. Next, the outputs of the TDNNs are aggregated into a statistics pooling layer, which computes their mean and standard deviation. Finally, these aggregated outputs are input into the hidden layers of DNNs, and the output of softmax on top of them is used to classify speakers using multi-class cross-entropy.

Last, a sentence-level representation was proposed for NLP tasks, such as Semantic Textual Similarity (STS) between sentences (Reimers and Gurevych, 2019). The representation, referred to as SBERT, was learned using a Siamese or triplet model—often referred to as a two-tower architecture (Huang et al., 2013a; Yang et al., 2020) with classification, regression, and triplet objective functions. One of the SSLRs, BERT (Devlin et al., 2019), was adopted and converted into a fixed-length representation for a sentence through different pooling strategies. The results showed that the average pooling strategy outperformed others, such as using the [*CLS*] token. In addition to SBERT, the average pooling strategy for utterance representation has gained popularity in many other tasks, including speaker identification, intent classification, and emotion recognition (Wang et al., 2018; wen Yang et al., 2021).

2.5 Word Error Rate Estimation

2.5.1 Definition of Terms

An ASR system is designed to transcribe speech, and its output consists of the best hypotheses generated by the system. In this thesis, the terms *ASR output* or *ASR hypotheses* are both collectively refer to ASR transcripts. In contrast, the ground-truth transcripts are referred to as *ASR references*. Since these ground-truth transcripts are manually generated by human experts, they are also referred to as manual transcripts. Although manual transcripts can technically refer to any transcripts created by a human, in this thesis, unless otherwise specified, they specifically refer to ASR reference or ground-truth transcripts.

2.5.2 Performance Assessment of Automatic Speech Recognition System

An ASR system's performance is assessed by measuring the error rate in its output. The output is a sequence of varying recognition units, such as a phoneme, a character or a word. Therefore, the error can be defined at these levels. The errors in ASR transcripts are identified by comparing them to the references. The measurement of these errors was recommended to be documented as the performance assessment of an ASR system in the 1980s (Pallett, 1985). Additionally, they included Correct Recognition Percent, referred to as Recognition Accuracy. These are summarised in Table 2.5.

Table 2.5 Correct Recognition, Substitution, Deletion and Inserted Percents (Pallett, 1985, p.382).

Correct Recognition Percent	$\frac{(\#\text{Correctly Recognised Words}) \times 100}{\#\text{Test Words}}$(Percent)
Substitution Percent	$\frac{(\#\text{Substituted Words}) \times 100}{\#\text{Test Words}}$(Percent)
Deletion Percent	$\frac{(\#\text{Deleted Words}) \times 100}{\#\text{Test Words}}$(Percent)
Insertion Percent	$\frac{(\#\text{Inserted Words}) \times 100}{\#\text{Test Words}}$(Percent)

These terms are now more commonly referred to as substitution, deletion, and insertion error rates. Examples are provided in Table 2.6.

Table 2.6 Examples of insertion, deletion, substitutions. [del] is a label of deletion.

Operation	**Word Sequence**
Insertion	ref: it said it expects to make a double quote hyp: it **is** said it expects to make a double quote
Deletion	ref: australians have got to recognize hyp: australians have **[del]** to recognize
Substitution	ref: then they expect a potential partner or lender hyp: **thin** they expect a potential partner or lender

Additionally, the sum of these error rates is frequently used as an evaluation metric for assessing ASR performance. For example, WER is a metric that sums all error rates at the word level:

$$\text{WER} = \frac{\text{Substitutions} + \text{Deletions} + \text{Insertions in a hypothesis}}{\text{Total number of words in a reference}} \tag{2.32}$$

2.5.2.1 Levenshtein distance

The number of substitutions, deletions and insertions is calculated using Levenshtein distance, which represents the similarity between two sequences. This is the minimum number of operations to transform one sequence to another. For example, an insertion followed by a deletion results in the same outcome as a substitution. In this case, it is counted as one substitution operation. The Levenshtein distance between two sequences *a* and *b* is defined as:

$$d_{\text{lev}}(a,b) = \begin{cases} \max(|a|,|b|) & \text{if } \min(|a|,|b|) = 0 \\ \min \begin{cases} d_{\text{lev}}(a[i-1],b[j]) + 1 & \text{(del.)} \\ d_{\text{lev}}(a[i],b[j-1]) + 1 & \text{(ins.)} \\ d_{\text{lev}}(a[i-1],b[j-1]) + \delta(a[i],b[j]) & \text{(sub.)} \end{cases} & \text{otherwise} \end{cases} \tag{2.33}$$

where $a[i]$ is the i-th recognition unit in a and $\delta(a[i],b[j])$ is an indicator function that is 0 if $a[i] = b[j]$ and 1 otherwise.

2.5.3 Word Error Rate Estimation

Negri et al. (2014) introduced a method for estimating the quality of ASR output, which was measured using WER, hereafter referred to as WER estimation, when manually transcribed transcripts are unavailable. For WER estimation without references, they categorised input features into two types: glass-box features, which are obtained from the internal ASR process, such as decoding, and black-box features, such as the length of silence, which are external to the ASR process. They trained a WER estimation model using both glass-box and black-box features for the best performance. Additionally, Ali and Renals (2018) introduced a feed-forward neural network, e-WER, to predict the number of errors and the word count per utterance. Ali and Renals (2020) extracted features using a phone recogniser to enhance the model's performance. However, these WER estimation models that utilised black-box features lagged behind those using glass-box features. With the introduction of SSLR models for speech and text (Chowdhury and Ali, 2023), such as XLS-R (Babu et al., 2022) and XLM-R (Conneau et al., 2020), the performance of the WER estimation model, e-WER3, has improved significantly in terms of Root Mean Square Error (RMSE) and Pearson Correlation Coefficient (PCC). These SSLR models, which rely on black-box features, have outperformed traditional WER estimation models using confidence scores, a glass-box feature.

2.5.3.1 e-WER3

e-WER3 is a WER estimator for multiple languages. Chowdhury and Ali (2023) generated ASR transcripts using a conformer-based ASR system (Gulati et al., 2020) trained on LibriSpeech (Panayotov et al., 2015) and extracted utterance and transcript features using XLS-R (Conneau et al., 2021) and XLM-R (Conneau et al., 2020). The hidden states of a Bidirectional LSTM (BiLSTM) in both directions over frame representations were concatenated to form an utterance representation, while a transcript representation was averaged over token representations. The overall architecture, including this aggregation of the representations, is illustrated in figure 2.7.

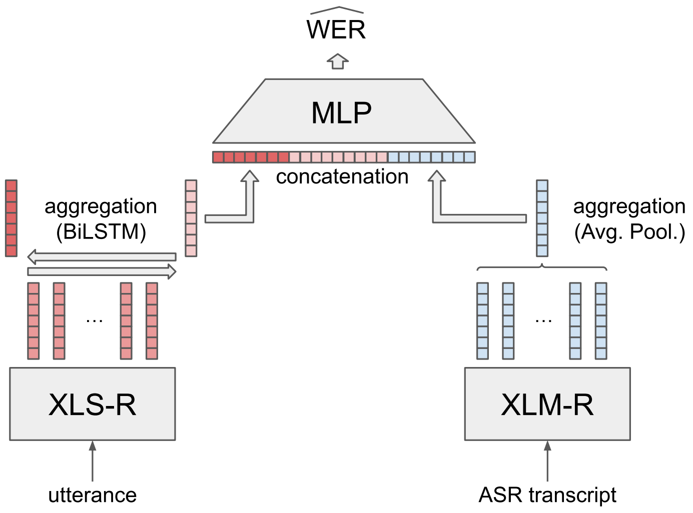


Figure 2.7 Overall architecture of e-WER3.

For data generation, ASR transcripts with a WER of 0 were selected up to the sum of the numbers in the second and third most frequent groups. The WER was predicted using fully connected layers on top of the concatenated representation. The result was 0.14 in RMSE and 0.66 in PCC on the English corpus Ted-lium 3 (TL3) (Hernandez et al., 2018), representing a 9% relative improvement in PCC over e-WER2 (Ali and Renals, 2020).

2.5.3.2 Training Process for Word Error Estimation

A training instance consists of an utterance, its ASR hypothesis and the WER between the utterance's corresponding manual transcript and the ASR hypothesis. The utterance and the ASR hypothesis are converted into features, either glass-box or black-box features, which are then input into the WER estimation model. This process is illustrated in figure 2.8.

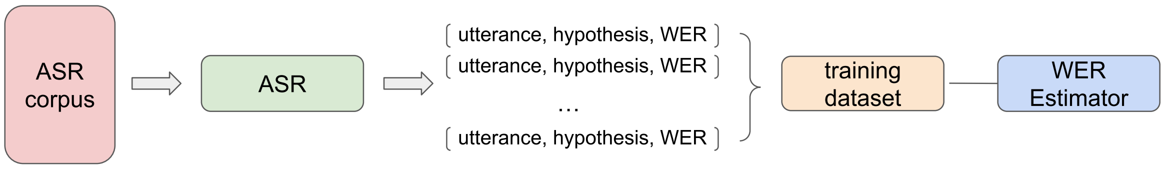

ASR corpus
ASR
[utterance, hypothesis, WER]
[utterance, hypothesis, WER]
...
[utterance, hypothesis, WER]
training dataset
WER Estimator

Figure 2.8 Illustration of Training Process for WER estimation.

2.6 Summary

This chapter provided an in-depth review of five key areas relevant to ASR: Semi-Supervised Learning for ASR, Data Selection for ASR, Deep Metric Learning, Representation Learning and WER estimation. It began by defining the key terminologies in semi-supervised learning and introduced algorithms that leveraged automatically generated ASR transcripts to improve model performance. These methods were particularly useful for training ASR systems in low-resource settings where transcribed data were scarce. Next, the chapter explored data selection strategies for ASR, discussing techniques such as confidence scores for word spotting and CTC segmentation. Additional methods for selecting data to reduce negative transfer due to domain mismatch were also introduced. Submodular function maximisation was highlighted as a strategy for efficiently selecting the most relevant data subsets. The chapter then delved into deep metric learning, a critical technique for representation learning that ensured similar data points were mapped closer together in the learned representation space. The fundamentals of multi-layer perceptrons, distance metrics, and loss functions, such as contrastive loss, triplet loss, and N-pair loss, were explained. These methods were essential for optimising models to align the edit distance of transcripts with the distance metric between utterance representations. After introduction to the self-supervised learning representation models for speech and text, the chapter concluded with a detailed discussion on WER estimation. Methods for calculating WER were outlined, followed by an exploration of WER estimation models, such as e-WER3, designed for scenarios where ground-truth transcripts were not available. The training processes for WER estimation were also covered, providing insights into the approaches for this task.

Chapter 3

Transcript Information in Utterance Representation

To utilise untranscribed data for semi-supervised learning in Automatic Speech Recognition (ASR), the linguistic information retained in speech representations could be useful for estimating the Word Error Rate (WER) of ASR transcripts. In this case, ASR transcripts with high WER need to be removed to reduce the propagation of errors. If the linguistic information related to the transcript of speech, hereafter referred to as transcript information, is retained in the speech representation and can be utilised for this purpose, then the WER of ASR transcripts could be estimated independently of the ASR using the representation retraining transcript information.

One challenging issue is that transcript information in the representation is not feasible to utilise. Because the transcript is sequential, the information is distributed across the sequence. Therefore, a method for aggregating this sequential information over the input is required to form a transcript representation. One approach to aggregation is to accumulate the context in one direction, or alternatively, in both directions. Long Short-Term Memory (LSTM) has been a popular method for this purpose (Hochreiter and Schmidhuber, 1997). Another approach is to employ a classification token as introduced in Natural Language Processing (NLP). This special token is employed by transformer-based representation models to capture the entire context for classification tasks (Devlin et al., 2019). This idea was further developed for matching sentences using a representation averaged over a token sequence (Reimers and Gurevych, 2019). It has also been applied to matching speech and text for Spoken Language Understanding (SLU) (Chung et al., 2021b; wen Yang et al., 2021). However, these works were limited to utilising semantic similarity between two inputs rather than focusing on information directly related to the transcripts themselves.

In this chapter, whether the transcript information, required to calculate WER, can be derived independently of ASR decoding is investigated to address part of the first research question—*Is WER estimation without ASR decoding effective for semi-supervised learning for ASR?* Since edit distance between transcripts is central to WER estimation, whether the edit distance can be learned through utterance and transcript representations is explored, which are extracted without reliance on ASR decoding.

To achieve this, two tasks are introduced:

1. Alignment Task (Section 3.1):
   This task involves ranking a set of utterances according to the edit distance—the number of operations required to transform one sequence to another—between their transcripts. If a model can be trained to successfully perform this task using utterance representations, it can be inferred that transcript information is present in the utterance representations. In this thesis, the alignment model is trained using Multi-distance N-pair loss. The results demonstrate that the distance metric, such as cosine distance, between utterance representations is highly correlated to the edit distance between their corresponding transcripts after training the model on the alignment task.

2. Matching Task (Section 3.2):
   In this task, transcript information is leveraged from both transcript and utterance representations. A model is trained on the matching task using deep metric learning, applied in both directions: utterances to transcripts and transcripts to utterances. Through this experiment, it is verified that the edit distance between transcripts can be represented by the distance metric between utterance and transcript representations.

3.1 Distant Alignment of Utterances with Multi-Distance N-pair Loss

Utterance representations have been widely used for various tasks. For example, i-vectors (Dehak et al., 2011) and x-vectors (Snyder et al., 2018) are popular representations for speaker recognition. These representations are expected to depend primarily on speaker characteristics rather than contents, as they are more related to consistent features across an entire utterance than to variable features over time. Nevertheless, some variable features still remain in these representations. The various types of information within x-vector were extensively probed, where the results showed that x-vectors retained lexical content in addition to speaker-related information (Raj et al., 2019).

So far, however, there has been little discussion about the transcript information within utterance representations in literature. It relates more to the text itself, while semantic information may refer to context. The first question about this type of information is how it can be utilised. If it is captured in utterance representations, it could be used to match an utterance to another where their corresponding transcripts are identical. If such matching is possible, an utterance could be distinguished from another when their corresponding transcripts differ. Additionally, the dissimilarity between the transcripts, such as the edit distance, could be estimated using utterance representations. To implement this idea, the dissimilarity between utterance representations is measured using a distance metric and aligned with the edit distance. Consequently, the performance of an alignment model would depend on the amount of transcript information in the utterance representation.

Regarding learning a distance from data, referred to as deep metric learning (see Section 2.3), the basic principle is that similar data points should be close together, while different data points should be farther apart. Based on this principle, the distance metric that captures the hidden relationship between data is searched for. One of the most widely used loss functions for this is the contrastive and triplet loss (see Equation (2.25) and (2.26)). An improved loss function using N-pair data was later proposed for faster convergence (see Equation (2.27)). Using this method, the distance metric of utterance representations is aligned with the edit distance of transcripts in a representation space.

For stable convergence, a multi-distance N-pair sampling technique is employed. As discussed in Section 2.3, three types of samples are required for metric learning: anchor, positive and negative samples. To generate N pairs of utterances for training, anchor samples are selected sequentially, while positive and negative samples are randomly selected from multiple distance groups, determined by the edit distance of their corresponding transcripts. This sampling technique helps ensure that the edit distances of samples in a training instance are more evenly distributed.

In this section, distant alignment between utterances using multi-distance N-pair loss is proposed as a method for measuring the amount of transcript information retained in utterance representation. The proposed model learns the distance metric between utterance representations using the N-pair loss with samples from N groups by the edit distance between their corresponding transcripts. The model's performance is evaluated on a distant alignment task, ranking utterances according to the edit distance of their corresponding transcripts. This performance will represent the amount of transcript information retrained in a specific utterance representation.

3.1.1 Multi-Distance N-pair Alignment Model

The overall process of the proposed method is as follows. First, sets of anchor, positive and negative samples are generated using the sampling technique. Next, all the utterances are encoded into utterance representations. Then, a model for distant alignment is trained using multi-distance N-pair loss. Finally, the accuracy of finding the positive utterance is evaluated using the representations aligned in the new representation space.

3.1.1.1 Sampling for Data Generation

An important consideration for a loss function in metric learning is the sampling technique used for the training dataset. If negative samples are too far from the anchor sample or positive samples are too close to it, the model may fail to converge during training. To alleviate this issue, utterances for the positive and negative samples are selected from N groups, which are divided based the edit distance of their corresponding transcripts.

Let t_i and t_j be two transcripts with their corresponding utterances u_i and u_j, respectively. Let $d_{\text{edit}}(t_i, t_j)$ be the edit distance between the transcripts. Now define N groups $G_{i,n}$ where $n = 1, 2, \ldots, N$, such that each group contains transcripts whose edit distance falls within the n-th range of edit distances when (u_i, t_i) is sampled as an anchor.

$$G_{i,n} = \{u_j | \frac{(n-1)}{10} \max(d_{\text{edit}}(t_i)) \leq d_{\text{edit}}(t_i, t_j) \leq \frac{n}{10} \max(d_{\text{edit}}(t_i))\} \tag{3.1}$$

where $\max(d_{\text{edit}}(t_i))$ is the maximum edit distance between t_i and other transcripts in the dataset.

For example, when an utterance of the anchor sample is selected from a train dataset, the edit distance between the transcript corresponding to the anchor and all the transcripts corresponding to the other utterances are calculated. Then, they are grouped into N groups by their edit distance. Next, the utterance from the closest group $G_{i,1}$ is used as the positive sample, while utterances from the other N-1 groups $G_{i,\{2\ldots10\}}$ are used as negative samples. This process is illustrated in Figure 3.1.

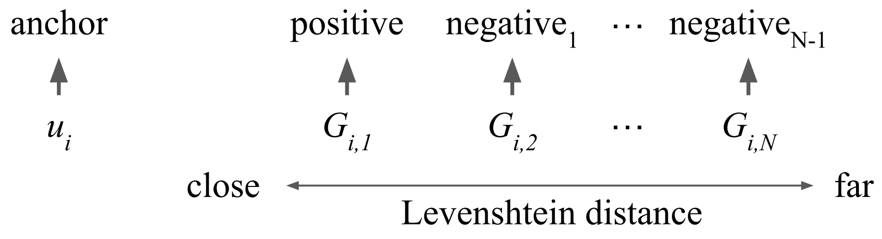


Figure 3.1 Anchor, positive and negative samples.

In some cases, transcripts of positive and negative samples are equally close to the transcript of their anchor. Two different utterances with identical transcripts could confuse the distant alignment model. To prevent this issue, the first negative sample's transcript should be farther from the anchor than the positive sample's transcript in terms of edit distance.

3.1.1.2 Model Architecture

A model is trained using deep metric learning with N-pairs from multi-distance groups. A set of data comprises an anchor, a positive and $N-1$ negative samples. For clarification, following the notations in Equation (2.27), the utterance representations of the anchor, positive and negative samples are denoted as $\mathbf{z} = f(u)$, $\mathbf{z}^+ = f(u^+)$ and $\mathbf{z}_i^- = f(u_i^-)$, respectively. Let $h(\cdot;\theta) : \mathcal{U} \to \mathbb{R}^d$ be a function taking an utterance u and generating its representation vector. Then, shared Deep Neural Networks (DNNs) is denoted as h and the output of the networks is denoted as $\mathbf{e} = h(\mathbf{z})$, $\mathbf{e}^+ = h(\mathbf{z}^+)$ and $\mathbf{e}_i^- = h(\mathbf{z}_i^-)$. The distance metric between utterance representations, $d_{\text{metric}}(\mathbf{e}_i, \mathbf{e}_j)$, would be proportional to the edit distance between their corresponding transcripts, $d_{\text{edit}}(t_i, t_j)$, if the representations contain transcript information. Thus, the utterance representations are learned by minimising $d_{\text{metric}}(\mathbf{e}, \mathbf{e}^+)$ and maximising $d_{\text{metric}}(\mathbf{e}, \mathbf{e}^-)$, to integrate transcript information. The model's architecture is illustrated in Figure 3.2.

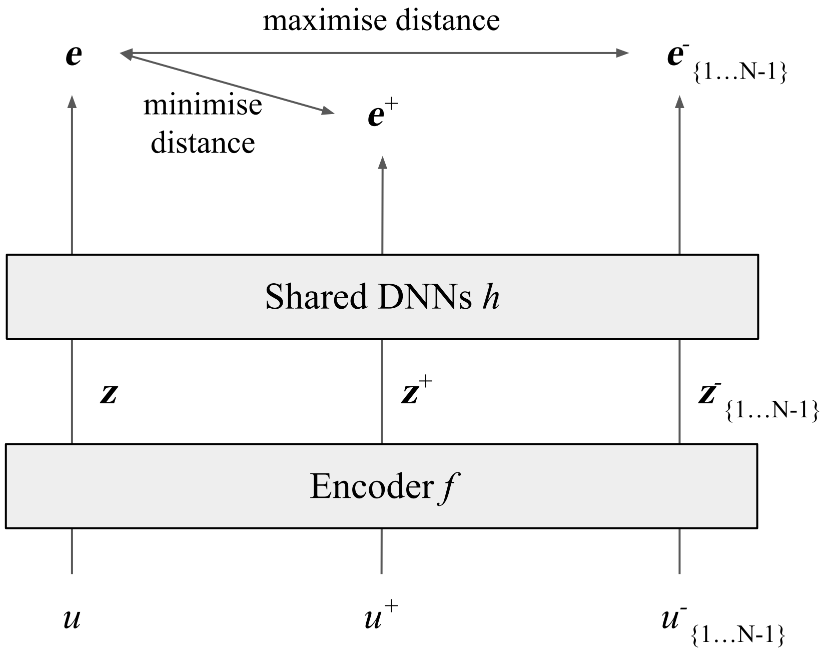


Figure 3.2 Model architecture for deep metric learning.

3.1.1.3 Multi-distance N-pair Loss Function

The multi-distance N-pair (MDN-pair) loss is based on Equation (2.27) with margin m introduced in Equation (2.26) where the positive and negative samples are chosen from N distance groups.

$$\mathcal{L}_{\text{MDN-pair}}(u, u^+, \{u_i^-\}_{i=1}^{N-1}, m; h) = \log\Big(1 + \sum_{i=1}^{N-1} \exp\big(\mathcal{L}_{\text{triplet}}(u, u^+, u_i^-, m; h)\big)\Big). \tag{3.2}$$

3.1.1.4 Evaluation Metrics

The distant alignment model is evaluated using Rank@1 accuracy and Spearman's Rank Correlation Coefficient (SRCC) ρ. First, Rank@1 accuracy refers to the ratio of correct predictions where the model correctly ranks the positive sample as the closest to the anchor, and if the distance between the anchor and the positive representation is the shortest, it is considered a correct prediction for Rank@1 accuracy. Let $\text{rank}(\mathbf{e}, \mathbf{e}^+)$ be the rank of the metric distance $d_{metric}(\mathbf{e}, \mathbf{e}^+)$. Rank@1 accuracy is calculated as:

$$\text{Rank@1 accuracy} = \frac{\text{the number of instances where } \text{rank}(\mathbf{e}, \mathbf{e}^+) = 1}{\text{total number of instances}}.$$

Second, SRCC measures the strength and direction of association between two ranked variables. It ranges from -1 to 1, where -1 and 1 represent negative and positive correlation, respectively, while 0 indicates no correlation. ρ is defined as:

$$\rho = 1 - \frac{6 \sum d_{\text{rank}}(i)^2}{M(M^2 - 1)}$$

where $d_{\text{rank}}(i)$ is the difference between the two ranks of the i-th observation and M is the total number of observations.

3.1.2 Experimental Setup

3.1.2.1 Datasets

The alignment model was trained and tested on two corpora of read speech and meeting speech: Wall Street Journal (WSJ) (Paul and Baker, 1992) and Augmented Multi-party Interaction (AMI) (Carletta et al., 2005), respectively. As shown in Table 3.1, the WSJ corpus consists of SI284, DEV93 and EVAL92/93 as training, validation and test sets, respectively, while the AMI corpus comprises Full-corpus-ASR (FCA) datastes[1] with Individual Head

[1]https://groups.inf.ed.ac.uk/ami/corpus/datasets.shtml

Microphones (IHM), which are used as training, validation and test sets: FCASA, FCASB and FCASC, respectively. Moreover, the average length of WSJ utterances is 7.58 seconds, while that of AMI utterances is 2.53 seconds. Lastly, all transcripts were converted to uppercase because case affected the value of edit distance.

Table 3.1 Numbers of utterances of WSJ and AMI datasets.

Corpus	**Dataset**	**Utterances**	**Words**	**Hours**
WSJ	SI284	37,414	654,004	81.48
	DEV93	1,016	17,130	1.08
	EVAL92, 93	1,091	18,464	0.70
AMI	FCASA	91,490	702,939	64.80
	FCASB	7,549	56,732	5.12
	FCASC	12,611	94,009	8.68

3.1.2.2 Data Generation

Given the computing resource constraints, the total number of positive and negative samples was set to 10. For 10-pair loss, a training instance consisted of 11 samples (an anchor and its transcript): one anchor, one positive and nine negative samples. First, an anchor sample was selected sequentially from the training set. Then, the Levenshtein Distance (LD) between the anchor transcript and the other transcripts was computed to determine the edit distance (see Section 2.5.2.1).

3.1.2.3 Feature extraction

The i-vector and x-vector representations for each utterance were set to 250 and 512 dimensions, respectively, following the configuration customised to the proprietary ASR system. The i-vector and x-vector extractors were trained on either the WSJ or AMI training datasets, and both were computed offline. Additionally, two Self-Supervised Learning Representation (SSLR) models, HuBERT Large (Hsu et al., 2021) and RoBERTa Large (Radford et al., 2023), were employed to extract utterance and transcript representations. The *CLS* tokens from their outputs were used as utterance and transcript representations, each with 1024 dimensions. For comparison, random representations were generated with dimensions of 250, 512, and 1024.

3.1.2.4 Distance metric

One of cosine, 1-norm and 2-norm distances was used for computing distance between two representations. Cosine distance was calculated by subtracting cosine similarity from the maximum possible cosine value.

3.1.2.5 Distant alignment model

The model for distant alignment consisted of three linear layers, each followed by the Exponential Linear Unit (ELU) activation function[2], except for the last layer. Learning rates were selected from 8e-05, 4e-05, 8e-06 and 4e-06. The learning rate was reset using a cosine annealing schedule every 15 epochs. Additionally, stochastic gradient descent with warm restarts (Povey et al., 2015) was employed for training. Margins for the distance metrics were selected from 1, 0.1, 0.001, and 0. The batch size was 10240. All models were implemented using the PyTorch library[3] (Paszke et al., 2019).

3.1.2.6 Evaluation

For hyper-parameter tuning, Rank@1 accuracy was measured. After that, Rank@1 accuracy as well as the correlation between the estimated ranks (determined by cosine distance) and the ground-truth ranks (determined by edit distance) was measured using SRCCas implemented in SciPy[4] (Virtanen et al., 2020).

3.1.3 Results

3.1.3.1 Distance Metrics

A distance metric was chosen based on the Rank@1 accuracy of the models. The amount of transcript information retained in a transcript representation is assumed to be greater than that in an utterance representation. Therefore, the best distance metric for distant alignment was explored using a representation of a transcript before training the model with a representation of an utterance. Three distance metrics were tested on the WSJ and AMI datasets: cosine, 1-norm and 2-norm distances. The marginal distance m affected the accuracy. The results are shown in Table 3.2. The alignment model performed best with cosine distance on both datasets, followed by 1-norm and 2-norm. The highest accuracies on the WSJ and AMI datasets were 39.23% and 68.19% with cosine distance, 31.44% and 66.80% with 1-norm

[2]https://pytorch.org/docs/stable/generated/torch.nn.ELU.html
[3]https://pytorch.org
[4]https://scipy.org

Table 3.2 Performance of the distant alignment model using transcript representations with various distance metrics.

Data set	**Metric**	**Rank@1 accuracy (%)**				
		m=1	**0.1**	**0.01**	**0.001**	**0**
WSJ	Cosine	30.80	**39.23**	38.04	36.02	23.28
	1-norm	31.44	29.97	28.69	26.49	26.40
	2-norm	30.25	33.27	31.99	28.51	25.57
AMI	Cosine	49.85	66.61	67.66	**68.19**	59.40
	1-norm	66.80	65.46	64.31	62.86	63.83
	2-norm	61.60	65.74	66.65	65.07	63.88

and 33.27% and 66.65% with 2-norm, respectively. In addition to the type of distance metric, the results indicate that the marginal distance m between $d_{\text{edit}}(\mathbf{e}, \mathbf{e}^{+})$ and $d_{\text{edit}}(\mathbf{e}, \mathbf{e}^{-})$ significantly affects accuracy. For example, the accuracy on the AMI dataset with cosine distance was 68.19% when $m = 0.001$, compared to 49.85% and 59.40% when $m = 1$ and $m = 0$, respectively. Based on these findings, cosine distance was selected as the distance metric for the following sections. The optimal margins for cosine distance were 0.1 and 0.001 for the WSJ and AMI datasets, respectively.

3.1.3.2 Rank@1 Accuracy

The distant alignment models using different utterance representations with cosine distance were compared based on Rank@1 Accuracy. As shown in Table 3.3, the accuracies for i-vector, x-vector and HuBERT improved on both the WSJ and AMI datasets, while the accuracies for random representations across three different dimensions showed accuracies of around 10%. As the number of samples for alignment is 10, this was an expected result. Therefore, these results indicate that the type of representation affects the accuracy, rather than the dimensionality. When the initial representation was HuBERT, the accuracies on the WSJ and AMI datasets were the highest, reaching 35.75% and 38.54%, respectively. Additionally, the accuracies of the x-vector representation were higher than those of the i-vector representation on both datasets.

3.1.3.3 Correlation coefficient between Edit Distance and Distance Metric

The SRCC between LD and cosine distance, chosen as the distance metric in Section 3.1.3.1, is shown in Table 3.4. First, the correlation coefficients without distant alignment is in column

Table 3.3 Performance of distant alignment models using different representations.

Initial Representation	**Rank@1 Accuracy (%)**	
	WSJ	**AMI**
Random (250)	9.99	10.25
Random (512)	10.08	9.94
Random (1024)	8.71	10.18
i-vector	19.16	27.33
x-vector	24.56	32.89
HuBERT	**35.75**	**38.54**

(A). The coefficient for RoBERTa was the highest, while the coefficient for i-vector was the lowest. The high accuracy with the transcript representation confirms that the alignment task is somewhat related to linguistic information in terms of edit distance. Second, the correlations in column (B) are between LD and cosine distance after training the model on the alignment task. The model's performance improved for all utterance and transcript representations, while it did not improve for randomly initialised representation. Particularly, the ρ for HuBERT on WSJ and AMI achieved 0.890 from 0.123 and 0.957 from 0.444, respectively, even surpassing that of RoBERT, which is transcript representation. This may be influenced by the fact that transcript representation focuses more on semantic information, which is a key aspect of NLP tasks. The observation that the correlation did not increase with random representations demonstrate that the transcript information is retained in the representations, not in the model.

3.1.4 Conclusion

To explore transcript information encoded in utterance representations, a distant alignment model was proposed. Utterance representations, including i-vector, x-vector and *CLS* token from HuBERT, were compared in the task of aligning the cosine distance of utterance representations with the edit distance of their corresponding transcripts using N-pair loss. Utterances for N-pairs were selected from N groups divided based on the edit distance of their transcripts. When the models with random and utterance representations were evaluated using Rank@1 accuracy, only the utterance representations were effective in this alignment task. Additionally, SRCC was measured between the cosine distance and the edit distance. The assumption that varying amounts of transcript information are retained in utterance representations was supported by the results, which showed that the correlation coefficients

Table 3.4 SRCC between Levenshtein distance of transcripts and cosine distance of utterance representations on the AMI test dataset. (A): w/o distant alignment, (B): w/ distant alignment

Dataset	**Input**	**Representation**	**Dimensions**	ρ	
				(A)	**(B)**
WSJ	-	Random	250	-0.003	0.041
	-	Random	512	0.000	-0.011
	-	Random	1024	-0.005	0.036
	Speech	i-vector	250	0.063	0.440
	Speech	x-vector	512	-0.168	0.586
	Speech	HuBERT	1024	0.123	**0.890**
	Text	RoBERTa	1024	0.197	0.845
AMI	-	Random	250	-0.001	0.000
	-	Random	512	0.004	-0.010
	-	Random	1024	-0.007	0.000
	Speech	i-vector	250	0.055	0.700
	Speech	x-vector	512	-0.285	0.812
	Speech	HuBERT	1024	0.444	**0.957**
	Text	RoBERTa	1024	0.753	0.934

and Rank@1 accuracy for ranking the utterance representations increased as the distant alignment model learned the edit distance between transcripts. This observation indicates that the amount of transcript information encoded in utterance representations can be utilised by aligning the distance metric with the edit distance. It can also be measured using the Rank@1 accuracy and SRCC. According to these evaluation metrics, the amount of transcript information encoded in the i-vector, x-vector and HuBERT representations ranked lowest, intermediate and highest, respectively.

3.2 Matching Unpaired Utterances and Transcripts Using Deep Metric Learning

So far, transcript information retained in an utterance representation has been utilised by aligning the distance metric between utterance representations and the edit distance between their corresponding transcripts. The result shows significant difference in the performance of distant alignment models using different representations. More transcript information can be utilised with self-supervised learning representation, such as HuBERT, than with representation designed for speaker recognition, such as i-vector. This opens up the possibility of using utterance representations to find pairs of utterances and transcripts without an ASR system from unpaired speech and text data.

However, there has been little discussion on generating paired datasets from unpaired speech and text data. If utterances and their corresponding transcripts can be paired without ASR systems, additional datasets for supervised learning could be generated. Cascading speech recognition with text retrieval (Lee et al., 2015) is a possible approach to this problem. This method utilised ASR transcripts for pairing. In other words, utterances were paired using the similarity between their ASR transcripts and unpaired transcripts. However, the retrieval performance depends on the performance of the ASR system. Furthermore, this dependency on the ASR system might limit the scalability of the method when applied to large datasets due to the required computational resources for ASR. Another approach is a matching model widely used for data across different modalities, such as image and text (Liong et al., 2017; Xu et al., 2019). This matching method employed deep metric learning, which uses a nonlinear representation model to learn a distance metric, such as cosine distance, using neural networks. While this method has been applied to audio-text retrieval for captioning audio (Elizalde et al., 2019; Mei et al., 2022), few studies have explored its use for transcription. The lack of utterance representations that retain transcript information might be one reason for limited attention to this task, which can be addressed by distant alignment of utterances as discussed in Section 3.1.

