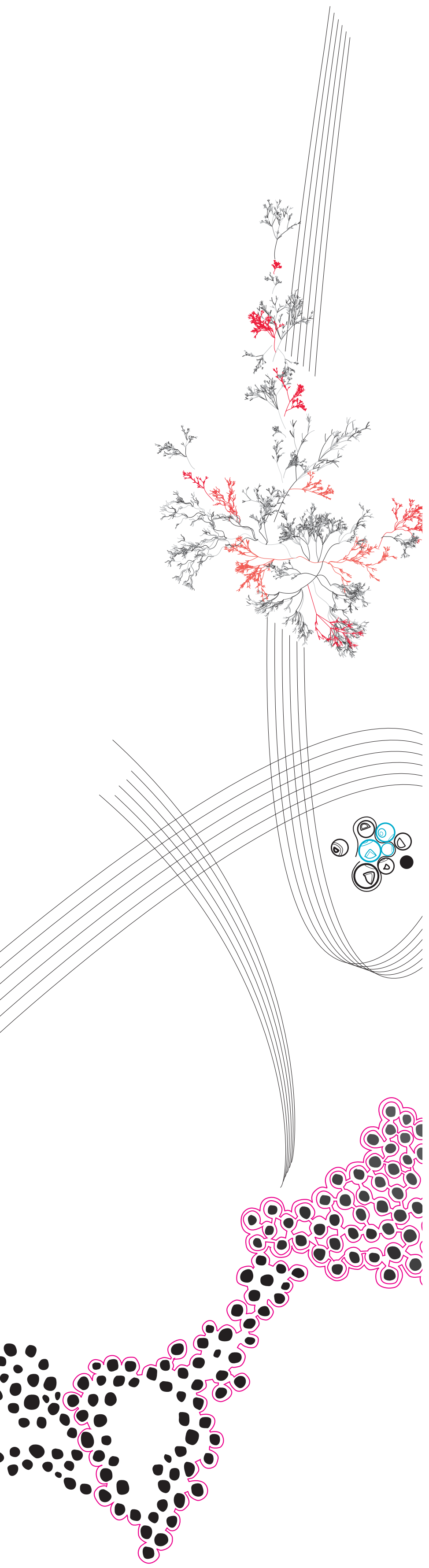
MSc Interaction Technology
Final Project



Fine-Tuning Pre-trained End-To-End Automatic Speech Recognition Models to Incorporate the Transcription of Laughter

Suzanne Spink

Supervisor: Dr. Khiet Truong, Estefanía Talavera Martínez, Dragos Balan

October, 2024

Department of Interaction Technology
Faculty of Electrical Engineering,
Mathematics and Computer Science,
University of Twente

UNIVERSITY OF TWENTE.

Contents

Abstract

This thesis aims to enhance Automatic Speech Recognition (ASR) by incorporating laughter detection, thereby broadening its applicability to more realistic and authentic real-world scenarios. There are various ASR models, but the pre-trained End-To-End models are particularly promising. These types of models can be fine-tuned on relatively little data. Two models were selected for fine-tuning and comparison: Whisper, a popular and high-performance model, and HuBERT, which emphasises phoneme sounds. Using two datasets that include spontaneous speech and laughter annotations - the AMI corpus and Switchboard - these models were pre-processed, normalised, fine-tuned and evaluated using the Word Error Rate (WER) for the ASR performance and F1-score, recall and precision for the laughter detection performance. The results indicated that the Whisper model performed best on the Switchboard dataset, achieving the highest F1-score (i.e. 0.901) and corresponding lowest WER (i.e. 0.161). On the AMI dataset, the results were more ambiguous. Neither model performed well enough for application on noisy datasets like AMI (i.e. both had an F1-score lower than 0.6). Still, HuBERT achieved the highest F1-score for laughter detection at 0.531. Whisper demonstrated a lower WER_L (i.e. word error rate including "laughter" annotations as a word) at 0.304, WER of 0.311 and a significantly higher precision of 0.949 (i.e. versus 0.785 precision for HuBERT), which is often critical for practical applications. Therefore, overall, Whisper is identified as the best-performing model for ASR in terms of laughter integration, particularly in applications focused on identifying laughter events without misinformation.

Keywords: interaction technology, laughter recognition, speech processing, automatic speech recognition (ASR), E2E models

Chapter 1

Introduction

Automatic Speech Recognition (ASR) methods have evolved in the last years and are becoming more robust and stable. They have progressed in many ways, where ASR was first only trained on read-aloud English speech, but the focus is currently on becoming robust in noisy conversational speech settings, e.g. meetings, and the ability to perform well with less data, e.g. when applying ASR to low-resource languages [1]. The type of model used has also progressed quickly in the last years, where statistical and hybrid models were the main types used [2], [3], but now End-To-End (E2E) models are considered the future of speech recognition [4]. These developments contribute to the use of ASR systems which have a high accuracy and can be deployed widely.

With this progression, there is more room for looking beyond the standard ASR transcriptions. With the increased focus on solving the challenges that come with transcribing conversational speech, the importance of transcribing paralinguistic elements is also emphasised [4], [5]. These elements include non-verbalisations like posture, gestures and eye contact, together with non-lexical elements (i.e. non-word sounds) like tone (e.g. sarcastic or serious), volume (e.g. increased when angry) and vocalisations (e.g. smiling). Often a combination of many factors is present to convey the context of words [5]. Currently, paralinguistic elements are often omitted in processing, where only lexical elements are included. As is the case when only the spoken text is transcribed: the overall atmosphere can be hard to gauge; the meaning of the text can be misinterpreted (e.g. sarcastic comments or jokes can be missed); and transcriptions are more prone to errors at the points where non-lexical elements are missing (e.g. when words are said while laughing).

The topic of transcribing paralinguistic elements in language is researched minimally. For the little research that has been done, the focus of research in ASR with paralanguage is primarily conducted on just audio signals (i.e. speech), so non-lexical elements, while exploring several ways of integration into various ASR models [1]. There are five types of non-lexical elements, so non-words, that can be distinguished. These are disfluencies, filler words, back channels, vegetative sounds, and affect bursts [6], [7]. Generally, plenty of research has been carried out into the first three, however, in the last two categories, i.e. vegetative sounds and affect bursts, even less research has been done. When considering the availability of data in different appropriate ASR datasets together with the lack of research in the area, an opportunity to conduct new research into the integration of laughter transcription in ASR systems can be identified. The ability to recognise laughter is an important asset to the ASR system, where it can potentially improve ASR accuracy by differentiating between speech and laughter (i.e. as the detection of disfluency has done [8], [9]); aid conversational agents give appropriate responses; and overall improve the conveying of meanings like nervousness or amusement.

In addition, in the up-and-coming E2E model laughter has only been considered in a separate model from the ASR pipeline. However, the unique benefit of E2E models is the opportunity to incorporate this directly in the pipeline by fine-tuning these pre-trained models to combine spoken words with laughter annotations.

As a result, this thesis will focus on integrating the transcription of laughter in the ASR pipeline by fine-tuning pre-trained E2E models. Currently, the two best models for this goal, are Whisper [10], [11] and HuBERT [12], [13]. Out of the two, Whisper has been pre-trained on the most data and is currently the most popular. However, HuBERT has been developed specifically to look into non-lexical elements and has already shown potential. Therefore, these two models are considered in this research.

This raises one main research question, namely:

How effectively can laughter transcription be integrated into established high-performing ASR systems through the fine-tuning of pre-trained End-To-End models?

To find the answer to this question, the research question has been broken down into three sub-questions. These are:

1. To what extent, if any, do the E2E ASR models Whisper and HuBERT currently transcribe laughter in conversational language?
2. To what extent, if any, can the E2E ASR models Whisper and HuBERT be fine-tuned on conversational speech including laughter annotations to improve the accuracy of laughter transcription?
3. How does fine-tuning the E2E ASR models Whisper and HuBERT on conversational speech with laughter annotations affect their overall performance in transcribing lexical elements?

All of the models are trained, evaluated and compared on the datasets *Switchboard* and *AMI*. Overall, the performance of this thesis will be evaluated on Word Error Rate, Laughter Detection F1-score, recall and precision, and qualitative analysis into laughter hits, deletion, substitution and insertion.

To answer the research questions, overall this thesis will consider three main parts. First, it is considered what happens at the positions in the audio where laughter is labelled in the test set, while the models did not learn the laughter label from the training set yet. With a qualitative analysis, there will be looked at if there are any deletions or substitutions for laughter and if so, if the substitutions are recognisable as laughter. It is expected that laughter is currently transcribed very minimally. However, it will take a different form which is not as a standard laughter label, but potentially recognisable as laughter (e.g. "haha"). It will also differ for each model. Exactly how this will differ is unclear, but Whisper is known to be more flexible, so it may perform better or recognise laughter more often as an event.

Secondly, both models will be fine-tuned and optimised to recognise laughter. The laughter event recognition is evaluated with F1-score, recall and precision. This is a fair and accurate method to show how well the laughter event has been integrated into the ASR pipeline. It is expected that this will significantly improve laughter recognition, as previous research has shown the potential of fine-tuning [14]–[16]. We cannot say anything about the difference in performance between Whisper and HuBERT. Both have their own benefits, as Whisper is larger and more flexible, while HuBERT was specifically developed for tasks that identify non-lexical elements, e.g. interruptions and emotions [13].

Lastly, if, how, and to what extent the Word Error Rate (WER) performance is affected by the fine-tuning for each of the models will be addressed. To evaluate this, the WER is considered for the model (1) without fine-tuning and (2) with fine-tuning but without laughter in the reference text. It is expected that this performance will be affected minimally, as the models are pre-trained on substantially more data than will be used in this research. Factors like catastrophic forgetting could play a role. Furthermore, knowing when laughter occurs, could even improve the performance of the ASR model, as it can aid in solving the difficulty of identifying the words that are spoken while laughing, as it can potentially recognise the words spoken while laughing more easily. However, a previous study has shown that this integration did not deteriorate the performance of ASR [6]. Lastly, HuBERT will be impacted more than Whisper due to its lack of flexibility.

Overall, the main research question can then be answered. It is expected that laughter is currently barely transcribed in the ASR system and that with the proven use of fine-tuning and the expected little impact it has on the ASR performance, laughter can be transcribed quite effectively.

This thesis is structured as follows. In Chapter 2, the literature on the topic of ASR is analysed, focusing on finding the gap of knowledge in the integration of laughter in E2E models. In Chapter 3 the Methodology is explained, showing the data pipeline, fine-tuning process and evaluation methods. In Chapter 4 the experiment Results are visualised and explained. In Chapter 5 the results are Discussed, focusing on three main parts. These points are: the behaviour of the baseline models (i.e. zero-shot and fine-tuned without laughter in the training data) on the test set including laughter, the laughter integration performance of the models when fully fine-tuned with laughter and the difference in lexical performance of the baseline and fully fine-tuned models. In Chapter 6, the research work is summarised and conclusions are drawn.

Chapter 2

Background

As mentioned in the introduction, research in Automatic Speech Recognition (ASR) has been prevailing in the last few years. ASR systems have shown they can be robust [17], [18], stable [19], and accurate [11], [20]. As a result of its success, ASR models have been applied to more complex tasks and opportunities have arisen for its applications in areas that could not be considered before, e.g. incorporating the emotions behind spoken text into transcriptions. The main reason for this rapid advancement was the development of the type of model used for these tasks, as the focus has shifted from Hybrid models to End-To-End (E2E) models. Some of these promising ASR E2E models are pre-trained and can easily be fine-tuned on smaller datasets. One of the challenges currently researched is making the ASR more accurate in real-life settings. This is done by using conversational speech for training datasets, as these are a far more accurate representation of real-life scenarios than e.g. cleaned read-aloud speech without interruptions or background noise. These will include many parameters and variables to take into account, e.g. overlapping and spontaneous speech. An important aspect of conversational speech, but also still under-researched in certain areas, is the detection and incorporation of non-lexical elements into transcriptions. This fine-tuning task has attracted attention recently, because these non-lexical vocalisations (e.g. laughter, disfluencies, and filler words), are very telling about the meaning of spoken words in context and/or the emotions behind them.

This Chapter will first give a general overview of the mechanics and development of ASR over the years, including an overview of commonly used databases in ASR, in Section 2.1. Next, it will discuss Hybrid models in Section 2.2, focusing on three types of Hybrid models and their applications. Then, E2E models will be discussed in Section 2.3, looking into the four most popular and best-performing ones and comparing their capabilities in an attempt to distinguish which is the best one to use for a certain task. Lastly, Section 2.4 will focus on one of the main challenges of training on conversational speech, namely the incorporation of non-lexical elements. It will look into their detection and the integration in both Hybrid and E2E methods, and their presence in three of the common ASR datasets.

2.1 Automatic Speech Recognition Overview

Before looking at the models that are used in ASR, it is important to understand what ASR is. This is most easy to understand by first considering the higher-level architecture. Then, the development over the years can be considered to understand why it progressed the way it

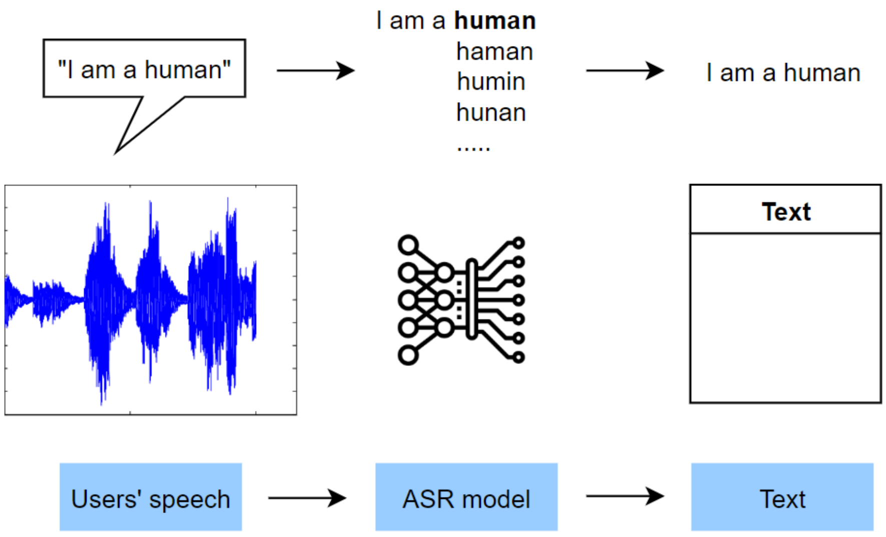


Figure 2.1: ASR visualised from a higher level.

did. Next, it is important to consider what limitations and parameters there are for the development of the ASR system. Lastly, the common ASR datasets are explained and described.

2.1.1 General ASR at the High-level

To understand how ASR works, the system can be considered from a higher level, as is visualised in Figure 2.1. First, a user will talk and this spoken text is recorded. This spoken text is an audio signal, which is then the input for the model, where the signal is interpreted and the trained model will make a prediction of what the most likely text outcome is. The outcome is a simple text string, called the hypothesis text.

To test how well the system works, the transcribed hypothesis text is evaluated during development. To evaluate the output performance, the hypothesis text is compared to the so-called "reference text". This reference text is used as the truth input text, so the manually annotated text of the audio. Often, both the hypothesis text and reference text are processed and normalised to match (e.g. "I'm" of the reference text becomes "I am", to match the hypothesis text output).

In most ASR research, the Word Error Rate (WER) is used to evaluate how accurate the hypothesis text is. This metric compares the hypothesis text to the reference text and considers per word if is correct. For example, if the reference text is *"This is an example sentence"*, and the hypothesis text is *"This is **the** example sentence"*, then the WER = errors / total words = 1/5 = 0.2%. Ideally, this would be 0, with no errors (i.e. WER = 0/5 = 0). There are more metrics that are often used, like Character Error Rate (CER) (i.e. the error rate calculated based on the comparison of reference and hypothesis text per individual character), Sentence Error Rate (SER) (i.e. error rate per sentence), Accuracy or F1-score, but this depends on the specific research topic and aim.

2.1.2 Historical Development

To develop any kind of ASR system, two main requirements need to be considered: the model and the dataset (i.e. speech) for the model to train on. Both of these have made significant improvements over the years, becoming more complex, extensive, and realistic. The way the model works has not only progressed but also the type of model has completely changed. This section will give a short overview of their development and influence on ASR.

Types of Speech

The type of research that can be done is highly dependent on what data is available that the model can train on and what complexity the ASR model can handle. In Figure 2.2, an overview of the development by NIST [1] is visualised. The first step of ASR was to train on read speech. Here it was often ensured that it was very clean data, as it had no background noise, interruptions, or other anomalies. The focus could be on simply transcribing speech. The Word Error Rate (WER), i.e. a common metric of performance in ASR which aims to be low, was quite low already around that time, but this data was extremely clean. This meant that in real-life scenarios, these models would not perform similarly, as spontaneous and continuous speech have many extra parameters that the model had not encountered before, e.g. background noise, stuttering, or different accents. This meant the model would get confused and could not be applied in real life yet. In addition, the focus was just on English text. The next step was taken around 1993, when conversational speech was also being researched. This was when the English dataset Switchboard [21] was created. The WER was initially extremely high for this type of speech, as the models had to adapt to a broader range of complexities compared to previous tasks. These complexities included handling multiple speakers, distinguishing between overlapping voices, and dealing with additional challenges like background noise. Fortunately, in the next ten years, with models adapting to and learning from these complexities, the WER was lowered massively. In 1995, also non-English conversational speech datasets were established, but their WER stayed even higher. Also around 1995, broadcast speech was introduced and both English and non-English news was used for ASR. This is cleaner than conversational speech, as the spontaneous aspects of speech were taken out like disfluencies, which resulted in a higher performance and lower WER. Lastly, Meeting Speech was introduced around the 2000s. The performance of this is even lower, due to the many other variables in the data that make it harder for the model to distinguish the individual speech from its surroundings, e.g. spontaneous and overlapping speech. The focus in the last ten years was on these types of natural and spontaneous speech, e.g. Meeting Speech, and managing its challenges. In this time, the models have become more flexible to different scenarios. They learn to handle these complexities by adapting to more realistic conversational patterns, improving their ability to recognise overlapping dialogue and individual speakers, sudden topic shifts, and unstructured speech, as well as filtering out background noise and other contextual distractions.