The problem of matching utterances and transcripts can be simplified by assuming segmented speech. If both speech and text are segmented and enough pairs of them exist, the model could work on the matching task at a practical level. In this section, a method for matching utterances and transcripts without dependence on an ASR system is proposed, using bidirectional NT-Xent loss (see Section 2.3.6). This method is trained to align the distance metric between utterance and transcript representations with the edit distance between their corresponding transcripts. By aligning the distances across utterance and transcript representations, the representations are mapped onto a shared representation space. The performance of the matching model is measured using Rank@1 accuracy. Additionally, the WER of the paired dataset is measured between the ground-truth transcript of an utterance and the transcript paired by the model.

The contributions of this work are summarised as follows:

- A method using deep metric learning for matching spoken utterances and transcripts is proposed without relying on an ASR system.
- Extensive experiments were conducted using bidirectional objective functions, BiNT-Xent and BiN-pair, along with separate and shared networks.

- A sampling technique drawing segments within a batch is introduced to improve computational efficiency in training data generation.
- The proposed model achieved a Rank@1 accuracy of 85.70% and a WER of 10.35% on the matching task.

3.2.1 Unpaired Utterance-Transcript Matching Model

3.2.1.1 Utterance-Transcript Matching Problem

Given M segments, each segment s_i consists of an utterance u_i and its corresponding transcript t_i . The transcripts of the M segments are assumed to be distinguishable from one another. The matching problem is to find the corresponding utterance or transcript for a given input. For example, when an utterance u_i and a set of transcripts $t_1, t_2, ..., t_M$ are given, the task is to identify t_i among the transcripts. By using the fact that the distance between the utterance u_i and its corresponding transcript t_i is the shortest, t_i can be found. This problem can be approached from both directions: from either utterances to transcripts or transcripts to utterances.

3.2.1.2 Sampling for Training Data

A training instance consists of an anchor, a positive and negative samples. In the case of the utterances-to-transcripts matching problem, u_i is given and set to the anchor sample. As the distance between a pair is the shortest, t_i is set as the positive sample. For generating instances, a batch B is randomly selected from the training data. A segment for an anchor and a positive is sequentially selected within the batch. After that, a fixed number S of transcripts most similar to the anchor's transcript are selected. For example, the transcripts are ordered by cosine distance to the anchor's transcript, and the first S transcripts are selected as negative samples. This process is repeated for the case of transcripts-to-utterances.

3.2.1.3 Model Architectures

An utterance and a transcript are encoded by pre-trained representation models, f and g, for speech and text, respectively. The utterance and transcript representations, $\mathbf{z}_u = f(u)$ and $\mathbf{z}_t = g(t)$, respectively, are then input into DNNs to learn representations for the distance metric. Let $h_\theta(\cdot) : \mathcal{U} \to \mathbb{R}^d$ be a function that takes an utterance representation $\mathbf{z}_u$ and generates its representation vector and $h_\omega(\cdot) : \mathcal{T} \to \mathbb{R}^d$ be a function that takes a transcript representation $\mathbf{z}_t$ and generates its representation vector. Their outputs are $\mathbf{e}_u = h_\theta(\mathbf{z}_u)$ and $\mathbf{e}_t = h_\omega(\mathbf{z}_t)$, respectively. The similarity $sim(\mathbf{e}_u, \mathbf{e}_t)$ is used to match the utterance and the transcript. If u_i is an anchor sample, then t_i is its positive sample; otherwise it is a negative sample. In the case of shared networks shown in Figure 3.3b, both $\mathbf{e}_u$ and $\mathbf{e}_t$ are output from the shared networks $h_\theta(\cdot)$.

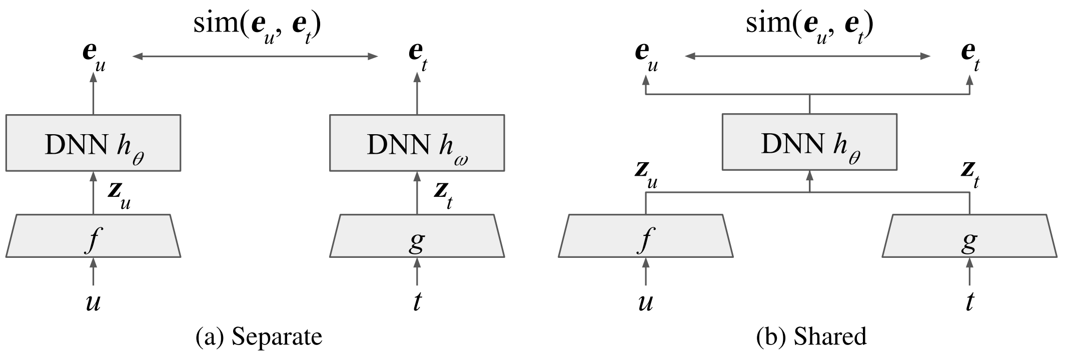


Figure 3.3 The architecture for deep metric learning.

3.2.1.4 Bidirectional Normalised Temperature-Scaled Cross Entropy Loss Function

The proposed method considers both directions of similarity: from utterances to transcripts and from transcripts and utterances. Therefore, either an utterance or a transcript can serve as an anchor for matching pairs. As a result, there are two sets of training instances. For example, the first set consists of anchor samples of utterances, positive samples of their corresponding transcripts and negative samples from other transcripts. The second set consists of anchor samples of transcripts, positive samples of their corresponding utterances and negative samples from other utterances. Let u_i be an anchor sample in the first set, t_i be its corresponding positive sample transcript and N_{u_i} and N_{t_i} be a set of the indexes of negative samples.

The bidirectional normalised temperature-scaled cross entropy loss (BiNT-Xent) is defined as follows:

$$\mathcal{L}^u_{\text{NT-Xent}}(B;h_\theta) = -\frac{1}{|B|}\Big(\sum_{u_i\in B}\log\frac{\exp(\text{sim}(\mathbf{e}_{u_i},\mathbf{e}_{t_i})/\tau)}{\sum_{j\in N_{u_i}}\exp(\text{sim}(\mathbf{e}_{u_i},\mathbf{e}_{t_j})/\tau)}\Big), \tag{3.3}$$

$$\mathcal{L}^t_{\text{NT-Xent}}(B;h_\omega) = -\frac{1}{|B|}\Big(\sum_{t_i\in B}\log\frac{\exp(\text{sim}(\mathbf{e}_{t_i},\mathbf{e}_{u_i})/\tau)}{\sum_{j\in N_{t_i}}\exp(\text{sim}(\mathbf{e}_{t_i},\mathbf{e}_{u_j})/\tau)}\Big), \tag{3.4}$$

$$\mathcal{L}_{\text{BiNT-Xent}}(B;h_\theta,h_\omega) = \mathcal{L}^u_{\text{NT-Xent}}(B;h_\theta) + \mathcal{L}^t_{\text{NT-Xent}}(B;h_\omega) \tag{3.5}$$

where $\text{sim}(\mathbf{e}_{u_i},\mathbf{e}_{t_j})$ is the cosine similarity between the i-th utterance and the j-th transcript representations, and τ is the temperature parameter.

Multi-distance N-pair loss (see Equation (3.2)) is also updated to consider both directions and is compared with BiNT-Xent.

$$\mathcal{L}^{u}_{\text{N-pair}}(B; h_\theta) = -\frac{1}{|B|}\Big(\sum_{u_i \in B} \mathcal{L}_{\text{MDN-pair}}(u_i, t_i, \{t_j\}_{j \in N_{u_i}}, m)\Big), \tag{3.6}$$

$$\mathcal{L}^{t}_{\text{N-pair}}(B; h_\omega) = -\frac{1}{|B|}\Big(\sum_{t_i \in B} \mathcal{L}_{\text{MDN-pair}}(t_i, u_i, \{u_j\}_{j \in N_{t_i}}, m)\Big), \tag{3.7}$$

$$\mathcal{L}_{\text{BiN-pair}}(B; h_\theta, h_\omega) = \mathcal{L}^{u}_{\text{N-pair}}(B; h_\theta) + \mathcal{L}^{t}_{\text{N-pair}}(B; h_\omega). \tag{3.8}$$

3.2.1.5 Sampling for Evaluation Data

In contrast to the sampling technique used for training data, evaluation data are generated using multi-distance N-pair sampling technique (see Section 3.1.1.1). By drawing samples from evenly distributed S distance groups, the difficulty of matching is expected to be similar. An evaluation instance consists of a segment for an anchor and a positive and S segments from S groups, divided by the edit distance between transcripts. Each segment from the test data is sequentially selected as an anchor. Then, all other segments were grouped into S sets based on the edit distance from the anchor segment's transcript. After that, S segments are drawn from the S groups. Any samples whose edit distance is the same as a positive sample are excluded for negative samples.

3.2.1.6 Evaluation Metrics

The performance of the model is measured using Rank@k accuracy (see Section 3.1.1.4) and WER (see Section 2.5.3). The mean values of accuracies from both directions are used for model comparison. Additionally, the WER between the transcripts of anchor and the predicted positive samples is measured. It is 0% if the positive sample ranks first. Otherwise, the WER is lower if the edit distance of a sample is closer to the anchor.

3.2.2 Experimental Setup

3.2.2.1 Dataset

The AMI corpus was selected for the matching task, as it consists of meeting speech with segments of various lengths. For example, there are mixes of short and long segments ranging from 0.03 to 33.19 seconds. For this experiment, three datasets with IHM were used: FCASA, FCASB, FCASC for training, validation, evaluation, respectively. Additionally, the model showed better performance on AMI between the datasets used for the distance alignment in Section 3.1.3.3.

Table 3.5 Statistics of AMI corpus.

Name	Segments	Hours	Min/Mean/Max (s)
FCASA	91,490	64.80	0.03/2.54/33.19
FCASB	7,530	5.12	0.03/2.44/24.22
FCASC	12,611	8.68	0.03/2.47/26.17

3.2.2.2 Model Architectures

HuBERT Large [5] and RoBERTa Large [6] were used as encoders for spoken utterances and transcripts, respectively. HuBERT Large was pre-trained on 60k hours of Libri-Light (Kahn et al., 2020b), while RoBERTa Large was pre-trained on 160GB of English text from various sizes and domains. The models were not updated during deep metric learning. They encode speech or text into 1024-dimensional vectors. The encoded outputs were input into two separate DNN models, each model consisting of two hidden layers with 1024 nodes, followed by ELU activation functions (Clevert et al., 2016) and an output layer. The models were trained with an alpha value of 0.01 for ELU with a maximum iteration of 40. For comparison, the same configuration was used for shared networks. All implementations were based on fairseq[7] (Ott et al., 2019), PyTorch (Paszke et al., 2019) and SpeechBrain[8](Ravanelli et al., 2021).

3.2.2.3 Hyper-parameters

Hyper-parameters, such as learning rate, were determined by grid search. The learning rate was set to 0.0005 for both BiNT-Xent and BiN-pair losses. The temperature for the BiNT-Xent loss function was determined within the range of [0.1, 1.0] as well as the margin for the BiN-pair loss function was selected within the range of [0.0001, 1.0].

3.2.2.4 Sampling for Data Generation

For each batch, segments were randomly selected from the training set. A segment was selected first. Either of its utterance or transcript served as the anchor or positive sample. Negatives samples were then selected, but segments with duplicate transcripts were not allowed. The batch sizes were 8, 16, 32, 64 and 128. The size was limited to 128 due

[5]https://github.com/facebookresearch/fairseq/blob/main/examples/hubert/README.md
[6]https://github.com/facebookresearch/fairseq/blob/main/examples/roberta/README.md
[7]https://ai.meta.com/tools/fairseq
[8]https://speechbrain.github.io

to the memory capacity of the computing resources and training time constraints. For the experiment on sampling size, the number of negative samples per batch was 7, 15 and 31. Negative samples were limited to half of the batch size because selecting all segments in the batch would be equivalent to having no negative sampling. For evaluation, the sampling size was set to 10.

3.2.3 Results

The performance of the matching models was measured using Rank@1 accuracy to find the optimal configuration: BiNT-Xent and BiN-pair losses; Shared and separate architectures; negative sampling size. Finally, the WER of the best model's output is computed.

3.2.3.1 BiNT-Xent vs BiN-pair

The Rank@1 accuracy of the matching model using BiNT-Xent and BiN-pair is shown in Table 3.6. The model with 2 layers of fully-connected feed forward neural networks and batch sizes of 64 and 32 achieved the highest accuracy using the BiNT-Xent and BiN-pair loss functions, with 85.59% and 84.89% accuracy, respectively. The model using BiNT-Xent outperformed the other using the BiN-pair loss. All subsequent experiments were conducted with the model using the BiNT-Xent loss and the batch size was set to 64.

Table 3.6 Rank@1 accuracy (%) with varying numbers of layers and batch sizes.

Loss Function	**Layers**	**Batch Size**				
		8	**16**	**32**	**64**	**128**
BiNT-Xent	2	83.23	84.61	85.26	**85.59**	85.51
	3	80.60	81.87	83.09	83.09	83.42
BiN-pair	2	82.46	84.26	**84.89**	84.67	84.52
	3	80.97	81.88	81.90	83.49	81.72

3.2.3.2 Shared vs Separate Architecture

The separate network architecture was compared with the shared network architecture. Additionally, the Rank@1 accuracies of the models using different equations as described in Section 3.2.1.4 are shown in Table 3.7. Equation (3.3) represents the loss when utterances are selected as anchors, while Equation (3.4) represents the loss when transcripts are selected as anchors. The table shows the Rank@1 accuracies for different combinations of these losses. It was observed that the highest accuracy was achieved when the loss was applied bidirectionally as in Equation (3.5), while the accuracy of the shared networks using Equation

(3.4) was the lowest. All subsequent experiments were conducted with the separate networks model using the bidirectional loss.

Table 3.7 Rank@1 accuracy(%) with architecture types and loss functions of different directions.

Loss Function	**Shared**	**Separate**
Equation (3.3)	85.27	85.14
Equation (3.4)	84.89	85.17
Equation (3.5)	85.47	**85.59**

3.2.3.3 Negative Sampling Size

When the batch size was 64, the model with a negative sampling size of 7 achieved the highest performance at 85.70%. This was higher than the accuracy of the model with the same configuration but a different number of negative samples. For example, the accuracies of the models with a negative sampling sizes of 15 and 31 are 84.99% and 85.29%, respectively. The results are organised in Table 3.8.

Table 3.8 Rank@1 accuracy (%) with negative sampling sizes.

Sampling Size	**7**	**15**	**31**
Rank@1	**85.70**	84.99	85.29

3.2.3.4 Performance on Utterance-Transcript Matching

The model with the optimal configuration achieved the results shown in Table 3.9. The accuracy of finding an utterance given a transcript was higher than that on finding a transcript given an utterance.

Table 3.9 Rank@k accuracy (%) on the AMI test dataset of the proposed method in each direction. Utt. and Tra. stand for utterance and transcript. For example, Utt.-to-Tra. means that an utterance is given and the corresponding transcript is matched.

Rank@1 (avg.)	**Direction**	**Rank@1**	**Rank@2**	**Rank@3**
85.70	Utt.-to-Tra.	85.11	94.37	97.23
	Tra.-to-Utt.	86.30	94.94	97.64

In addition to Rank@k accuracy, the WER of the output of the matching model was also measured. This can be interpreted as the accuracy of the data generated using the utterance and transcript matching model. To compute the WER, the transcript of the anchor was treated as the reference transcript and the transcript predicted as the positive sample was treated as the hypothesis from an ASR system. For example, there were four substitutions between the reference, ['WHICH', 'IS', 'THE', 'HIGHEST', 'EXACTLY'], and the hypothesis ['YOU', 'OH', 'GO', 'AWAY']. In the result, substitutions were most observed.

Table 3.10 Evaluation the proposed method in Word Error Rate.

WER	**Insertion**	**Deletion**	**Substitution**
10.35%	2.29%	2.04%	6.02%

3.2.4 Conclusion

In this section, a method for matching an utterance and a transcript using deep metric learning was proposed. The model trained on the AMI corpus using bidirectional NT-Xent loss with negative sampling achieved 85.70% accuracy on a task of finding an utterance or a transcript among 10 samples when their corresponding transcript or utterance was given. The matching model could also be useful for generating paired data sets for supervised learning for ASR without the need for ASR systems. Furthermore, it could function as a weak ASR model when utterance and transcript segments are available without pairing information. When the performance of the model was measured using WER between the paired utterance and transcript, it was 10.35%.

3.3 Summary

Transcript information at the utterance level was explored in this chapter. First, the amount of transcript information retained in the representation was measured by aligning the cosine distance of the utterance representations with the edit distance of the transcripts. The correlation between these distances demonstrated a strong positive association when self-supervised speech representations, such as HuBERT, were used. To leverage this transcript information, a method to match utterances and transcripts was proposed. The results demonstrated that the edit distance between transcripts could be transferred into a distance metric between utterance representations. This finding suggested the potential for generating paired data from unpaired speech and text data using the matching method.

Chapter 4

Word Error Rate Estimation of Automatically Transcribed Data

When humans assess the quality of something, such as an image, video, speech and translation, the assessment is subjective because the only reliable way to evaluate the quality perceived by humans is to ask for their opinion. However, subjective quality assessments are impractical for most applications due to the involvement of humans, making the process time-consuming and costly. As a result, quality has been quantified and evaluated implicitly through automatic measurements for image (Damera-Venkata et al., 2000), video (Seshadrinathan et al., 2010), speech (Voran, 1999) and translation (Su et al., 1992). For example, one of the most commonly used metrics for machine translation is BiLingual Evaluation Understudy (BLEU) (Papineni et al., 2002).

In the case of Automatic Speech Recognition (ASR), evaluation metrics are typically based on quantity rather than quality. One of the most popular metrics, such as Word Error Rate (WER), measures the ratio between the number of errors in ASR output and the total number of words in the manual transcript. However, in the context of data selection for semi-supervised learning, the Error Rates (ERs) of the system's output can be interpreted as an indicator of transcription quality. For example, an ASR system trained on transcripts with fewer errors should outperform one trained on transcripts with more errors. Therefore, for semi-supervised learning, the quality of ASR transcripts with a low error rate is considered higher than that of transcripts with a high error rate. Although it is referred to as *quality*, it can be measured by comparing the ASR output to a gold-standard transcript. For instance, automatically transcribed utterances can be compared to manually transcribed ones. When gold-standard transcripts are available, the quality of ASR transcripts can be easily quantified using typical ASR system evaluation metrics, such as WER.

However, manual transcription remains still costly and time-consuming, leading to research in quality estimation without reference transcripts. This approach has been applied not only to ASR transcripts but also to Machine Translation (MT), which was introduced by the workshop on statistical machine translation (Callison-Burch et al., 2012). The task's potential includes determining whether output requires human post-editing or selecting the best translation from the output of multiple systems. Similarly, Louis and Nenkova (2013) proposed a quality estimation method for summary content without gold-standard human summaries. For speech recognition, researchers have utilised confidence measures to assess the quality of ASR system output since the early 1990s (Jiang, 2005). Confidence in the model's output indicates the reliability of its recognition decision, and it can be computed using posterior probability (Young, 1994) or likelihood ratio (Rose et al., 1995). This information is derived directly from ASR decoding, making confidence estimation inherently dependent on the ASR system.

These methods, based on confidence measures, address the need for manual transcripts by estimating the quality of ASR output during decoding. However, there has been a growing demand for quality estimation without access to the ASR system, especially as more transcripts are collected without access to the internal process of proprietary or industrial ASR systems. This demand has led to the development of reference-free quality estimation methods that do not rely on ASR decoding. To eliminate dependency on information obtained from ASR decoding, Negri et al. (2014) divided features into two types: those that require ASR decoding (glass-box features) and those that do not (black-box features). For example, word confidence and acoustic probability were glass-box features, whereas the number of silences per second and the percentage of content words in the hypothesis were black-box features. Experimental results demonstrated that these black-box features could help improve the performance of the Quality Estimation (QE) model when combined with glass-box features. Similarly, Ali and Renals (2018) explored quality estimation using black-box features via neural networks.

In this chapter, the first research question—*Is WER estimation without ASR decoding effective for semi-supervised learning for ASR?*—is further explored by investigating methods for WER estimation without relying on ASR decoding. Three models for WER estimation are introduced, each building upon the strengths of the previous approaches. First, the Deep Metric Learning (DML)-based WER (DML-WER) estimation model is presented in Section 4.1. This model computes WER by learning a distance metric between utterance-level and transcript-level representations using Deep Neural Networks (DNNs). It builds on the findings of the previous chapter, where the cosine distance between utterance and transcript representations was shown to correlate with the edit distance between their corresponding transcripts. Next, a fast estimation model, Fe-WER, is introduced in Section 4.2. Unlike DML-WER, Fe-WER employs an Multi-Layer Perceptron (MLP) to directly estimate WER from the representations without relying on a learned distance metric. Finally, this model is extended to estimate not only the overall WER but also the insertion, deletion, and substitution ERs, resulting in the Multi-Target Regression for Error Rates (MTR-ER4) model in Section 4.3.

4.1 Deep Metric Learning for Word Error Rate Estimation

WER is a commonly used metric for evaluating ASR systems. It represents the ratio of substitution, insertion, and deletion errors in an ASR transcript compared to the total number of words in an manual transcript. In certain scenarios, it can be highly beneficial to estimate the WER of ASR output, especially when the ground-truth transcript is unavailable. For example, a WER estimation model can be employed to rank ASR transcripts (Jalalvand et al., 2015) or to select unlabelled data for ASR self-training (Chen et al., 2020b; Kahn et al., 2020a; Xu et al., 2021). Another application is filtering out training data with high-WER transcripts, especially when such data are collected from the internet. To achieve optimal ASR performance with the selected data, high-WER samples typically need to be excluded from ASR training, especially in the case of recent ASR models, such as Whisper (Radford et al., 2023), which are trained on large amounts of data collected from the internet. When dealing with large amounts of data, the computational efficiency of a WER estimator becomes crucial.

One straightforward approach to estimate the WER of ASR output is to rely on confidence scores generated by the ASR system itself (Jeon et al., 2020; Kumar et al., 2020). This method avoids the need to build an additional WER prediction model. However, this approach carries a risk of bias and—as will be shown—tends to underperform compared to dedicated WER estimation methods. Additionally, confidence scores are not well aligned with WER, particularly due to their inability to predict deletion errors effectively.

In the previous chapter, it was demonstrated that the edit distance used for WER is correlated with the cosine distance between utterance-level representations. Based on this finding, WER estimation models are developed by using utterance representations generated through average pooling over frame-level representations, as this approach has been reported to outperform the *CLS* token in Natural Language Processing (NLP) tasks, such as semantic textual similarity, as shown by Reimers and Gurevych (2019). Additionally, to transform the edit distance into WER, it needs to be normalised by the number of words in the manual transcript. While a complex estimation model could be built to estimate this word count, leveraging existing information from the data itself is often a more reliable approach. For example, rather than estimating the number of words in a spoken utterance, the duration of the utterance could be used. Although duration may not be exactly proportional to word count, it provides a reasonable correlation. Assuming that duration is correlated with word count, WER can be estimated directly using the cosine distance between utterance-level representations. Thus, the utterance and transcript matching model is extended in this section to estimate WER by incorporating utterance length information.

4.1.1 Proposed Method

4.1.1.1 Architecture

The proposed method consists of two parts: learning representations for a matching task and deep metric learning for WER estimation. The first part involves mapping representations into a new space using the matching model investigated in Section 3.2. The second part is a WER estimation model based on deep metric learning illustrated in Figure 4.1. This model uses two encoders, f and g, for an utterance and an ASR transcript, respectively. Each input is mapped into a shared feature space by the pre-trained matching model trained, where the cosine distance between the representations is correlated with the edit distance between the manual transcript and the ASR transcript for the utterance. When estimating WER, the duration of the utterance is also considered. The shared DNNs consist of multi-layer neural

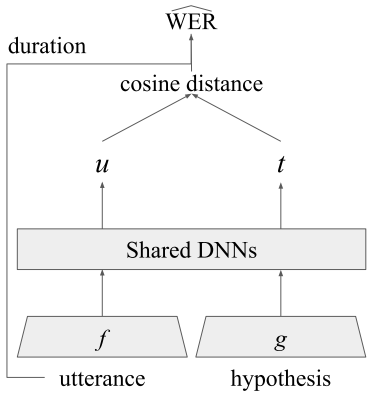


Figure 4.1 Deep Metric Learning for WER.

networks. The cosine distance between the mapped representations u and t is converted into the edit distance as follows:

$$\widehat{\text{ED}}(u,t) = \tau \ln \frac{2 + d_{\cos}(u,t)}{2 - d_{\cos}(u,t) + \varepsilon} \tag{4.1}$$

where τ is a temperature parameter for smoothing and $d_{\cos}(\cdot,\cdot)$ is the cosine distance. Since WER is the ratio of the edit distance to the number of words, the utterance duration is used instead of estimating the number of words to avoid adding uncertainty.

$$\widehat{\mathrm{WER}} = \frac{\widehat{\mathrm{ED}}(u_i,t_j)}{\mathrm{Duration}_i} \tag{4.2}$$

4.1.1.2 Training objective

The Mean Squared Error (MSE) between the actual WER and the estimated $\widehat{\mathrm{WER}}$ is used as the objective function to train the MLP. Here, WER represents the error rate between manual transcripts and ASR transcripts and $\widehat{\mathrm{WER}}$ is the estimation by the model.

$$\mathrm{MSE} = \frac{\sum_{i=1}^{N}(\mathrm{WER}_i - \widehat{\mathrm{WER}}_i)^2}{N}$$

where N is the number of instances in the dataset and i is the index of each instance.

4.1.1.3 Evaluation metrics

Pearson Correlation Coefficient (PCC) and Root Mean Square Error (RMSE) are used as evaluation metrics. A PCC value close to 1 indicates that two variables change in the same direction, while a value close -1 indicates that they change in opposite directions.

$$\frac{\sum_{i=1}^{N}(\mathrm{WER}_i - \mu_{\mathrm{WER}})(\widehat{\mathrm{WER}}_i - \mu_{\widehat{\mathrm{WER}}})}{\sqrt{\sum_{i=1}^{N}(\mathrm{WER}_i - \mu_{\mathrm{WER}})^2 \sum_{i=1}^{N}(\widehat{\mathrm{WER}}_i - \mu_{\widehat{\mathrm{WER}}})^2}}$$

where μ_{WER} is the mean of WER. Lastly, the ratio between the WER weighted by the number of words in reference transcripts $\mathrm{WER}_{\mathrm{wrd}}$ and $\widehat{\mathrm{WER}}_{\mathrm{dur}}$ is also measured.

$$\mathrm{WERR} = \frac{|\mathrm{WER}_{\mathrm{wrd}} - \widehat{\mathrm{WER}}_{\mathrm{dur}}|}{\mathrm{WER}_{\mathrm{wrd}}}.$$

4.1.2 Experimental Setup

4.1.2.1 Data

The TED-LIUM3 (TL3) (Hernandez et al., 2018) corpus was used as for WER estimation. This corpus consists of audio and closed captions extracted from TED Talks. The audio was segmented using an ASR system, and only the segments that perfectly align with the closed captions at the word level were included. As a result, the segments in the corpus were used without any additional preprocessing, such as voice activity detection. The segments were selected for the corpus when the Whisper large-v2 [1] was employed for transcribing the corpus, as it demonstrated comparable performance on TL3 and is publicly available, ensuring reproducibility. For text normalisation, the transcripts were lower-cased and punctuation was removed using NeMo Text Processing[2] (Zhang et al., 2021). The transcribed data were highly imbalanced due to the high volume of utterances with no errors. To address this issue, the number of ASR transcripts with a WER of 0 was restricted to the sum of the numbers of transcripts in the second and third histogram bins order of frequency. Additionally, utterances up to 10 seconds in length were selected, and WER was clamped between 0% and 100%. Whisper's text normaliser was modified to prevent the replacement of numeric expressions with Arabic numerals. The statistics of the selected data are summarised in Table 4.1.

Table 4.1 Statistics of the sets of data selected. ASR transcripts were generated by Whisper large-v2.

Dataset	**# Seg.**	**Total Dur. (h)**	**Avg. Dur.**	**Avg. # Wrd.**	**Avg. WER**	**Std. Dev. of WER**	**$\mathrm{WER}_{\mathrm{wrd}}$**
test	842	1.41	6.05	16.72	0.1429	0.1997	0.1088
dev	1,034	1.70	5.93	17.72	0.1532	0.2247	0.1225
train	123,255	200.55	5.86	17.04	0.2434	0.3209	0.2029

4.1.2.2 Self-Supervised Learning Representations

Self-Supervised Learning Representations (SSLRs) for utterances and ASR transcripts were selected based on their performance on benchmarks, including the Speech processing Universal PERformance Benchmark (SUPERB) (wen Yang et al., 2021), General Language Understanding Evaluation (GLUE) (Wang et al., 2018) and SuperGLUE (Wang et al., 2019). These benchmarks assess models on various tasks, such as phoneme recognition and paraphrase detection. Additionally, two SSLR models in (Chowdhury and Ali, 2023) used for WER estimation with Bidirectional LSTM (BiLSTM) were included. Summary information on these models, including model size and the number of parameters, is provided in Table 4.2.

[1]https://github.com/openai/whisper
[2]https://github.com/NVIDIA/NeMo-text-processing

Table 4.2 Summary information of SSLRs.

Input Type	Model	Abbr.	Size	# Parameters
Utterance	data2vec (Baevski et al., 2022)	DA	Large	313M
	HuBERT (Hsu et al., 2021)	HU	Large	316M
	WavLM (Chen et al., 2022)	WA	Large	317M
	XLSR-53 (Conneau et al., 2021)	XS	Large	315M
Transcript	DeBERTa-V3 (He et al., 2021a)	DE	Large	283M
	GPT-2 (Radford et al., 2019)	GP	Medium	355M
	RoBERTa (Radford et al., 2023)	RO	Large	355M
	XLM-R (Conneau et al., 2020)	XM	Large	560M

4.1.2.3 Baseline Word Error Rate Estimators

The proposed model was compared with a method based on confidence score (WER-CS). For sequence-level confidence scoring, the log probability of Whisper large-v2 over the output tokens was averaged and subtracted from 1. Two strategies were employed: greedy decoding only and full decoding. The full decoding strategy included a beam size of 5, greedy decoding with the 5 best hypotheses and sampling temperature settings ranging from 0 to 1 in increments of 0.2.

4.1.2.4 Representations Mapped by a Matching Model

Distant alignment models were trained using combinations of different SSLRs on a task of matching an utterance to its corresponding transcript. The model had a 2-layer separate architecture with BiNT-Xent loss and a batch size of 64. For more details, readers can refer to Section 3.2.

4.1.2.5 Hyper-parameters

The model employed an MLP for the shared DNNs. It consists of an input layer, two hidden layers and an output layer. All layers were 1024-dimensional with an Exponential Linear Unit (ELU) activation function. hyper-parameters were determined by grid search: a learning rates were selected from [0.001, 0.0001, 0.00001], temperature from [1, 8, 32] and an alpha value for the ELU function from [1,4,16].

4.1.3 Results

WER models with deep metric learning showed varying performance depending on the combination of utterance-level and transcript-level representations. The first observation from Table 4.3 is the performance difference between two groups: one group consists of combinations of SSLR representations, while the other group uses representations mapped by the model trained on the matching task.

On the left side of the table, the performance of WER estimation models varies significantly based on the representations employed. For example, the RMSE of the WER estimation model using XS is approximately 0.69, while the RMSE of the model using DA ranges from 0.1511 to 0.3675. The effect of the ASR transcript representation on RMSE is not as pronounced as that of the utterance representation, but it is still noticeable. For example, the RMSE of the WER estimation model using DA is approximately 0.1511 and 0.1663 with XM and RO, while 0.3004 and 0.3675 with GP and DE, respectively.

On the right side of the table, the performance of the WER models is within a relatively narrow range. For example, RMSE ranges from 0.0965 to 0.1285 and PCC from 0.8298 to 0.8946 across all types of representations, while on the left side, they range from 0.1511 to 0.6960 and from 0.7394 to 0.0895, respectively. These results imply that deep metric learning for WER is more stable when the representations are mapped onto a new space by the matching model than when they are not. Otherwise, the performance optimisation of the models is highly dependent on the combination of representations.

4.1.3.1 Comparison with Baselines

The proposed models were compared with the baseline, WER-CS, as described in Section 4.1.2.3. They outperformed the baselines in RMSE by at least 36.99%. In terms of PCC, the performance of DML-WER with mapped representations supassed both baselines. The comparison results are shown in Table 4.4.

4.1.4 Conclusion

A WER estimation model using deep metric learning with mapped representations is proposed. The results demonstrate that the representations mapped by a separately trained matching model contribute to the stable convergence of performance. Additionally, the proposed method with mapped representations outperformed the confidence score baseline in both RMSE and PCC. The proposed model achieved 0.0965 in RMSE and 0.8946 in PCC on the TL3 dataset.

Table 4.3 Results of WER estimation using deep metric learning with different SSLRs combinations on TL3 dev. All models were trained with three seeds.

SSLR		w/o Mapping		w/ Mapping	
Utt.	**Hyp.**	**RMSE↓**	**PCC↑**	**RMSE↓**	**PCC↑**
DA	DE	.3675±.002	.4945±.001	.1285±.001	.8298±.002
DA	GP	.3004±.001	.5849±.001	.1190±.000	.8492±.001
DA	RO	.1663±.000	.7467±.000	.1161±.000	.8569±.001
DA	XM	**.1511±.000**	**.7394±.000**	.1165±.001	.8580±.003
HU	DE	.4438±.001	.2777±.000	.1108±.001	.8689±.002
HU	GP	.3179±.001	.3264±.000	.1124±.001	.8659±.002
HU	RO	.3381±.000	.3039±.000	.1122±.001	.8691±.003
HU	XM	.3400±.001	.3129±.000	.1060±.000	.8803±.002
WA	DE	.4814±.000	.2538±.000	.1079±.002	.8753±.004
WA	GP	.4017±.000	.2789±.000	.1044±.000	.8857±.000
WA	RO	.4798±.000	.2228±.000	.1020±.000	.8941±.002
WA	XM	.4729±.000	.2616±.000	**.0965±.000**	**.8946±.013**
XS	DE	.6960±.000	.1201±.002	.1169±.001	.8525±.002
XS	GP	.6931±.000	.0895±.002	.1172±.001	.8537±.002
XS	RO	.6930±.001	.1748±.000	.1145±.000	.8600±.001
XS	XM	.6924±.000	.1327±.000	.1098±.001	.8710±.003

Table 4.4 RMSE and PCC of baseline systems on TL3 test. WER_{wrd} is a reference WER weighted by words. $\widehat{\text{WER}}_{\text{dur}}$ is the WER estimate weighted by duration. † is the proposed method.