Types of Architecture

The type of model used for ASR has taken a long journey too. In the past, ASR models usually combined two different kinds of models, namely acoustic and language models. Acoustic models use the raw waveform to predict which phoneme most likely corresponds to it, while the language models aid these predictions by giving probabilities based on language, e.g. grammar. How the two are combined can be seen in Figure 2.3

The first type of acoustic model was the classically combined Hidden Markov Model (HMM) with the Gaussian Mixture Model (GMM), which results in the HMM-GMM. The HMM is a probabilistic model that can model sequential data, i.e. in ASR that is the temporal sequence

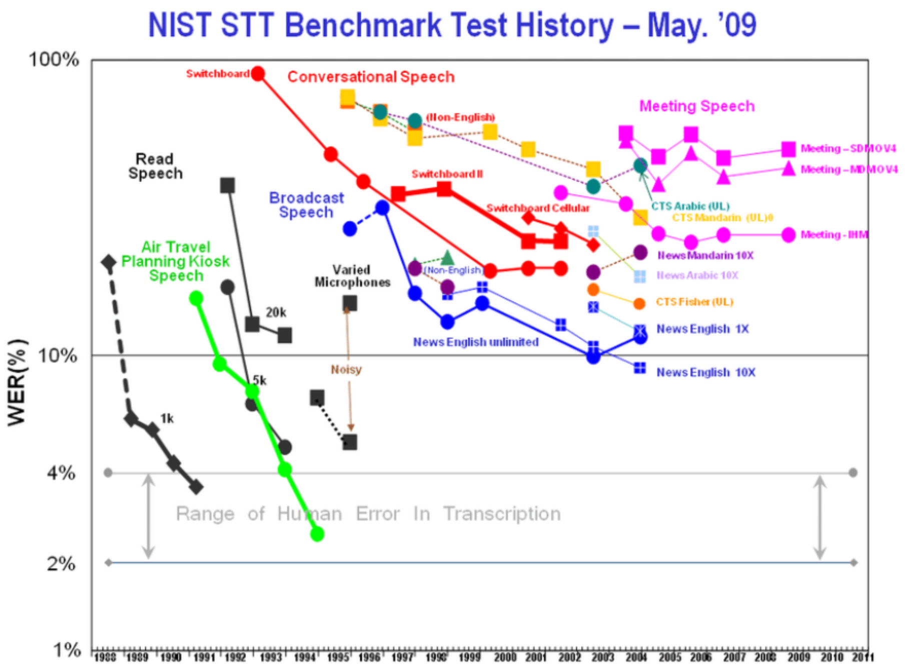


FIGURE 2.2: Overview of Types of Speech and Datasets over the years 1988 to 2009 [1].

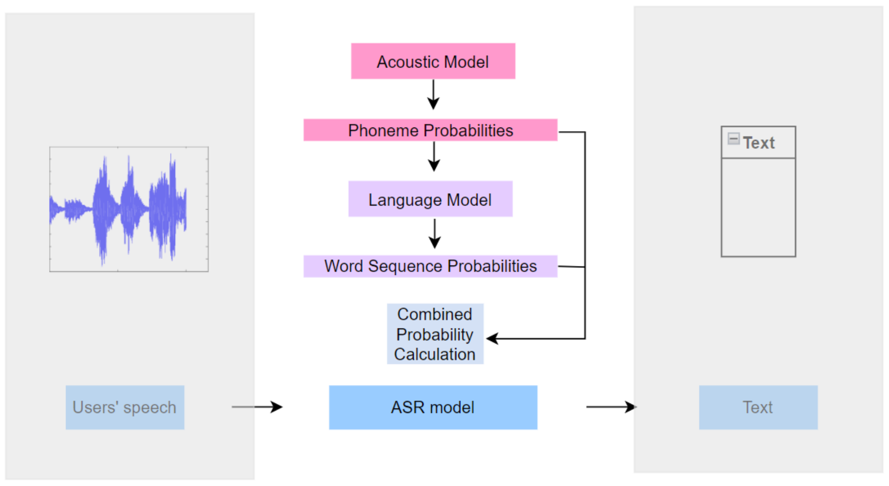


FIGURE 2.3: Acoustic Model with Language Model.

of speech signals, well and works with observations and states, whereas the GMM commonly represents the probability distribution of continuous data and models the acoustic features. The n-gram language model was most often combined with these acoustic models. This uses a sequence of n tokens (i.e. divided words) to estimate probabilities of what the n-gram would be. In the case of the sentence "This is a language model" and a 2-gram (i.e. bigrams), this would result in ["This is", "is a", "a language", "language model"]. Based on the frequency of the occurrence of the n-grams in the training data and other methods (e.g. smoothing or interpolation), a prediction is given for the correct sequence.

Around the time when the Deep Learning era arose, the focus shifted to different types of acoustic models. These new types of acoustic models and language models were together called Hybrid models. Here, the statistical GMM from the classic HMM-GMM approach was replaced with other acoustic models, resulting in, for example, an HMM-Deep Neural Network or HMM-Support Vector Machine. [2]. To this day, Hybrid models are most often used in existing applications [2], [3]. However, lately, there has been an increase in interest in E2E models. These models can directly convert raw speech into words and can efficiently train on lots of labelled data. As a result, there is no explicit distinction between language models and acoustic models anymore. In addition, E2E models are very flexible, as they can easily be adapted to different applications or tasks by fine-tuning certain types of pre-trained models. As a result, E2E models are currently considered to be the future of ASR for most tasks.

2.1.3 Limitations and Variables in Developing ASR

There are a few variables and limitations that make ASR challenging or suppress the quick progression of ASR performance. Being aware of these challenges can help understand the issues that arise in the development of ASR. However, they all come down to the same problem: insufficient data. There are a few parameters and variables you need to take into account to make a dataset accurate and robust in real-life scenarios.

The first is the role of noise. Depending on the scenario, e.g. if you are outside, in an echoing room, or if there is a lot of background noise, the model needs to be able to distinguish between these elements and the speaker. However, due to ease of recording and other concerns like privacy, most data is recorded inside, with the knowledge of participants and with good microphones. This gives biased audio clips that are not completely natural or spontaneous. In practice, this means that if the model is trained on just this data, the system will not be robust when applied to a different scenario. Especially when ASR systems were relatively new, data had to be as clean as possible to be able to optimise the models. Nowadays, datasets are created with the thought in mind to be as inclusive as possible, and might also contain unclean data, to optimise the advanced models further.

Secondly, read-aloud speech and spontaneous speech can be differentiated between. Read speech has sentences with little noise or disfluencies, hence there are few outliers or complex structures that the model did not know. However, spontaneous speech, e.g. meetings, can overlap or include stuttering, inconsistent grammar, and other non-lexical vocalisations. These are much harder to recognise for the model, for instance, due to its interdisciplinary nature of dealing with fields from linguistics, acoustics and human interaction [22]. To recognise these, the models must take into account more complex patterns and both linguistic and contextual elements beyond straightforward speech.

Lastly, many datasets focus on clean English speech, as this is one of the most used and highresourced languages in the world. However, there are many possible variations of accents, languages, and speech disabilities (i.e. low-resource contexts), with many environmental parameters (e.g. the convoluted nature of conversational speech) that could be transcribed. Due to the little data available on these and its variations, it makes it hard for ASR systems to accurately transcribe them [23].

As a result, an ASR system can seldom be robust in every scenario, due to the many variables that come into play, e.g. developmental costs, speaker recruitment and privacy concerns, when creating a new dataset. However, essential is also the fact that even if all of the data were available, these models still have complexity constraints and developers would have time and budget constraints. Training and optimising would take an immense amount of time.

2.1.4 Common ASR datasets

The performance of the models is extremely dependent on which dataset is used for training and testing. It contains the information that the models need to learn and evaluate. There are a few datasets that are most commonly used in general ASR tasks.

However, the requirements for the type of data that the dataset contains can differ, where the focus can vary from a specific language, accent, to other non-lexical vocalisations, e.g. disfluencies, while speaking. Currently, the most commonly used datasets in the field of ASR are LibriSpeech [24], CommonVoice [25], and Switchboard [21]. LibriSpeech has an English open-sourced version and Multilingual version. The English dataset contains two main subsets: 'clean' (i.e. cleaned speech) and 'other' (i.e. more challenging speech). It consists of a 1000 hours of read speech from the LibriVox (i.e. free Audiobooks) project. The Multilingual Librispeech (MLS) was also extracted from LibriVox's read-aloud Audiobooks, but consisted of 8 European languages. However, one negative aspect is that the training input audio is not from a natural environment. This makes both the model and its results biased and less robust in a real-world environment with other variables, e.g. background noise. Moreover, no additional labelling that can be useful in real-world applications is present, like laughter or disfluencies, as this is not present in read-aloud speech. Another open-sourced dataset is CommonVoice, which consists of 7335 hours of validated audio in 60 languages by volunteers. This results in more diverse, yet still simple read-aloud data. Also Switchboard is often used, as this contains natural speech, i.e. not reading out loud. The audio clips were from two-sided telephone conversations but collected from a controlled environment, where speakers were guided through various topics with a robot operator. Many models are evaluated and compared on standard ASR tasks with these datasets. Lastly, a slightly different dataset is Voxpopuli [26]. This contains 400K hours of unlabelled European parliament speech recordings in 23 languages from the years 2009 to 2020, with additional accented speech.

There is currently a new focus in the field of ASR, where robustness is vital to the application of ASR in real-world scenarios. As a result, there is a focus on conversational speech data. This includes the annotation of paralanguistic elements, like laughing, disfluencies, and backchannel words. In this case, the ICSI Meetings Dataset [27], AMI Corpus [28], or The SSPNet Vocalization Corpus [29] are used most often. In addition, the large Switchboard dataset, as mentioned above, also contains disfluencies and labels on laughter events. The ICSI Meetings Dataset contains 72 hours of speech from meetings and its transcriptions are extremely thorough. They include annotations of sounds like laughter, coughs, and other noises, e.g. door slams and notations of muttering. The AMI Corpus is quite similar, as they also contain meeting recordings of approximately 100 hours, with annotations like dialogue acts and head movements. The SSPNet Vocalization Corpus is less thorough but also includes some extra annotations. It consists of 2763 English audio clips from telephone conversations. Besides spoken text, they have annotated laughter and filler events. Another task that is currently being researched is recognising disfluencies [30]–[33], which can help anticipate and solve errors in ASR and increase inclusivity. These tasks either use the bigger datasets with natural speech as mentioned above, where disfluencies naturally occur, or a specific dataset that focuses on a special kind of disfluency. For example, in the case of dysarthric speech recognition, the UA Speech Dataset [34] or Domotica dataset [35] were used in different research.

Therefore, the dataset used depends on the needs of the ASR task. There are a few big ones that are used for standard ASR training tasks. Overall, more data is better, but these datasets do not always contain the information necessary. In those cases, smaller datasets will suffice if they contain (more of) the information necessary to train on. This is because the quality and relevance (i.e. when targeted and domain-specific) of the data are often more important than sheer quantity.

2.2 Statistical & Hybrid ASR Models

The first type of ASR model was the GMM-HMM acoustic model with the n-gram language model. When ASR was beginning to show potential and Deep Learning was introduced, the new types of models that were being developed were the Hybrid models. The main factor that identifies a Hybrid model, is the transition from the type of acoustic model used. The language model is still mostly the same, but more advanced techniques are used, e.g. neural network or transformer-based instead of the n-gram model. Due to thorough research into this area and its proven use, many established systems still use the Hybrid model [2], [3].

There are quite some types of models and they have been combined in many ways, each with their benefits and disadvantages. However, we can identify the three most common acoustic model types in ASR and these will be discussed below. They are the Hidden Markov Model (HMM), Artificial Neural Networks (ANN), and Support Vector Machines (SVM). [20]

2.2.1 Statistical GMM-HMM Models

The HMM-based model has been seen as one of the most successful and flexible of models for ASR over the last few decades. In the 1970's, the discrete density HMM was first introduced by Carnegie-Mellon and IBM [36], after which the continuous density HMM was soon followed by Bell Labs [37], [38]. With an increase in computational power, the performance of the model advanced quickly [39]. As a result, it is most commonly used, even now [3], [20].

At first, the HMM-based model was most commonly paired with the GMM model in ASR. This resulted in a good base, where the HMM models probabilities of transitions between phonemes, and the GMM predicts the likelihood of observing acoustic features for a specific phoneme. At this time, the focus of the research was still on enhancing the accuracy of speech recognition and optimising the model. The HMM-based model has been optimised in many ways throughout the last years. One research study [40] used the particle swarm optimization algorithm instead of the standard Viterbi algorithm. They showed equal performance of error rates and quicker optimisation results, with a recognition error rate of 0.73% for both algorithms with 8 as the number of reference words. Another research [41] improved the HMM-based model with the minimum mean square error principle, compared to a conventional spectral subtraction. The results showed an improvement in flexible noise robustness and real-time implementation possibilities in objective signal-to-noise ratio (SNR) and subjective evaluations. For example, for white noise, the HMM-based systems have a 2.5 dB and up SNR advantage over their Spectral Subtraction baseline system; matching subjective evaluation by personal preferences. Additionally, connectionist components can improve state-of-the-art HMM models, where the multilayer perceptron (MLP)-HMM had a word accuracy of 0.8 to 2.5% lower than the context-dependent HMM [42]. This best result had a recognition error of 5%.

The HMM-based model has progressed from a simple HMM-GMM model to a well-performing and optimised model for ASR. The inherent nature of the HMM-GMM also has its own limitations, with model complexity issues (i.e. needing extensive resources) [43], overfitting [44], struggling to deal with speech variations (e.g. speaking rate and accents) [45], and incorrect parameters and assumptions that might not fit every scenario (e.g. the use of the Gaussian distribution or the fixed-length context window) [43]. The optimised HMM-based model gave quite a steady performance in the end, but there were still many problematic factors in its application. Issues like noise robustness, dealing with spontaneous speech, and the lack of availability of diverse datasets also still contributed to holding back its development into a fully functioning system. [46], [47] This resulted in a need for Hybrid models.

2.2.2 Artificial Neural Network

More recently, there has been an increase in the popularity of Neural Network (NN). There are many types, one of which is the Convolutional Neural Network (CNN). There are several advantages to using a CNN, like the possibility of local filtering and pooling [48]. A pre-trained Deep Neural Network (DNN) and a Recurrent Neural Network (RNN) have also been compared, with a focus on various noisy environments. The RNN gave the best results, with a WER score of 0.70% on the very clean Aurora2 dataset. However, the RNN had much higher WER scores in noisy environments, showing optimisation is still necessary for real-world applications. [49] With this introduction of ANNs, the performance of ASR quickly increased. As a consequence, there was a new focus on the necessity for environmental robustness. This has been researched more thoroughly over the years, with one research [50] combining linear filtering, multiple feature types, and feature transformations. This showed a relative performance gain of 7.24% to 9.83% over other DNN research in various acoustic environments, with the lowest WER of 36.7%. Therefore, it seems that the additions to NNs are an important potential asset when making ASR robust in real-world environments.

While DNNs yield the optimal results for pattern recognition [20], they have shown to have some limitations, like overfitting [51] and finding long time-sequences difficult to handle [46]. As a result, they have been combined with the conventional HMM-based model as a hybrid NN-HMM model to overcome these limitations. For example, the ANN-HMM increases flexibility and minimises inaccuracies [46], [47], with one research [46] finding a relative WER reduction of 46.34% over other HMM-based models. Another research [51] compared the GMM-HMM with a DNN-HMM, which showed that the overfitting issue was solved with the use of DNN's hidden layers.

There are a lot of opportunities in this field, due to the various types of NNs and the many parameters that can be optimised. The NNs by themselves are a trending topic too and a lot of research is currently conducted on their optimisation. As a result, the Hybrid DNN-HMM still potentially has room for improvement too.

2.2.3 Support Vector Machines

Less commonly used in ASR, but still yielding good results, are SVMs. The advantages of the SVM are that they can optimise the representational and discriminate ability of the classifier simultaneously [44]. One study [44] tested their system on the Switchboard dataset and took the WER down from 41.6% for the state-of-the-art HMM-based model to 40.6% for their SVM. Two years later, with further research, they [52] found that the main problem was a segmentation issue, where a potential of 36.1% WER could be reached, compared to a 38.6% WER for the HMM-based model.

However, SVM's need a lot of computational power, making them impractical in real-world applications. Various researches have tried to solve this problem. One study [53] suggested using a weighted least squares training procedure. The results were promising in noisy and clean environments but did not match up to the state-of-the-art performance of an HMM-based system. Another more recent research [54] applied the particle swarm optimization algorithm to the SVM, as was done with the HMM. The main aim of this research was to minimise power consumption, yet a 99% accuracy in recognition success rate was also found.

To solve some of these issues, the Hybrid of SVM-HMM has also been researched. One study [55] suggests a system with two main stages, the first consisting of the HMM for speech segmentation, the second using the time instance derivation from the HMM to extract feature vectors. The SVM is then applied to classify the feature vectors. This was tested on the MOCHA dataset and consists out of clean data and white noise. On clean data, this resulted in a recognition accuracy of 99.35% for the SVM-based model, compared to the 99.34% on the conventional HMM-based baseline model. A slightly bigger difference could be seen when testing on white noise, where the SVM-based model reached a highest recognition accuracy of 50.73%, compared to 49.4% on the HMM-based model. They showed that improvements are very minimal, which might be due to inaccuracies in the first stage, i.e. the segmentation stage. Another research [56] examines limitations in SVM-HMM classifiers, namely the binary classification properties of the SVM and the necessity of each word being present in the training vocabulary data. The approaches used to solve these issues were to, (1) cascade SVM classifiers and (2) an HMM-based synthesis approach, respectively. Another research [57] found slight improvement with their SVM-HMM system, with a 96.96% recognition accuracy compared to a 96.4% accuracy by the conventional HMM-based system. However, due to computational costs, this is not a viable system in real-life scenarios as it is.

The SVM-HMM Hybrid is seen as having potential in the field, where a high recognition accuracy has been found in several scenarios. However, there are several issues and limitations that hold back its further development. Many of these have already been considered and while some a resolved in certain scenarios, the bigger issues like computing complexity limitations remain. Due to the resource constraint of researchers, this also makes its further research unappealing.

2.3 End-To-End ASR Models

To reduce complexity and increase flexibility in ASR systems, end-to-end (E2E) models have been developed in recent years [4]. The main difference lies in the architecture, where the Hybrid ASR system uses several models for the task, while the E2E approach directly maps the speech to text without intermediate modeling. Unlike Hybrid models, these E2E models are trained on thousands of hours of labeled data, called pre-training. This model can then be re-trained more quickly on a new and much smaller dataset using what it already knows, which is called fine-tuning. This can be done in various ways, as there are several types of E2E models. These types can be divided into two main categories: explicit alignment algorithms (e.g. Connectionist Temporal classification (CTC), Recurrent Neural Network Transducer (RNN-T), and Recurrent Neural Aligner (RNA)) and implicit alignment algorithms (e.g. Attention-based Encoder-Decoder). Explicit algorithms are most appropriate for streaming ASR, as they can be used for live-streaming data. However, while implicit algorithms yield high results, they need data on past, present, and future situations, which makes them more suitable for ASR that has already been recorded. [2], [4], [58], [59].

Most recently, the best performing and most popular E2E ASR models are Whisper [60], XLSR [61] & XLS-R [62] based on Wav2Vec2.0 [63] and HuBERT [13]. [64], [65]. All models besides Whisper are explicit alignment algorithms, but only Whisper uses an implicit encoder-decoder model, while all the others use the CTC architecture [66] (i.e. encoder-only model). Below, these models are each explained and their performance is discussed while considering their merits and flaws. Lastly, an attempt is made to distinguish the best model for a certain fine-tuning task, by summarising various types of tasks of fine-tuning research that compare several models.

2.3.1 Whisper

This model is an open-source and multilingual model, claiming to be robust against many variables in audio (e.g. background noise and accents), and has been trained on 680 000 hours of audio. It uses a self-supervised end-to-end approach, as can be seen in Figure 2.4. The input audio is split every 30 seconds and converted to a log-Mel spectrogram, before inputting into the encoder-decoder transformer. It is known to be versatile in application since it is not fine-tuned to one dataset. However, this also means that it does not outperform all existing approaches [60].