	SSLRs	**RMSE↓**	**PCC↑**	$\textbf{WER}_{\textbf{wrd}}$	$\widehat{\textbf{WER}}_{\text{dur}}$	**WERR↓**
WER-CS + full	-	0.2611	0.5654	8.40%	31.85%	2.7916
WER-CS + greedy	-	0.2546	0.6944	10.88%	33.34%	2.0643
DML-WER w/o Mapping	DA,XM	0.1604	0.6460	10.88%	13.07%	0.2023
†DML-WER w/ Mapping	WA,XM	0.0956	0.8784	10.88%	10.66%	0.0202

4.2 Fast Word Error Rate Estimation

As demonstrated in the previous section, the WER estimation model using deep metric learning (DML-WER) demonstrated that WER can be estimated solely from utterance and transcript representations, without relying on ASR decoding. While DML-WER achieved promising results, it has several limitations. First, the method requires a two-step training process: representations are first mapped into a new space through unpaired matching tasks, followed by the training of separate networks to estimate WER. Additionally, both steps depend on complex sampling techniques and require careful hyper-parameter fine-tuning to ensure stable convergence. These challenges constrain the broader applicability of the approach.

Recently, a similar approach has been proposed to directly estimate the WER of ASR system output. For example, e-WER3 (Chowdhury and Ali, 2023) employs bidirectional long short-term memory (BiLSTM) networks to extract features from speech, while the features for text are averaged over tokens. Then, WER is directly estimated using an MLP with these features. Although this method has made impressive progress in WER estimation by simplifying the overall process, there are still several questions that have not been fully studied.

Firstly, e-WER3 relies on BiLSTMs, which are computationally intensive for long sequences, such as spoken utterances. This limits its scalability for training with long speech segments. Secondly, the performance of the estimator depends on the input features for both speech and text. Thus, exploring different combinations of SSLRs for speech and text is crucial for optimising WER estimation performance. Lastly, performance needs to be analysed across various data attributes, such as utterance lengths and speaker variation, in addition to standard evaluation metrics.

In this section, a Fast estimation model for WER (Fe-WER) consisting of speech and text encoders, feature aggregators and a WER estimator, is proposed. The SSLRs, aggregated by average pooling, are used to directly estimate WER with an MLP. This method will be evaluated from both accuracy and efficiency perspectives. The contributions of this section are as follows:

1. This section proposes a WER estimation model, Fe-WER, using average pooling, which outperforms the baseline model in computational efficiency without sacrificing performance.

2. Experimental results demonstrate that the combination of HuBERT (Hsu et al., 2021) and XLM-R (Conneau et al., 2020) achieves the best performance for WER estimation.

3. A comparative analysis of the distributions of reference WER and WER estimates is presented including an examination of the average values per speaker.

4.2.1 Proposed Method

4.2.1.1 Architecture

Fe-WER (see Figure 4.2) is based on a two-tower architecture, which maps different representations into a shared space to capture the similarity between two inputs. The proposed model consists of two aggregators—one for speech and another for text—and fully connected layers that perform the WER estimation. The aggregators convert the features extracted by SSLRs into sequence-level representations. These two sequence-level representations are concatenated and input into an MLP consisting of fully connected layers with a Rectified Linear Unit (ReLU) activation function. A sigmoid function is applied to the output. The

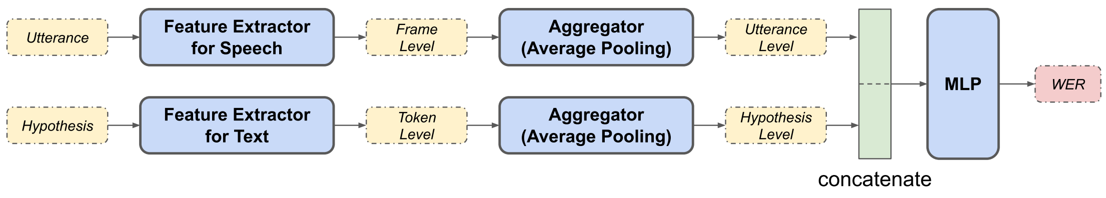

Utterance
Feature Extractor for Speech
Frame Level
Aggregator (Average Pooling)
Utterance Level
Hypothesis
Feature Extractor for Text
Token Level
Aggregator (Average Pooling)
Hypothesis Level
MLP
WER
concatenate

Figure 4.2 Illustration of the proposed method for WER estimation

WER estimate $\widehat{\text{WER}}$ is defined as:

$$\widehat{\text{WER}} = \text{MLP}(\text{concat}(\text{Avg}(f(s)), \text{Avg}(g(t))))$$

where Avg is the average pooling function, $f(\cdot)$ and $g(\cdot)$ are encoders for speech and text, respectively, and s and t are a spoken utterance and an ASR transcript, respectively.

4.2.1.2 Weighted Word Error Rate Estimate

When reference transcripts are available, WER is weighted by the number of words in the reference transcripts, accounting for the length of utterances. However, for weighted WER estimation, where reference transcripts are not available, the number of words in the reference transcripts cannot not be utilised. Previously, the WER of a dataset was estimated by computing the mean of all WER values from ASR transcripts, regardless of their length. Therefore, in the absence of reference transcripts, WER is weighted using the duration of the utterances instead of the number of words. The weighted WER estimate is defined as:

$$\widehat{\text{WER}}_{\text{dur}} = \frac{\sum_{i=1}^{N}(\widehat{\text{WER}}_i \times \text{Duration}_i)}{\sum_{i=1}^{N}(\text{Duration}_i)}$$

where i is the index of a pair consisting of an utterance and its corresponding hypothesis. When WER is weighted by the number of words in the reference transcripts, it is denoted as WER_{wrd}.

4.2.2 Experiment Setup

For the comparison of WER estimation models, the datasets and the SSLR models used for DML-WER in Section 4.1 were employed.

4.2.2.1 Baseline Word Error Rate Estimators

A method using BiLSTM was implemented as a baseline. A single-layer BiLSTM was used to aggregate SSLR representations, with input and hidden feature sizes matching the input dimensions. For further details, readers can refer to e-WER3 (Chowdhury and Ali, 2023).

4.2.2.2 Fast estimation of Word Error Rate

Average pooling over the frame or token dimension was used as the aggregator. hyper-parameters were selected via grid search. Fe-WER includes an MLP with two hidden layers and one output layer, activated by ReLU and Sigmoid functions, respectively. Each layer's output is normalised, and dropout (0.1) is applied to the hidden layers. The fully connected layers consist of three layers with 600, 32, and 1 nodes on top of 2048-dimensional input features. The model was trained with an Adam optimiser (learning rate: 1e-3), a cosine annealing scheduler and early stopping at 40 epochs.

4.2.3 Results

First, BiLSTM and average pooling aggregators are compared across different combinations of SSLRs. Next, the WER estimation models with the best SSLR combination are compared with a baseline using confidence scoring. This is followed by an analysis of WER estimation at the utterance level and a comparison of inference speed.

4.2.3.1 Aggregators

Aggregators using BiLSTM and average pooling were compared with combinations of SSLRs in Section 4.1.2.2. First, RMSE and PCC tend to improve with average pooling in 13 out of 16 combinations. Second, the best combinations were DA and XM for BiLSTM and HU and XM for average pooling. The latter outperformed the former by 0.0099 in RMSE and 0.0228 in PCC on Ted-lium 3 (TL3) dev. Results are summarised in Table 4.5.

Table 4.5 Results of BiLSTM and Average pooling aggregators with different SSLRs combinations on TL3 dev. All models were trained with three seeds.

SSLR		**BiLSTM**		**Avg. Pool.**	
Utt.	**Hyp.**	**RMSE↓**	**PCC↑**	**RMSE↓**	**PCC↑**
DA	DE	.1185±.001	.8490±.004	.1213±.000	.8425±.001
DA	GP	.1254±.005	.8405±.008	.1185±.001	.8512±.002
DA	RO	.1193±.002	.8491±.008	.1190±.002	.8486±.004
DA	XM	**.1111±.008**	**.8700±.018**	.1137±.001	.8637±.002
HU	DE	.1216±.002	.8398±.004	.1105±.002	.8702±.005
HU	GP	.1233±.002	.8387±.005	.1093±.001	.8741±.001
HU	RO	.1227±.004	.8363±.011	.1123±.003	.8676±.006
HU	XM	.1212±.011	.8418±.032	**.1012±.003**	**.8928±.007**
WA	DE	.1289±.005	.8200±.014	.1164±.002	.8551±.003
WA	GP	.1270±.003	.8245±.009	.1111±.002	.8709±.006
WA	RO	.1210±.004	.8420±.013	.1167±.002	.8561±.004
WA	XM	.1172±.005	.8520±.015	.1099±.002	.8734±.005
XS	DE	.1289±.003	.8191±.011	.1216±.002	.8412±.006
XS	GP	.1200±.003	.8467±.008	.1155±.001	.8585±.002
XS	RO	.1285±.003	.8226±.006	.1161±.003	.8567±.007
XS	XM	.1199±.005	.8474±.009	.1101±.001	.8717±.003

4.2.3.2 Comparison with Baselines

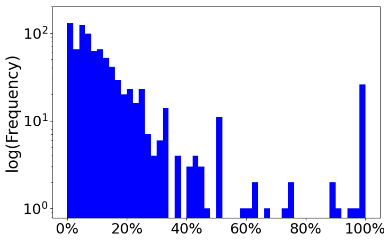
The proposed model, which uses an average pooling aggregator with HU and XM, is compared to the baseline, a model using BiLSTM with DA and XM. The proposed model outperformed the baseline in RMSE and PCC by at least 14.10% and 1.22%, respectively. The comparison results are shown in Table 4.6.

Table 4.6 RMSE and PCC of baseline systems on TL3 test. $\mathrm{WER}_{\mathrm{wrd}}$ is a reference WER weighted by words. $\widehat{\mathrm{WER}}_{\mathrm{dur}}$ is the WER estimate weighted by duration. † is the proposed method.

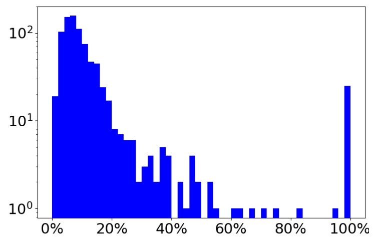
	SSLRs	RMSE↓	PCC↑	$\mathrm{WER}_{\mathrm{wrd}}$	$\widehat{\mathrm{WER}}_{\mathrm{dur}}$	WERR↓
BiLSTM	DA,XM	0.1071	0.8793	10.88%	10.96%	0.0073
†Avg. Pool.	HU,XM	0.0920	0.8900	10.88%	10.39%	0.0450

4.2.3.3 Distributions of Word Error Rate Reference and Word Error Rate Estimates

The histograms of reference WERs and WER estimates on TL3 test are visualised in Figure 4.3. The distribution of Fe-WER estimates is similar to that of the reference WERs. However, the frequency of reference WERs peaks in the [0%, 2%) range in Figure 4.3(a), while the estimates peak in the [4%, 8%) range in Figure 4.3(b). This discrepancy may be due to the Sigmoid function outputting small values rather than 0. Additionally, WER estimates around 20% are less frequent than reference WERs. In this range, three or more insertions in a row are frequently observed in the ASR transcripts. Recognising these words as one insertion error could have led to the lower estimates.



(a) reference WER



(b) WER estimate

Figure 4.3 Histograms on TL3 test

4.2.3.4 Average Word Error Rate Reference and Estimate per Speaker

The distributions of average reference WER and WER estimate per speaker are similar (see Figure 4.4). The high average reference WER for Speaker 5 is due to the majority of shorter utterances, which have a low resolution of WER. For example, the WER of a spoken utterance for a single word is either 0 or at least 100%. For Speaker 16, the average WER estimate is higher than the average reference WER target. The phenomenon of high WER estimate was discussed in Section 4.2.3.3.

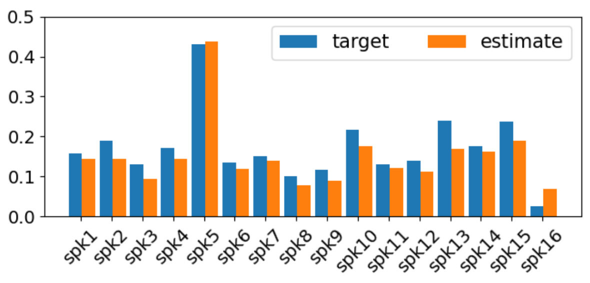


Figure 4.4 Average WER per each speaker.

4.2.3.5 Inference Speed

The inference time of the WER estimators was measured on a single NVIDIA RTX A6000 GPU with a batch size of 1, including encoding time. The model using BiLSTM had an inference time of 18.64 seconds, while the proposed method's inference time was significantly shorter at 5.42 seconds, reducing the inference time by approximately 70.92%. The details are summarised in Table 4.7. These results demonstrate that Fe-WER is more efficient, making it preferable for applications requiring quick WER estimation.

Table 4.7 RTF of BiLSTM and Avg. Pool. with HU and XM on TL3 test. Total duration is about 5,223 seconds. RTF: total time ÷ total duration. † is the proposed method.

	BiLSTM	†Avg. Pool.
Feature extraction		
+ utterance	2.72	2.72
+ transcript	0.93	0.93
Aggregation	5.28	ε
Feedforward	9.71	1.77
Total	18.64	5.42
RTF	0.003569	0.001038

4.2.4 Conclusion

In this section, a fast WER estimator has been proposed. The model consists of SSLR encoders for speech and text, average pooling aggregators and an MLP estimator. The WER estimator outperforms the e-WER3 baseline by relative 17.93% and 5.82% in RMSE and PCC, respectively. Additionally, the distributions of reference WER and WER estimates were analysed across utterance lengths and speakers. Furthermore, the experimental results show that the inference speed has significantly improved, for example, 4.6 times faster than the e-WER3 baseline, without any degradation in performance.

4.3 Multi-Target Regression for Error Rate Estimation

WER is computed by first aligning a reference transcript and an ASR transcript. Then, three types of errors in ASR are identified: insertions, deletions and substitutions. Several studies have been conducted to estimate these separate ERs. For example, Ogawa et al. (2016) investigated the task of error type classification, distinguishing between correct, insertion, deletion and substitution. Additionally, there have been a few studies focused on detecting deletions in ASR transcripts. Seigel and Woodland (2014) adapted confidence scores to detect deletions in ASR output. Another study (Ragni et al., 2018) predicted the deletion ER by using bidirectional recurrent neural networks. Deletion estimation was also combined for Word- and Utterance-level confidence (Qiu et al., 2021).

However, estimating each of the error types that constitute WER using SSLRs with average pooling has not yet been explored. In this work, we present a new Multi-Target Regression (MTR) model for WER estimation and individual ERs. MTR is a method used to predict multiple numeric values simultaneously (Spyromitros-Xioufis et al., 2016). By sharing neural network parameters, the dependencies between targets are leveraged, enabling the model to learn the estimation task (Reyes and Ventura, 2019).

For ER estimation, three architectures built upon Fe-WER 4.2.1 using DNNs are compared. First, Fe-WER in Section in Section 4.2.1 is extended to estimate insertion, deletion and substitution ERs. Second, overall WER is added as a target with network parameters shared across all tasks. Next, non-shared parameters for WER estimation are trained jointly with individual ERs. Alternatively, individual ERs and WER are estimated by predicting the raw edit counts including the word count. Among these architectures, the MTR model for WER and the three ERs, referred to as MTR-ER4, show the best performance in terms of average RMSE and PCC across transcripts produced by different ASR systems.

The aims of this section are:

- To present an ER estimation method for ASR transcripts, which provides insertion, deletion, substitution rates and WER
- To demonstrate the benefits of using shared parameters for multiple ER targets

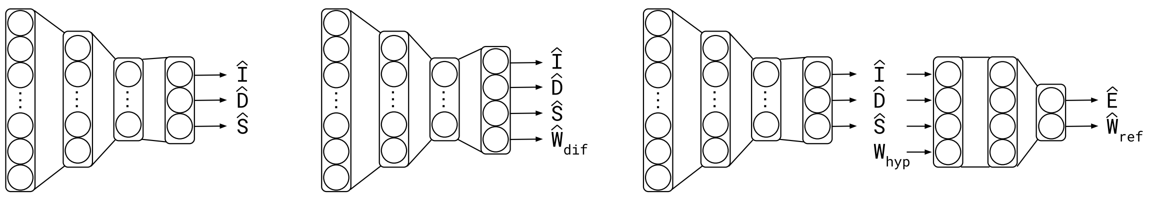
4.3.1 Proposed Method

WER is computed using the substitution S, insertion I and deletion D counts in an ASR transcript in relation to a reference transcript, as well as the number of words in the reference transcript, W_{ref}.

$$\text{WER} = \frac{S+I+D}{W_{\text{ref}}} = \text{ER}_S + \text{ER}_I + \text{ER}_D \tag{4.3}$$

where $\text{ER}_I, \text{ER}_D, \text{ER}_S$ are the insertion, deletion, substitution ERs respectively, e.g., $\text{ER}_I = \frac{I}{W_{\text{ref}}}$. For WER estimation, there are two approaches: Error Count (EC) estimation, ER estimation. The first approach involves predicting the individual error counts, such as S, I and D. For W_{ref}, the number of words in an ASR transcript, W_{hyp}, can be utilised. For example, the number of words in a reference transcript, W_{ref}, can be calculated by subtracting I from and adding D to W_{hyp}. Another approach is to predict ERs directly: $\text{ER}_I, \text{ER}_D, \text{ER}_S$. In this case, the word count is not needed. EC estimation and ER estimation models using MTR are noted as MTR-EC and MTR-ER and they are illustrated in Figure 4.5 and Figure 4.6, respectively.

4.3.1.1 Word Error Rate Estimation with Multi-target Regression for Error Counts



(a) 3 targets (b) 4 targets (c) Joint Training

Figure 4.5 Illustration of MTR for EC estimation models. $\widehat{I}$, $\widehat{D}$, $\widehat{S}$ and $\widehat{E}$ are the numbers of insertions, deletions, substitutions and the total number of them. W_{hyp} and $\widehat{W}_{\text{ref}}$ are the counts of words in an ASR transcript and words in a reference transcript, respectively. $\widehat{W}_{\text{dif}}$ is the difference between W_{hyp} and W_{ref}. For simplicity, the feature extraction is omitted.

MTR-EC3 MTR-EC with three targets (MTR-EC3) in Figure 4.5(a) predicts $\widehat{I}$, $\widehat{D}$, $\widehat{S}$ simultaneously. The training objective is to minimise the sum of MSE of each count.

$$\sum_{y \in \{I,D,S\}} \frac{1}{N} \sum_{i=1}^{N} (y_i - \widehat{y}_i)^2 \tag{4.4}$$

where N is the total number of utterances in the training data. In this case, WER is estimated using the predicted counts.

$$\widehat{\text{WER}} = \frac{\widehat{I}+\widehat{D}+\widehat{S}}{W_{\text{hyp}} - \widehat{I} + \widehat{D}}. \tag{4.5}$$

MTR-EC4 In Figure 4.5(b), the difference $\widehat{W}_{\text{dif}}$ between W_{ref} and W_{hyp} is predicted by MTR-EC with four targets (MTR-EC4) in addition to the other ECs. The training objective is to minimise MSE of all counts and $\widehat{W}_{\text{dif}}$ in a similar way to Equation (4.4). In this case, WER is estimated using the predicted counts.

$$\widehat{\text{WER}} = \frac{\widehat{I}+\widehat{D}+\widehat{S}}{W_{\text{hyp}}+\widehat{W}_{\text{dif}}}. \tag{4.6}$$

JT-MTR-EC In Figure 4.5(c), the outputs of MTR-EC3, along with W_{hyp}, are input to MLPs to predict the total error count $\widehat{E}$ and $\widehat{W}_{\text{ref}}$. MTR-EC3 and the additional MLPs are jointly trained to minimise MSE of all counts, including $\widehat{E}$ and $\widehat{W}_{\text{ref}}$.

$$\sum_{y\in\{I,D,S,E,W_{\text{ref}}\}} \frac{1}{N}\sum_{i=1}^{N}(y_i-\widehat{y}_i)^2 \tag{4.7}$$

where E is the sum of all ECs. WER is estimated using the final outputs.

$$\widehat{\text{WER}} = \frac{\widehat{E}}{\widehat{W}_{\text{ref}}}. \tag{4.8}$$

4.3.1.2 Word Error Rate Estimation with Multi-target Regression for Error Rates

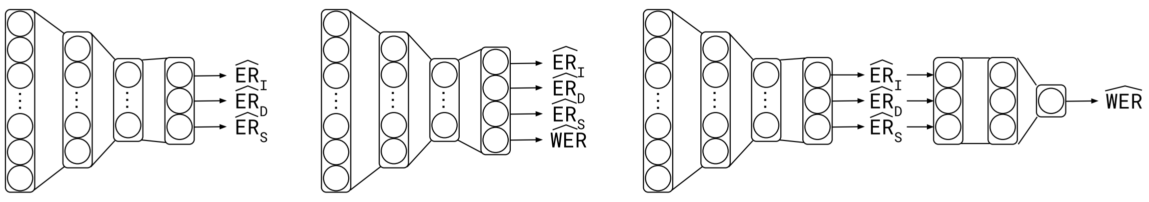


Figure 4.6 Illustration of MTR for ER estimation models. $\widehat{\text{ER}}_I$, $\widehat{\text{ER}}_S$, $\widehat{\text{ER}}_D$, $\widehat{\text{WER}}$ stands for insertion, deletion, substitution and word error rate, respectively. For simplicity, the feature extraction is omitted.

MTR-ER3 Similar to Section 4.3.1.1, $\widehat{\text{ER}}_I$, $\widehat{\text{ER}}_S$ and $\widehat{\text{ER}}_D$ are predicted simultaneously as depicted in Figure 4.6(a). The training objective of MTR-ER with three targets (MTR-ER3) is to minimise the MSE of the ERs. As defined in Equation (4.3), WER can be estimated by summing all the ERs as follows:

$$\widehat{\text{WER}} = \widehat{\text{ER}}_I + \widehat{\text{ER}}_D + \widehat{\text{ER}}_S. \tag{4.9}$$

MTR-ER4 Unlike MTR-EC4, $\widehat{\text{WER}}$ is directly predicted by MTR-ER with four targets (MTR-ER4). In addition to the ERs, WER is also added as a target. In Figure 4.6(b), the only difference from Section 4.3.1.2 is the size of the output layer. The training objective is to minimise the MSE of all the rates including WER.
JT-MTR-ER Additional MLPs are used to estimate $\widehat{\text{WER}}$ by using the ER estimates. As the ER estimates are enough for computing WER, additional information, such as W_{hyp}, is not added to the input to the MLP different to JT-MTR-EC.

4.3.1.3 Evaluation Metrics

For evaluation, the RMSE and PCC of each ER are measured. The PCC is defined as:

$$r_{\text{ER}} = \frac{\sum_{i=1}^{N}(\text{ER}_i - \mu_{\text{ER}})(\widehat{\text{ER}}_i - \mu_{\widehat{\text{ER}}})}{\sqrt{\sum_{i=1}^{N}(\text{ER}_i - \mu_{\text{ER}})^2 \sum_{i=1}^{N}(\widehat{\text{ER}}_i - \mu_{\widehat{\text{ER}}})^2}} \tag{4.10}$$

where $\text{ER} \in \{\text{ER}_I, \text{ER}_D, \text{ER}_S, \text{WER}\}$. For comparison across multiple ASR transcripts, the average RMSE and PCC over ASR transcripts are used.

4.3.2 Experiment Setup

4.3.2.1 Datasets

The same ASR corpus used in the previous section, TL3, is selected for WER estimation in this experiment. This choice ensures consistency and allows for direct comparison with a baseline system, Fe-WER. For training data, TL3 datasets are transcribed by various ASR system. The statistics of TL3 before preprocessing for WER estimation are summarised in Table 4.8.

Table 4.8 Statistics of TL3 datasets.

Dataset	**# Seg.**	**Total Dur. (h)**	**Avg. Dur.**	**Avg. # Wrd.**
test	2,582	4.589	6.398	17.12
dev	2,710	4.606	6.118	18.16
train	262,971	444.619	6.087	17.42

4.3.2.2 ASR Systems

To evaluate the performance of the ER estimation models, different types of ASR systems were employed to transcribe the utterances. Training data for WER estimation was generated by transcribing the TL3 train set using different ASR systems including the Whisper (Radford et al., 2023) transcripts used for the baseline: Conformer (Gulati et al., 2020), HuBERT (Hsu et al., 2021), Whisper, Chain (Povey et al., 2015), wav2vec 2.0 (Baevski et al., 2020), Transducer (Graves, 2012). These models and their weights were downloaded from online resources[3][4][5][6][7], while the Chain model was trained on 100 hours of LibriSpeech (Panayotov et al., 2015) using LF-MMI (Povey et al., 2016). The data was augmented by adjusting the speed and volume of the audio.

4.3.2.3 Data Selection

To address the imbalanced data distribution, the data selection strategy from Section 4.1.2.1 is adopted. Utterances shorter than 10 seconds were selected. As a result of this data selection process, the distributions of WER on the ASR transcripts are different. The statistics of the evaluation datasets are summarised in Table 4.9.

Table 4.9 Collections of ASR transcripts for evaluation.

Model	**Trained on**	**Language Model**	**# Seg.**	**Avg. WER(%)**	**Std. WER(%)**	**Avg. ER$_I$(%)**	**Avg. ER$_D$(%)**	**Avg. ER$_S$(%)**
Conformer	LS 960h	Transf.	1486	16.92	17.73	5.42	1.89	10.08
HuBERT	LS 960h	-	1305	12.65	13.28	1.57	1.95	9.13
Whisper	680k from net.	Transf.	842	14.29	19.97	5.82	4.29	4.73
Chain	LS 100h	3-gram	1939	23.41	18.79	5.28	3.95	14.41
Transducer	LS 960h	RNN	2270	16.43	15.78	2.61	2.07	6.54
wav2vec 2.0	LS 960h	-	1490	11.19	14.66	4.23	2.15	10.27

4.3.2.4 Error Rate Estimation Models

The ER estimation models are built upon the Fe-WER architecture described in Section 4.2 except the size of the output layer. The output sizes of the proposed models was various

[3]https://huggingface.co/speechbrain
[4]https://github.com/facebookresearch/fairseq
[5]https://github.com/openai/whisper
[6]https://github.com/kaldi-asr/kaldi
[7]https://github.com/speechbrain/speechbrain

according to the targets. The output sizes of MTR-EC3, MTR-EC4 and JT-MTR-EC were 3, 4 and 2, respectively. Those of MTR-ER3, MTR-ER4 and JT-MTR-ER were 3, 4 and 1, respectively. They are illustrated in Figure 4.5 and 4.6. For joint training, the output of EC or ER estimators were input into their own WER estimation layers including a hidden layer and an output layer. The size of the hidden layer matched the output of EC or ER estimators. Batch normalisation was applied to the hidden layer. The output was used for computing WER and the predicted WER was clamped between [0%, 100%].

4.3.2.5 Feature Extraction

The representations for speech and text were extracted using HuBERT (Hsu et al., 2021) and XLM-R (Conneau et al., 2020). The sizes of both models were large with 316M and 560M parameters, respectively. Their outputs were both 1024-dimensional vectors per frame and token, respectively, and averaged over time.

4.3.2.6 Hyper-parameters

An MLP consists of input, two hidden layers and an output layers. The size of the layers was selected from 300, 600 and 900 for the first hidden layer, while 16, 32 and 64 were used for the second hidden layer. The learning rate was chosen from the range 1e-4, 3e-4, 7e-4, 1e-3, 3e-3 and 7e-3 as well as the batch size from the range 128, 256, 512, 1024 and 2048. The output layer was activated by either HardSigmoid or Sigmoid. All hyper-parameters were selected via grid search. For the count estimation model, the activation function of the output layer was ReLU, since the targets were counts of insertions, deletions, substitutions and words, which are positive integers.

4.3.3 Results

The models were evaluated on both WER and ER estimation. First, the models were evaluated based on WER estimation performance for comparison with the baseline, Fe-WER. The performance was measured across six types of ASR transcripts and the results are summarised in Table 4.11 to 4.10. Second, they were then evaluated for ER estimation and the results are shown in Table 4.13 to 4.18.

4.3.3.1 Word Error Rate Estimation

For WER estimation, the MTR-ER4 model outperformed all the others, including the baseline, in terms of average RMSE and PCC across the six ASR transcripts. When comparing MTR-

ER and MTR-EC models, the overall performance of MTR-EC models lagged behind that of MTR-ER models. Additionally, MTR-EC4 and MTR-ER4 outperformed MTR-EC3 and MTR-ER3, respectively. Based on the results, it can be concluded that the performance of WER estimation models improves when predicting ER_S, ER_I, ER_D and WER simultaneously, rather than estimating the separate ERs or WER.

Table 4.10 Mean performance of ER estimation models on WER estimation.

	Mean	
	RMSE↓	PCC↑
Baseline		
Fe-WER	0.1044	0.7725
Proposed methods		
MTR-EC3	0.1118	0.7311
MTR-EC4	0.1110	0.7422
JT-MTR-EC	0.1295	0.6187
MTR-ER3	0.1156	0.7647
MTR-ER4	**0.1028**	**0.7795**
JT-MTR-ER	0.1040	0.7736

Table 4.11 Performance of ER estimation models on WER estimation of Conformer, HuBERT and Whisper's outputs.

	Conformer		HuBERT		Whisper	
	RMSE↓	PCC↑	RMSE↓	PCC↑	RMSE↓	PCC↑
Baseline						
Fe-WER	0.1124	0.7739	0.0872	0.7573	0.0949	0.8796
Proposed methods						
MTR-EC3	0.1236	0.7198	0.0959	0.6990	0.1007	0.8650
MTR-EC4	0.1194	0.7446	0.0880	0.7558	0.1075	0.8504
JT-MTR-EC	0.1429	0.6270	0.1048	0.6285	0.1129	0.8316
MTR-ER3	0.1311	0.7839	0.0892	0.7484	0.1076	0.8902
MTR-ER4	0.1131	0.7726	0.0869	0.7598	0.0917	0.8890
JT-MTR-ER	0.1141	0.7659	0.0862	0.7621	0.0972	0.8804

Table 4.12 Performance of ER estimation models on WER estimation of Chain, Transducer, and wav2vec 2.0's outputs.

	Chain		Transducer		wav2vec 2.0	
	RMSE↓	PCC↑	RMSE↓	PCC↑	RMSE↓	PCC↑
Baseline						
Fe-WER	0.1223	0.7595	0.1124	0.7739	0.0872	0.7573
Proposed methods						
MTR-EC3	0.1284	0.7350	0.1236	0.7198	0.0959	0.6990
MTR-EC4	0.1285	0.7428	0.1194	0.7446	0.0880	0.7558
JT-MTR-EC	0.1450	0.6574	0.1429	0.6270	0.1048	0.6285
MTR-ER3	0.1375	0.7420	0.1311	0.7839	0.0892	0.7484
MTR-ER4	0.1188	0.7768	0.1131	0.7726	0.0869	0.7598
JT-MTR-ER	0.1207	0.7671	0.1141	0.7659	0.0862	0.7621

4.3.3.2 Error Rate Estimation

In contrast, the MTR-ER3 model performed better than the MTR-ER4 model in ER estimation. MTR-ER3 outperformed MTR-ER4 by relative 1.95%, 5.42% and 21.06% on ER_S, ER_I and ER_D estimation, respectively. Adding WER as a target to have negatively impacted MTR-ER4's performance in ER_D resulting relatively large errors. Weighted MSE between ERs and WER could be helpful in preventing MTR-ER4's performance degradation in ER_D, particularly in terms of PCC.

From the perspective of individual ER estimation, both MTR-ER3 and MTR-ER4 showed relatively low PCCs in ER_D estimation. These low PCCs were observed when the average ER of an ASR hypothesis, as shown in Table 4.9, was low. For example, the average ER_Ds of Conformer, HuBERT, Transducer and wav2vec 2.0 hypotehses are below 2.5% and their corresponding PCCs for MTR-ER4 were 0.1195, 0.2385, 0.3695 and 0.1814, respectively. Additionally, the PCC for MTR-ER4 on ER_I was lowest for the HuBERT hypothesis, where the average ER_I was 1.57%.

Table 4.13 Performance of MTR-EC3 models on ER_S, ER_I, ER_D estimation.

ASR	RMSE↓			PCC↑		
	ER_S	ER_I	ER_D	ER_S	ER_I	ER_D
Conformer	0.0101	0.0137	0.0018	0.6284	0.6495	0.2366
HuBERT	0.0045	0.0027	0.0028	0.7408	0.0948	0.3060
Whisper	0.0216	0.3813	0.0030	0.5546	0.7636	0.8704
Chain	0.0101	0.0084	0.0028	0.6452	0.6084	0.7027
Transducer	0.0066	0.0108	0.0025	0.5551	0.5058	0.3647
wav2vec 2.0	0.0065	0.0083	0.0022	0.7286	0.4092	0.4057
mean	0.0099	0.0709	0.0025	0.6421	0.5052	0.4810

Table 4.14 Performance of MTR-EC4 models on ER_S, ER_I, ER_D estimation.

ASR	RMSE↓			PCC↑		
	ER_S	ER_I	ER_D	ER_S	ER_I	ER_D
Conformer	0.0096	0.0242	0.0019	0.6526	0.6124	0.2750
HuBERT	0.0041	0.0027	0.0025	0.7751	0.0960	0.4380
Whisper	0.0172	0.0475	0.0047	0.6189	0.9312	0.8529
Chain	0.0097	0.0080	0.0026	0.6602	0.6297	0.7332
Transducer	0.0078	0.2157	0.0022	0.4784	0.4660	0.5249
wav2vec 2.0	0.0068	0.0066	0.0023	0.7175	0.5841	0.4064
mean	0.0092	0.0508	0.0027	0.6505	0.5532	0.5384

Table 4.15 Performance of JT-MTR-EC models on ER_S, ER_I, ER_D estimation.

ASR	RMSE↓			PCC↑		
	ER_S	ER_I	ER_D	ER_S	ER_I	ER_D
Conformer	0.0106	0.0163	0.0019	0.6259	0.5568	0.1556
HuBERT	0.0072	0.0035	0.0039	0.5313	0.0469	0.0356
Whisper	0.0095	0.1107	0.0065	0.7085	0.9264	0.7086
Chain	0.0107	0.0137	0.0034	0.6115	0.2698	0.6148
Transducer	0.0064	0.0126	0.0028	0.5841	0.0682	0.2503
wav2vec 2.0	0.0080	0.0096	0.0025	0.6515	0.3779	0.1908
mean	0.0087	0.0277	0.0035	0.6188	0.3743	0.3260

Table 4.16 Performance of MTR-ER3 models on ER_S, ER_I, ER_D estimation.

ASR	RMSE↓			PCC↑		
	ER_S	ER_I	ER_D	ER_S	ER_I	ER_D
Conformer	0.0078	0.0074	0.0019	0.7318	0.7353	0.1522
HuBERT	0.0045	0.0025	0.0029	0.7657	0.2251	0.2545
Whisper	0.0052	0.0059	0.0027	0.8466	0.8442	0.8808
Chain	0.0080	0.0075	0.0027	0.7248	0.6465	0.7140
Transducer	0.0056	0.0038	0.0020	0.6431	0.5755	0.5650
wav2vec 2.0	0.0053	0.0069	0.0022	0.7846	0.5463	0.3960
mean	0.0061	0.0057	0.0024	0.7494	0.5955	0.4938

Table 4.17 Performance of MTR-ER4 models on ER_S, ER_I, ER_D estimation.

ASR	RMSE↓			PCC↑		
	ER_S	ER_I	ER_D	ER_S	ER_I	ER_D
Conformer	0.0086	0.0079	0.0019	0.6986	0.7220	0.1195
HuBERT	0.0039	0.0027	0.0029	0.7825	0.0790	0.2385
Whisper	0.0060	0.0065	0.0033	0.8249	0.8245	0.8551
Chain	0.0083	0.0076	0.0029	0.7092	0.6452	0.6834
Transducer	0.0059	0.0038	0.0026	0.6296	0.5489	0.3695
wav2vec 2.0	0.0058	0.0066	0.0026	0.7658	0.5699	0.1814
mean	0.0064	0.0059	0.0027	0.7351	0.5649	0.4079

Table 4.18 Performance of JT-MTR-ER models on ER_S, ER_I, ER_D estimation.

ASR	RMSE↓			PCC↑		
	ER_S	ER_I	ER_D	ER_S	ER_I	ER_D
Conformer	0.0090	0.0091	0.0020	0.7011	0.6665	0.2112
HuBERT	0.0039	0.0028	0.0028	0.7948	0.0277	0.3083
Whisper	0.0071	0.0074	0.0030	0.7955	0.8067	0.8670
Chain	0.0089	0.0079	0.0031	0.6823	0.6076	0.6686
Transducer	0.0055	0.0045	0.0031	0.6435	0.4232	0.2487
wav2vec 2.0	0.0061	0.0063	0.0024	0.7569	0.6027	0.2942
mean	0.0068	0.0063	0.0027	0.7290	0.5224	0.4330

4.3.4 Conclusion

In this section, an ER estimation model using MTR was presented. The model with four ER targets, referred to as MTR-ER4, not only outperformed the baseline on WER estimation but also provided individual ERs on ASR transcripts generated by six different ASR systems. A lower performance was observed in ER_D estimation when the average ER was below 2.5%. However, MTR-ER3, which estimates three ER targets excluding WER, showed better performance on ER_D estimation than MTR-ER4 with an improvement by 21.06%.

4.4 Summary

Several methods for WER estimation of ASR transcripts were explored in this chapter, focusing on improving performance without relying on full ASR decoding. First, DML was applied to estimate WER by aligning the cosine distance between utterance-level representations with the edit distance between their corresponding transcripts. The results showed that this approach effectively estimated WER and outperformed the baseline using confidence scores. Second, a fast WER estimation method was introduced, providing computational efficiency through average pooling with SSLRs to estimate WER directly. This method not only outperformed previous baseline models in terms of RMSE and PCC, but also significantly reduced inference time, making it suitable for applications requiring WER estimation on large datasets. Finally, this chapter introduced the MTR approach, which extended WER estimation to include individual ER estimations, such as substitution, insertion and deletion ER. The MTR-ER4 model, in particular, showed strong performance across ASR transcripts generated by various ASR systems. This extended task improved WER estimation and provided valuable insights into the quality of ASR transcripts.

Chapter 5

Unsupervised Domain Similarity Measurements

The domain of data for Automatic Speech Recognition (ASR) can be defined in many ways. It is often used interchangeably with an ASR corpus (Doulaty et al., 2015), as such corpora are typically collected according to specific speaking styles, subjects and recording conditions. Additionally, subsets of an ASR corpus can be regarded as distinct domain datasets based on common features, such as age, gender and dialects. Another possible feature is the noise environment in which speech was recorded, e.g., on buses, in cafes, in pedestrian areas or at street junctions (Meng et al., 2017). Beyond the acoustic characteristics of speech, an audio signal also includes content. Therefore, the domain of data can also be defined from a linguistic perspective, where a domain might refer to a genre, e.g., comedy or documentary (Deena et al., 2019) or a topic, e.g., biomedical or computer science (Beltagy et al., 2019).

When ASR is considered as a task, the meaning of *domain* should be explored from an ASR performance perspective. To begin, the discussion is started with data used for evaluation, referred to as target data and data used for training, referred to as source data. If the distribution of the target data is identical to that of the source data, the model optimised on the source data is likely to perform equivalently on the target data. Otherwise, ASR performance typically degrades. For example, Doulaty et al. (2015) reported that domain mismatch between source and target data leads to performance degradation in ASR systems. Similarly, Chen et al. (2023) conducted experiments on semi-supervised learning using an untranscribed ASR corpus that did not match the target domain. Their results showed that the performance of an ASR system was degraded from 4.85% to 5.52% when a mismatched corpus was added to the source data compared to a system trained solely on a matched corpus.

Thus, it could be assumed that as the distributions of source and target data become more similar, the performance of the ASR system improves and vice versa.

To understand the relationship between ASR performance and domain match, the discrepancy between data distributions must be measured. One common measure for this is Kullback–Leibler (KL) divergence (Kullback, 1997). This measure has been used to quantify distributional discrepancy in datasets for ASR (Asami et al., 2015; Gouvêa and Davel, 2011). However, KL divergence poses computational challenges when finding optimal subsets of data. For example, Gouvêa and Davel (2011) used KL divergence based on phonemes or n-grams in transcripts to select data. Given M utterances, they selected each utterance from an initial set of N utterances and compared it with each utterance in the remaining $M-N$ utterances to find the pair that most reduced the KL divergence. Consequently, KL divergence had to be computed $M \times (M-N)$ times, leading to exponential growth in complexity as the dataset size increased.

Another challenge in measuring domain discrepancies is the selection of features from the various acoustic characteristics, such as speakers, speaking styles, noise and recording devices. However, the labels for these features are not always available from ASR corpora, especially when data are sourced from the internet. Furthermore, certain feature used for data selection, such as speech versus non-speech (Asami et al., 2015), may be difficult to obtain for datasets recorded in complex environments, for example, public spaces with babble. In cases where data domain is not labelled, an unsupervised method for defining the domain of target data is necessary, followed by a measurement of domain similarity for data selection.

The second research question—*How can the similarity between an utterance and a set of target data be measured?*—is addressed in this chapter. To tackle this overarching question, it is further divided into two specific research questions:

- Can the acoustic and linguistic domains of target data for ASR be defined in an unsupervised manner?
- How can the domain similarities be measured between an utterance and a dataset?

In this chapter, data selection methods using acoustic and linguistic domain similarities will be explored. In Section 5.1, Acoustic Domain Similarity (ADS) is computed using noise-contrastive loss ratios. In Section 5.2, Linguistic Domain Similarity (LDS) is obtained using an n-gram Language Model (LM) trained on the tokens of spoken utterances.

5.1 Acoustic Domain Similarity

The acoustic characteristics of speech plays a significant role in ASR performance. In many cases, discrepancies between the acoustic features of source and target datasets can results in degradation of ASR accuracy. Accurately quantifying the acoustic domain similarity between datasets is therefore crucial for improving overall system performance. In this section, ADS is introduced as a measure of how closely the acoustic characteristics of two datasets align. This can be computed without the need for explicit labels or manual annotations by leveraging unsupervised methods. This approach is particularly useful when working with unlabelled data, where labels for acoustic features are not readily available.

If this measurement effectively represents ADS, it will be helpful for selecting utterances close to target data without labels. To demonstrate the effectiveness of this measurement, an unsupervised submodular data selection method using noise-contrastive loss ratios is proposed. This method is based on a noise-contrastive loss function learned by predicting future latent representations conditioned on past latent representations (Oord et al., 2018; Schneider et al., 2019). By measuring the difference between these representations, the similarity between source and target acoustic domains can effectively be quantified. This measure can then be used for selecting in-domain data, reducing performance degradation, and improving the robustness of ASR systems when applied to unseen or mismatched datasets.

To address the challenge of finding the optimal subset, data selection methods using submodular function within a budget (Asami et al., 2015; Lin and Bilmes, 2009; Wei et al., 2014). These methods require monotonicity of a function that converts a set into a measurable value, allowing for the optimal subset to be found within the budget. The noise-contrastive loss ratio function is monotonic and non-decreasing as utterances are added, making it suitable as a submodular function for data selection.

Experimental results show that models trained on datasets selected using the proposed method outperform selection methods based on log-likelihoods produced by Gaussian Mixture Model (GMM)-Hidden Markov Model (HMM) models, in terms of Word Error Rate (WER). When selecting a fixed amount, e.g., 10 hours of data, the difference between the two methods on Tedtalks (TD) was a relative 20.23% WER. Additionally, the method can minimise negative transfer while maintaining or improving on the performance of models trained on the full training set. Results show that WER on the WSJCAM0 (WS0) dataset was reduced by 6.26% relative when selecting 85% of the data from the full dataset.

The remainder of this section is organised as follows: Section 5.1.1 defines acoustic domain similarity. Section 5.1.2 details the proposed approach. Section 5.1.3 outlines the experimental setup. The results are presented in Section 5.1.4, and the section concludes with Section 5.1.5.

5.1.1 Definition

To begin, the concepts of domain and task are defined. The definitions used by Pan and Yang (2010) are followed, where a domain $\mathcal{D}$ comprises a feature space $\mathcal{X}$ and a marginal probability distribution $P(X)$, where X is a instance set $\{x_1,...,x_n\}$ in $\mathcal{X}$.

$$\mathcal{D} = \{\mathcal{X}, P(X)\}. \tag{5.1}$$

For example, an audio signal can be mapped into the feature space $\mathcal{X}$. Then, $P(X)$ represents the distribution modelled using X. A task $\mathcal{T}$ involves another space $\mathcal{Y}$ containing labels Y and a predictive function $f(\cdot)$, which can be interpreted as the conditional probability of the label $y \in Y$ conditioned on $x \in X$. Thus, $\mathcal{T}$ is defined as follows:

$$\mathcal{T} = \{\mathcal{Y}, f(X)\} = \{\mathcal{Y}, P(Y|X)\}. \tag{5.2}$$

When $\mathcal{T}$ is an Acoustic Modelling (AM) task, a label $y \in Y$ can be a token for transcription. Therefore, a domain for a specific task can be defined as follows:

$$\mathcal{D}^{\mathcal{T}} = \{\mathcal{X}, \mathcal{Y}^{\mathcal{T}}, P(X, Y^{\mathcal{T}})\} \tag{5.3}$$

where $P(X, Y^{\mathcal{T}}) = P(X)P(Y^{\mathcal{T}}|X)$.

A domain on which a model is trained is referred to as the source domain $\mathcal{D}_S$, while a domain on which a model is evaluated is referred to as the target domain $\mathcal{D}_T$. The training and test data are referred to as source domain data D_S and target domain data D_T, respectively. Then, the datasets for AM are defined as follows:

$$\begin{aligned} D_S^{\text{AM}} &= \{X_S, Y_S^{\text{AM}}\}, \\ D_T^{\text{AM}} &= \{X_T, Y_T^{\text{AM}}\}. \end{aligned} \tag{5.4}$$

The source domain for AM can be defined as follows:

$$\mathcal{D}_S^{\text{AM}} = \{\mathcal{X}_S, \mathcal{Y}_S^{\text{AM}}, P(X_S, Y_S^{\text{AM}})\} \tag{5.5}$$

where X_S and Y_S^{AM} are the sets of X and Y in $\mathcal{D}_S$ for AM, respectively. In this case, $P(X_S, Y_S^{\text{AM}})$ can be modelled using D_S^{AM} and used for data selection. One example of employing $P(X_S, Y_S)$ for data selection is a confidence score. If data from $\mathcal{D}_T^{\text{AM}}$ are given for training, the data selection problem can be addressed using $P(X_T, Y_T^{\text{AM}})$.

However, in most cases, D_T are not available for training and Y_T^{AM} is not provided. To define the target domain for AM without Y_T^{AM}, an alternative approach is to adopt the concept of Self-Supervised Learning Representations (SSLRs) to produce Y_T^{SSLR}. For example, when $x_t \in X_T$ represents the current frame, the next frame x_{t+1} is assigned to $y_t \in Y_T^{\text{SSLR}}$. Then, $P(X_T, Y_T^{\text{SSLR}})$ can be modelled using Y_T^{SSLR} in the place of Y_T^{AM}. Thus, the target domain for AM is replaced with the target domain for SSLR:

$$\mathcal{D}_T^{\text{SSLR}} = \{\mathcal{X}_T, \mathcal{Y}_T^{\text{SSLR}}, P(X_T, Y_T^{\text{SSLR}})\}. \tag{5.6}$$

Although the target domain can be defined in a self-supervised learning manner, the task changes from AM to SSLR. Since the aim of this thesis is to improve the performance of an ASR system, it is necessary to confirm that adding data selected using $\mathcal{D}_T^{\text{SSLR}}$ improves the performance of the ASR system. Literature reports that the performance of SSLR models on their own tasks correlates positively with the performance of ASR. In addition to the these findings, experiments will be conducted to confirm this assumption using $\mathcal{D}_T^{\text{SSLR}}$.

From a performance perspective, an instance is considered *in-domain* if the performance of an ASR system improves or, at least, does not degrade when it is added to the training data. Under this definition, a variable correlated with ASR performance improvement can serve as a measure of the in-domain relevance of data, referred to as domain similarity, specifically for ASR. If $P(X_T, Y_T^{\text{SSLR}})$ is correlated with ASR performance, this probability can be interpreted as the likelihood that an instance (x, y) is in-domain.

$$\text{Domain Similarity for ASR} \propto P(x, y^{\text{SSLR}}) \tag{5.7}$$

This probability is maximised during the training stage of the representation model and the representation becomes more useful as the probability is maximised. To support this assumption, the performance of the ASR system needs to be analysed using data selected by $P(x, y^{\text{SSLR}})$. For example, data selection can be based on the following measure where $\mathbf{o}_t$ is an observation at time t, $\mathbf{c}_t$ is a context embedding and $\mathbf{z}_{t+k}$ is a latent embedding at time $t+k$:

$$\text{ADS}(\mathbf{o}_{1:T}) \propto \frac{1}{K}\sum_{k=1}^{K}\frac{1}{T-k}\sum_{t=1}^{T-k}\log P(\mathbf{c}_t, \mathbf{z}_{t+k}). \tag{5.8}$$

where T is the length of an observation sequence and K is the maximum number of future steps.

5.1.2 Proposed Method

5.1.2.1 Loss ratios

As a submodular function (see Section 2.2.4), the mean of the ratios of all frames in an utterance is used. For simplicity, let $\mathcal{L}(\mathbf{z}_t)$ denote the noise-contrastive loss function based on the InfoNCE loss introduced in Equation (2.30).

$$\mathcal{L}(\mathbf{z}_t) = \sum_{k=1}^{K} \mathcal{L}(\{x_i\}_{i=1}^{t}, x_{t+k}, N_{t,k}; \mathbf{W}_k, \theta, \omega) \tag{5.9}$$

where k is the future step for prediction and $\mathbf{z}_t$ represents $\{x_i\}_{i=1}^{t}$. First, the ratios between the values of a loss function $\mathcal{L}_P$ trained on the data pool D_P and those of another loss function $\mathcal{L}_T$ trained on a target dataset D_T can be calculated as follows:

$$\mathrm{LR}(\mathbf{z}_{1:T}) = \frac{1}{T}\sum_{t=1}^{T} \frac{\mathcal{L}_P(\mathbf{z}_t) + \alpha}{\mathcal{L}_T(\mathbf{z}_t) + \alpha} \tag{5.10}$$

where α is a constant added to prevent overflow or underflow of the loss ratio and $\mathbf{z}_t$ is an embedding of an observation at time t. The accumulated loss ratio of all utterances in a subset S, which is included in P, is then defined as a submodular function:

$$f_{LR}(S) = \sum_{\mathbf{z}_{1:T} \in S} \big(\mathrm{LR}(u)\big) \tag{5.11}$$

This modular function f_{LR} is non-negative because it is the sum of means of ratios between non-negative losses as described in Equation (2.30). Moreover, it is normalised, meaning that function's value is zero when the input is null.

5.1.2.2 Negative transfer minimisation

When a budget is not given, the data size can be reduced without degrading WER performance by minimising negative transfer. Negative transfer refers to the performance degradation that occurs when a dataset from another domain is added to a training set. To maximise the performance of an ASR model, negative transfer should be minimised. As the optimal data subset is selected based on the value of the submodular function in Equation (5.11), a dataset prone to negative transfer can be filtered using a threshold. This threshold is determined through grid search.

Additionally, the value of $\mathcal{L}_T(\mathbf{z}_t)$ reflects how well the model fits the target dataset. If an utterance is subject to negative transfer, $\mathcal{L}_T(\mathbf{z}_t)$ would be high. Thus, another threshold based on loss values is also explored in the same method.

5.1.3 Experimental Setup

Similar to the previous study by Doulaty et al. (2015), a Data Pool (DP) was generated for selection. It consisted of four datasets: AMI (Cieri et al., 2004), Fisher (FS) (Carletta et al., 2005), TD (Ng et al., 2014) and WS0 (Robinson et al., 1995). The training data pool totalled 40 hours, with 10 hours from each dataset. For pre-training, two 1-hour sets from each dataset were designated as target and test datasets. The experiment involved three stages: pre-training, data selection, building ASR systems.

5.1.3.1 Pre-training

A wav2vec model[1] was modified to support noise-contrastive loss ratios. To accommodate the small size of the target dataset, the kernel sizes and strides of encoder layers were changed from (10,8,4,4,4) and (5,4,2,2,2) to (16,16) and (8,8), respectively. Additionally, the kernel sizes of aggregation layers were reduced from (2,3,...,13) to (2,3,4). The number of prediction steps was set to 6 instead of 12. Each model was trained for up to 200 epochs, with early stopping applied after 10 epochs of no improvement. The models were trained separately on each target dataset and the training data pool.

A GMM-HMM system was trained to obtain log-likelihood values. Due to the limited amount of training data, one hour, the accuracy of the system was lower compared to the hybrid ASR system described in Section 5.1.3.3. The training data were then decoded using these models, and the log-likelihood of the utterances in the decoding was used for data selection.

5.1.3.2 Data Selection

After pre-training, frame-level losses were calculated for both the target and training dataset. To prevent the ratios from overflowing, the losses were adjusted by adding a constant α. The mean loss value for each utterance was used as a score for data selection. A greedy method was employed to select data based on these scores, sorting the utterances and selecting data within the given budget. When no budget was provided, noise-contrastive loss ratios and

[1]https://github.com/facebookresearch/fairseq/blob/main/examples/wav2vec/README.md

losses were used to minimise negative transfer and reduce the amount of the selected data. The thresholds for noise-contrastive loss ratios and losses were empirically determined.

5.1.3.3 Hybrid Automatic Speech Recognition System

Hybrid GMM-HMM and neural network systems were built for speech recognition, based on the system described by Povey et al. (2015). For alignment, monophone, triphone, linear discriminant analysis and maximum likelihood linear transform, and speaker adaptive training models were trained sequentially. Neural networks were then trained using labelled frames generated by the GMM-HMM models. Additionally, the LM was trained on the merged text from all datasets.

5.1.4 Results

5.1.4.1 Baseline

An ASR system was built using the entire data pool, which consisted of 40 hours of data, to serve as the baseline system. The model was trained on all the data, and each test dataset, along with the combined dataset, was scored separately, as shown in Table 5.1.

Table 5.1 WER(%) of a baseline system.

Feature	**AMI**	**FS**	**TD**	**WS0**	**DP**
MFCC	26.69	35.72	24.58	9.90	25.04

5.1.4.2 Data Selection

The results of the data selection process are shown in Table 5.2. The first column is the target dataset on which the pre-trained model was trained on. When segments were selected based on the noise-contrastive loss ratios between a pre-trained model on AMI and another model on the data pool, 3263, 3503 and 3521 segments of AMI were selected for sets of 10, 20 and 30 hours from the pool, respectively. In contrast, 2023, 2810 and 3222 segments were selected based on log-likelihood, respectively. Data from the same corpus as the target data tended to be selected more often using noise-contrastive loss ratios than log-likelihood.

Given 10, 20 and 30-hour budgets, most of the selected data for the first 10 hours were from the same dataset as the target data. As the budget increased, data from the different datasets were selected more frequently, as the amount of each dataset was limited to 10 hours. However, one of the datasets was less frequently selected up to 30 hours. For example, when the target dataset was WS0, only 334 utterances from FS were selected for the 30-hour dataset, which was relatively lower than the other data selection results. This had a negative impact on performance on the whole training dataset.

Table 5.2 Numbers of selected segments by noise-contrastive loss ratios (CLR) and log-likelihood (LL). The total numbers for AMI, FS, TD and WS0 were 3526, 3330, 3244 and 3685, respectively.

Target Dataset	**Hours of Subset (CLR/LL)**			**Selected Dataset**
	10h	**20h**	**30h**	
AMI	3263/2023	3503/2810	3521/3222	AMI
	14/131	291/774	1083/1863	FS
	195/306	1811/1089	2725/2020	TD
	16/1008	1320/2261	3070/3262	WS0
FS	0/13	669/1616	2209/2717	AMI
	3257/3301	3328/3325	3329/3325	FS
	65/18	2615/1399	3123/2455	TD
	0/0	15/349	1479/1646	WS0
TD	103/1385	1524/2250	2797/2899	AMI
	362/162	1789/781	2686/1807	FS
	2773/1100	3181/2099	3219/2779	TD
	0/720	152/1662	1471/2781	WS0
WS0	104/845	2166/2492	3299/3208	AMI
	0/4	4/337	334/1699	FS
	28/57	1222/625	3116/1861	TD
	3527/2680	3684/3653	3685/3685	WS0

5.1.4.3 Loss Ratios vs Log-Likelihood

The data selected in Table 5.2 were used to train ASR models. The performance of each system was evaluated in terms of WER. The results show that models trained on datasets selected using noise-contrastive loss ratios outperformed those selected using log-likelihood from the hybrid system. For example, as seen in Table 5.3, models trained on datasets selected by noise-contrastive loss ratios generally performed better, except in the 20-hour and 30-hour cases when the target datasets were FS and AMI, respectively.

Table 5.3 WER(%) on selected datasets for different training subset sizes. The best WER performance for each target dataset is highlighted in bold.

Method	Target	10h	20h	30h	40h
CLR	AMI	31.71	28.62	27.02	**26.69**
	FS	39.57	37.12	**35.49**	35.72
	TD	28.07	25.54	**24.43**	24.58
	WS0	11.14	9.57	**9.32**	9.90
LL	AMI	34.51	29.56	26.95	**26.69**
	FS	40.02	36.80	36.56	**35.72**
	TD	35.19	28.37	26.42	**24.58**
	WS0	11.27	9.90	**9.89**	9.90

5.1.4.4 Negative Transfer Minimisation

For the grid search of thresholds, varying amounts of data were selected based on noise-contrastive loss ratios and loss values to verify the ADS assumption that the SSLR loss contributes to ASR performance improvement. Based on the results in Table 5.3, between 80% and 95% of the data pool were selected for training.

Table 5.4 WER(%) on selected datasets for negative transfer minimisation by noise-contrastive loss ratios and losses. CL stands for noise-contrastive loss.

Method	Target Dataset	80%	85%	90%	95%
CLR	AMI	26.98	26.79	**25.91**	26.35
	FS	35.83	36.96	35.83	**35.72**
	TD	24.97	25.25	24.94	**24.34**
	WS0	9.66	9.71	**9.51**	9.66
CL	AMI	27.19	26.55	**25.78**	27.36
	FS	**35.02**	36.11	35.75	35.50
	TD	25.09	24.61	**24.34**	24.59
	WS0	9.56	**9.28**	9.66	9.52

As shown in Table 5.4, the optimal subsets for WER performance were found to be between 80% and 95% of the entire dataset. The best performances for AMI, FS, TD and WS0 on datasets selected by CLR were 25.91%, 35.72%, 24.34% and 9.51%, respectively. When datasets were selected using CL, the results were 25.78%, 35.02%, 24.34% and 9.28% for AMI, FS, TD and WS0, respectively. These WERs were competitive with the baseline performance achieved by the CLR method within budget constraints in Table 5.3. For example, on the 80% FS dataset, selected by CL, the WER was 35.02%, while the baseline performance was 35.72%. In other words, comparable WER performance can be achieved with a smaller dataset by eliminating data that may cause negative transfer.

5.1.5 Conclusion

An unsupervised data selection with a submodular function based on noise-contrastive loss ratios has been explored. For data selection based on budget, this method outperforms a method using log-likelihood produced by the models trained on a target dataset. Furthermore, the amount of data selected from the data pool can be minimised without performance degradation by avoiding negative transfer. The performance of the ASR models trained on the training dataset selected by noise-contrastive losses and ratios is comparable to the baseline performance on the whole training dataset.

5.2 Linguistic Domain Similarity

In the previous discussions, it was highlighted that the performance of an ASR system is dependent on the probability modelled for an acoustic domain using SSLR. However, the system's performance is also influenced by the linguistic domain, such as topics (Bertoldi et al., 2001). Lefevre et al. (2001) demonstrated that the linguistic domain affects ASR performance. For example, an acoustic model trained on a broadcast news dataset performed better on other datasets when paired with an LM trained on those datasets, compared to using an LM trained solely on the broadcast news dataset.

To verify the effectiveness of this measure for LDS, the edit distance between discrete units of an utterance pair whose transcripts are identical is measured. Edit distance quantifies how dissimilar two sequences are by counting the operations required to convert one sequence to another. Insertion, deletion and substitution edits are widely used for this purpose, in a measure known as Levenshtein Distance (LD) (see Section 2.5.2.1). In this thesis, LD is used interchangeably with edit distance.

It is assumed that the discrete units are close to linguistic tokens when the sum of the edit distances of all the pairs U is minimised. Additionally, an LM trained with discrete units can be compared to one trained with transcripts using perplexity, a common evaluation metric for LMs (Bahl et al., 1983). If there is a strong correlation between the perplexities of the two models, the assumption underlying LDS would be confirmed.

In this section, a data selection method using LDS with discrete units is proposed. Discrete units are generated using Textlesslib (Kharitonov et al., 2022) with a sliding window approach. Then, an n-gram language mode is trained on the target data $\mathcal{T}$. The averaged probability of discrete units from the LM is used as LDS for data selection. The effectiveness of this method is evaluated by comparing the edit distance between utterances with identical transcripts and by measuring the model's perplexity on different datasets.

5.2.1 Definition

Since ASR performance improves when the source domain for the LM is similar to the target domain for the LM, it becomes essential to select training data the LM based on the target domain for the LM. Therefore, the target domain for the LM is defined as follows:

$$\mathcal{D}_T^{\text{LM}} = \{\mathcal{X}_T, \mathcal{Y}_T^{\text{LM}}, P(X_T, Y_T^{\text{LM}})\} \tag{5.12}$$

and the domain similarity for ASR is related to the probability derived from the definition of $\mathcal{D}_T^{\text{LM}}$:

$$\text{Domain Similarity for ASR} \propto P(x, y^{\text{LM}}). \tag{5.13}$$

However, one challenge is that X_T and Y_T are not available because they represent tokens of transcripts used for ASR. In the case of semi-supervised learning, a straightforward approach is to use ASR transcripts of $\mathcal{T}$ generated by an ASR system. However, these transcripts may contain noise due to the limited performance of the ASR system. Although the nature of linguistic information in speech is not fully understood, there have been some proposals using quantisation techniques to exploit linguistic information for speech resynthesis (van den Oord et al., 2017) and speech-to-speech translation (Lee et al., 2022). Moreover, Kharitonov et al. (2022) released a library that includes pre-trained models for speech tokenisation, which quantises acoustic features and generates sequences of discrete units. These units can serve as linguistic tokens instead of using ASR transcripts.

Inspired by the use of discrete units for LM, the previous and current tokens obtained from the target domain are treated as X_T and Y_T. The joint probability $P(X_T, Y_T)$ is then modelled for a next-token prediction task. If the input consists of n tokens, the task resembles n-grams modelling. In summary, the LDS of an utterance $t_{1:L}$ is defined as:

$$\text{LDS}(t_{1:L}) \propto \frac{1}{L-n+1} \sum_{l=n}^{L} \log p(t_{l-n+1}, \ldots, t_l) \tag{5.14}$$

where t represents a discrete unit for speech and L is the length of the token sequence of the utterance.

5.2.2 Proposed Method

To effectively calculate linguistic similarity, word-level representations are preferred. In this approach, a sliding window function is applied to continuous representations after encoding them with HuBERT but prior to K-means quantisation. The tokens returned by HuBERT correspond to frame-level representations. However, this frame rate limits the context captured by HuBERT, making it less suitable for n-gram LMling. By averaging over w frames using sliding windows, this limitation is addressed, incorporating a broader linguistic context.

$$t_i = \arg\min_k ||\mathbf{c}_k - \frac{1}{w}\sum_{j=i}^{i+w} \mathbf{z}_j||^2 \tag{5.15}$$

where $\mathbf{c}_k$ is the centroid of a cluster k and w is the window size. Additionally, to identify the best hyper-parameters for extracting discrete units, the Edit Distance Ratio (EDR) is computed with U, a set of pairs of discrete unit sequences with identical transcripts. The edit distance between two sequences is calculated by determining the minimum number of single-character edits required to transform one sequence into the other. This distance is then normalised by the total length of both sequences for symmetry.

$$\mathrm{EDR}(U) = \frac{\sum_{(t^l,t^m)\in U} d(t^l,t^m)}{\sum_{(t^l,t^m)\in U} |t^l| + |t^m|}. \tag{5.16}$$

where t^l and t^m represent the token sequences of the l-th and m-th utterances, respectively, and EDR is the edit distance ratio. The hyper-parameters θ^{S2T} for a speech-to-unit task that yield the lowest EDR values are then chosen to extract discrete units from the target data, which are used to train an n-gram LM.

$$\arg\min_{\theta^{S2U}} \mathrm{EDR}(U). \tag{5.17}$$

LDS measures domain similarity between source and target data from a linguistic perspective. The similarity is computed using an LM trained on manual transcripts, denoted as $\mathcal{M}$. Alternatively, the linguistic probability is computed using discrete speech representation from an LM trained on spoken utterances in the target domain.

$$P(t_i|t_{i-n+1},\ldots,t_{i-1}) = \frac{\mathrm{count}(t_{i-n+1},\ldots,t_i)}{\mathrm{count}(t_{i-n+1},\ldots,t_{i-1})} \tag{5.18}$$

where t is a discretised token, i is an index and n is the length of dependency.

5.2.2.1 An Evaluation Metric

An evaluation metric for the LM using discrete units is the Pearson Correlation Coefficient (PCC), which is defined as follows:

$$\rho = \frac{\sum(\mathrm{WER}(t^l,t^m) - \mu_{\mathrm{WER}})(\mathrm{DER}(t^l,t^m) - \mu_{\mathrm{DER}})}{\sqrt{\sum(\mathrm{WER}(t^l,t^m) - \mu_{\mathrm{WER}})^2 \sum(\mathrm{DER}(t^l,t^m) - \mu_{\mathrm{DER}})^2}} \tag{5.19}$$

where DER represent the discrete unit error rate. The Error Rate (ER) is calculated as follows:

$$\mathrm{ER}(x^l,x^m) = \frac{d(x^l,x^m)}{|x^l|} \tag{5.20}$$

where x^l and x^m are the tokenised units of the l-th and m-th utterances.

5.2.3 Experimental Setup

5.2.3.1 Datasets

The Wall Street Journal (WSJ) (Paul and Baker, 1992) corpus consists of read speech from experts reading excerpts from the WSJ. The SI284 dataset from WSJ is used for the experiment on LDS. The labels 5k and 20k represents the vocabulary in the datasets.

Table 5.5 Hours of WSJ Datasets.

Training	**Dev**		**Eval**			
SI284	**93_5k**	**93_20k**	**92_5k**	**92_20k**	**93_5k**	**93_20k**
81.49	1.12	1.09	0.67	0.71	0.43	0.41

5.2.3.2 Text Standardisation

Transcripts were standardised using Whisper Normaliser [2] and Nemo Text Processing[3]. General rules for standardising all corpora, including WSJ, were applied as follows. First, words in brackets and parentheses were removed. Second, exceptions were handled using a custom mapping list and Nemo Text Processing, e.g., "401K" to "four o one k". Third, symbols were removed using Whisper Normaliser, e.g., "won't" to "will not". Next, all

[2]https://github.com/openai/whisper

[3]https://github.com/NVIDIA/NeMo-text-processing

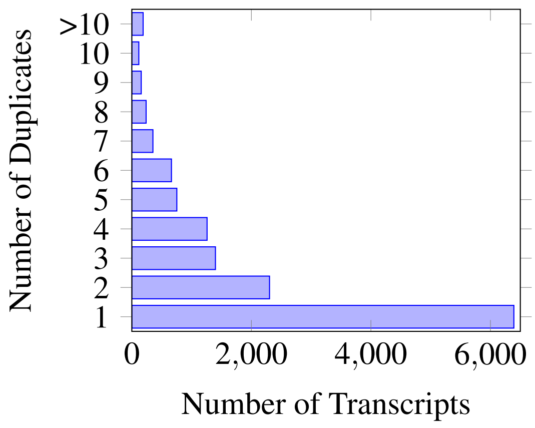


Figure 5.1 Histogram of identical transcripts in WSJ. y-axis is the number of utterances whose transcripts are identical. x-axis is the number of transcripts.

letters were lower-cased. Lastly, periods were removed. Some examples of the standardised transcripts are provided in Table 5.6.

Table 5.6 Examples of text standardisation. In each block, a transcript in the upper line is original text and another in the bottom line is the result of standardisation.

Type	**Transcript**
Remove brackets Remove symbols Lower-case	Pratt &AMPERSAND [pau] [pau] [pau] Whitney said last week it might lay off employees after the first of the year because of decreased demand for jet engines and parts .PERIOD
	pratt ampersand whitney said last week it might lay off employees after the first of the year because of decreased demand for jet engines and parts period
Mapping Remove periods Remove symbols	But at least they won’t be asking What’s this weird tasting stuff says Mr. Thompson
	but at least they will not be asking what is this weird tasting stuff says mister thompson

5.2.3.3 Hyper-parameters

In WSJ, some utterances share identical transcripts but are read by different speakers. For example, there are 2304 unique transcripts, shared by two utterances. The distribution of these utterances is plotted in Figure 5.1. Among the utterances with identical transcripts, 7433 pairs were selected for fine-tuning hyper-parameters on discrete units.