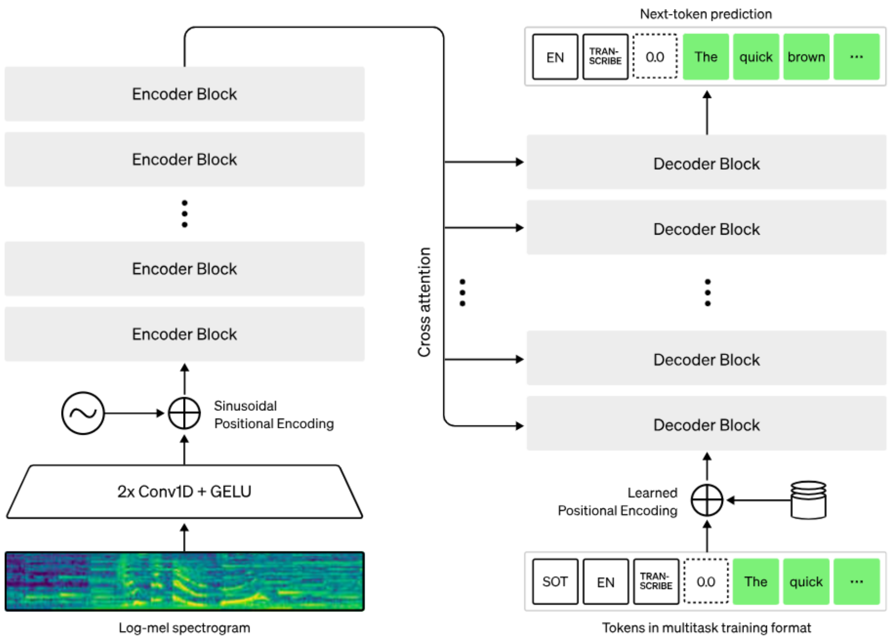


FIGURE 2.4: Illustration of Whisper [11] architecture, from [60].

Whisper performs well overall. On the LibriSpeech clean dataset the human-level error rate was 5.8%. A pre-trained Conformer model has managed to achieve a WER of 1.4% [67], which is the lowest thus far. Whisper cannot match this, but the model used was fully optimised to fit this dataset. The optimised Whisper only achieved a WER of 2.5% on LibriSpeech clean [17], but is more diverse and flexible [60]. Applying Whisper to other datasets showed potential and it sometimes performed much better than many other models.

The influence of the dataset size was also researched, where the biggest dataset gave the best results. However, when using just 8% of the data, Whisper already reached a WER score for English of 10.9%, while for the full dataset, it reached a WER of 9.9%. This difference is very small, so gains are minimal for larger datasets. This should be taken into account when the aim of the application is easy implementation. In the case of multilingual data, the WER was 36.4% for 8% of the data, but 29.2% for all the data, which is a more significant difference [17]. On the other hand, but equally importantly, it has also been shown that Whisper is known to quickly overfit on small datasets [68]. Therefore, the size of the dataset has to be chosen carefully.

2.3.2 Wav2Vec 2.0

Like Whisper, this model is a self-supervised neural net framework, as shown in Figure 2.5. It consists of three parts. The first is the feature encoder, which takes raw audio data and with a 1D CNN, normalization, and a GELU activation function, it gives feature vectors as output. The second part is the transformer, which uses relative positional embeddings. The model passes random mask feature vectors to a transformer and can be fine-tuned by adding a linear layer to the network. Lastly, there is the quantization module, which converts the continuous data into discrete data. [63] However, by itself, Wav2Vec 2.0 is only a feature extraction framework. When adding the softmax layer and using CTC, the model can be used for ASR tasks.

On the clean LibriSpeech dataset, Wav2Vec 2.0 achieved a WER of 1.8, but a 3.3 WER on LibriSpeech Other. One aim of this model was to find if it could be trained on less labelled data. This showed promising results, where the combination of 53k hours of unlabelled data with 10 minutes of labelled data still achieved a WER of 4.8 on LibriSpeech Clean and a WER of 8.2 on LibriSpeech Other. [63]

On basic tasks, more recently, Wav2Vec 2.0 is often outperformed. For instance, a zero-shot Whisper model performs with 55.2% fewer errors on average than Wav2Vec 2.0, when considering 13 datasets including AMI, Switchboard and Common Voice. However, when Wav2Vec 2.0 is optimised, it is still a model with potential.

2.3.3 XLSR

This self-supervised E2E model is based on the Wav2Vec 2.0 model, using the CTC. However, the Wav2Vec 2.0 model was only trained on one language, and the XLSR (and its variations) was trained on more extensive and multilingual data. The XLSR architecture can be seen in Figure 2.5. The first version is the XLSR, with the biggest being the XLSR-53, which is a large model pre-trained on 53 languages. [61] Later, the XLSR-53 model is further developed into the XLS-R model [62] with cross-lingual pre-training in 128 languages and half a million hours of speech audio.

The XLSR-10 Large, an earlier version than XLSR-53 which was trained on just 10 languages,

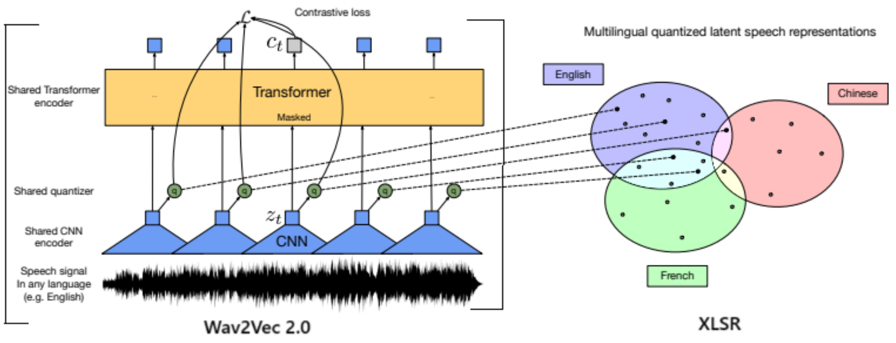


FIGURE 2.5: Illustration of Wav2Vec 2.0 model [63] architecture and addition of XLSR model [61], from [61].

was tested on the Common Voice dataset, as this dataset is multilingual and consists of 38 mainly European languages. This model reduced the Phoneme Error Rate (PER) by 72% [61] which is comparable to state-of-the-art m-CPC [69]. On the BABEL dataset, which is also multilingual but includes Asian and African languages, the XLSR-10 Large improved WER by 16% on state-of-the-art BLSTM-HMM [70]. Overall, the performance differed greatly per language but got competitive results to state-of-the-art for both datasets. This was especially the case for low-resource languages, that do not have a lot of data available. For instance, the WER on BABEL for Tagalog was 40.6% in the best state-of-the-art [69], but XLSR-10 Large reached WER of 37.3% and XLSR-53 reached WER of 33.2%. For Swahili, state-of-the-art reached WER of 35.5% [69], where XLSR-10 matched this, but XLSR-53 improved it to a WER of 26.5% [61].

The next and latest version of this model is the XLS-R. This model can be deployed on various types of ASR tasks and has also been trained on other tasks, e.g. speech translation and language identification. It has also improved state-of-the-art work on many common ASR datasets. For instance, to continue the example of Tagalog and Swahili, the XLS-R reached a new lowest WER of 29.3% and 21.0% respectively. Across several languages, the XLS-R improves the XLSR-53 with 1 WER at equal capacity and goes up to a WER of 2.9 for additional capacity. In addition, it has been shown to match the performance of models trained on just English. For the same capacity, Wav2Vec 2.0 significantly outperforms the XLS-R. However, with increased capacity (i.e. comparing the 0.3B vs 1B version of XLS-R) and little training data, the XLS-R outperforms Wav2Vec 2.0. For example, at 10 mins labeled data, wav2Vec 2.0 has a WER of 32.1% on the clean LibriSpeech dataset, but XLS-R reaches 29.1% WER. However, on more training data, the models are comparable. For example, on 10 hours of labeled data, Wav2Vec 2.0 has a WER of 5.6% and XLS-R a WER of 5.9%. [62]

2.3.4 HuBERT

The HuBERT model is based on the Facebook's AI method and is a Hidden Unit BERT [13]. BERT stands for Bidirectional Encoder Representations from Transformers and learns contextual relations between words by interpreting entire sequences of text at once and masking 15% of the words. [71] However, this means that masks must be of a set length, and this is often not the case for non-lexical elements. Therefore, HuBERT aims to make room for research on the delivery of words, i.e. identifying lexical and nonlexical information in audio. It uses an offline k-means clustering step to find the prediction loss over the masked areas. As a result, it

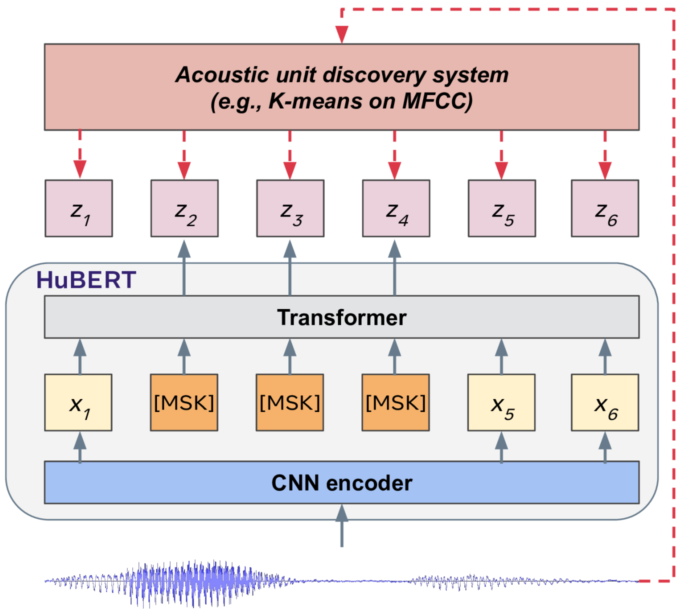


FIGURE 2.6: Illustration of HuBERT [12] architecture, from [13]
.

combines acoustic and language models. [65] This is visualised in Figure 2.6.