These discrete units were generated using Textless-lib (Kharitonov et al., 2022). The library provides a pre-trained representation model built upon the HuBERT base model, which was trained on 960 hours of LSP. In Textless-lib, HuBERT acted as the encoder. Its outputs were averaged over w frames before passing them to the quantiser for clustering. To set the boundaries between windows, the window slid by s steps. Windows were then assigned to one of the c clusters pre-defined using K-means clustering. The values for w, s, and c were selected from the ranges [1, 5, 10, 25], [1, 2, 5] and [50, 100, 200, 500], respectively. For fine-tuning the hyper-parameters, EDR was computed on discrete units extracted from 7433 pairs of WSJ utterances with identical manual transcripts. The discrete units yielding the lowest EDR were used to train a 3-gram LM[4] on the discrete units of the WSJ training set. The 3-gram LM was chosen for two main reasons. First, with relatively long utterances averaging 7.8 seconds, the 3-gram model is capable of capturing sufficient contextual information by modelling dependencies across sequences of three consecutive tokens. Simpler models like 1-gram or 2-gram fail to account for this level of context, making them unsuitable for this task. Second, the moderate vocabulary size, ranging from 50 to 500 tokens, means that higher-order models such as 4-gram or 5-gram would likely overfit the data without providing significant performance improvements. Witten-Bell discounting[5] was adopted for smoothing, which relies on the probability of observing an unseen word in the current context.

5.2.4 Results

Table 5.7 shows that EDR increases as the discrete unit size increases.

Table 5.7 EDR with 7,433 pairs selected from WSJ, where w and s are both set to 1.

50	**100**	**200**	**500**
0.2301	0.2469	0.2614	0.2947

Additionally, the results of grid search for the optimal parameters w and s are shown in Figure 5.2. The figure demonstrates that a window size of 5 and a step size of 1, along with a vocabulary size of 50, resulted in the lowest EDR of 0.2272. These optimal values were then used to extract discrete units from WSJ.

Using the optimal hyper-parameters, a SRILM 3-gram LM was trained on the test data and used to compute probabilities within the training data. When the LM was trained using words, the perplexity of the LM was 196.70 and 442.52 on WSJ Eval 92_5k and 92_20k,

[4]http://www.speech.sri.com/projects/srilm

[5]http://www.speech.sri.com/projects/srilm/manpages/ngram-discount.7.html

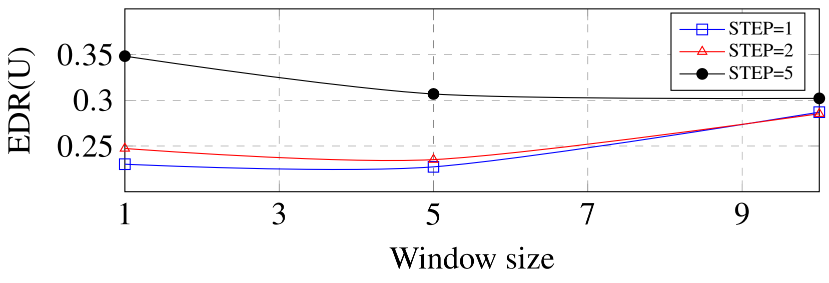


Figure 5.2 EDR with different window sizes and steps when the vocabulary size is 50.

respectively. The difference reduces when the LM was trained using discrete units; however, the results indicate that perplexity using discrete units remains high on both WSJ Eval 92_20k and Eval 93_20k. The results are summarised in Table 5.8.

Table 5.8 Perplexity of an LM trained on a WSJ training datasets and evaluated on a WSJ evaluation datasets.

Token	**92_5k**	**92_20k**	**93_5k**	**93_20k**
word	196.70	442.52	226.67	440.70
discrete unit	7.92	8.04	7.94	8.04

To verify the correlation between the edit distance of discrete unit sequences and that that of word sequences, PCC was measured. The utterances of pairs for EDR were randomly shuffled. Then, the edit distances of the pairs were measured at both the discrete unit and word levels. The PCC between the two edit distances was 0.8366. When normalised by the length of the first utterance, it increased to 0.8560. As a PCC above 0.5 is generally considered to indicate a moderate to strong correlation, the edit distance using discrete units can potentially replace the edit distance using words.

Additionally, an example of the edit distance and ERs are provided in Table 5.9. The edit distance between utterances whose transcripts are identical is lower than that between utterances whose transcripts are different.

5.2.5 Conclusion

In this section, a method for extracting linguistic information using discrete units was proposed. Textless-lib, which provides a pre-trained model based on HuBERT and K-means clustering, was employed to extract sequences of discrete units from audio files without requiring ASR decoding. The edit distance between the discrete unit sequences showed a correlation with that of word sequences as well as the error rate. LMs were trained on discrete

Table 5.9 An example of edit distances and token error rate at a word and a discrete unit level. $d_{\text{edit}}(A,\cdot)$ is edit distance between utterance A and B or C. $DER(t^l,\cdot)$ is discrete unit error rate between utterance A and B or C.

Utterance	**Sequence**	$d_{\textbf{edit}}(t^1,\cdot)$	**DER**$(t^1,\cdot)$
t^1	'revenue', 'for', 'the', 'latest', 'quarter', 'rose', 'sixty', 'one', 'percent', 'to', 'four', 'hundred', 'seven', 'million', 'dollars', 'from', 'two', 'hundred', 'fifty', 'three', 'million', 'dollars'	-	-
t^2	'revenue', 'for', 'the', 'latest', 'quarter', 'rose', 'sixty', 'one', 'percent', 'to', 'four', 'hundred', 'seven', 'million', 'dollars', 'from', 'two', 'hundred', 'fifty', 'three', 'million', 'dollars'	0	0.0
t^3	'revenue', 'fell', 'two', 'percent', 'in', 'the', 'latest', 'period', 'to', 'one', 'point', 'five', 'billion', 'dollars', 'from', 'one', 'point', 'five', 'three', 'billion', 'dollars'	16	0.7273
t^1	20, 13, 15, 33, 18, 17, 0, 3, 30, 26, 45, 30, 17, 35, 48, 41, 19, 43, 26, 34, 11, 5, 32, 29, 26, 24, 17, 29, 35, 11, 14, 21, 47, 35, 11, 14, 5, 45, 10, 38, 3, 44, 37, 17, 11, 36, 21, 16, 12, 33, 20, 33, 20, 18, 47, 35, 30, 29, 24, 11, 14, 21, 0, 3, 28, 27, 19, 31, 28, 0, 8, 42, 19, 24, 34, 36, 30, 17, 35, 3, 44, 49, 47, 29, 26, 24, 30, 47, 35, 44, 45, 4, 30, 17, 45, 4, 27, 19, 16, 31, 28, 46, 0, 8, 42, 16, 24, 16, 36, 9, 39, 20, 13, 20	-	
t^2	20, 2, 13, 15, 33, 18, 17, 0, 3, 26, 45, 30, 35, 48, 41, 19, 43, 4, 26, 11, 36, 5, 10, 29, 24, 10, 17, 29, 34, 11, 14, 21, 47, 45, 11, 5, 45, 10, 38, 3, 28, 44, 37, 17, 11, 14, 21, 43, 16, 12, 46, 44, 18, 49, 47, 35, 30, 29, 24, 32, 31, 7, 26, 17, 35, 34, 36, 14, 21, 0, 3, 35, 26, 28, 27, 0, 19, 4, 28, 49, 0, 8, 42, 19, 16, 34, 36, 30, 17, 3, 46, 49, 47, 45, 38, 32, 0, 31, 26, 17, 7, 30, 47, 35, 30, 49, 45, 4, 30, 17, 45, 27, 0, 19, 4, 28, 46, 0, 8, 42, 16, 34, 36, 9, 39, 20, 39, 20	114	0.4561
t_3	20, 39, 20, 39, 22, 13, 15, 33, 18, 10, 17, 0, 3, 30, 47, 31, 28, 26, 45, 46, 30, 0, 42, 44, 49, 47, 45, 7, 37, 17, 11, 14, 21, 43, 31, 28, 44, 49, 47, 28, 48, 35, 41, 19, 43, 4, 26, 34, 11, 36, 5, 37, 32, 45, 24, 4, 16, 7, 46, 33, 49, 47, 35, 10, 38, 3, 28, 44, 37, 32, 29, 38, 31, 4, 44, 30, 8, 40, 3, 30, 44, 37, 0, 35, 19, 45, 4, 28, 49, 0, 8, 42, 19, 16, 24, 34, 36, 9, 1, 2, 15, 18, 30, 17, 0, 8, 3, 27, 46, 10, 38, 31, 28, 44, 37, 32, 29, 43, 4, 44, 30, 8, 40, 16, 30, 44, 30, 48, 30, 48, 17, 45, 4, 7, 37, 47, 0, 42, 19, 4, 31, 28, 26, 0, 8, 42, 19, 16, 34, 36, 9, 39, 22, 39, 22, 20	107	0.9386

units and words for comparison in perplexity. The perplexity was high on large-vocabulary datasets, while it was relatively low on smaller-vocabulary datasets.

5.3 Summary

This chapter explored methods for measuring domain similarity in an unsupervised manner, specifically in the context of ASR. It began by defining domains and tasks mathematically and then extended these concepts to define the domain for a specific task. The probability distributions of domains were linked to two key concepts: ADS and LDS. These measures were associated with the notion of in-domain data, which was crucial for enhancing ASR performance.

To validate the use of ADS, an unsupervised data selection method based on a submodular function and contrastive loss ratios was proposed. This method calculated the ratio of two probability distributions—one learned from a target dataset and the other from a general data pool. Data from the target domain were prioritised for selection, demonstrating that the proposed approach successfully identified in-domain data from a multi-domain data pool.

For LDS, a method for extracting linguistic information from speech using discrete units was introduced. Discrete units were derived through a combination of a pre-trained HuBERT model and K-means clustering, bypassing the need for ASR decoding. LMs were then trained on both word sequences and discrete units. These models were evaluated by comparing their perplexity and examining correlations between edit distances and error rates. The results indicated a strong correlation between edit distances and error rates, and the LMs exhibited consistent trends in perplexity across different datasets.

In summary, both ADS and LDS provided effective measures of domain similarity in data selection for ASR. The findings suggested that these unsupervised approaches could improve ASR performance by selecting more relevant data and better capturing linguistic patterns in the absence of manual transcripts.

Chapter 6

Semi-Supervised Learning for ASR with Utterance Selection

Methods for Automatic Speech Recognition (ASR) have been investigated for decades, leading to the development of model architectures that improve both performance and generalisation. Simultaneously, the demand for data has grown with these advancements, with ASR systems now requiring and utilising more than a thousand hours of training data (Ardila et al., 2020). However, manually transcribed utterances are not available across all domains, e.g., different recording conditions or topics, and manual transcription still remains costly and time-consuming. As an alternative, researchers have explored semi-supervised learning (Veselý et al., 2013) methods, leveraging untranscribed utterances to enhance ASR performance.

One of the early attempts at semi-supervised learning was proposed by Zavaliagkos et al. (1998b), who transcribed spoken utterances using an ASR system, referred to as a seed model, to generate ASR transcripts. They found that ASR transcripts up to 20% corrupted could still improve the performance of an ASR system when used as training data. To ensure the quality of these ASR transcripts, data were selected using information from ASR decoding, such as confidence score (Veselý et al., 2013). Although this method has proven useful for data selection, Georgescu et al. (2021) reported that overconfident data often contribute barely to ASR performance improvement. Moreover, methods relying on confidence scores not only require access to the internal process of the ASR system, which is not always possible, especially in industrial systems, but also overlook deletions in ASR output. To address these issues, Negri et al. (2014) proposed a quality estimation method for ASR using features obtainable without ASR decoding. Moreover, Word Error Rate (WER) estimation models using self-supervised representations for speech and text were introduced in Chapter 5.

When selecting data for ASR from multiple domains, performance degradation can occur due to domain mismatches between source and target data (Doulaty et al., 2015), especially between distinct domains such as read speech and telephone speech. In fact, results from (Chen et al., 2023) indicated that domain mismatch can have a greater impact on ASR performance than the volume of training data. To address this, a data selection method for ASR that accounts for domain mismatch by using the loss from a representation learning model was proposed in Section 5.1.

This chapter integrates the methods proposed in the previous chapters to address all three overarching research questions introduced in Chapter 1:

- Is WER estimation without ASR decoding effective in semi-supervised learning for ASR?
- How can the similarity between an utterance and a set of target data be measured?
- Do low-WER and domain-relevant data selection methods complement each other in semi-supervised learning for ASR?

To answer these questions, the methods developed for WER estimation and domain similarity measurement are combined and applied to a semi-supervised learning framework for ASR. Specifically, a WER estimation model, MTR-ER in Section 4.3, is employed to evaluate and select data with low-WER from a pool of automatically transcribed utterances. Similarly, the domain similarity measurement methods introduced in Section 5.1 are used to identify data that is acoustically and linguistically relevant to the target domain. By integrating these selection criteria, both low-WER and domain-relevant data are selected for training an ASR system.

This chapter demonstrates how these selection methods complement each other, ensuring that the selected data is not only of high quality but also well-matched to the target domain. The effectiveness of this combined approach is evaluated by training an ASR system on the selected data and comparing its performance against baseline methods. Through this process, all three research questions are addressed, providing a unified framework for semi-supervised learning in ASR that leverages WER estimation and domain similarity measurements.

The aims of this work are:

- Development of data selection methods for semi-supervised learning in ASR, utilising an untranscribed data pool consisting of multiple domain data
- Application of a WER estimation model with MTR-ER4 for selecting low-WER ASR transcripts

- Measurement of acoustic and linguistic domain similarities for better alignment between the source and target domains

6.1 Semi-Supervised Learning for ASR with Utterance Selection

6.1.1 Semi-Supervised Learning

Semi-supervised learning for ASR is a method to train an ASR system with ASR transcripts. The overall steps for Semi-supervised learning for ASR with Utterance Selection are as follows:

Algorithm 2 Algorithm of Semi-Supervised Learning for ASR with Utterance Selection.

```
data: ASR model $A_\theta$, ASR corpus $D_M$, data pool of spoken utterances $D_U$
result: Trained ASR model $A_\theta$
Initialise $A_\theta$ by training it with $D_M$
repeat
    Train $A_\theta$ with $D_M$
    Transcribe $D_U$ using $A_\theta$ to generate $D_A$
    Select $D_W$ from $D_A$ using $\widehat{\text{WER}}$ with $\tau_W$ (see Section 6.1.2)
    Select $D_S$ from $D_A$ using ADS with $\tau_A$ or LDS with $\tau_L$ (see Sections 6.1.3 and 6.1.4)
    Compute $D'_S = \{D_W \cap D_S\}$
    Train $A'_\theta$ with $\{D_M \cup D'_S\}$
until $A_\theta$ converges or the maximum number of iterations is reached
```

6.1.2 Word Error Rate Estimation

WER of ASR transcripts is estimated using MTR-ER4, a WER model with Multi-Target Regression (MTR) for Error Rate (ER) estimation, as described in Fig 6.1.

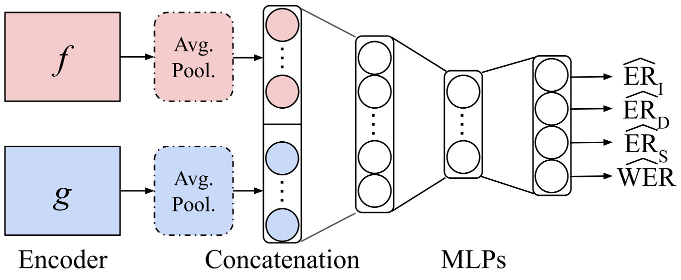


Figure 6.1 Illustration of MTR-ER4 Architecture.

This model consists of encoders, $f(\cdot)$ and $g(\cdot)$ for speech and text, respectively, along with an Multi-Layer Perceptron (MLP) for WER estimation. The estimation is given by:

$$\widehat{\mathrm{ER}} = \mathrm{MLP}(\mathrm{concat}[\mathrm{Avg}(f(X^l)); \mathrm{Avg}(g(A^l))]) \tag{6.1}$$

where $\widehat{\mathrm{ER}}$ is the ER estimation and Avg$(\cdot)$ is an average pooling function. X^l and A^l represent the spoken utterance and the ASR transcript of the l-th pair, respectively. The model predicts insertion, deletion and substitution error rates, as well as WER, to improve generalisation across domains.

The model is trained by minimising the Mean Squared Error (MSE) across all error rates:

$$\mathrm{MSE}_j = \frac{1}{N}\sum_{i=1}^{N}(\mathrm{ER}_j - \widehat{\mathrm{ER}_j})^2 \tag{6.2}$$

where j represents the error types: insertion, deletion, substitution and WER.

6.1.3 Acoustic Domain Similarity

Acoustic Domain Similarity (ADS) between an utterance and a target dataset is measured using the mutual information between a latent representation of a future observation $\mathbf{x}_{t+k}$ and a context representation $\mathbf{c}_t$ for speech, where k and t represent the future step and the time of the observation, respectively. For simplicity, let $\mathcal{L}_k(\mathbf{z}_t)$ denote the noise-contrastive loss function based on the InfoNCE loss introduced in Equation (2.30).

$$\mathcal{L}_k(\mathbf{x}_t) = \mathcal{L}(\{x_i\}_{i=1}^{t}, x_{t+k}, N_{t,k}; \mathbf{W}_k, \theta, \omega) \tag{6.3}$$

where k is the future step for prediction and $\mathbf{z}_t$ represents $\{x_i\}_{i=1}^{t}$. The utterance-level loss of a sequence $X_{1:T}$ is given as:

$$\mathcal{L}(X_{1:T}) = \frac{1}{T-k}\sum_{t}^{T-k}\sum_{k}\mathcal{L}_k(\mathbf{x}_t) \tag{6.4}$$

where $X_{1:T} = \{\mathbf{x}_1 ... \mathbf{x}_t ... \mathbf{x}_T\}$. ADS is then computed as:

$$\mathrm{ADS}(X^l) = \frac{\frac{1}{|D_T|}\sum_{X^m \in D_T}\mathcal{L}(X^m)}{\mathcal{L}(X^l)} \tag{6.5}$$

where X^l is an utterance in a data pool D_U and $|D_T|$ is the number of utterances in the test data D_T from the target domain $\mathcal{D}_T$. The average loss of target domain utterances is divided by the utterance loss to normalise the value so that it is proportional to the probability of similarity. If the utterance loss is close to the average loss of the target data, the value approaches 1. A threshold τ_A for filtering is determined based on the amount of training data available and the distribution of ADS in D_T.

6.1.4 Linguistic Domain Similarity

Linguistic Domain Similarity (LDS) measures the similarity between an utterance and a target dataset from a linguistic perspective. Instead of using an Language Model (LM) trained on manual transcripts $\mathcal{M}$, the LM is trained on discrete units from spoken utterances in the target domain. To effectively calculate linguistic similarity, a word-level unit is preferable. For the context at the word level, a sliding window is applied to the continuous representations before K-means clustering (see Equation (5.15)). To fine-tune the hyper-parameters for extracting discrete units, the Edit Distance Ratio (EDR) between utterances with identical transcripts is minimised (see Equation (5.16)). The hyper-parameters yielding the lowest EDR are chosen to extract discrete units from the target data for training an n-gram LM. The probability of the token sequence $T_{1:S}$ of an utterance is computed as:

$$P(T_{1:S}) \approx \sum_{i=1}^{S} \log(P(t_i|t_{i-n+1},...,t_{i-1})) \tag{6.6}$$

and LDS of the utterance is defined as:

$$LDS(T^l) = \frac{\frac{1}{|D_T|}\sum_{T^m \in D_T} P(T^m)}{P(T^l)}. \tag{6.7}$$

where T^l and T^m are the token sequences of utterances X^l and X^m, respectively. A threshold τ_L for filtering is determined similarly to ADS.

6.1.5 Data Selection for Speech Recognition

Training data for semi-supervised learning were selected using WER estimate and either ADS or LDS. One straight strategy is to use thresholds for the measures. For example, data with WER estimate lower than 20% and ADS lower than τ_A or ADS lower than τ_L are selected.

6.2 Experimental Setup

6.2.1 Datasets

For semi-supervised learning in ASR, CHiME-5 (CH5) (Boeddecker et al., 2018) was used as the target data. It consists of 50 hours of conversational speech recorded in a home environment using multi-channel microphones. Among the various channels, the recordings from the reference microphone array were used for this experiment. The training data from CH5 was utilised to train a seed model for ASR. The data pool was composed of spoken utterances from five ASR corpora: AMI (Carletta et al., 2005), LibriSpeech (LSP) (Panayotov et al., 2015), Switchboard (SWB) (Godfrey et al., 1992), Ted-lium 3 (TL3) (Hernandez et al., 2018) and Wall Street Journal (WSJ) (Paul and Baker, 1992). The AMI corpus comprises meeting recordings involving up to four participants in an office environment and provides automatically generated word- and phoneme-level timings for the transcripts. A subset of the AMI corpus, Full-corpus-ASR, was used in this experiment. Additionally, recordings captured using independent headset microphones were selected for this experiment, as opposed to those recorded with multiple distant microphones. The LSP corpus offers approximately 1000 hours of read speech from books. The SWB corpus encompasses two-sided telephone conversations. TL3 is a corpus of 452 hours of audio talks, and the Wall Street Journal (WSJ) corpus is composed of read speech with machine-readable texts from Wall Street Journal news articles.

	Corpus	**Training**	**Dev**	**Test**
Target	CH5	37.76	6.07	5.37
Data Pool	AMI	64.80	5.12	8.68
	LSP	961.25	10.51	10.75
	SWB	311.26	4.61	4.59
	TL3	444.62	6.13	3.57
	WSJ	81.485	2.20	2.22

Table 6.1 Hours of ASR corpora.

6.2.2 Seed Model for Automatic Transcription

A HuBERT large model[1] pre-trained on 60k hours of Libri-Light (Kahn et al., 2020b) was fine-tuned on CH5 according to the publicly released recipe. Default hyper-parameters applied, except for the mask length, which was set to 2 to accommodate short utterances.

[1]https://github.com/facebookresearch/fairseq

All manual transcripts were standardised using the Whisper normaliser [2] and NeMo text processing tool [3].

6.2.3 Word Error Rate Estimation Models

For WER estimation, the Fe-WER model was compared with MTR-ER4, which estimates insertion, deletion, substitution and WER. Both models were trained on CH5 Training and evaluated on the datasets in the data pool. The transcripts used for training the WER estimation models were generated using the seed model in Section 6.2.2. To minimise data imbalance, the number of ASR transcripts with a WER of 0 was limited to the sum of the second and third most frequent bins in a 100-bin histogram. Features for speech and text were extracted using HuBERT large and XLM-R large models. Hyper-parameters were determined via grid search, with input and output layers fixed at 2048 and 1, respectively. Hidden layers were chosen from the ranges [300, 600, 900] and [16, 32, 64], respectively. An Adam optimiser with a Cosine Annealing scheduler was used and learning rates were selected from [1e-4, 3e-4, 7e-4, 1e-3, 3e-3, 7e-3].

6.2.4 Acoustic Domain Similarity

For ADS, wav2vec (Schneider et al., 2019) was employed, where the density ratio $f_k(\mathbf{x}_{t+k}, \mathbf{c}_t)$ from Equation (6.4) was implemented. This model was pre-trained on the target data D_T, CH5 Test; then the ADS of each utterance in the data pool D_U was computed.

Using the default hyper-parameters of the wav2vec large [4], kernel sizes were set to (16, 16) and strides to (8, 8). The context networks were comprised of three layers with increasing kernel sizes (2, 3, 4) to fit the smaller amount of data in D_T. Early stop was implemented after 15 epochs of no improvement.

6.2.5 Linguistic Domain Similarity

Using Textless-lib (Kharitonov et al., 2022), discrete units were extracted from speech. Hyper-parameters were determined based on EDR with WSJ as described in Section 5.2.4. The optimal vocabulary and window sizes were 50 and 5, respectively, with a sliding window step size of 1. An SRILM 3-gram LM was trained on the discrete units that yielded the lowest EDR and was used to generate LDS for the data pool. For the details of experimental setup, readers are referred to Section 5.2.3.

[2]https://github.com/openai/whisper
[3]https://github.com/NVIDIA/NeMo-text-processing
[4]https://github.com/facebookresearch/fairseq

6.2.6 Data Selection for Speech Recognition

To control for the influence of the amount of selected data, it was fixed at 8 and 32 hours, ensuring that there remained room for performance improvement when additional supervised data were added. To satisfy this setting, the data were selected up to those hours from the data ordered by the WER estimate. To put a constraint using domain similarity on the data, either of ADS or LDS is combined to the data selection method. For example, when the data are ordered by WER estimate and selected up to 8 hours from the beginning, the utterances whose domain similarities were lower than their thresholds were filtered out. The thresholds were set to the maximum values of the bottom 10% in ADS and LDS for the CH5 Test.

6.3 Results

6.3.1 Seed Model for Automatic Transcription

The performance of the HuBERT model based on the amount of supervised training data is shown in Figure 6.2. The model trained on 32 hours of data was selected as the seed model because its performance improvement using the remaining data would be compared to that achieved with the selected data.

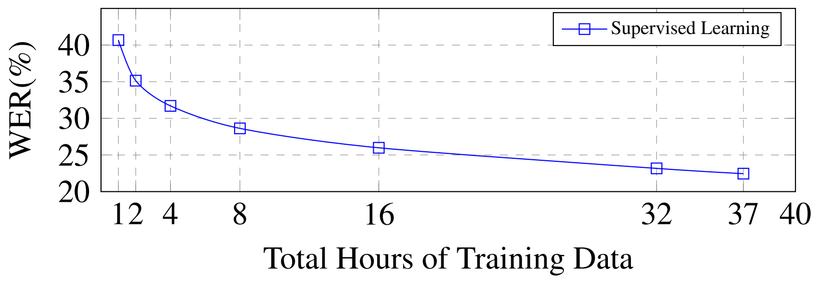


Figure 6.2 Performance of HuBERT with Supervised Learning.

The performance of the seed model was evaluated both with and without an LM for decoding. The LM was trained on the manual transcripts of the training data used for the seed model. As shown in Table 6.2, the improvement on AMI, CH5, and SWB was minor, whereas there was a significant degradation on LSP, TL3, and WSJ. For example, while the WER on AMI improved by an absolute 1.44%, it degraded by an absolute 11.04% on the test-clean set of LSP. Since this experiment focuses on semi-supervised learning, where untranscribed data are utilised by transcribing them with an ASR system, greater emphasis is placed on the model's performance on out-of-domain data rather than in-domain data. To achieve lower WER transcripts across multiple domains on average, a language model was not used, as

ASR performance without it tends to be more robust across various out-of-domain datasets. Consequently, when transcribing the data pool using the seed model, Viterbi decoding was adopted.

Table 6.2 WER(%) of the seed model with and without an LM.

Language Model	Test Dataset								
	AMI	CH5	LSP		SWB		TL3	WSJ	
			clean	other	callhome	swbd		eval92	eval93
Viterbi	28.57	23.17	10.27	18.87	40.89	36.97	12.88	13.34	14.76
KenLM (3g)	27.13	21.04	21.31	24.50	35.69	34.45	18.82	23.78	25.14

6.3.2 Word Error Rate Estimation

To verify that multi-target regression is effective on a multi-domain data pool, the MTR-ER4 model's performance was compared with the Fe-WER model's performance on all the test datasets. These models were trained on 32 hours of CH5 Training; then their performance was compared in Root Mean Square Error (RMSE) and Pearson Correlation Coefficient (PCC) as well as in terms of cumulative mean of WER reference.

6.3.2.1 Out-of-domain Performance

First of all, the evaluation results are summarised in Table 6.3 and 6.4. While Fe-WER outperformed on CH5 Test, the results are varying on the other datasets. For example, Fe-WER outperformed MTR-ER4 on CH5, LSP-other, TL3 and both WSJ datasets in RMSE, while MTR-ER4 outperformed on AMI, LSP-clean and both SWB datasets.

Table 6.3 Performance of Fe-WER on all the datasets.

WER Model	Test Dataset								
	AMI	CH5	LSP		SWB		TL3	WSJ	
			clean	other	callhome	swbd		eval92	eval93
RMSE	0.3521	0.2567	0.1061	0.1519	0.4449	0.3520	0.1352	0.0969	0.1002
PCC	0.5686	0.6842	0.5554	0.6799	0.5601	0.6506	0.5557	0.5749	0.6899

For comparison of their performance on out-of-domain datasets, the results except CH5 were averaged and summarised in Table 6.5. Both the RMSE and PCC of MTR-ER4 are better than those of Fe-WER.

Table 6.4 Performance of MTR-ER4 on all the datasets.

WER Model	**Test Dataset**								
	AMI	**CH5**	**LSP**		**SWB**		**TL3**	**WSJ**	
			clean	**other**	**callhome**	**swbd**		**eval92**	**eval93**
RMSE	0.3232	0.2749	0.1016	0.1663	0.4246	0.3323	0.1511	0.1023	0.1208
PCC	0.6161	0.6577	0.6027	0.6993	0.6658	0.6394	0.5701	0.5789	0.6566

Table 6.5 Mean RMSE and PCC of WER estimation models on out-of-domain datasets.

	RMSE↓	**PCC↑**
Fe-WER	0.2174	0.6044
MTR-ER4	**0.2153**	**0.6286**

6.3.2.2 Cumulative Mean of Word Error Rate Reference

Figure 6.3 shows the cumulative mean of WER reference increasing continuously when ASR transcripts are selected in order using MTR-ER4's estimate. However, Fe-WER's cumulative mean of WER reference initially decreases until reaching 0.10 before increasing. The spike in the low range of WER estimate could be critically harmful on the ASR performance and a weakness in selecting a small amount of data from the data pool. Based on these results, MTR-ER4 was used to select low-WER utterances from the data pool.

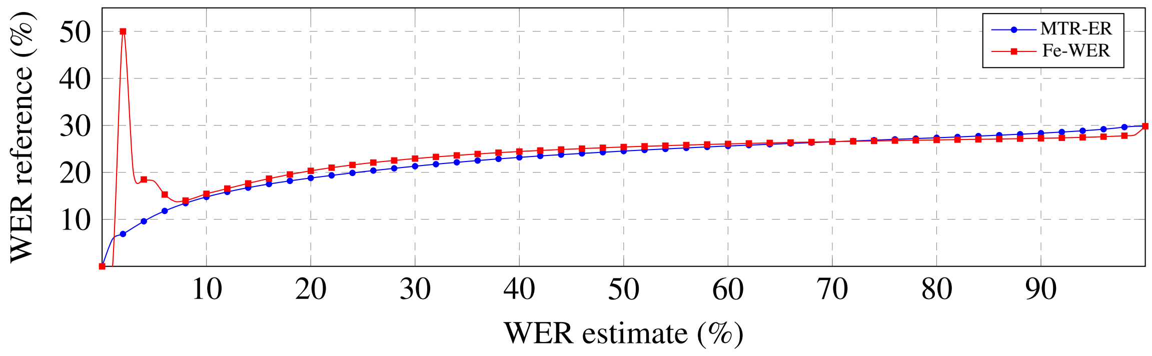


Figure 6.3 Cumulative mean of reference WER according to a threshold for WER estimate.

6.3.2.3 Distribution of Word Error Rate Estimates

The WER of data transcribed using the seed model is estimated using MTR-ER4. The distribution of WER estimates is as shown in Figure 6.4. It is relative left-skewed and overlaps significantly with that in CH5 Test and is denser than the target data distributions in the range [0%, 10%].

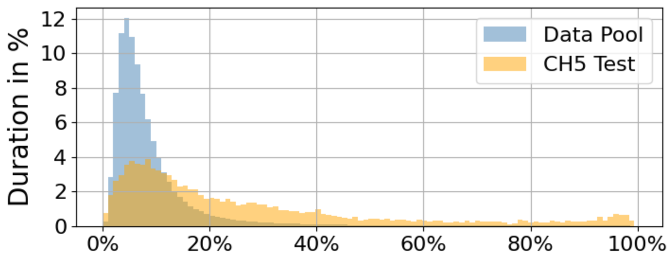
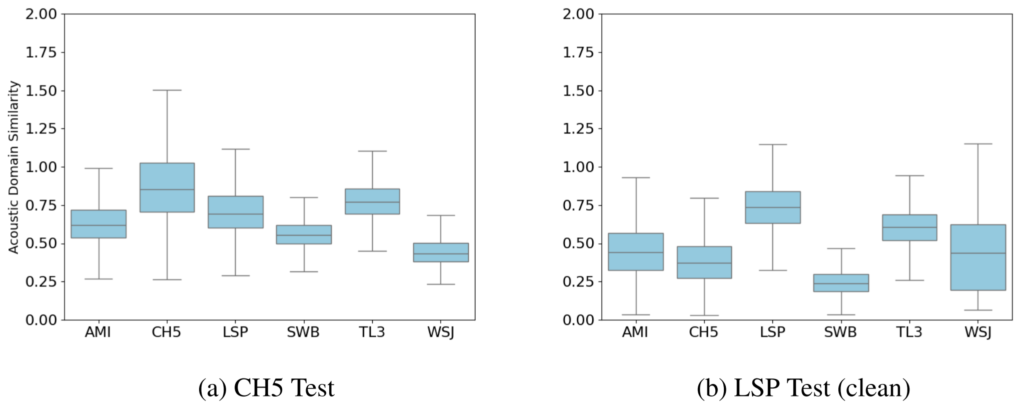


Figure 6.4 Distribution of WER estimate.

6.3.3 Acoustic and Linguistic Domain Similarity

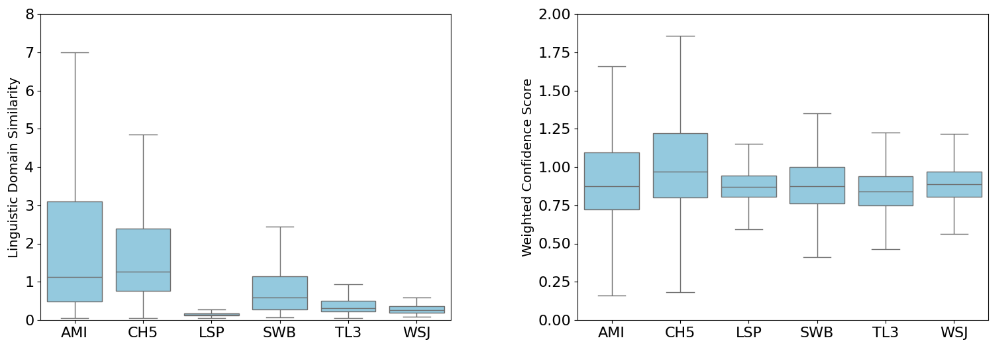
The ADS distribution is plotted in 6.5. The box plots shows the distribution of ADS values in the data pool when the target data were from CH5 Test and LSP Test. The mean ADS for CH5 is the highest in Figure 6.5(a), while the mean ADS for LSP is the highest in Figure 6.5(b). This observation indicates that the distributions are influenced by the target data. For CH5 as the target, TL3 appears to be the most acoustically similar, while WSJ is the least similar. However, the exact reason for the high mean ADS of TL3 when the target is CH5 remains unclear. It may be influenced by shared acoustic characteristics between TL3 and CH5, such as recording conditions or speaker demographics, but further investigation would be required to confirm this.



(a) CH5 Test (b) LSP Test (clean)

Figure 6.5 ADS distributions when target data are CH5 Test and LSP Test (clean).

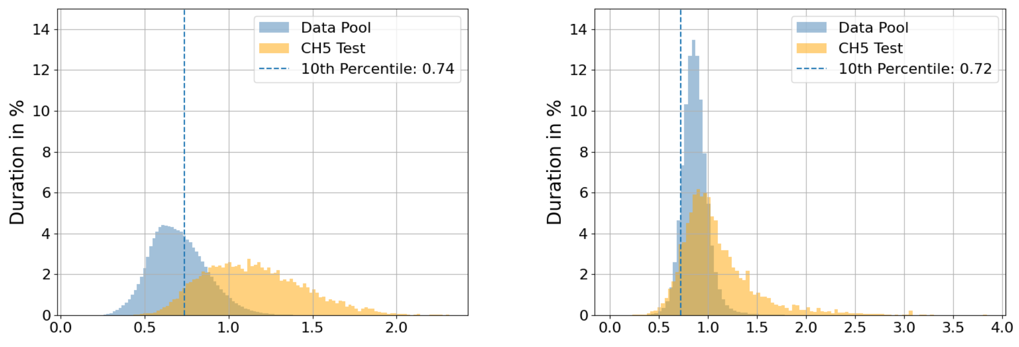
The LDS and Weighted Confidence Score (WCS) distributions are plotted in Figure 6.6. For LDS, the mean LDS for CH5 is highest among the datasets as the same as the mean ADS for CH5. However, the mean of LDS for LSP is lower, indicating it is the most dissimilar from CH5. Considering that LSP is an ASR corpus consisting of audio books and CH5 is a daily conversation at home during dinner, this dissimilarity between two datasets seems to be reasonable. This is the same as the mean WCS for CH5 in Figure 6.6(b). However, the difference between the mean and the others are not as significant as the difference in Figure 6.6(a).



(a) LDS (b) WCS

Figure 6.6 LDS and WCS distributions.

Thresholds for data selection were determined based on the amount of data chosen for semi-supervised learning. Utterances were selected in order by the WER estimate up to 8 hours and 32 hours of data. The thresholds τ_A and τ_L for ADS and LDS, respectively, were set according to the distribution of domain similarity in the test data D_T. Specifically, the thresholds for ADS and LDS were defined as the maximum values within the bottom 10% in CH5 Test data, which were 0.74 and 0.72, respectively. The distributions of ADS and LDS in both the data pool and the target data are illustrated in Figure 6.7.



(a) ADS (b) LDS

Figure 6.7 Distribution of ADS and LDS.

The distribution of ADS in the data pool is relative left-skewed. The majority of utterances have ADS values lower than 1, with a value of 1 indicating the mean ADS of the utterances in the target data. Regarding LDS, the distributions in the data pool overlap significantly with those in the target data and they are denser than the target data distributions.

6.3.4 Data Selection for Speech Recognition

Data for ASR training were selected based on WER estimate and either ADS or LDS. First, data were selected using one of the domain similarities. The duration of data selected using ADS and LDS in Figure 6.7 was 715.18 and 1667.14 hours with WER of 15.34% and 18.34%, respectively. After that, data were selected from the utterances ordered by WER estimate up to 8 and 32 hours. Therefore, the automatically transcribed data selected for semi-supervised learning is highlighted in grey (see Figure 6.8).

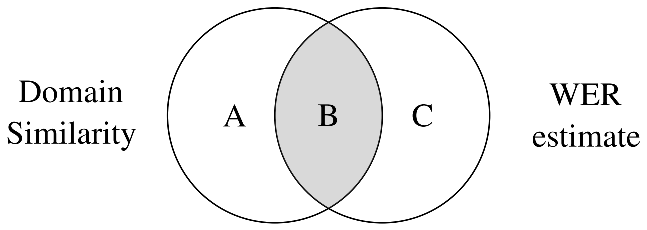


Figure 6.8 Venn Diagram of data selected using domain similarity and WER estimate.

Additionally, the duration of selected data according to the target amounts is summarised in Table 6.6.

Table 6.6 Duration of selected data based on ADS and LDS at target amounts of 8 and 32 hours.

Domain Similarity	**Threshold**	**Duration (hours)**		
		A	B	C
ADS	**0.7368**	707.18	8	14.06
		683.18	32	46.09
LDS	**0.7229**	1659.14	8	0.99
		1635.14	32	4.01

For comparison, training data for an ASR system were selected using five strategies. First, data were selected randomly. Second, they were selected using WCS. The WCS is a weighted confidence score at the character level. The third one is a method using WER estimate in order. The other two methods are combinations of WER and one of the domain similarities.

The statistics of data selected using these strategies are shown in Table 6.7 and 6.8. First, the WER reference increased when data were selected using WCS. For example, the mean WER reference on the automatically transcribed LSP data increased from 13.39% to 67.76% in 8-hour data selection, although the mean WCS was significantly higher than that of LSP using the other strategies. Second, a reduction in WER is observed when data were selected based on WER estimate. In the data selected using the random strategy, there was a similar amount of data from LSP, SWB and TL3, as their durations were dominant. Among these, the mean WER reference on SWB was relatively high, 58.40%. When data were selected based WER estimate, the mean WER reference decreased to 10.62% in 8-hour data selection, while it increased to 77.13% with WCS. Third, when ADS was combined with WER estimate, most WSJ data were filtered out, leaving only 7 utterance in the 32-hour data selection. In contrast, data were not filtered out much using LDS because the distributions of WER were significantly overlapped as shown in Figure 6.7(b).

Table 6.7 Statistics for 8 hours of data selected from a data pool.

Data Selection	**Corpus**	**# Utt.**	**Dur. (h)**	**Avg. WER Ref. (%)**	**Avg. WER Est. (%)**	**Avg. ADS**	**Avg. LDS**	**Avg. Conf.**
Random	AMI	437	0.30	38.54	30.45	0.64	0.95	0.56
	LSP	1,175	4.01	13.39	8.57	0.71	0.87	0.50
	SWB	1,029	1.28	58.40	28.50	0.57	0.91	0.55
	TL3	1,204	2.05	20.39	12.23	0.78	0.86	0.54
	WSJ	164	0.37	13.12	6.88	0.44	0.89	0.36
WCS	AMI	3,872	2.07	35.32	50.23	0.65	1.05	0.91
	LSP	272	0.75	67.76	45.46	0.65	1.02	0.91
	SWB	6,106	3.25	77.13	61.52	0.54	1.03	0.90
	TL3	2,438	1.92	47.48	65.25	0.65	1.09	0.91
	WSJ	10	0.01	25.00	57.88	0.35	1.15	0.89
$\widehat{\text{WER}}$	AMI	587	0.79	4.68	0.93	0.61	0.83	0.47
	LSP	200	0.67	2.01	1.06	0.73	0.90	0.43
	SWB	1,291	2.55	10.62	0.95	0.58	0.90	0.47
	TL3	1,105	2.37	3.99	1.04	0.82	0.82	0.45
	WSJ	724	1.62	3.70	0.91	0.40	0.91	0.29
$\widehat{\text{WER}}$ & ADS	AMI	146	0.21	5.50	1.19	0.81	0.87	0.49
	LSP	407	1.43	2.75	1.37	0.89	0.90	0.45
	SWB	163	0.34	9.09	1.24	0.79	0.90	0.46
	TL3	2,786	6.02	4.43	1.31	0.87	0.83	0.47
	WSJ	1	0.00	6.67	1.32	0.74	0.88	0.47
$\widehat{\text{WER}}$ & LDS	AMI	484	0.69	4.97	0.96	0.62	0.89	0.48
	LSP	228	0.77	2.11	1.09	0.73	0.92	0.43
	SWB	1,306	2.64	10.37	0.97	0.58	0.92	0.47
	TL3	1,045	2.27	3.89	1.07	0.83	0.86	0.46
	WSJ	733	1.65	3.70	0.93	0.40	0.93	0.29

Table 6.8 Statistics for 32 hours of data selected from a data pool.

Data Selection	**Corpus**	**# Utt.**	**Dur. (h)**	**Avg. WER Ref. (%)**	**Avg. WER Est. (%)**	**Avg. ADS**	**Avg. LDS**	**Avg. Conf.**
Random	AMI	1,557	1.13	36.23	28.27	0.63	0.97	0.55
	LSP	4,831	16.59	13.10	8.56	0.72	0.88	0.50
	SWB	4,190	5.24	56.95	27.20	0.56	0.90	0.54
	TL3	4,549	7.69	20.98	12.76	0.78	0.86	0.53
	WSJ	617	1.34	14.78	7.32	0.44	0.89	0.33
WCS	AMI	10,071	5.30	35.34	43.18	0.65	1.08	0.85
	LSP	1,738	4.64	38.85	27.63	0.67	1.01	0.83
	SWB	22,282	13.86	76.21	52.91	0.55	0.99	0.84
	TL3	9,080	8.13	42.68	47.39	0.68	1.03	0.84
	WSJ	49	0.07	18.31	35.63	0.37	1.14	0.83
$\widehat{\text{WER}}$	AMI	1,577	2.13	5.76	1.29	0.62	0.84	0.48
	LSP	1,643	5.77	2.96	1.49	0.73	0.89	0.45
	SWB	3,810	7.91	11.86	1.32	0.59	0.90	0.48
	TL3	5,662	12.22	4.89	1.43	0.82	0.82	0.47
	WSJ	1,760	3.96	4.44	1.26	0.41	0.91	0.30
$\widehat{\text{WER}}$ & ADS	AMI	424	0.61	7.43	1.69	0.82	0.87	0.50
	LSP	2,414	8.69	3.59	1.88	0.88	0.89	0.46
	SWB	520	1.11	13.67	1.72	0.79	0.90	0.49
	TL3	9,901	21.57	5.87	1.78	0.87	0.83	0.48
	WSJ	7	0.02	6.16	1.88	0.77	0.92	0.30
$\widehat{\text{WER}}$ & LDS	AMI	1,301	1.86	5.98	1.34	0.63	0.90	0.49
	LSP	1,826	6.45	2.95	1.53	0.73	0.90	0.45
	SWB	3,939	8.34	11.69	1.37	0.59	0.91	0.48
	TL3	5,208	11.35	4.79	1.47	0.83	0.86	0.47
	WSJ	1,772	4.01	4.41	1.30	0.41	0.92	0.30

6.3.5 Automatic Speech Recognition Performance

The selected data were merged with the supervised data used to train the seed model, and then another HuBERT model was trained on the combined dataset. The model required several thousand epochs to fully train until it became overfitted, which took more than a week on 8 NVIDIA RTX A6000 GPU cards. To complete the training within a practical time frame, the number of epochs was limited to 500 for the comparison of data selection methods. With 32 hours of training data, this amounted to approximately 264 GPU hours—about one and a half days with 8 GPUs, which was considered feasible. The ASR performance with different data selection methods is shown in Table 6.9, 6.10 and 6.11.

With the limited number of epochs, the ASR performance with 32 hours of supervised data was 24.48%. This dataset was combined with ASR transcripts generated by the seed model for semi-supervised learning. The 8 hours of ASR transcripts selected randomly helped improve the ASR performance by an absolute 0.83%. As the amount of selected data increased, the performance improvement also increases. For example, when 32 hours of ASR transcripts were used, the improvement was an absolute 1.512%. The confidence score-based method showed a small gain with 8-hour data selection selection but degraded ASR performance with 32-hour data selection selection.

The ASR performance improved with data selection using WER estimate. The performance improvement with 8 and 32 hours of ASR transcripts was an absolute 1.23% and 1.81%, respectively. The ASR performance was further enhanced when the data selected using both WER estimation and ADS. The improvement was an absolute 1.36% and 1.96% with 8 and 32 hours of ASR transcripts, respectively. The ASR performance improvement of 1.96% represents relative 8%, which is more than marginal. However, the combination of WER estimation and LDS did not yield as much improvement as the combination of WER estimation and ADS.

Table 6.9 Supervised ASR performance on CH5 Dev. The number of epochs is limited to 500.

Data Selection	**Manu. + Auto. (hours)**	**WER(%) of Auto. Trans.**	**WER(%) of HuBERT**
Supervision	32	-	24.480

Table 6.10 Semi-supervised ASR performance on CH5 Dev with 8 hours of data selected using different methods. The number of epochs is limited to 500.

Data Selection	**Manu. + Auto. (hours)**	**WER(%) of Auto. Trans.**	**WER(%) of HuBERT**
random	32 + 8	19.14	23.65
WCS	32 + 8	74.05	23.89
$\widehat{\text{WER}}$	32 + 8	5.89	23.25
$\widehat{\text{WER}}$ & ADS	32 + 8	4.56	**23.12**
$\widehat{\text{WER}}$ & LDS	32 + 8	5.96	23.25

Table 6.11 Semi-supervised ASR performance on CH5 Dev with 32 hours of data selected using different methods. The number of epochs is limited to 500.

Data Selection	**Manu. + Auto. (hours)**	**WER(%) of Auto. Trans.**	**WER(%) of HuBERT**
random	32 + 32	19.00	22.96
WCS	32 + 32	57.10	23.43
$\widehat{\text{WER}}$	32 + 32	6.38	22.67
$\widehat{\text{WER}}$ & ADS	32 + 32	5.81	**22.52**
$\widehat{\text{WER}}$ & LDS	32 + 32	6.41	22.69

To verify the results, the models were evaluated on CH5 Test with additional epochs. They were trained for 3720 epochs, approximately 10 days of training on the 8 GPUs, with 32 hours of training data. Based on the ASR performance in Figure 6.2, the ASR performance with 40 and 64 hours of supervised data is expected to be 22.24% and 21.07%, respectively. The datasets selected for Semi-supervised learning with WER estimation and ADS were used for training. The ASR performance with 8 and 32 hours of selected data was 22.86% and 22.61% in WER, respectively. If the WERs with semi-supervised learning are compared with the estimates from supervised learning, the relative improvement to the supervised learning is 33.33% and 26.66%, while they were -16.13% and -1.90% when data were selected randomly, respectively.

Table 6.12 Supervised ASR performance on CH5 Test.

Data Selection	**Manu. + Auto. (hours)**	**WER(%) of HuBERT**	**WER Reduction**
Supervision	32	23.17	
(extrapolation)	40	22.24	100%
(extrapolation)	64	21.07	100%

Table 6.13 Semi-supervised ASR performance on CH5 Test with 8 hours of data selected using different methods.

Data Selection	**Manu. + Auto. (hours)**	**WER(%) of HuBERT**	**WER Reduction**
random	32 + 8	23.32	-16.13%
$\widehat{\text{WER}}$ & ADS	32 + 8	22.86	33.33%

Table 6.14 Semi-supervised ASR performance on CH5 Test with 32 hours of data selected using different methods.

Data Selection	**Manu. + Auto. (hours)**	**WER(%) of HuBERT**	**WER Reduction**
random	32 + 32	23.21	-1.90%
$\widehat{\text{WER}}$ & ADS	32 + 32	**22.61**	26.66%

6.4 Conclusion

In this chapter, the application of semi-supervised learning for ASR using utterance selection based on domain similarity and WER estimation was explored. First, by using methods such as ADS and LDS, it was aimed to identify utterances most relevant to the target domain for training. Second, Fe-WER and MTR-ER4 were compared in out-of-domain performance and cumulative mean WER to determine a better method for data selection. Then, based on the results, MTR-ER4 was combined with ADS and LDS for data selection. The results demonstrated that integrating WER estimation with domain similarity metrics can yield meaningful improvements in ASR performance with semi-supervised learning. Specifically, the combination of WER estimation and ADS proved to be the most effective approach, offering a relative 2.42% improvement with 32 hours of data. While the method based on WER estimation and LDS did not perform as well, this highlights the importance of tailoring selection strategies to the specific characteristics of the target domain. Future work may further optimise these methods or explore their application to larger and more diverse datasets.

Chapter 7

Robust Error Rate Estimation for Automatically Transcribed Data

In Automatic Speech Recognition (ASR), accurately estimating the error rate of transcriptions is essential for evaluating system performance. This chapter explores two distinct approaches to improving error rate estimation for ASR transcripts in different contexts, with a focus on their effectiveness in semi-supervised learning for ASR, addressing part of the first research question.

The first approach addresses the limitations of traditional Word Error Rate (WER) estimators, particularly the reliance on specific ASR systems or domains. This leads to issues with adaptability and reduced accuracy when applied to unseen or out-of-domain data. To overcome these challenges, an Automatic Speech Recognition System-Independent Word Error Rate Estimation (SIWE) method is introduced. This method generates hypotheses for WER estimation by simulating ASR system output, making it independent of any specific ASR system. By considering phonetically similar or linguistically likely alternatives, SIWE achieves robust WER estimation even on out-of-domain datasets, surpassing traditional WER models in performance metrics such as Root Mean Square Error (RMSE) and Pearson Correlation Coefficient (PCC).

The second approach focuses on the issue of degraded error rate (ER) estimation performance when working with short utterances. Prior work has shown that traditional word-level WER estimators struggle in these scenarios. In response, a Character Error Rate Estimation (Fe-CER) model is proposed for ASR of short utterances. Fe-CER utilises character-level tokenisation to achieve higher resolution in ER estimation on short utterances, providing more precise predictions than word-level models. It is compared against models using phonemes, byte-pair encoding tokens, and words, and demonstrates superior performance in normalised RMSE and PCC.

These methods aim to enhance the robustness of ASR quality estimation, whether by improving performance across different domains and systems or by addressing the challenges posed by short utterances. This chapter will explore the principles behind these two methods and their comparative advantages in real-world ASR scenarios.

7.1 ASR System-Independent Word Error Rate Estimation

Previous WER estimators (Chowdhury and Ali, 2023; Negri et al., 2014), including Fe-WER, have limitations due to their dependency on characteristics of ASR transcripts in training datasets. For example, Whisper tends to produce more insertions in its output. These dependencies may cause performance degradation or a narrow the range of WER estimator uses.

Typically, training datasets for WER estimators are generated based on speech datasets with ground-truth references. These datasets are composed of pairs of speech utterances, hypotheses and target WERs. The WERs are derived by comparing a hypothesis with the reference transcript and computing the edit distance. The hypotheses are generated using a specific ASR system, which makes the WER estimators dependent on the ASR system used to generate the training datasets. In this section, these WER estimators are referred to as system-dependent WER estimators. When estimating the quality of output from another ASR system, system-dependent WER estimators do not perform well and need to be re-trained, which reduces the usefulness. Additionally, these estimators are likely to suffer performance degradation on out-of-domain test data, where unseen errors may occur.

The work in this section aims to address these issues by proposing a System-Independent Word Error Rate Estimation (SIWE) method. Instead of generating training datasets using ASR systems, a range of data augmentation methods are proposed to generate plausible hypotheses. The data augmentation methods insert errors into ground-truth references using three types of strategies: substitution, insertion and deletion errors. The training datasets generated by the proposed methods help SIWE achieve state-of-the-art WER estimation performance. On in-domain test data, SIWE reaches the same performance level as system-dependent WER estimators. Furthermore, SIWE outperforms system-dependent WER estimators on out-of-domain test data.

The main objectives of this work can be summarised as follows:

- A System-Independent WER estimator is proposed, offering a broader range of applicability than system-dependent WER estimators.

- A novel data augmentation method for generating training datasets for WER estimation is introduced.
- The proposed SIWE model matches the performance of system-dependent estimators on in-domain test data and outperforms them on out-of-domain test data.

7.1.1 Proposed Method

7.1.1.1 Hypothesis Generation Strategy

There are three main strategies for hypothesis generation: random selection, phonetic similarity and linguistic probability. These approaches mimic the errors of an ASR system, an acoustic and a language model, respectively. The relation between hypothesis generation strategies and edit types is described in Table 7.1.

Table 7.1 Hypothesis generation strategies and edit types. del.: deletion, sub.: substitution, ins.: insertion.

Strategy	**Del.**	**Sub.**	**Ins.**
Random selection	✓	✓	✓
Phonetic similarity		✓	
Linguistic probability			✓

Random Selection Strategy: Positions for insertions, deletions and substitutions are selected randomly, aiming for a specific target WER.
Phonetic Similarity Strategy: ASR systems often produce errors between phonetically similar words, such as *grief* and *brief*. This strategy generates substitutions by considering phonetic similarity. A word in the reference transcript is converted into a phoneme sequence, e.g., speech to *S P IY CH*. Then, the edit distance between two words is calculated. After calculating all the distances to the other words in a vocabulary list, the top *n* words similar to the reference word are listed. When a word is substituted with another word, the word for substitution is selected from the phonetic similar word list.
Linguistic Probability Strategy: Insertions can occur due to grammatical corrections rather than the errors of the acoustic model. For example, in the phrase *[I do not] know if he is alive*, the words in brackets could be inserted by an ASR model using a language model to predict the most probable words in context.

7.1.1.2 Hypothesis Generation Method

Random Sampling of Transcripts: The first approach to hypothesis generation is to draw samples randomly from the transcripts of the ASR corpus. In other words, the spoken utterances and transcripts are paired randomly. This method guarantees that the vocabulary list, the number of words and the distribution of words in the training, validation and evaluation dataset do not change during hypothesis generation.
Random Sampling of Words: Words in the reference transcript are replaced with randomly chosen words from the vocabulary, while maintaining the overall distribution of words and transcript length. However, this method does not target a specific WER. A more controlled method for setting WER is introduced next.
Edit Generation: To achieve a target WER, individual edits (insertions, deletions, substitutions) are generated. WER is calculated as the ratio of edit errors to the number of words in the reference. Tokens for deletion *[del]* and substitution *[sub]* are first inserted into the transcript. For substitutions, phonetically similar words are chosen using a phonetic similarity matrix. Insertion tokens *[ins]* are then replaced with linguistically probable words, selected based on probabilities obtained from a language model trained on the reference transcripts. The example of the hypothesis generation is described in Table 7.2.

Table 7.2 Example of hypothesis generation.

Step	**Text**										
reference	on	the		morning	of	september	eleventh		two	thousand	and
token(del/sub)	on	the		[sub]	of	[del]	eleventh		two	[sub]	and
replacement	on	the		talking	of		eleventh		two	gunned	and
token(ins)	on	the	[ins]	talking	of		eleventh	[ins]	two	gunned	and
hypothesis	on	the	one	talking	of		eleventh	down	two	gunned	and

7.1.1.3 Data Augmentation

Training data are augmented by merging hypothesis sets of different WERs, where each set is generated individually. Instances may be duplicated when combined, but they are retained to maintain the desired amount of data. Thus, the total amount of data remains constant even when multiple datasets are merged.

7.1.1.4 Evaluation Metrics

RMSE and PCC are used as evaluation metrics for WER estimation. These metrics are common in recent WER estimation studies (Chowdhury and Ali, 2023). RMSE measures the average difference between estimates and actual WERs, while PCC assesses the relationship between them. PCC ranges from -1 to 1, with a value of 1 indicating that estimates increases as targets increase, and a value of 0 indicating no relationship.

7.1.2 Experimental Setup

7.1.2.1 Automatic Speech Recognition Corpora

The Ted-lium 3 (TL3) corpus (Hernandez et al., 2018) was used for training WER estimators. This corpus has been utilised for for the WER estimation task in previous studies: (Chowdhury and Ali, 2023) and Section 4.2. While the TL3 test dataset was used for in-domain evaluation, three evaluation datasets from different domains were used to access the models on out-of-domain data. First, the SC set of Full-corpus-ASR partition of meetings (FCASC) is an evaluation dataset for the AMI corpus (Carletta et al., 2005). It is a multiparty meeting corpus recorded in business meetings with three or four participants, where they played the roles of employees in business situations with or without pre-defined scenarios. Second, the 2000 HUB5 English Evaluation Speech dataset[1] is an English conversational telephone speech dataset. It consists of 20 unreleased conversations from the Switchboard study (Godfrey et al., 1992) as well as 20 conversations from CALLHOME American English Speech[2]. The Switchboard conversations are between two people discussing daily topics, while the CALLHOME conversations are between family members or close friends. Last, the Wall Street Journal (WSJ) corpus is a read speech corpus based on mainly WSJ materials (Paul and Baker, 1992). One of the features of WSJ is its transcripts with verbalised punctuation, e.g., COMMA. For WSJ, there are multiple evaluation datasets with different vocabulary sizes: eval92 5k, eval92 20k, eval93 5k, eval93 20k. For the following experiment, these evaluation datasets were merged into a single dataset for the following experiments. The evaluation datasets from different domains were named as AMI test, SWB/CH test and WSJ test.

7.1.2.2 Automatic Speech Recognition Systems

The training data for WER estimation were generated by transcribing the TL3 training set with different ASR systems as listed in Section 2.1.2: Whisper (Radford et al., 2023), wav2vec 2.0 (Baevski et al., 2020), Chain (Povey et al., 2015), Conformer (Gulati et al., 2020), Transducer (Graves, 2012). These models and their weights were downloaded from

[1]https://catalog.ldc.upenn.edu/LDC2002S09
[2]https://catalog.ldc.upenn.edu/LDC97S42

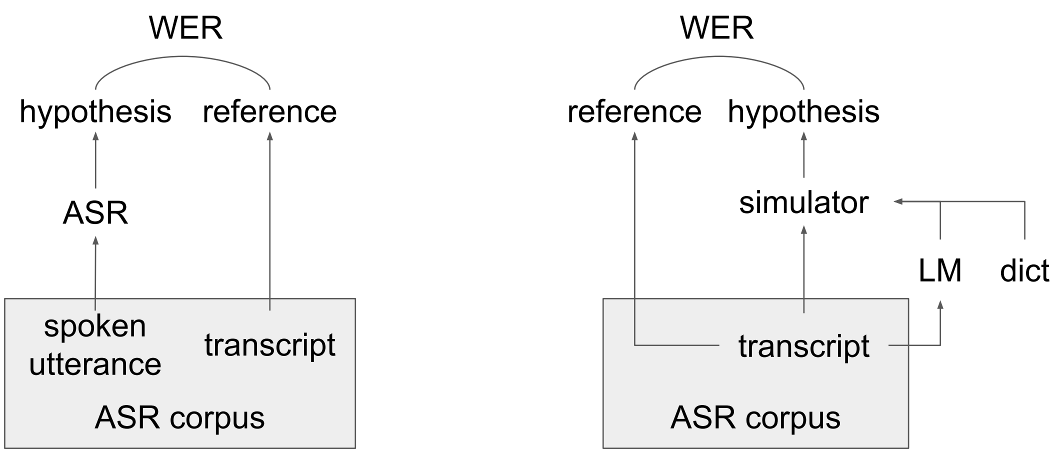
publicly available repositories[3][4][5][6][7], except for the Chain model. The Chain model was trained on 100 hours of LibriSpeech (Panayotov et al., 2015) using LF-MMI (Povey et al., 2016), along with augmented versions created by altering speed and volume. The details of each model, including size and their training data, are summarised in Table 7.3.

Table 7.3 Summary of ASR systems.

	Model	**Trained on**	**Language Model**
ASR1	Whisper	680k from the internet	Transformer
ASR2	wav2vec 2.0	LS 960h	None
ASR3	Chain	LS 100h	3-gram
ASR4	Conformer	LS 960h	Transformer
ASR5	Transducer	LS 960h	RNN

7.1.2.3 Training Datasets

Training datasets for WER estimation were generated in two ways as depicted in Figure 7.1: transcribing the TL3 training dataset using an ASR model and simulating ASR output.



(a) ASR hypothesis (b) Hypothesis generation

Figure 7.1 Illustration of ASR hypothesis and Hypothesis generation.

[3]https://github.com/openai/whisper
[4]https://github.com/facebookresearch/fairseq
[5]https://github.com/kaldi-asr/kaldi
[6]https://huggingface.co/speechbrain
[7]https://github.com/speechbrain/speechbrain

First, the training dataset was transcribed using the ASR models listed in Section 7.1.2.2. To prevent imbalanced data from low WER utterances, the number of utterances with a WER of 0 was limited to the sum of the second and the third most common WER. Second, hypotheses were generated using the hypothesis generation methods described in Section 7.1.1.2 and Table 7.4. For GEN1, spoken utterances and transcripts were randomly and uniformly paired. For GEN2, each word was replaced by another word randomly selected based on the word distribution of the TL3 training dataset. GEN3 was a method to generate hypotheses aiming for target WERs of 2%, 4%...100%. From GEN4 to GEN7, phonetic similarity was used, while GEN8 employed linguistic probability. When the target WER is specified, the number of edits, such as insertions, deletions and substitutions, was distributed almost equally. For example, in GEN7, the WER of insertions, deletions and substitutions were 3.06%, 3.33% and 3.29%, respectively, when the target WER was 10%. Substitutions were generated based on the length of the phonetically similar word list, which varied from 10, 30, 50 and 100 for GEN4 to GEN7. For linguistic probability, a 3-gram language model was trained on the TL3 training data using the SRI Language Modeling toolkit[8]. Several datasets were merged for data augmentation. For example, the hypotheses generated for 10%, 20%, ..., 100% WER were merged into a dataset called GEN7W10-100 when using the GEN7 method. Datasets generated by ASR systems and hypothesis generation methods are summarised in Table 7.5.

Table 7.4 Hypothesis generation methods. PS*m*: phonetically similar *m* words, LS*m*: linguistically similar *m* words

	Methods and Strategies
GEN1	random sampling of transcript
GEN2	random sampling of word
GEN3	edit generation
GEN4	edit generation, PS10
GEN5	edit generation, PS30
GEN6	edit generation, PS50
GEN7	edit generation, PS100
GEN8	edit generation, PS100, LS100

[8]http://www.speech.sri.com/projects/srilm/

Table 7.5 Training datasets for WER estimation. HYPn: transcribed by ASRn, GENnWm: generated by GENn with target WER of m.

Dataset	Hours	WER(%)	Std. Dev.
HYP1 train	256.22	19.13	0.3184
HYP2 train	372.62	16.18	0.1939
HYP3 train	434.78	31.72	0.2419
HYP4 train	356.19	15.80	0.2019
HYP5 train	376.52	17.36	0.1864
GEN1 train	444.62	128.98	2.9786
GEN2 train	444.62	98.90	0.0320
GENnWm train	444.62	approx. m	

7.1.2.4 Evaluation Datasets

The evaluation datasets were generated by the ASR systems listed in Table 7.3. Instances with a WER of 0 were filtered out as described in Section 7.1.2.3. As a result, the total duration of the audio varied based on the ASR system used. The WER estimation models were also evaluated on the out-of-domain test sets: AMI test, Switchboard (SWB) test and WSJ test. Each test set was transcribed using ASR1–5. The WER of the evaluation datasets is summarised in Table 7.6 and 7.7.

Table 7.6 WER on in-domain evaluation datasets.

Dataset	Hours	WER(%)	Std. Dev.
HYP1 test (TL3)	1.97	9.79	0.1935
HYP2 test (TL3)	3.41	12.43	0.1501
HYP3 test (TL3)	4.18	20.99	0.1806
HYP4 test (TL3)	3.38	12.26	0.1673
HYP5 test (TL3)	3.53	13.00	0.1445

7.1.2.5 Word Error Rate Estimator

An Multi-Layer Perceptron (MLP) was employed to predict WER. The layer sizes of the model were [2048, 600, 32, 1]. The outputs of the hidden layers were dropped out at a rate of 0.1. The Adam optimiser was used with a learning rate of 0.007. The cosine scheduler with 15 iterations was applied for learning rate annealing. For feature extraction, the HuBERT

Table 7.7 WER on out-of-domain evaluation datasets.

Dataset	**Hours**	**WER(%)**	**Std. Dev.**
HYP1 test (AMI)	7.34	27.54	0.4068
HYP2 test (AMI)	8.16	38.83	0.3542
HYP3 test (AMI)	8.68	59.46	0.3184
HYP4 test (AMI)	8.18	39.34	0.3555
HYP5 test (AMI)	8.68	40.87	0.3613
HYP1 test (SWB/CH)	2.75	19.70	0.3522
HYP2 test (SWB/CH)	3.37	33.36	0.3569
HYP3 test (SWB/CH)	3.56	65.58	0.2889
HYP4 test (SWB/CH)	3.29	34.12	0.3590
HYP5 test (SWB/CH)	3.56	65.61	0.3575
HYP1 test (WSJ)	2.13	03.07	0.0748
HYP2 test (WSJ)	1.60	12.93	0.1166
HYP3 test (WSJ)	2.22	13.88	0.1332
HYP4 test (WSJ)	1.50	12.58	0.1123
HYP5 test (WSJ)	1.62	13.06	0.1120

large (Hsu et al., 2021) and XLM-R large (Conneau et al., 2020) models were adopted. The averaged representations are 1024-dimensional features.

7.1.3 Results

7.1.3.1 Evaluation on In-Domain Datasets

The performance of the WER estimators trained on ASR hypotheses is summarised in Table 7.8. The estimators performed best when the evaluation dataset was generated with the same ASR system used to transcribe the training dataset. For example, when the estimator was trained on HYP1 train (TL3), its performance was the best on HYP1 test (TL3).

For ASR system-independent WER estimation, the estimators were trained on data augmented with the datasets listed in Table 7.5. SIWE1 and SIWE2, trained using the random sampling methods for data generation, GEN1 and GEN2, respectively, showed relatively poor results, as shown in Table 7.9. However, the RMSE and PCC of SIWE3–8 were comparable to those of ASR system-dependent WER estimators, Fe-WER1–5. SIWE8 outperformed the other SIWE models on most evaluation datasets. When comparison Fe-WERs and SIWEs, the RMSE and PCC values, averaged across HYP1–5 test without

Table 7.8 RMSE and PCC of WER estimators trained and evaluated on ASR hypotheses of TL3. Fe-WER*n* is an estimator trained on HYP*n* train.