HuBERT has been (1) tested on low- and high-resource datasets, (2) compared to other similar methods, e.g. wav2vec 2.0 and (3) considered in regard to different Language Models (LM), i.e. 4-gram and Transformer. With just 10 minutes of pre-training, HuBERT (LM: Transformer) reached a WER of 4.7% on Librispeech test-clean and 7.6% on test-other. This is a lower WER of 0.1% and 0.6% than wav2vec 2.0 respectively. For high-resource datasets, i.e. in this research the 960 hours of pretraining labeled LibriSpeech data was used, HuBERT does not improve over wav2vec 2.0 much. The fine-tuned HuBERT X (LM: Transformer) reached a WER of 1.8% on test-clean and 2.9% on test-other, while wav2vec 2.0 reached 1.8% and 3.3% respectively. Therefore, it shows that HuBERT is especially effective in low-resource scenarios. In addition, no matter the amount of pretraining data, the Transformer LM gave much better results than the 4-gram LM for HuBERT. For example, on 10 minutes of pre-training, HuBERT (LM: 4-gram) got a WER of 6.6% on test-clean and 10.3% on test-other, which is much higher than mentioned above. [65]

2.3.5 Fine-tuning for Low-Resource and Miscellaneous Tasks

Some research has already been conducted on analysing the ins and outs of fine-tuning these models to optimise the performance of other tasks. However, an important question that arises is, which model can best be used for which fine-tuning task. In Table 2.1 all fine-tuning research in the area of low-resource speech, language, or accents can be found. In Table 2.2, more miscellaneous fine-tuning research is summarised, e.g. high-resource language and (audio event) specialisations.

The varying results are striking, even in similar research, where there is not one model that seems to outperform the others. When fine-tuned on Kazakh [72], Wav2Vec 2.0 outperforms

HuBERT and Whisper Large. Moreover, when fine-tuned on Luxembourgish [73], Wav2Vec 2.0 has better WER results, but Whisper is concluded to be more appropriate in application when thoroughly considered in context. In addition, Whisper often outperforms all other models [15], [17], [74], [75]. However, in some specific scenarios, e.g. a dialectal transfer task [16] and dysarthric speech [14], XLSR-53 has its benefits over the others. Lastly, in one very specific tasks, HuBERT surprisingly outperforms the other models, i.e. when creating acoustic word embeddings [76]. Moreover, HuBERT often performs very similarly to Wav2Vec 2.0 [14], [16], [77]. However, HuBERT is a relatively new model, so very little focused research on its capabilities has yet been done.

Thus, it seems to be very specific to the data, parameters, and variables to know which model fits best to a task. In addition, it is important to thoroughly look into the text and not just evaluation metrics, as they do not always show the full picture, e.g. for some texts the importance of accurately transcribed punctuation or numbers can weigh heavier than just WER.

2.4 Non-lexical Elements in ASR

These conventional ASR methods have currently reached an impressive performance. However, one of the current main challenges is to make the ASR system more complete by accurately representing real-life speech. The lack of detail and extensive variety in ASR training sets results in scenarios where the ASR system might function very well in one scenario, but not in another. Therefore, it is very important to have an appropriate and well-represented dataset. This consists out of conversational speech, as this closely resembles real life and other aspects that help improve this representation of real-life settings, e.g. background noise, with overlapping and spontaneous speech [81]. One new factor of importance in conversational speech, which was less relevant in read-aloud speech, is to identify non-lexical elements. This not only helps the system perform better (i.e. to accurately transcribe words that were stuttered or laughed through), but also aids it in accurately interpreting the correct type of tone, intention, or context from only speech to text (e.g. to identify the emotional state of the speaker) [82].

This chapter will differentiate between (1) detecting non-lexical elements in Hybrid ASR models, where elements are first separately detected and then added to the pipeline, and (2) non-lexical elements integration in E2E ASR models, where these two steps are taken together. Lastly, (3) three appropriate ASR datasets will be examined to see what research potential there still is within the ASR-paralanguage field.

2.4.1 Detection of Non-lexical Elements Using Hybrid Methods

Important non-lexical elements in speech processing are those that act as a social cue. A few different types are most commonly identified, namely disfluencies, fillers, backchannels, affect bursts and vegetative states. [6], [7] Almost every category has been researched quite extensively as a separate entity in the past, with a general aim to extract these vocalisations from audio. The next step, which some studies include in their work and others do not, is to incorporate this into the ASR pipeline. The section will focus on the detection of these elements and subsequent integration with Hybrid ASR models.

Disfluencies

Disfluencies that occur in natural speech can be distinguished by speech that falls into repetition, is not smooth, or not continuous [83]. This is the most extensively researched type of paralanguage in ASR, where many types of models have been applied. This is probably since this research not only helps with social speech interpretation but also aims to decrease errors in basic ASR performance. This is because disfluencies like hesitations, repetitions, and false starts often introduce inconsistencies in the audio signal, which can lead to errors in word alignment and recognition.

One commonly used dataset for disfluencies is Switchboard. As a result, many studies can easily be compared on this dataset. Over the years, the performance of disfluency detection gradually improved. In 2013, one study reached a F-score of 84.1% [32] on Switchboard with the Max-Margin Markov Networks model. In 2015, a research [31] improved this performance to a F-score of 85.4% with the use of semi-Markov Conditional Random Field (CRF) with prosodic feature incorporation. In 2016, the Attention-based model [84], using a Bidirectional Long-Short Term Memory Neural Network (BLSTM) to encode the source sentence, got an F-score of 86.7%. In 2017, the Transition-based model [85] reached a F-score of 87.5%. Then in 2019, performance went up again to 89.0% [86], with research using a non-autoregressive neural machine translation (NMT) model with a transformer. The performance of this high-scoring NMT model [86] was also tested on the Chinese corpus. It performed similarly to state-of-the-art but performed much worse than Switchboard, due to the small dataset and inherent complicated structure of the language, with an F-score of 52.8% [86].

Other research focused more on differentiating between types of disfluency to detect. For instance, one study in 2014 used a BiSLTM for disfluency detection and got an 85.9 % F-score, which matches state-of-the-art performance. However, it was better at detecting non-repetitions than previous research, which is noturiously hard to detect, with a F-score of 66.7 %, compared to 61.1 % F-score of the CRF of previous work [87]. [30]

Fillers

Fillers are one of the most common vocalisations [88] including words like "uhm" and "ah". The specific words and sounds used differ per language [89], but all have the main function of giving the speaker time to think [90]. They are often considered a subgroup of disfluency, as the identification of filler words also helps with error detection in ASR. However, due to its specific use and applications in other tasks (e.g. speaker identification), there has been an extra focus on it, and has often been considered separately. To identify fillers in speech, many types of models have been applied over the years.

Already back in 2006, research was done on identifying filler words. One study [33] used an HMM to find filler words in two different datasets. On the Broadcast News dataset, the human-generated reference transcript (REF) had a NIST error rate of 18.11% and Speech-Recognition Output (STT) of 56.00. On the Conversational telephone Speech dataset, the REF was at 26.98 and STT at 41.66. Furthermore, another method that has been used to identify filler words is with the SParseval tool [91]. One study, also in 2006 [92], reached an F-score of 93.1% in identifying filler words on the Switchboard dataset. Interestingly, they also note that deleting filler words earlier, before parsing to the ASR, did not yield significantly better results, improving from 87.8% to 88.9%.

A surge in research in the area occured in 2013, due to the start of the INTERSPEECH challenge. This yearly challenge aims to get a comprehensible overview of both established benchmarks and innovative work currently done in the field of speech communication [93]. In 2013, the topic was computational paralinguistics, focusing on social signals, conflict, emotion, and autism [94]. In the social signals sub-category, filler words and laughter were researched. Datasets are provided for the research and this sub-category, this was the SSPNet Vocalisation Corpus. The baseline was first set with a SVM, which resulted in an 83.6% Area Under Curve (AUC) for filler words. The highest performing research that participated in this challenge [95] used a DNN with filtering and masking techniques and got an AUC of 89.7%.

In 2016, one study [96] used a DNN-based system to identify filler words. From the Confusion Matrix, the highest True Positive score of 82.62% can be extracted for System E and the biggest issue was that filler words were sometimes classified as garbage. A CNN was subsequently used for classification and the overall AUC improved with 4.85% and 6.01% absolute on the CTS dataset and UT-Opinion dataset, respectively.

Backchannels

This category contains words like "yeah" and "right" to acknowledge or encourage the speaker [97]. Notably, backchannel responses vary immensely with culture [98], [99] and most research has been done on American and European populations, so it has to be kept in mind that not all results can be compared fairly, and carry a bias [100].

ANN's are implemented widely in this field. One research study compared 2 types of networks on the Switchboard dataset. They applied a feed-forward network, resulting in an F1-score of 0.327, and an LSTM network, resulting in an F1-score of 0.375.[97] More specifically, the prediction of backchannels (i.e. either on identifying backchannels or determining when backchannels should happen), has become very popular [100]–[103]. One research study [103] focused on identifying backchannels and the ASR used the hybrid DNN-HMM system with the Soundboard dataset. They found that including personal embeddings that mimic behaviour improved performance. The best accuracy they reached was 58.9%. Another research [100] used semi-supervised learning on a Hindi dataset consisting of not just audio, but also video, to insert backchannels. They found that only minimal annotated data was needed to train a ResNet and Random Forest classifier. With the use of quantitative analysis, they uncovered that 95% of participants found that the model outperformed the random prediction. They also introduced listener embeddings to emulate various backchanneling behaviours.

Most other research focused on when backchannels should happen. One research study applied an NN on Switchboard [102] to predict the timing of backchannels, which found that more layers in the network resulted in slightly higher F1-scores. While the F-score was low, with the highest at 0.06, this metric is not appropriate for this type of research. A subjective evaluation metric was needed to conclude if the backchannels generated were natural and here they found that the (subjectively) best-rated network was the 3s-trained NN. Another research [101] used multimodal output features and two models (i.e. HMM and CRF) to predict and generate backchannel responses, both audio and visual.

Vegetative Sounds

A category that is researched very little, is vegetative sounds. These are mostly involuntary sounds that include vocalisations like coughing, yawning, and snoring [7]. Often, these are sounds that are not annotated in ASR datasets, as it is more of a nuanced observation and hard to pick up on audio. In addition, this information was often not seen as relevant or simply did not even occur in the recording at all.

However, during the COVID pandemic, there has been a surge in (comparative) research regarding coughing and sneezing [104]. Often, only cough samples are extracted, but sometimes this is also combined with speech processing and other vocal characterizers like sneezing. The difficulty remains when comparing its performance with state-of-the-art, as it cannot be compared to similar ASR datasets. In addition, very little ASR research has been done into crying, probably due to the lack of data and overall need for this. However, a specific interest was taken in infants crying, mostly to identify different types of crying like hunger, colic, diaper, and sad [105]–[107]. One research study reached an average accuracy of 94.77% in differentiating between these and other categories [107]. Moreover, yawning has also barely been researched. There has been a specific focus on the scenario of a driver yawning [108]–[111], but most research focused only on video, to see if the mouth was open, and not on audio.

Affect Bursts

The last set that can be identified within nonverbal vocalisation types, is affect bursts. These include vocalisations like laughing, crying, and screaming. More recently, these types of bursts have been annotated more, but still not that much variation in data is available. Also, these bursts occur relatively infrequently. Probably due to the lack of data, this field is under-researched regarding its integration with ASR.

The main vocal characterizer that is researched is laughter, as this can be found in annotated ASR datasets, although minimally. In 2007, one research study [112] focused on developing a laugh detector by using different feature types, i.e. spectral and prosodic, and classification techniques, i.e. GMM, SVM and MLP. The Equal Error Rate (EER) was around 3% for differentiating between laughter and speech on the ICSI dataset. They also found that mixing the GMM with SVM and fusing feature types improved performance the most. Alternatively, often an NN has been used, where another research in 2007 [113] used an NN for laughter recognition and reached an EER of 7.9% on the same ICSI dataset. More recently, another study [114] used a DNN to detect laughter and reached an accuracy of 88.1% on the SSPNet testset. With a CNN, another research [96] improves their ASR AUC performance by 8.15% and 11.9% absolute for laughter on the CTS dataset and UT-Opinion dataset respectively. The highest True Positive for laughter was 73.3%, where the biggest mistake made was that laughter was sometimes classified as garbage.

In the last few years, different models have been optimised for laughter detection, focusing on the right combination of features, classifiers and training data. In 2021, one study [18] compared various datasets used in laughter detection to make it robust in noisy environments. The next year, in 2022, a more general study [115] was conducted on the comparison of laughter detection in the current state-of-the-art. They found limitations that have to be addressed, e.g. shortcomings in sufficiently large and naturalistic datasets and the use of the correct type of evaluation metrics. However, during this time the focus was on E2E systems, as will be elaborated on in the next section.

2.4.2 Non-lexical Elements in E2E ASR Models

There are two main methods to incorporate non-lexical elements into E2E ASR. One way is to fine-tune existing E2E ASR models. As far as the writer is aware, very little research has been done in this field regarding non-lexical elements of speech. One study investigated social signal detection in combination with E2E ASR, specifically investigating laughter and filler words [116]. This research compared the CER with the final ASR performance, showing the lexical impact. Without labels, the CER was at 19.41% and with labels inserted on both sides, this was at 19.69%. This showed that it is possible to incorporate labels, i.e. social cue information, into the transcript without the ASR performance deteriorating.

Another way is to separately detect these non-lexical elements and add them to the ASR pipeline at the end. The overall number of studies performed in this area is minimal, probably due to the combination of the lack of data and since this type of model has only recently been extensively researched and developed. Disfluencies, including hesitations and fillers, have been the focus of the field, as their removal and subsequent cleaner sentences have the potential to improve ASR performance.

One study [117] on disfluency removal in E2E ASR models is often used as the baseline in this field. This research was the first to investigate the direct integration in three E2E ASR models, i.e. CTC, Sequence-to-Sequence (Seq2Seq), and Transformer, and found that these models can indeed directly omit disfluencies from the transcripts. Each ASR model was trained twice, first with a basic ASR pipeline and later adding the disfluency detection model, being trained on disfluent speech and transcripts, and secondly by integrating the E2E models directly, which was only trained on disfluent speech but fluent transcripts. However, for all three models, the ASR E2E performance got worse compared to the baseline system of no disfluency integration, when tested on the Corpus of Spontaneous Japanese (CSJ) dataset. For the CTC model without any disfluency removal and the CTC model with the disfluency detection model, i.e. called CTC base and CTC pipeline respectively, WER was at 12.5%. However, the CTC E2E model performed worse, at 14.3% WER. For the Seq2Seq base and Seq2Seq pipeline, this was at 11.2 % WER, but at 12.2% WER for Seq2Seq E2E. Lastly, for the Transformer, the base and pipeline were at 11.2 % WER, but at 13.8 % for Transformer E2E.

The next year, two other studies concurrently considered the labeling of hesitations and integration into E2E ASR. One research study [9] considered the same dataset, i.e. CSJ, and used the ESPnet2. The new model improved the most over the baseline model for eval3, i.e. the largest dataset. The baseline model got a CER of 7.3% and SER of 48.4%, while their ASR hesitation model with Speed Perturbation (SP) got a CER of 4.1% and SER of 34.2%. This seems substantial and while it does show potential, the biggest improvement was caused by the SP. The baseline with SP got a CER of 4.7 % and SER 37.5%. The other research [8], focused on increasing accuracy by adding disfluency training data and replacing partial words. There was an improved ASR performance, with a relative WER improvement of 22% for the test set of disfluencies and 16% on the test set of stuttering speech.

However, a follow-up research [118] considered filler and hesitation labeling with E2E ASR models in spontaneous speech. Their proposed method got an improved CER of 10.3% and SER of 32.8%, compared to the baseline disfluency removal of 2020 [117] with a CER of 12.9 % and SER of 49.9%. These results are also compared to the ASR model without disfluency labeling, which got a much higher CER of 16.0% and SER of 63.8%. This study shows the potential of integrating disfluency labeling and removal to improve spontaneous ASR performance.

Some other interesting research was also conducted. One study looked into using different end-to-end models and features for laughter detection, but did not include any research on the ASR performance. They found that using the BLSTM CTC model gave the best results.

In addition, a combination of spectral and prosodic features yielded the best results with 81.83 % accuracy for laughter detection [119], which affirms the results from the earlier research in 2007 [112]. Moreover, another study [120] used a Seq2Seq DNN with a two-layer BLSTM, several feed-forward layers, and a Rectified Linear Unit (ReLU). , to prove that it is more beneficial to divide non-verbal vocalisations into separate types of classes, i.e. laugh, breath, whistle, sip, etc, than simply differentiating between "laughter" and "other", as is often done. Also, they achieved the highest performance of 1.58% EER on the ICSI dataset with multi-label classification and the Seq2Seq model.

Overall, there is very limited research on the integration of non-lexical element detection with E2E ASR, but there is potential in the field. In addition, it is quite telling that when adding social cue labels like laughter, the ASR performance does not deteriorate. This shows potential improvement for adding other non-verbal social cues to ASR transcriptions. Moreover, no research has been done yet into the fine-tuning of E2E ASR models for non-lexical elements, but there was potential in related areas, so there could be opportunities here.

2.4.3 Integration of Non-lexical Elements in Existing Datasets

To find the gap in research it is not only important to consider what research has already been done but also to analyse what specific gaps there are in this research. One way to do this is to consider common datasets in this research and to see how prevalent certain non-lexical elements are, to see what would impact the development of the ASR field the most. Therefore, we analyse three different datasets, based on the count of the frequency of the occurrence of the five paralanguage categories, i.e. as described in Chapter 2.4.1. Disfluencies will not be analysed in the datasets, as this has been researched a lot already in the past. The other four categories will be considered: filler words (i.e. uh, um, hmm), backchannels (i.e. yeah, right, uh-huh), affect bursts (i.e. laughter and crying), and vegetative sounds (i.e. coughing and yawning). The exact words will depend on what and how the elements are labeled in their respective datasets.

The results are then compared to the research done above, to find what potential research still is to be done. The three datasets that will be analysed are the three most common and largest ASR datasets used in non-lexical vocalisation detection tasks, as identified in Chapter 2.1.4. These are: ICSI Meetings, AMI Corpus and Switchboard.

Two notes are of importance. First, the word "right" can be used as a word in a sentence too, so this count might not be an accurate representation of the occurrence of backchannel words. A more in-depth analysis of its use in context should be done when researching this word as backchannels, which was not done due to time constraints. Secondly, each dataset annotates the words differently, especially for the filler words. For example, in the Switchboard dataset, the word "huh" never occurs, while this is the biggest dataset. When analysing the text, it is clear that the "huh" from AMI and ICSI, is "hmm"or "uh" in Switchboard. Therefore, a direct comparison cannot be made and there must be looked at how these directly translate to each other. As a result, in the case of this analysis, a broader analysis is done and there will mainly be looked at the total count of filler and backchannel words.