Model	**Evaluated on**									
	HYP1 test		**HYP2 test**		**HYP3 test**		**HYP4 test**		**HYP5 test**	
	RMSE↓	**PCC↑**	**RMSE↓**	**PCC↑**	**RMSE↓**	**PCC↑**	**RMSE↓**	**PCC↑**	**RMSE↓**	**PCC↑**
Fe-WER1	**0.0926**	**0.8806**	0.1234	0.6524	0.1876	0.5375	0.1404	0.6658	0.1333	0.5491
Fe-WER2	0.1350	0.7210	**0.0928**	**0.7962**	0.1494	0.6807	0.1151	0.7584	0.1069	0.6921
Fe-WER3	0.1404	0.7003	0.1139	0.7454	**0.1148**	**0.7790**	0.1166	0.7239	0.1165	0.6745
Fe-WER4	0.1275	0.7565	0.1084	0.7323	0.1459	0.6556	**0.1090**	**0.7611**	0.1125	0.6656
Fe-WER5	0.1402	0.6908	0.1017	0.7662	0.1303	0.7148	0.1105	0.7517	**0.1064**	**0.6997**

duration weighting, showed that SIWE8 performed better than Fe-WER1 by 0.0056 and 0.0289, respectively.

Table 7.9 RMSE and PCC of WER estimators trained on GEN hypotheses and evaluated on ASR hypotheses of TL3. SIWE*n* is trained on GEN*n*W*10–100*.

Model	**Evaluated on**									
	HYP1 test		**HYP2 test**		**HYP3 test**		**HYP4 test**		**HYP5 test**	
	RMSE↓	**PCC↑**	**RMSE↓**	**PCC↑**	**RMSE↓**	**PCC↑**	**RMSE↓**	**PCC↑**	**RMSE↓**	**PCC↑**
SIWE1	0.8730	0.0754	0.8447	0.1022	0.7776	0.0516	0.8494	0.0949	0.8416	0.0661
SIWE2	0.8797	0.1964	0.8507	0.2703	0.7845	0.2011	0.8436	0.0829	0.8484	0.1838
SIWE3	0.1542	0.6897	0.1320	0.5994	0.1558	0.6551	0.1626	0.5257	0.1331	0.5707
SIWE4	0.1471	0.7144	0.1222	0.6470	0.1488	0.6721	0.1583	0.5433	0.1315	0.5588
SIWE5	0.1472	0.7225	0.1235	0.6481	0.1503	0.6851	0.1621	0.5294	0.1324	0.5708
SIWE6	0.1484	0.7122	0.1252	0.6305	0.1495	0.6820	0.1612	0.5282	0.1307	0.5756
SIWE7	0.1443	0.7282	0.1195	0.6739	0.1479	0.6830	0.1598	0.5361	0.1283	**0.5959**
SIWE8	**0.1268**	**0.7914**	**0.1133**	**0.7083**	**0.1369**	**0.7152**	**0.1455**	**0.6204**	**0.1266**	0.5946

7.1.3.2 Evaluation on Out-of-Domain Datasets

The WER estimators were evaluated on out-of-domain datasets listed in Table 7.7. For simplicity, the RMSE and PCC values were averaged over HYP1–5 test without duration weighting. The mean RMSE and PCC are shown in Table 7.10. On AMI and SWB/CH tests, the performance of SIWE7 and SIWE8 was better than that of all Fe-WERs, while the results on WSJ test were mixed. On WSJ test, the RMSE and PCC values of all SIWE models were better than those of Fe-WER1, although they were worse than those of Fe-WER4.

Table 7.10 Mean RMSE and PCC of WER estimators evaluated on AMI, SWB/CH and WSJ test.

Model	**AMI test**		**SWB/CH test**		**WSJ test**	
	RMSE↓	**PCC↑**	**RMSE↓**	**PCC↑**	**RMSE↓**	**PCC↑**
Fe-WER1	0.3620	0.5705	0.3581	0.5356	0.2727	0.2623
Fe-WER2	0.4623	0.4523	0.4196	0.4653	0.1186	**0.5274**
Fe-WER3	**0.3286**	**0.6097**	0.3250	**0.5645**	0.2675	0.3944
Fe-WER4	0.5289	0.1741	0.4940	0.1601	**0.1158**	0.4926
Fe-WER5	0.3400	0.5978	**0.3209**	0.5601	0.1657	0.5060
SIWE7	**0.2822**	**0.6764**	**0.2645**	**0.6673**	**0.2177**	0.3628
SIWE8	0.2897	0.6518	0.2882	0.6224	0.2549	**0.3947**

7.1.4 Discussion

7.1.4.1 Data Generation Method

This section discusses the effect of each generation method for WER estimation on both in-domain and out-of-domain data. First, the edit generation method, GEN3, significantly improved the performance of the estimators compared to GEN1 and GEN2. In Table 7.9, the mean RMSE of SIWE3 was 0.1542, while those of SIWE1 and SIWE2 were 0.8730 and 0.8797, respectively. The larger amount of training data contributed to the improved performance of the estimator because the training dataset for SIWE3 was the merged dataset GEN3W*10–100*. Additionally, all PCC values of the estimators trained on GEN3–7 were over 0.5, while those of SIWE1 and SIWE2 were below 0.3.

Second, using phonetic similarity for substitution also brought an additional gain in the estimators' performance. As the size of the phonetically similar word list increased from 10 to 100, the RMSE and PCC of SIWE improved as well. Furthermore, when linguistic probability was added to the generation method, GEN8, the performance notably improved on all evaluation datasets. For example, the mean RMSE and PCC of the SIWE8 on HYP1–5 test were 0.1298 and 0.6860, while those of SIWE7 were 0.1400 and 0.6434, respectively.

However, the performance improvement from GEN8 was not achieved on out-of-domain data. As shown in Table 7.10, the results of SIWE8 lagged behind those of SIWE3–7 on AMI and SWB/CH tests. This suggests that the linguistic information learned from the TL3 corpus might not be effective for WER estimation on different corpora.

7.1.4.2 Data Augmentation

The performance of SIWEs was generally better than that of Fe-WERs on out-of-domain data, except for WSJ test. To further analyse, SIWEs were trained on datasets with different ranges of target WER. For example, training data with WERs of 2%, 4%, up to 20%. The amount of data remained unchange compared to the datasets used in Table 7.10 as the same number of datasets were merged. Performance improved when the WER range of the training dataset was closer to the WER of evaluation datasets. For example, SIWE7W2–20 outperformed the others in terms of both mean RMSE and PCC when the mean WER of HYP1–5 test (WSJ) was 11.10%. Similarly, SIWE7W42–60 outperformed the others on SWB/CH test, where the mean of WER of HYP1–5 test (SWB/CH) was 43.67%. These results shown in Table 7.11 outperform those of SIWE7 in Table 7.10.

Table 7.11 Mean RMSE and PCC of SIWE7 trained on the datasets comprising different target WER ranges. Mean WERs of HYP1–5 test of AMI, SWB/CH and WSJ are 0.4121, 0.4367 and 0.1110, respectively.

WER range	**AMI test**		**SWB/CH test**		**WSJ test**	
	RMSE↓	**PCC↑**	**RMSE↓**	**PCC↑**	**RMSE↓**	**PCC↑**
W2–20	0.3454	0.6618	0.3243	0.6321	**0.1092**	**0.4546**
W12–30	0.3106	0.6818	0.2886	0.6592	0.1193	0.4219
W22–40	0.2957	0.6771	0.2714	0.6678	0.1452	0.3929
W32–50	0.2835	0.6869	0.2636	0.6720	0.1673	0.3783
W42–60	0.2765	**0.6900**	**0.2593**	**0.6731**	0.1961	0.3871
W52–70	**0.2747**	0.6827	0.2642	0.6657	0.2610	0.3682
W62–80	0.2755	0.6844	0.2673	0.6663	0.3063	0.3700
W72–90	0.2803	0.6811	0.2746	0.6651	0.3191	0.4036
W82–100	0.3023	0.6870	0.3125	0.6627	0.4588	0.3704

7.1.5 Conclusion

Hypothesis generation and data augmentation methods for ASR system-independent WER estimation were proposed in this section. The estimator trained on hypotheses generated by this proposed method outperformed baseline models on out-of-domain datasets. Linguistic information, as a hypothesis generation strategy, contributed to improving the performance of WER estimators on in-domain datasets. The use of phonetic similarity for substitutions improved the performance of the estimators on both in-domain and out-of-domain datasets. Additionally, on out-of-domain datasets, the performance of estimators was further marginally improved when hypotheses were generated with a target WER close to that of the evaluation dataset.

7.2 Character Error Rate Estimation for ASR of Short Utterances

Despite WER-based quality estimation usefulness, it has been shown to face performance degradation in more complex settings, particularly when the training dataset contains short utterances with high variance of WER, especially when the dataset comprises relatively short utterances. For example, when WER is measured of a one word utterance, the quantisation of WER means that it will be 0%, 100%, 200%, etc. The high quantisation noise in this case leads to high means and variances. Inspired by this observation, various tokenisation strategies are explored for Error Rate (ER) estimation on short utterances in this study.

Transcripts are tokenised into smaller units than words, such as phonemes, characters or Byte Pair Encoded (BPE) tokens, to reduce the quantisation noise. Using these units, Phone Error Rate (PER) (Fang et al., 2020; Li et al., 2020), Character Error Rate (CER) (Conneau et al., 2023; Sato et al., 2022; Zhang et al., 2020a) and Token Error Rate (TER) (Chuang et al., 2021; Zhou et al., 2020) are applied for ER estimation in ASR in this section. As transcripts are tokenised in a smaller unit, the denominator of error rate tends to be larger. As a result, the metric becomes higher resolution metric. This approach has been found to improve ER estimation performance not only on short utterance but also across a wider range of utterance lengths. The tokenisation strategies are evaluated on several ASR corpora: TL3 (Hernandez et al., 2018) and CHiME-5 (CH5) (Barker et al., 2018).

The key outcomes of this work are:

- An ER estimation model using CER (Fe-CER) for ASR corpora consisting of relatively short utterances.
- A comparison of ER estimation models using different tokenisation strategies, evaluated using normalised root mean square error (nRMSE) as the evaluation metric.
- Experimental results on a range of corpora.

7.2.1 Proposed Method

The Fe-WER model was adopted to predict CER, PER and TER as well as WER (see Section 4.2.1.1).

7.2.1.1 Tokenisation

WER is widely used as a target for ER estimation for ASR, providing an analysis on word-level tokenisation. To address the low resolution of WER on short utterances, other tokenisation schemes, such as CER, PER and TER, are introduced for ER estimation. First, CER offers a finer-grained analysis by estimating error rates at the character level instead of word level. Second, PER is determined by mapping words to phonemes, which are the distinct units of sound in a language. Last, TER is calculated using BPE tokens in the transcripts. An example of tokenisation is shown in Table 7.12. A normalised sentence, "i started reading about alzheimer is and tried to familiarise myself with the research", is assumed before tokenisation.

Table 7.12 Examples of different tokenisation levels for error rate estimation.

	Tokenisation
WER	[’i’, ’started’, ’reading’, ’about’, ’alzheimer’, ’is’, ’and’, ’tried’, ’to’, ’familiarize’, ’myself’, ’with’, ’the’, ’research’]
CER	[’i’, ’ ’, ’s’, ’t’, ’a’, ’r’, ’t’, ’e’, ’d’, ’ ’, ’r’, ’e’, ’a’, ’d’, ’i’, ’n’, ’g’, ’ ’, ’a’, ’b’, ’o’, ’u’, ’t’, ’ ’, ’a’, ’l’, ’z’, ’h’, ’e’, ’i’, ’m’, ’e’, ’r’, ’ ’, ’i’, ’s’, ’ ’, ’a’, ’n’, ’d’, ’ ’, ’t’, ’r’, ’i’, ’e’, ’d’, ’ ’, ’t’, ’o’, ’ ’, ’f’, ’a’, ’m’, ’i’, ’l’, ’i’, ’a’, ’r’, ’i’, ’z’, ’e’, ’ ’, ’m’, ’y’, ’s’, ’e’, ’l’, ’f’, ’ ’, ’w’, ’i’, ’t’, ’h’, ’ ’, ’t’, ’h’, ’e’, ’ ’, ’r’, ’e’, ’s’, ’e’, ’a’, ’r’, ’c’, ’h’]
PER	[’AY’, ’S’, ’T’, ’AA’, ’R’, ’T’, ’AH’, ’D’, ’R’, ’EH’, ’D’, ’IH’, ’NG’, ’AH’, ’B’, ’AW’, ’T’, ’AE’, ’L’, ’Z’, ’HH’, ’AY’, ’M’, ’ER’, ’IH’, ’Z’, ’AH’, ’N’, ’D’, ’T’, ’R’, ’AY’, ’D’, ’T’, ’UW’, ’F’, ’AH’, ’M’, ’IH’, ’L’, ’Y’, ’ER’, ’AY’, ’Z’, ’M’, ’AY’, ’S’, ’EH’, ’L’, ’F’, ’W’, ’IH’, ’DH’, ’DH’, ’AH’, ’R’, ’IY’, ’S’, ’ER’, ’CH’]
TER	[b’i’, b’ started’, b’ reading’, b’ about’, b’ al’, b’zheimer’, b’ is’, b’ and’, b’ tried’, b’ to’, b’ familiar’, b’ize’, b’ myself’, b’ with’, b’ the’, b’ research’]

The target error rates are calculated by first performing dynamic programming alignment, and then using the equation:

$$\mathrm{ER}_x = \frac{S_x + D_x + I_x}{N_x} \tag{7.1}$$

where x represents W(Word), C(Character), P(Phoneme) and T(Token) and N_x is the number of x in the reference. S, D and I are the number of substitutions, deletions and insertions, respectively.

7.2.1.2 Training Objective

The objective function for model training employs the Mean Squared Error (MSE) between the target ER and the ER estimate, where ER represents the ER between references and hypotheses, and $\widehat{ER}$ is the model's estimation. Here, N denotes the number of instances in the dataset, and i serves as the index for each instance.

$$\mathrm{MSE}_x = \frac{\sum_{i=1}^{N}(\mathrm{ER}_{x,i} - \widehat{\mathrm{ER}}_{x,i})^2}{N}. \tag{7.2}$$

7.2.1.3 Evaluation

The performance of the ER estimation models with WER, CER, PER and TER is evaluated on test datasets by normalised RMSE and PCC. As described in Section 4.1.1.3, ER estimation models have been evaluated using RMSE and PCC. RMSE represents the average difference between targets and estimates. However, when comparing the performance of ER estimation models using different metrics, their results cannot be directly compared using RMSE due to the different distributions of the target values. To account for this, RMSE is normalised by the standard deviation σ of the target error rate. The variance of the target ER can be considered as the MSE of a model that always predicts the mean of the target. By normalising RMSE by the standard deviation of the target ER, the improvement of the presented model over one that predicts the mean can be measured. An nRMSE of less than 1 indicates improvement, with lower nRMSE values reflecting greater performance gains. The normalised RMSE is defined as:

$$\mathrm{nRMSE}_x = \frac{\sqrt{\mathrm{MSE}_x}}{\sigma_x} \tag{7.3}$$

PCC quantifies the degree of linear association between two variables. It ranges from -1 to 1, where -1 indicates a complete negative linear correlation, 0 implies no correlation, and +1 signifies a perfect positive correlation.

$$\mathrm{PCC} = \frac{\sum_{i=1}^{N}(\mathrm{ER}_{x,i} - \mu_{\mathrm{ER}_x})(\widehat{\mathrm{ER}}_{x,i} - \mu_{\widehat{\mathrm{ER}_x}})}{\sqrt{\sum_{i=1}^{N}(\mathrm{ER}_{x,i} - \mu_{\mathrm{ER}_x})^2 \sum_{i=1}^{N}(\widehat{\mathrm{ER}}_{x,i} - \mu_{\widehat{\mathrm{ER}_x}})^2}} \tag{7.4}$$

where μ_{ER_x} is the mean of ER_x.

7.2.2 Experimental Setup

7.2.2.1 Data

Two datasets are used in the experiments: TL3 (Hernandez et al., 2018) and CH5 (Barker et al., 2018). TL3 is a dataset containing speech from public talks on various topics, while CH5 is an ASR corpus of daily conversation in home environments and consists of relatively short utterances. The statistics of TL3 and CH5 are summarised in Table 7.13. The average duration of CH5 utterances is shorter than that for TL3. The ER estimation models will be evaluated on CH5 to determine whether a model using different tokenisation strategies outperforms the Fe-WER model when trained on the dataset with short utterances. The reference transcripts are pre-processed to be lower-case and have punctuation removed. They are transcribed using the Whisper large model (Radford et al., 2023). The hypotheses are pre-processed in the same way.

Table 7.13 Statistics of the two datasets.

		# Segments	**Total Dur. (h)**	**Avg. Dur. (s)**
TL3	train	123,255	200.55	5.86
	dev	1,034	1.70	5.93
	test	842	1.41	6.04
CH5	train	70,483	35.86	1.83
	dev	6,200	4.41	2.56
	test	9,918	5.23	1.90

7.2.2.2 Error Rate Estimation Using Characters, Phonemes and Tokens

The performance of ER estimation models using CER, PER and TER is compared with Fe-WER. The WER, CER, PER and TER estimators utilise average pooling over either the frame or token dimensions as an aggregation method. The models comprise an MLP with two hidden layers and an output layer, and activation functions applied to the concatenated feature layer. Additionally, batch normalisation is applied to the output of each layer, except for the output layer, and dropout is implemented on the hidden layers.

The hyper-parameters for these models are selected through grid search. The optimiser, activation function for the hidden and output layers, and hidden layer dimensions are described in Table 7.14. The estimators are trained using a cosine annealing scheduler with early stopping after 40 epochs.

For tokenisation strategies, the hypothesis is split by space for WER because the text comprises only English letters and spaces without punctuation after text normalisation. For CER, spaces are treated as character because their position affects pronunciation. Words are mapped to phonemes using the dictionary provided by the TL3 corpus[9] for both TL3 and CH5 for PER. While spaces are ignored for tokenisation at the phoneme level, they are included in BPE tokens. For TER, tiktoken[10] has been adopted, which was used for Whisper (Radford et al., 2023) and GPT models(Brown et al., 2020).

Table 7.14 Model hyper-parameters for training on TL3 and CH5 datasets.

		Learning Rate	**Activation Function**	**Layers**
WER	TL3	0.001	Sigmoid	600, 32
	CH5	0.0003	Sigmoid	300, 16
CER	TL3	0.007	Sigmoid	600, 16
	CH5	0.0007	Sigmoid	300, 16
PER	TL3	0.007	Clamp	300, 32
	CH5	0.0003	Sigmoid	600, 32
TER	TL3	0.003	Sigmoid	600, 32
	CH5	0.0007	Sigmoid	300, 16

7.2.3 Results

To test whether there is a correlation between WER and the other ERs, PCC was calculated as shown in Table 7.15. The results indicate that WER is highly correlated with these metrics, showing a strong positive linear relationship between the variables.

Table 7.15 PCC between WER and CER, PER and TER.

	WER/CER	**WER/PER**	**WER/TER**
TL3	0.9429	0.9383	0.9704
CH5	0.9598	0.9565	0.9846

The performance of ER estimation models using different metrics were measured by RMSE and PCC in percentage as shown in Table 7.16, because these results are compared

[9]https://www.openslr.org/51
[10]https://github.com/openai/tiktoken

with the mean and standard deviation of WER, CER, PER and TER in percentage as shown in Table 7.17. While Fe-WER outperformed the other models on TL3 in terms of PCC, Fe-CER achieved the best performance in terms of RMSE. On CH5, Fe-CER outperformed all other models in both RMSE and PCC. The detailed results are presented in Table 7.16.

Table 7.16 RMSE and PCC of ER estimation models.

		WER(%)	**CER(%)**	**PER(%)**	**TER(%)**
TL3	RMSE	8.77	**8.03**	8.08	9.86
	PCC	**89.95**	89.89	89.94	87.96
CH5	RMSE	32.02	**28.21**	29.56	32.58
	PCC	61.54	**66.95**	65.05	60.59

The higher RMSE of Fe-WER on TL3 compared to that of Fe-CER could be due to the different distributions of target ERs. For example, the average and standard deviation of WER on TL3 were 0.1429 and 0.1997, respectively, whereas those of CER were 0.1061 and 0.1816, respectively. The average and standard deviation of target ERs are summarised in Table 7.17.

Table 7.17 Average and standard deviation of target ERs.

		WER(%)	**CER(%)**	**PER(%)**	**TER(%)**
TL3	Average	14.29	10.61	10.66	15.79
	Std. Dev.	19.97	18.16	18.39	20.63
CH5	Average	36.65	31.74	32.46	37.89
	Std. Dev.	40.36	37.83	38.67	40.67

To facilitate a fair comparison between the ER estimation models using different metrics, the RMSE was normalised by the standard deviation as described in Section 7.2.1.3. In terms of nRMSE, Fe-WER outperformed the other models, contrasting with the results on TL3 in Table 7.16. The nRMSEs values of the models are exhibited in Table 7.18.

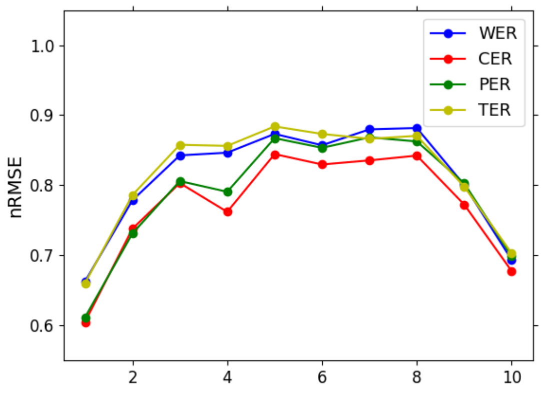
On CH5, Fe-CER's nRMSE was found to be relatively 6% lower than that of Fe-WER and its PCC was relative 8.79% higher. On TL3, However, Fe-WER outperformed the other models in both nRMSE and PCC. These results suggest that Fe-CER is particularly effective when trained on datasets with relatively short utterances, such as CH5.

Table 7.18 nRMSE of ER estimation models.

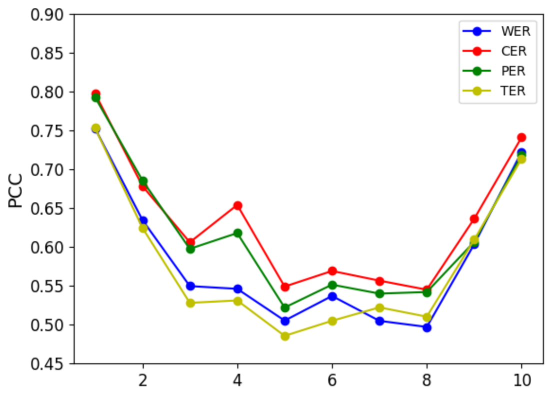
		WER(%)	CER(%)	PER(%)	TER(%)
TL3	nRMSE	**43.91**	44.21	43.94	47.79
CH5	nRMSE	79.33	**74.57**	76.44	80.10

7.2.4 Analysis

The different models were compared to assess performance improvements on short utterances. For the analysis, utterances were sorted by duration in ascending order. Then, they were split into 10 groups. Figure 7.2 displays the nRMSE and PCC values for the 10 bins of utterances CH5 evaluation datasets.



(a) nRMSE



(b) PCC

Figure 7.2 nRMSE and PCC of ER estimates on CH5. The ordered utterances are grouped into 10 bins, with each bin containing an equal number of utterances. The utterances are first sorted by their length before binning.

The ER estimation model using CER showed better performance on CH5. The nRMSE of Fe-CER was relatively 8.72% lower than that of Fe-WER and its PCC was relatively 5.96% higher. Fe-CER outperformed Fe-WER particularly in the fourth bin of CH5 with nRMSE and PCC improvements of relative 9.97% and 19.75%, respectively. The performance of Fe-CER in terms of nRMSE and PCC was superior when the model was trained on datasets with relatively short utterances, such as CH5.

7.2.5 Conclusion

For evaluating ASR output on short utterances, an ER estimation model using CER is proposed. The performance of ER estimation models with various tokenisation strategies was evaluated using nRMSE and PCC. The results demonstrate that Fe-CER outperforms Fe-WER on the CH5 dataset, which consists of relatively short utterances, by 6.00% and 8.79% relative in both nRMSE and PCC, respectively. In the analysis of performance of the ER estimation models across utterance durations, Fe-CER consistently outperforms the other models on CH5.

7.3 Summary

In this chapter, two approaches to improving error rate estimation for ASR transcripts were proposed: system-independent WER estimation and CER estimation for short utterances.

The first section introduced the SIWE model, which addressed the limitations of system-dependent WER estimators by generating training datasets through hypothesis generation. By employing a variety of error insertion strategies—such as phonetic similarity and linguistic probability—the SIWE model achieved comparable results to system-dependent estimators on in-domain data and demonstrated superior performance on out-of-domain datasets. This flexibility made SIWE a robust solution for WER estimation, particularly in real-world scenarios where ASR systems varied or domain-specific data were unavailable.

In the second section, the CER estimation model, Fe-CER, was presented to address the limitations of WER-based estimation on short utterances. Traditional WER models generated high quantisation noise when applied to short utterances, leading to performance degradation. By tokenising transcripts at the character level, Fe-CER offered a higher resolution for error rate estimation and outperformed Fe-WER on the CH5 dataset, which consists of short utterances.

Together, these two approaches offered a robust framework for error rate estimation. SIWE provided a flexible, system-independent solution applicable to diverse domains, while Fe-CER excelled in handling short utterances. Future work could focus on integrating these approaches for broader applications or exploring additional error generation strategies and tokenisation methods to further improve estimation accuracy in challenging ASR environments.

Chapter 8

Conclusion

This thesis explored several innovative approaches to improving the performance of Automatic Speech Recognition (ASR) systems through semi-supervised learning with domain similarity and WER estimation. The focus has been on addressing key challenges associated with leveraging untranscribed data, selecting domain-relevant utterances, and estimating the accuracy of ASR transcripts, all of which are crucial for enhancing ASR systems in real-world applications.

In the chapter on Transcript Information in Utterance Representation, the study investigated how much linguistic information from transcripts is retained in utterance representations. The aim was to understand if this information could be used for accuracy estimation of ASR output, traditionally independent of ASR decoding. A model was developed to align a distance between utterance representations and an edit distance of their corresponding transcripts. The results demonstrated a strong correlation, confirming that transcript information is indeed preserved in utterance representations. This finding was leveraged to propose a method for matching spoken utterances and their transcripts, opening up new possibilities for not only generating a training dataset from unpaired utterances and transcripts but also estimating Word Error Rate (WER) without ASR decoding, potentially simplifying accuracy estimation for ASR output in real-world applications.

In the chapter on Word Error Rate Estimation of Automatically Transcribed Data, three main approaches for WER estimation are explored: Deep Metric Learning (DML) for WER (DML-WER), the Fast estimation for WER Estimator (Fe-WER) and Multi-Target Regressions for Error Rate (ER) estimation. These models aimed to estimate the accuracy of ASR output by calculating WER between reference and ASR transcripts using deep learning models trained on transcript and utterance representations. Fistly, DML-WER learned a distance metric between utterance and transcript representations using deep metric learning, which was found to correlate strongly with actual WER. The second approach, Fe-WER, employed a more efficient estimation model based on average pooling with Self-Supervised Learning Representations (SSLRs). Fe-WER demonstrated both higher performance in terms of Root Mean Square Error (RMSE) and Pearson Correlation Coefficient (PCC), as well as improved computational efficiency. Finally, MTR-ER4 extended WER estimation to include individual error rates, such as substitution, insertion, and deletion rates. The results showed that MTR-ER4 achieved strong performance across multiple datasets, offering a more granular approach to estimating transcript accuracy by providing insights into the types of errors present.

In the chapter on Unsupervised Domain Similarity Measurements, two domain similarity measures—Acoustic Domain Similarity (ADS) and Linguistic Domain Similarity (LDS)—are introduced along with their applications in improving ASR performance. ADS presents a method to measure the acoustic similarity between data, essential for improving ASR performance, especially when handling untranscribed data. The method uses unsupervised techniques, leveraging contrastive loss ratios. A submodular data selection strategy is proposed to identify a representative subset of data using the domain similarity. The ADS-based selection method enhances the performance of ASR systems, particularly when training data are selected from a multi-domain data pool. LDS was introduced to quantify how closely the linguistic characteristics of source and target datasets align. Discrete units derived from acoustic representations are used as linguistic tokens. LDS is calculated through an n-gram language model. The method is particularly useful in semi-supervised learning scenarios where reference transcripts are unavailable. Experiments demonstrate a strong correlation between edit distances of discrete unit sequences and word sequences, confirming the effectiveness of discrete units in capturing linguistic domain similarity.

In the chapter on Semi-Supervised Learning for Automatic Speech Recognition with Utterance Selection, the goal is to improve the performance of ASR systems by utilising both manually transcribed and automatically generated transcripts. A key challenge in this approach is ensuring the accuracy of the ASR transcripts, as higher-accuracy transcripts lead to better performance improvements. The study introduced methods for estimating WER independently of ASR system outputs, allowing for more reliable selection of high-accuracy transcripts without relying on confidence scores derived from ASR decoding. Additionally, the issue of domain mismatch was addressed by developing techniques to measure ADS and LDS between utterances and the target data. These measures enabled the selection of in-domain data for training, which is critical for avoiding performance degradation in ASR systems. The findings demonstrated that integrating WER estimation with domain similarity metrics—particularly ADS—significantly improved ASR performance.

In the chapter on Robust Error Rate Estimation for Automatically Transcribed Data, significant contributions to the field of ASR output ER estimation are presented, introducing approaches that enhance both the generalisability and precision of the estimations. The innovations outlined—ASR system-independent Word Error Rate estimation (SIWE) and Character Error Rate Estimation (Fe-CER) for short utterances—address key challenges in accuracy estimation of ASR output across diverse datasets and utterance lengths. The System-Independent Word Error Rate Estimation (SIWE) demonstrates that it is possible to decouple error rate estimation from specific ASR systems. By employing novel data augmentation strategies, SIWE is able to simulate plausible ASR errors in a way that reduce system dependencies, offering a more flexible solution for real-world ASR applications where ground-truth references are not available. The introduction of Fe-CER underscores the importance of fine-grained tokenisation strategies for improving error estimation in cases where traditional WER-based metrics showing performance degradation, especially with short utterances. By tokenising at the character level, Fe-CER reduces the high variance and quantisation noise, thereby offering more accurate and reliable error estimates in such cases.

While these contributions represent significant advancements in the field, several promising directions remain for future research. One such direction is the further analysis of factors influencing WER estimation performance. Although the proposed models perform well, their behaviour under varying conditions is not yet fully understood. Investigating the impact of factors such as utterance length, acoustic conditions, and the average WER of the reference dataset could help identify potential limitations and inform refinements to enhance the robustness and generalisability of WER estimation models. Additionally, further research is needed to examine the relationship between human-defined acoustic domains and the unsupervised acoustic domain similarity metrics proposed in this thesis. Identifying which human-defined acoustic categories correlate with computed ADS values could improve model interpretability and bridge the gap between machine-driven metrics and human perception of acoustic environments. Finally, a key direction for future work is the unification of the metrics developed in this thesis—WER estimation, ADS and LDS—into a cohesive framework. This includes developing methods to automatically determine optimal thresholds for these metrics, facilitating seamless integration into ASR systems. A unified framework would streamline the deployment of semi-supervised learning techniques, optimise data selection, and enhance ASR performance across diverse applications.

References

Afshan, A., Kumar, K., and Wu, J. (2021). Sequence-level confidence classifier for ASR utterance accuracy and application to acoustic models. In *Interspeech 2021*, pages 4084–4088.

Ali, A. and Renals, S. (2018). Word error rate estimation for speech recognition: e-WER. In Gurevych, I. and Miyao, Y., editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 20–24, Melbourne, Australia. Association for Computational Linguistics.

Ali, A. and Renals, S. (2020). Word error rate estimation without ASR output: e-WER2. In *Interspeech 2020*, pages 616–620.

Amodei, D., Ananthanarayanan, S., Anubhai, R., Bai, J., Battenberg, E., Case, C., Casper, J., Catanzaro, B., Cheng, Q., Chen, G., Chen, J., Chen, J., Chen, Z., Chrzanowski, M., Coates, A., Diamos, G., Ding, K., Du, N., Elsen, E., ..., and Zhu, Z. (2016). Deep Speech 2 : End-to-end speech recognition in English and Mandarin. In Balcan, M. F. and Weinberger, K. Q., editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 173–182, New York, New York, USA. PMLR.

Ardila, R., Branson, M., Davis, K., Kohler, M., Meyer, J., Henretty, M., Morais, R., Saunders, L., Tyers, F., and Weber, G. (2020). Common Voice: A massively-multilingual speech corpus. In Calzolari, N., Béchet, F., Blache, P., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4218–4222, Marseille, France. European Language Resources Association.

Asami, T., Masumura, R., Masataki, H., Okamoto, M., and Sakauchi, S. (2015). Training data selection for acoustic modeling via submodular optimization of joint Kullback-Leibler divergence. In *Interspeech 2015*, pages 3645–3649.

Babu, A., Wang, C., Tjandra, A., Lakhotia, K., Xu, Q., Goyal, N., Singh, K., von Platen, P., Saraf, Y., Pino, J., Baevski, A., Conneau, A., and Auli, M. (2022). XLS-R: Self-supervised cross-lingual speech representation learning at scale. In *Interspeech 2022*, pages 2278–2282.

Baevski, A., Hsu, W.-N., Xu, Q., Babu, A., Gu, J., and Auli, M. (2022). data2vec: A general framework for self-supervised learning in speech, vision and language. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S., editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 1298–1312. PMLR.

Baevski, A., Zhou, Y., Mohamed, A., and Auli, M. (2020). wav2vec 2.0: A framework for self-supervised learning of speech representations. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 12449–12460. Curran Associates, Inc.

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

Barker, J., Watanabe, S., Vincent, E., and Trmal, J. (2018). The fifth 'CHiME' speech separation and recognition challenge: Dataset, task and baselines. In *Interspeech 2018*, pages 1561–1565.

Baum, L. E. and Petrie, T. (1966). Statistical inference for probabilistic functions of finite state Markov chains. *The Annals of Mathematical Statistics*, 37(6):1554 – 1563.

Beltagy, I., Lo, K., and Cohan, A. (2019). SciBERT: A pretrained language model for scientific text. In Inui, K., Jiang, J., Ng, V., and Wan, X., editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620, Hong Kong, China. Association for Computational Linguistics.

Benajiba, Y., Sun, J., Zhang, Y., Jiang, L., Weng, Z., and Biran, O. (2019). Siamese networks for semantic pattern similarity. In *IEEE ICSC*, pages 191–194.

Bertoldi, N., Brugnara, E., Cettolo, M., Federico, M., and Giuliani, D. (2001). From broadcast news to spontaneous dialogue transcription: Portability issues. In *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221)*, volume 1, pages 37–40 vol.1.

Boeddecker, C., Heitkaemper, J., Schmalenstroeer, J., Drude, L., Heymann, J., and Haeb-Umbach, R. (2018). Front-end processing for the CHiME-5 dinner party scenario. In *Proc. 5th International Workshop on Speech Processing in Everyday Environments (CHiME 2018)*, pages 35–40.

Bourlard, H. and Morgan, N. (1989). A continuous speech recognition system embedding MLP into HMM. In Touretzky, D., editor, *Advances in Neural Information Processing Systems*, volume 2. Morgan-Kaufmann.

Bridle, J. S. (1990). Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In Soulié, F. F. and Hérault, J., editors, *Neurocomputing*, pages 227–236, Berlin, Heidelberg. Springer Berlin Heidelberg.

Bromley, J., Guyon, I., LeCun, Y., Säckinger, E., and Shah, R. (1993). Signature verification using a "Siamese" time delay neural network. In Cowan, J., Tesauro, G., and Alspector, J., editors, *NIPS*, volume 6. Morgan-Kaufmann.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., ..., and Amodei, D. (2020). Language models are few-shot learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Callison-Burch, C., Koehn, P., Monz, C., Post, M., Soricut, R., and Specia, L. (2012). Findings of the 2012 workshop on statistical machine translation. In Callison-Burch, C., Koehn, P., Monz, C., Post, M., Soricut, R., and Specia, L., editors, *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 10–51, Montréal, Canada. Association for Computational Linguistics.

Carletta, J., Ashby, S., Bourban, S., Flynn, M., Guillemot, M., Hain, T., Kadlec, J., Karaiskos, V., Kraaij, W., Kronenthal, M., Lathoud, G., Lincoln, M., Lisowska, A., McCowan, I., Post, W., Reidsma, D., and Wellner, P. (2005). The AMI meeting corpus: A pre-announcement. In *Proceedings of the Second International Conference on Machine Learning for Multimodal Interaction*, MLMI'05, page 28–39, Berlin, Heidelberg. Springer-Verlag.

Chapelle, O., Scholkopf, B., and Zien, A. (2006). *Semi-Supervised Learning*. The MIT Press.

Chen, S., Wang, C., Chen, Z., Wu, Y., Liu, S., Chen, Z., Li, J., Kanda, N., Yoshioka, T., Xiao, X., Wu, J., Zhou, L., Ren, S., Qian, Y., Qian, Y., Wu, J., Zeng, M., Yu, X., and Wei, F. (2022). WavLM: Large-scale self-supervised pre-training for full stack speech processing. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1505–1518.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. (2020a). A simple framework for contrastive learning of visual representations. In III, H. D. and Singh, A., editors, *ICML*, volume 119 of *Mach. Learn. Res.*, pages 1597–1607. PMLR.

Chen, Y., Ding, W., and Lai, J. (2023). Improving noisy student training on non-target domain data for automatic speech recognition. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5.

Chen, Y., Wang, W., and Wang, C. (2020b). Semi-supervised ASR by end-to-end self-training. In *Interspeech 2020*, pages 2787–2791.

Chopra, S., Hadsell, R., and LeCun, Y. (2005). Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546 vol. 1.

Chowdhury, S. A. and Ali, A. (2023). Multilingual word error rate estimation: e-WER3. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5.

Chuang, S.-P., Chang, H.-J., Huang, S.-F., and Lee, H.-y. (2021). Non-autoregressive Mandarin-English code-switching speech recognition. In *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 465–472.

Chung, S.-W., Chung, J. S., and Kang, H.-G. (2020). Perfect match: Self-supervised embeddings for cross-modal retrieval. *IEEE J. Sel. Topics Signal Process.*, 14(3):568–576.

Chung, Y.-A., Zhang, Y., Han, W., Chiu, C.-C., Qin, J., Pang, R., and Wu, Y. (2021a). w2v-BERT: Combining contrastive learning and masked language modeling for self-supervised speech pre-training. In *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 244–250.

Chung, Y.-A., Zhu, C., and Zeng, M. (2021b). SPLAT: Speech-language joint pre-training for spoken language understanding. In *Proc. 2021 Conf. North Amer. Chapter Assoc. Comput. Linguist.: Human Lang. Technol.*, pages 1897–1907, Online. Association for Computational Linguistics. [Online]. doi: 10.18653/v1/2021.naacl-main.152.

Cieri, C., Miller, D., and Walker, K. (2004). The Fisher corpus: A resource for the next generations of speech-to-text. In Lino, M. T., Xavier, M. F., Ferreira, F., Costa, R., and Silva, R., editors, *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal. European Language Resources Association (ELRA).

Clark, K., Luong, M.-T., Le, Q. V., and Manning, C. D. (2020). ELECTRA: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*.

Clevert, D., Unterthiner, T., and Hochreiter, S. (2016). Fast and accurate deep network learning by exponential linear units (ELUs). presented at Int. Conf. Learn. Repr. (ICLR).

Conneau, A., Baevski, A., Collobert, R., Mohamed, A., and Auli, M. (2021). Unsupervised cross-lingual representation learning for speech recognition. In *Interspeech 2021*, pages 2426–2430.

Conneau, A. et al. (2020). Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Conneau, A., Ma, M., Khanuja, S., Zhang, Y., Axelrod, V., Dalmia, S., Riesa, J., Rivera, C., and Bapna, A. (2023). FLEURS: Few-shot Learning Evaluation of Universal Representations of Speech. In *2022 IEEE Spoken Language Technology Workshop (SLT)*, pages 798–805.

Damera-Venkata, N., Kite, T., Geisler, W., Evans, B., and Bovik, A. (2000). Image quality assessment based on a degradation model. *IEEE Transactions on Image Processing*, 9(4):636–650.

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.

Deena, S., Hasan, M., Doulaty, M., Saz, O., and Hain, T. (2019). Recurrent neural network language model adaptation for multi-genre broadcast speech recognition and alignment. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 27(3):572–582.

Dehak, N., Kenny, P. J., Dehak, R., Dumouchel, P., and Ouellet, P. (2011). Front-end factor analysis for speaker verification. *IEEE Trans. Audio, Speech, Language Process.*, 19(4):788–798. [Online]. doi: 10.1109/TASL.2010.2064307.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186, Minneapolis, Minnesota. ACL.

Doulaty, M., Saz, O., and Hain, T. (2015). Data-selective transfer learning for multi-domain speech recognition. In *Interspeech 2015*, pages 2897–2901.

Elizalde, B., Zarar, S., and Raj, B. (2019). Cross modal audio search and retrieval with joint embeddings based on text and audio. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4095–4099.

Fang, A., Filice, S., Limsopatham, N., and Rokhlenko, O. (2020). Using Phoneme Representations to Build Predictive Models Robust to ASR Errors. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, page 699–708, New York, NY, USA. Association for Computing Machinery.

Fralick, S. (1967). Learning to recognize patterns without a teacher. *IEEE Transactions on Information Theory*, 13(1):57–64.

Fukushima, K. (1975). Cognitron: A self-organizing multilayered neural network. *Biological cybernetics*, 20(3):121–136.

Georgescu, A.-L., Manolache, C., Oneaţă, D., Cucu, H., and Burileanu, C. (2021). Data-filtering methods for self-training of automatic speech recognition systems. In *2021 IEEE Spoken Language Technology Workshop (SLT)*, pages 1–7.

Glorot, X., Bordes, A., and Bengio, Y. (2011). Deep sparse rectifier neural networks. In Gordon, G., Dunson, D., and Dudík, M., editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 315–323, Fort Lauderdale, FL, USA. PMLR.

Godfrey, J., Holliman, E., and McDaniel, J. (1992). SWITCHBOARD: Telephone speech corpus for research and development. In *[Proceedings] ICASSP-92: 1992 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 517–520 vol.1.

Gody, R. and Harwath, D. (2023). Unsupervised fine-tuning data selection for ASR using self-supervised speech models. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5.

Gorisch, J. and Schmidt, T. (2024). Evaluating workflows for creating orthographic transcripts for oral corpora by transcribing from scratch or correcting ASR-output. In Calzolari, N., Kan, M.-Y., Hoste, V., Lenci, A., Sakti, S., and Xue, N., editors, *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 6564–6574, Torino, Italia. ELRA and ICCL.

Gouvêa, E. and Davel, M. H. (2011). Kullback-Leibler divergence-based ASR training data selection. In *Interspeech 2011*, pages 2297–2300.

Graves, A. (2012). Sequence transduction with recurrent neural networks. *arXiv preprint arXiv:1211.3711*. presented at *Int. Conf. of Machine Learning (ICML) Workshop*.

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*, ICML '06, page 369–376, New York, NY, USA. Association for Computing Machinery.

Gulati, A., Qin, J., Chiu, C.-C., Parmar, N., Zhang, Y., Yu, J., Han, W., Wang, S., Zhang, Z., Wu, Y., and Pang, R. (2020). Conformer: Convolution-augmented transformer for speech recognition. In *Interspeech 2020*, pages 5036–5040.

Gutmann, M. U. and Hyvärinen, A. (2012). Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of Machine Learning Research*, 13(11):307–361.

He, P., Gao, J., and Chen, W. (2021a). DeBERTaV3: Improving deberta using ELECTRA-style pre-training with gradient-disentangled embedding sharing.

He, P., Liu, X., Gao, J., and Chen, W. (2021b). DeBERTa: Decoding-enhanced BERT with disentangled attention. In *International Conference on Learning Representations*.

He, Y., Sainath, T. N., Prabhavalkar, R., McGraw, I., Alvarez, R., Zhao, D., Rybach, D., Kannan, A., Wu, Y., Pang, R., Liang, Q., Bhatia, D., Shangguan, Y., Li, B., Pundak, G., Sim, K. C., Bagby, T., Chang, S.-y., Rao, K., and Gruenstein, A. (2019). Streaming end-to-end speech recognition for mobile devices. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6381–6385.

Hernandez, F., Nguyen, V., Ghannay, S., Tomashenko, N., and Estève, Y. (2018). TED-LIUM 3: Twice as much data and corpus repartition for experiments on speaker adaptation. In Karpov, A., Jokisch, O., and Potapova, R., editors, *Speech and Computer*, pages 198–208, Cham. Springer International Publishing.

Higuchi, Y., Moritz, N., Roux, J. L., and Hori, T. (2021). Momentum pseudo-labeling for semi-supervised speech recognition. In *Interspeech 2021*, pages 726–730.

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A.-r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T. N., and Kingsbury, B. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Hoffer, E. and Ailon, N. (2015). Deep metric learning using triplet network. In Feragen, A., Pelillo, M., and Loog, M., editors, *Similarity-Based Pattern Recognition*, pages 84–92, Cham. Springer International Publishing.

Hori, T., Astudillo, R., Hayashi, T., Zhang, Y., Watanabe, S., and Le Roux, J. (2019). Cycle-consistency training for end-to-end speech recognition. In *ICASSP*, pages 6271–6275.

Householder, A. S. (1941). A theory of steady-state activity in nerve-fiber networks: I. definitions and preliminary lemmas. *The bulletin of mathematical biophysics*, 3:63–69.

Hsu, W.-N., Bolte, B., Tsai, Y.-H. H., Lakhotia, K., Salakhutdinov, R., and Mohamed, A. (2021). HuBERT: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, 29:3451–3460.

Huang, P.-S., He, X., Gao, J., Deng, L., Acero, A., and Heck, L. (2013a). Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management*, CIKM '13, page 2333–2338, New York, NY, USA. Association for Computing Machinery.

Huang, Y., Yu, D., Gong, Y., and Liu, C. (2013b). Semi-supervised GMM and DNN acoustic model training with multi-system combination and confidence re-calibration. In *Interspeech 2013*, pages 2360–2364.

Jalalvand, S., Negri, M., Falavigna, D., and Turchi, M. (2015). Driving ROVER with segment-based ASR quality estimation. In Zong, C. and Strube, M., editors, *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1095–1105, Beijing, China. Association for Computational Linguistics.

Jeanrenaud, P., Ng, K., Siu, M., Rohlicek, J., and Gish, H. (1993). Phonetic-based word spotter: various configurations and application to event spotting. In *3rd European Conference on Speech Communication and Technology (Eurospeech 1993)*, pages 1057–1060.

Jeon, W., Jordan, M., and Krishnamoorthy, M. (2020). On modeling ASR word confidence. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6324–6328.

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

Kahn, J., Lee, A., and Hannun, A. (2020a). Self-training for end-to-end speech recognition. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7084–7088.

Kahn, J., Rivière, M., Zheng, W., Kharitonov, E., Xu, Q., Mazaré, P., Karadayi, J., Liptchinsky, V., Collobert, R., Fuegen, C., Likhomanenko, T., Synnaeve, G., Joulin, A., Mohamed, A., and Dupoux, E. (2020b). Libri-Light: A benchmark for ASR with limited or no supervision. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7669–7673.

Kaya, M. and Bilge, H. Ş. (2019). Deep metric learning: A survey. *Symmetry*, 11(9). [Online]. doi: 10.3390/sym11091066.

Kemp, T. and Waibel, A. (1998). Unsupervised training of a speech recognizer using TV broadcasts. In *Proc. 5th International Conference on Spoken Language Processing (ICSLP 1998)*, page paper 0758.

Kemp, T. and Waibel, A. (1999). Unsupervised training of a speech recognizer: Recent experiments. In *Proc. 6th European Conference on Speech Communication and Technology (Eurospeech 1999)*, pages 2725–2728.

Kharitonov, E., Copet, J., Lakhotia, K., Nguyen, T. A., Tomasello, P., Lee, A., Elkahky, A., Hsu, W.-N., Mohamed, A., Dupoux, E., and Adi, Y. (2022). textless-lib: A library for textless spoken language processing. In Hajishirzi, H., Ning, Q., and Sil, A., editors, *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: System Demonstrations*, pages 1–9, Hybrid: Seattle, Washington + Online. Association for Computational Linguistics.

Khurana, S., Moritz, N., Hori, T., and Roux, J. L. (2021). Unsupervised domain adaptation for speech recognition via uncertainty driven self-training. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6553–6557.

Krause, A. and Golovin, D. (2014). Submodular function maximization. In Bordeaux, L., Hamadi, Y., and Kohli, P. E., editors, *Tractability: Practical approaches to hard problems*, page 71–104. Cambridge University Press. [Online]. doi: 10.1017/CBO9781139177801.004.

Krishnan Parthasarathi, S. H. and Strom, N. (2019). Lessons from building acoustic models with a million hours of speech. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6670–6674.

Kullback, S. (1997). *Information theory and statistics*. Courier Corporation.

Kumar, A., Singh, S., Gowda, D., Garg, A., Singh, S., and Kim, C. (2020). Utterance confidence measure for end-to-end speech recognition with applications to distributed speech recognition scenarios. In *Interspeech 2020*, pages 4357–4361.

Kumar, A. K., Waldekar, S., Sahidullah, M., and Saha, G. (2022). Robust acoustic domain identification with its application to speaker diarization. *International Journal of Speech Technology*, 25(4):933–945.

Lagos, N. and Calapodescu, I. (2024). Unsupervised multi-domain data selection for ASR fine-tuning. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 10711–10715. IEEE.

Lamel, L., Gauvain, J., and Adda, G. (2001). Investigating lightly supervised acoustic model training. In *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221)*, volume 1, pages 477–480 vol.1.

Lamel, L., Gauvain, J.-L., and Adda, G. (2002). Lightly supervised and unsupervised acoustic model training. *Computer Speech & Language*, 16(1):115–129.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *nature*, 521(7553):436–444.

Lee, A., Chen, P.-J., Wang, C., Gu, J., Popuri, S., Ma, X., Polyak, A., Adi, Y., He, Q., Tang, Y., Pino, J., and Hsu, W.-N. (2022). Direct speech-to-speech translation with discrete units. In Muresan, S., Nakov, P., and Villavicencio, A., editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3327–3339, Dublin, Ireland. Association for Computational Linguistics.

Lee, C., Rabiner, L., Pieraccini, R., and Wilpon, J. (1990). Acoustic modeling for large vocabulary speech recognition. *Computer Speech & Language*, 4(2):127–165.

Lee, L.-s., Glass, J., Lee, H.-y., and Chan, C.-a. (2015). Spoken content retrieval—beyond cascading speech recognition with text retrieval. *IEEE/ACM TASLP*, 23(9):1389–1420.

Lefevre, F., Gauvain, J.-L., and Lamel, L. (2001). Towards task-independent speech recognition. In *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221)*, volume 1, pages 521–524 vol.1.

Li, J. (2022). Recent advances in end-to-end automatic speech recognition. *APSIPA Transactions on Signal and Information Processing*, 11(1):–.

Li, T., Meng, Q., and Sun, Y. (2023). Improved noisy iterative pseudo-labeling for semi-supervised speech recognition. In *2022 IEEE Spoken Language Technology Workshop (SLT)*, pages 167–173.

Li, X., Dalmia, S., Li, J., Lee, M., Littell, P., Yao, J., Anastasopoulos, A., Mortensen, D. R., Neubig, G., Black, A. W., and Metze, F. (2020). Universal phone recognition with a multilingual allophone system. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8249–8253.

Lin, H. and Bilmes, J. (2009). How to select a good training-data subset for transcription: submodular active selection for sequences. In *Interspeech 2009*, pages 2859–2862.

Liong, V. E., Lu, J., Tan, Y.-P., and Zhou, J. (2017). Deep coupled metric learning for cross-modal matching. *IEEE Trans. Multimedia*, 19(6):1234–1244.

Long, Y., Li, Y., Wei, S., Zhang, Q., and Yang, C. (2019). Large-scale semi-supervised training in deep learning acoustic model for ASR. *IEEE Access*, 7:133615–133627.

Louis, A. and Nenkova, A. (2013). Automatically assessing machine summary content without a gold standard. *Computational Linguistics*, 39(2):267–300.

Lu, Z., Wang, Y., Zhang, Y., Han, W., Chen, Z., and Haghani, P. (2022). Unsupervised data selection via discrete speech representation for ASR. In *Interspeech 2022*, pages 3393–3397.

Ma, J. and Schwartz, R. (2008). Unsupervised versus supervised training of acoustic models. In *Interspeech 2008*, pages 2374–2377.

Mai, L. and Carson-Berndsen, J. (2022). Unsupervised domain adaptation for speech recognition with unsupervised error correction. In *Interspeech 2022*, pages 5120–5124.

Mei, X., Liu, X., Sun, J., Plumbley, M., and Wang, W. (2022). On metric learning for audio-text cross-modal retrieval. In *Interspeech 2022*, pages 4142–4146.

Meng, Z., Chen, Z., Mazalov, V., Li, J., and Gong, Y. (2017). Unsupervised adaptation with domain separation networks for robust speech recognition. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 214–221.

Mithun, N. C., Li, J., Metze, F., and Roy-Chowdhury, A. K. (2018). Learning joint embedding with multimodal cues for cross-modal video-text retrieval. In *Proc. 2018 ACM Int. Conf. Multimedia Retrieval*, ICMR '18, pages 19–27, New York, NY, USA. Association for Computing Machinery. [Online]. doi: 10.1145/3206025.3206064.

Mohamed, A., Lee, H.-y., Borgholt, L., Havtorn, J. D., Edin, J., Igel, C., Kirchhoff, K., Li, S.-W., Livescu, K., Maaløe, L., Sainath, T. N., and Watanabe, S. (2022). Self-supervised speech representation learning: A review. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1179–1210.

Moritz, N., Hori, T., and Roux, J. L. (2021). Semi-supervised speech recognition via graph-based temporal classification. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6548–6552.

Negri, M., Turchi, M., C. de Souza, J. G., and Falavigna, D. (2014). Quality estimation for automatic speech recognition. In Tsujii, J. and Hajic, J., editors, *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1813–1823, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Ng, R. W. M., Doulaty, M., Doddipatla, R., Aziz, W., Shah, K., Saz, O., Hasan, M., AlHaribi, G., Specia, L., and Hain, T. (2014). The USFD SLT system for IWSLT 2014. In Federico, M., Stüker, S., and Yvon, F., editors, *Proceedings of the 11th International Workshop on Spoken Language Translation: Evaluation Campaign*, pages 86–91, Lake Tahoe, California.

Novotney, S., Schwartz, R., and Ma, J. (2009). Unsupervised acoustic and language model training with small amounts of labelled data. In *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 4297–4300.

Ogawa, A., Hori, T., and Nakamura, A. (2016). Estimating speech recognition accuracy based on error type classification. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(12):2400–2413.

Oord, A. v. d., Li, Y., and Vinyals, O. (2018). Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748.*

Ott, M., Edunov, S., Baevski, A., Fan, A., Gross, S., Ng, N., Grangier, D., and Auli, M. (2019). fairseq: A fast, extensible toolkit for sequence modeling. In Ammar, W., Louis, A., and Mostafazadeh, N., editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Pallett, D. S. (1985). Performance assessment of automatic speech recognizers. *Journal of Research of the National Bureau of Standards*, 90(5):371.

Pan, S. J. and Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359.

Panayotov, V., Chen, G., Povey, D., and Khudanpur, S. (2015). Librispeech: An ASR corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). BLEU: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, page 311–318, USA. Association for Computational Linguistics.

Park, C., Chen, M., and Hain, T. (2024a). Automatic speech recognition system-independent word error rate estimation. In Calzolari, N., Kan, M.-Y., Hoste, V., Lenci, A., Sakti, S., and Xue, N., editors, *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 1979–1987, Torino, Italia. ELRA and ICCL.

Park, C., Kang, H., and Hain, T. (2024b). Character error rate estimation for automatic speech recognition of short utterances. In *Proceedings of 32nd European Signal Processing Conference (EUSIPCO 2024)*, pages 131–135, Lyon, France.

Park, D. S., Zhang, Y., Jia, Y., Han, W., Chiu, C.-C., Li, B., Wu, Y., and Le, Q. V. (2020). Improved noisy student training for automatic speech recognition. In *Interspeech 2020*, pages 2817–2821.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., ..., and Chintala, S. (2019). PyTorch: An imperative style, high-performance deep learning library. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R., editors, *Adv. Neural Inf. Process. Syst.*, volume 32. Curran Associates.

Paul, D. B. and Baker, J. M. (1992). The design for the Wall Street Journal-based CSR corpus. In *Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York, February 23-26, 1992*.

Peddinti, V., Povey, D., and Khudanpur, S. (2015). A time delay neural network architecture for efficient modeling of long temporal contexts. In *Interspeech 2015*, pages 3214–3218, Dresden, Germany. [Online]. doi: 10.21437/Interspeech.2015-647.

Povey, D., Peddinti, V., Galvez, D., Ghahremani, P., Manohar, V., Na, X., Wang, Y., and Khudanpur, S. (2016). Purely sequence-trained neural networks for ASR based on Lattice-Free MMI. In *Interspeech 2016*, pages 2751–2755.

Povey, D., Zhang, X., and Khudanpur, S. (2015). Parallel training of DNNs with natural gradient and parameter averaging. presented at Int. Conf. Learn. Repr. (ICLR).

Qiu, D., He, Y., Li, Q., Zhang, Y., Cao, L., and McGraw, I. (2021). Multi-task learning for end-to-end ASR word and utterance confidence with deletion prediction. In *Interspeech 2021*, pages 4074–4078.

Radford, A. (2018). Improving language understanding by generative pre-training. *OpenAI blog*.

Radford, A., Kim, J. W., Xu, T., Brockman, G., McLeavey, C., and Sutskever, I. (2023). Robust speech recognition via large-scale weak supervision. In *International conference on machine learning*, pages 28492–28518. PMLR.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language models are unsupervised multitask learners. *OpenAI blog*.

Ragni, A., Li, Q., Gales, M. J. F., and Wang, Y. (2018). Confidence estimation and deletion prediction using bidirectional recurrent neural networks. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 204–211.

Raj, D., Snyder, D., Povey, D., and Khudanpur, S. (2019). Probing the information encoded in x-vectors. In *2019 IEEE Autom. Speech Recognit. Understanding Workshop (ASRU)*, pages 726–733, Sentosa, Singapore. [Online]. doi: 10.1109/ASRU46091.2019.9003979.

Ravanelli, M., Parcollet, T., Plantinga, P., Rouhe, A., Cornell, S., Lugosch, L., Subakan, C., Dawalatabad, N., Heba, A., Zhong, J., Chou, J.-C., Yeh, S.-L., Fu, S.-W., Liao, C.-F., Rastorgueva, E., Grondin, F., Aris, W., Na, H., Gao, Y., ..., and Bengio, Y. (2021). SpeechBrain: A general-purpose speech toolkit. *arXiv preprint arXiv:2106.04624*.

Reimers, N. and Gurevych, I. (2019). Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Reyes, O. and Ventura, S. (2019). Performing multi-target regression via a parameter sharing-based deep network. *International journal of neural systems*, 29(09):1950014.

Robinson, T., Fransen, J., Pye, D., Foote, J., and Renals, S. (1995). WSJCAMO: A British English speech corpus for large vocabulary continuous speech recognition. In *1995 International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 81–84 vol.1.

Rose, R., Juang, B., and Lee, C. (1995). A training procedure for verifying string hypotheses in continuous speech recognition. In *1995 International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 281–284 vol.1.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *nature*, 323(6088):533–536.

Sato, H., Ochiai, T., Delcroix, M., Kinoshita, K., Kamo, N., and Moriya, T. (2022). Learning to enhance or not: Neural network-based switching of enhanced and observed signals for overlapping speech recognition. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6287–6291.

Schneider, S., Baevski, A., Collobert, R., and Auli, M. (2019). wav2vec: Unsupervised pre-training for speech recognition. In *Interspeech 2019*, pages 3465–3469.

Schroff, F., Kalenichenko, D., and Philbin, J. (2015). FaceNet: A unified embedding for face recognition and clustering. In *CVPR*.

Scudder, H. (1965). Adaptive communication receivers. *IEEE Transactions on Information Theory*, 11(2):167–174.

Seigel, M. and Woodland, P. (2014). Detecting deletions in ASR output. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2302–2306.

Seshadrinathan, K., Soundararajan, R., Bovik, A. C., and Cormack, L. K. (2010). Study of subjective and objective quality assessment of video. *IEEE Transactions on Image Processing*, 19(6):1427–1441.

Singh, S., Hou, F., and Wang, R. (2023). A novel self-training approach for low-resource speech recognition. In *INTERSPEECH 2023*, pages 1588–1592.

Siu, M., Gish, H., and Richardson, F. (1997). Improved estimation, evaluation and applications of confidence measures for speech recognition. In *Proc. 5th European Conference on Speech Communication and Technology (Eurospeech 1997)*, pages 831–834.

Snyder, D., Garcia-Romero, D., Sell, G., Povey, D., and Khudanpur, S. (2018). X-vectors: Robust DNN embeddings for speaker recognition. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5329–5333.

Sohn, K. (2016). Improved deep metric learning with multi-class N-pair loss objective. In Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., and Garnett, R., editors, *Adv. in Neural Inf. Process. Syst.*, volume 29. Curran Associates.

Spyromitros-Xioufis, E., Tsoumakas, G., Groves, W., and Vlahavas, I. (2016). Multi-target regression via input space expansion: treating targets as inputs. *Machine Learning*, 104(1):55–98.

Su, K.-Y., Wu, M.-W., and Chang, J.-S. (1992). A new quantitative quality measure for machine translation systems. In *Proceedings of the 14th Conference on Computational Linguistics - Volume 2*, COLING '92, page 433–439, USA. Association for Computational Linguistics.

Swietojanski, P., Ghoshal, A., and Renals, S. (2013). Revisiting hybrid and GMM-HMM system combination techniques. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6744–6748.

Thomas, S., Seltzer, M. L., Church, K., and Hermansky, H. (2013). Deep neural network features and semi-supervised training for low resource speech recognition. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6704–6708.

Ulasik, M. A., Hürlimann, M., Germann, F., Gedik, E., Benites, F., and Cieliebak, M. (2020). CEASR: A corpus for evaluating automatic speech recognition. In Calzolari, N., Béchet, F., Blache, P., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 6477–6485, Marseille, France. European Language Resources Association.

van den Oord, A., Vinyals, O., and kavukcuoglu, k. (2017). Neural discrete representation learning. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Veselý, K., Hannemann, M., and Burget, L. (2013). Semi-supervised training of deep neural networks. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 267–272.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., ..., and Contributors, S. . (2020). SciPy 1.0: Fundamental algorithms for scientific computing in Python. *Nature Methods*, 17:261–272. [Online]. doi: 10.1038/s41592-019-0686-2.

Voran, S. (1999). Objective estimation of perceived speech quality. I. Development of the measuring normalizing block technique. *IEEE Transactions on Speech and Audio Processing*, 7(4):371–382.

Wang, A., Pruksachatkun, Y., Nangia, N., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. (2019). SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. (2018). GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Watanabe, S., Hori, T., Kim, S., Hershey, J. R., and Hayashi, T. (2017). Hybrid CTC/Attention architecture for end-to-end speech recognition. *IEEE Journal of Selected Topics in Signal Processing*, 11(8):1240–1253.

Wei, K., Liu, Y., Kirchhoff, K., and Bilmes, J. (2014). Unsupervised submodular subset selection for speech data. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4107–4111.

Weintraub, M. (1995). LVCSR log-likelihood ratio scoring for keyword spotting. In *1995 International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 297–300 vol.1.

wen Yang, S., Chi, P.-H., Chuang, Y.-S., Lai, C.-I. J., Lakhotia, K., Lin, Y. Y., Liu, A. T., Shi, J., Chang, X., Lin, G.-T., Huang, T.-H., Tseng, W.-C., tik Lee, K., Liu, D.-R., Huang, Z., Dong, S., Li, S.-W., Watanabe, S., Mohamed, A., and yi Lee, H. (2021). SUPERB: Speech processing universal performance benchmark. In *Interspeech 2021*, pages 1194–1198.

Wenzek, G., Lachaux, M.-A., Conneau, A., Chaudhary, V., Guzmán, F., Joulin, A., and Grave, E. (2020). CCNet: Extracting high quality monolingual datasets from web crawl data. In Calzolari, N., Béchet, F., Blache, P., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4003–4012, Marseille, France. European Language Resources Association.

Wu, P., Shi, J., Zhong, Y., Watanabe, S., and Black, A. W. (2021). Cross-lingual transfer for speech processing using acoustic language similarity. In *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 1050–1057.

Xu, Q., Baevski, A., Likhomanenko, T., Tomasello, P., Conneau, A., Collobert, R., Synnaeve, G., and Auli, M. (2021). Self-training and pre-training are complementary for speech recognition. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3030–3034.

Xu, Q., Likhomanenko, T., Kahn, J., Hannun, A., Synnaeve, G., and Collobert, R. (2020). Iterative pseudo-labeling for speech recognition. In *Interspeech 2020*, pages 1006–1010.

Xu, X., He, L., Lu, H., Gao, L., and Ji, Y. (2019). Deep adversarial metric learning for cross-modal retrieval. *World Wide Web*, 22(2):657–672.

Yang, G., Zhong, Z., Ding, M., Sebe, N., and Ricci, E. (2023). Self-training transformer for source-free domain adaptation. *Applied Intelligence*, 53(13):16560–16574.

Yang, J., Yi, X., Zhiyuan Cheng, D., Hong, L., Li, Y., Xiaoming Wang, S., Xu, T., and Chi, E. H. (2020). Mixed negative sampling for learning two-tower neural networks in recommendations. In *Companion Proceedings of the Web Conference 2020*, WWW '20, page 441–447, New York, NY, USA. Association for Computing Machinery.

Yarowsky, D. (1995). Unsupervised word sense disambiguation rivaling supervised methods. In *33rd Annual Meeting of the Association for Computational Linguistics*, pages 189–196, Cambridge, Massachusetts, USA. Association for Computational Linguistics.

Young, S. (1994). Detecting misrecognitions and out-of-vocabulary words. In *Proceedings of ICASSP '94. IEEE International Conference on Acoustics, Speech and Signal Processing*, volume ii, pages II/21–II/24 vol.2.

Zavaliagkos, G., McDonough, J., Miller, D., El-Jaroudi, A., Billa, J., Richardson, F., Ma, K., Siu, M., and Gish, H. (1998a). The BBN Byblos 1997 large vocabulary conversational speech recognition system. In *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP '98 (Cat. No.98CH36181)*, volume 2, pages 905–908 vol.2.

Zavaliagkos, G., Siu, M.-H., Colthurst, T., and Billa, J. (1998b). Using untranscribed training data to improve performance. In *Proc. 5th International Conference on Spoken Language Processing (ICSLP 1998)*, page paper 1007.

Zhang, P., Liu, Y., and Hain, T. (2014). Semi-supervised DNN training in meeting recognition. In *2014 IEEE Spoken Language Technology Workshop (SLT)*, pages 141–146.

Zhang, S., Do, C.-T., Doddipatla, R., and Renals, S. (2020a). Learning noise invariant features through transfer learning for robust end-to-end speech recognition. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7024–7028.

Zhang, Y., Bakhturina, E., Gorman, K., and Ginsburg, B. (2021). NeMo inverse text normalization: From development to production. In *Interspeech 2021*, pages 4468–4472.

Zhang, Y., Park, D. S., Han, W., Qin, J., Gulati, A., Shor, J., Jansen, A., Xu, Y., Huang, Y., Wang, S., Zhou, Z., Li, B., Ma, M., Chan, W., Yu, J., Wang, Y., Cao, L., Sim, K. C., Ramabhadran, B., ..., and Wu, Y. (2022). BigSSL: Exploring the frontier of large-scale semi-supervised learning for automatic speech recognition. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1519–1532.

Zhang, Y., Qin, J., Park, D. S., Han, W., Chiu, C.-C., Pang, R., Le, Q. V., and Wu, Y. (2020b). Pushing the limits of semi-supervised learning for automatic speech recognition. *arXiv preprint arXiv:2010.10504*.

Zhou, X., Yılmaz, E., Long, Y., Li, Y., and Li, H. (2020). Multi-encoder-decoder transformer for code-switching speech recognition. In *Interspeech 2020*, pages 1042–1046.

Zhu, H., Gao, D., Cheng, G., Povey, D., Zhang, P., and Yan, Y. (2023). Alternative pseudo-labeling for semi-supervised automatic speech recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:3320–3330.

Žmolíková, K., Karafiát, M., Veselý, K., Delcroix, M., Watanabe, S., Burget, L., and Černocký, J. (2016). Data selection by sequence summarizing neural network in mismatch condition training. In *Interspeech 2016*, pages 2354–2358.