ICSI Meetings

The ICSI Meetings [27] dataset has elaborate annotations, both in vocal and nonvocal sounds. In Table 2.3, an overview of the elements that were annotated is given, i.e. vocalisations, filler words and backchannels. For vegetative sounds, they have labelled the vocal sounds yawning and coughing, but very minimally. For the affective bursts, only laughing was annotated; even differentiating between the type of laughter, e.g. "breath-laugh" and "laughing while talking". In total, as many as 11 513 laughter events occurred. Various filler words were identified and in total, almost 30 000 occurences were counted. Moreover, a bit more than 25 000 backchannel words were identified when counting all variations of the backchannel lexical elements, i.e. mm, yeah, and right. This dataset contained 70 hours of speech. This shows that for filler words and backchannels, there is plenty of data available. In addition, there is also a lot of data on laughter, but significantly less on coughing and yawning.

AMI Corpus

In Table 2.4, an overview is given of the vocalisations, filler words, and backchannels in the AMI dataset [28]. The AMI Corpus included a few vocal sounds, but did not differentiate between many different kinds, e.g. just "laugh" and not "laughing while talking" and the total number of laughter vocal sounds was high, where there were 16 524 laughter counts. For the vegetative sounds, "coughing" was annotated a bit over 1 100 times and "yawning" was not annotated at all. There were almost 40 000 filler words and 38 000 backchannel words. There were a 100 hours of speech in this dataset and the focus of these audio recordings was on multi-party interactions. Also for this dataset, it is the case that there is a lot of opportunity to research all the types discussed, but again there is less data on coughing.

Switchboard

The Switchboard dataset [21] has a lot of labels for non-lexical elements and contains 260 hours of speech. However, the variety of labelling was not very high. An overview can be seen in Table 2.5. It has no vegetative sounds and only one type of affect burst, namely laughter. However, they also labeled if this was during speech and what words were said while laughing. In total, a bit more than 35 000 laughter events occurred. In addition, many filler words and backchannel words could be identified, with almost 125 000 and 100 000 events respectively. As a result, a lot of research could be done on all types of non-lexical elements discussed.

2.5 Conclusion

After analysing the literature and three relevant datasets, some conclusions can be drawn as to why my specific research should be conducted. This can be seen in the way of (1) topic and (2) execution method. The importance of researching paralanguage has been shown along the way in this chapter, where conversational speech is used to train the model to more accurately represent human interaction in ASR. However, exactly what part to focus on is still debatable. In addition, the execution method, i.e. the type of model and specific method, used for this identified topic will be discussed.

2.5.1 Paralanguage Gap in Research

As identified in the previous sections, five types of paralanguage can be considered. When finding the gap in research, one can look at (1) the under-researched areas and (2) how commonly they occur in speech. Disfluencies have been the most popular in research and are very versatile (i.e. difficult to analyse and distinguish), hence they are omitted from this discussion.

There is a lot of data on filler words and backchannel words as well. While this shows potential, these have also been researched in various scenarios. In addition, these are less interesting from a paralanguage perspective. The main aim of its detection would be to help conversational robots interpret and identify speakers and aid the ASR in gauging context or emotion recognition. The remaining two are vegetative sounds and affect bursts, which were unequivocally researched the least. In theory, these include all vocalisations like laughing, crying, yawning, coughing, etc. However, in many datasets this is often not present.

In the three large ASR datasets discussed, some of these vegetative sounds and affect bursts were present. The ICSI corpus did contain a few instances of yawning and coughing, but showed that this was not often present in spontaneous speech. However, the much larger number of laughter occurrences highlights the significant impact it can have on the development of ASR systems when laughter is integrated. There were more laughter events (i.e. 11 500 events) and extra information was even available, i.e. if laughing occurred during talking or not. In the AMI corpus, yawning was never annotated, but more coughing events were labeled. With 1 100 events, these show that integrating it into ASR would have a much smaller impact. Many more laughter events occurred, at 16 500, but less information is available on the type of laughter. The Switchboard dataset was by far the largest, with more than double the hours of labeled speech as the AMI corpus. The size of the dataset is reflected in the high number of laughter events recognised, with just over 35 000 events. They even labeled if laughing occurred during talking and if so, during which word. There is no labeling on other affect bursts besides laughter.

Consequently, the greatest potential for advancing ASR systems lies in effectively detecting and incorporating laughter events. It has been the least researched but is relatively very prevalent in spontaneous speech. The Switchboard corpus has the most occurrences and even has some extra information, hence this dataset is probably the most informative for the model. The ICSI corpus and AMI corpus both had reasonably similar amounts and types of data, with 11 500 and 16 500 laughter event counts, respectively and were all recorded in meetings. Each has its benefits, where ICSI has extensive details that can be extra informative to the model, while the AMI corpus has more laughter events. When this extra information is not used in the training process, AMI will be the preferred choice. This is also the case for this research, as the focus will be on integrating laughter transcriptions into the ASR system by tagging the laughter event.

2.5.2 ASR Model Type Opportunities

The type of models used for ASR systems have developed from statistical and hybrid models to E2E models. The E2E model has been shown to behave the best, with high performance and robustness. While E2E models have been used to transcribe laughter, this was only done as a separate event and later added to the pipeline. However, the technique of fine-tuning has shown to have good results too. This is an under-researched field, where no type of paralanguage has yet been used for fine-tuning as far as the author is aware. This allows for interesting insights and makes room for potential other research. As a result, this type of model will be used for the integration of laughter transcription in ASR systems.

Exactly which model should be used can also be debated. Each model has its merits and can be argued for in different applications. In the case of paralanguage, there is one model that stands out. HuBERT has a particular focus on "cues from how those words are delivered, e.g., speaker identity, emotion, hesitation, and interruptions. " [13]. As a result, this is the first model that will be used in this research. Secondly, the Whisper model is currently the largest and most popular. It is known to be flexible, trained on lots of data, and can handle background noise quite well. Therefore, this will be the second model used and will be compared to the performance of HuBERT.

2.5.3 Potential Limitations

Some limitations should be taken into consideration before starting with the research and training the model. These could affect the performance, scope and applicability of this research.

First of all, the focus of this research is on transcribing laughter. To do this, laughter events are tagged. However, this focus is only on laughing out loud, not on laughing while talking or smiling. This could potentially confuse the system, as the model could get confused by the similarities. Therefore, this research will also look specifically at what happens at those points where laughing while talking occurs.

Secondly, laughter will sound quite differently per person, context, intensity, and acoustically in the room. This can make it challenging for ASR system to identify all types of laughter consistently and to make it robust for laughter that it was not trained on. This also means that laughter will sound differently in the AMI dataset and Switchboard. Beneficially, both datasets consist of conversational speech in a similar context, so this means that a fair comparison can be made between the datasets. The main consideration would be when applying this system in other scenarios or on other types of data.

Thirdly, a recurring issue in the literature is that there is no standardised evaluation method for this kind of research. In most NLP and ASR research WER is used for the transcription and accuracy is used for the identification of paralanguage, but this does not exactly fit the requirements for this research. Some researchers have developed unique metrics to precisely assess their results, but this approach complicates comparisons with other studies and the evaluation of their effectiveness. Due to the various opinions and difficulty with this topic and the general lack of similar research, it is best to keep to what is used most often. This way a new baseline with this new research can be set and still compared to the state of the art.

Fourthly, fine-tuning is inherently problematic. Several issues may arise, like catastrophic forgetting. In addition, a balance must be found between the high performance of ASR, with a good WER, and a high accuracy in transcribing laughter. Often, these two can clash. Moreover, the issue of overfitting may occur. This could limit the generalisation to other datasets or real-life scenarios. Also, this is a time and computationally complex problem. Hence, this could mean that model refinement cannot be carried out to the maximum capacity within the time this research must be conducted.

Lastly, ethical and bias concerns should always be taken into consideration. The data used in this research has been collected ethically and with consent for its open-sourced use. However, the data only consists of a few people and the English language. It should be noted that this means that the final fine-tuned ASR system is not representative of people from all backgrounds, genders, accents, or other languages. A lot more training and fine-tuning with more inclusive datasets must be done before this system can be applied in real life. It is meant as the first step in finding opportunities for further research in making the ASR system more accurate in representing real life and gauging emotions and contexts.

Research papers	ASR Models (Types)				
Topic	Whisper	Wav2Vec2	XLS (R-53 & -R)	HuBERT	Conclusion
Fine-tuning on dysarthric speech [14]		Wav2Vec	x	x	Features extracted from **XLSR** model yielded best results for English, Spanish, and Italian, seeing as it contains more variations of similar phonemes than other models. On the UA-Speech dataset, wav2vec2 got a WER of 29.3% and CER of 27.7%; HuBERT a WER of 29.7% and CER of 28.2%; XLSR got a WER of 26.1% and CER of 24.1%.
Fine-tuned on Dutch dysarthric speech [15]	Small		x		**Whisper** performs better than XLSR-53 for all impairment severity groups, i.e. severe, moderate and mild. The best results was reached on the mild group, the performance decreasing with severity, where Whisper had a WER of 37.51% and XLSR-53 of 43.54%
Developing Maltese (language) with ASR [78]	Small, Large	x	x		**XLS-R** performs best, with near 2% CER and 8.53% WER on the CommonVoice dataset. Whisper-Small and Large were very similar, with CER between 5 and 10 % and WER around 20%.
Fine-tuning on Kazakh (language) [72]	Large	x	x		**Wav2Vec2** had the lowest CER and WER at 2.8% and 8.7%. Whisper achieved 4.1% and 19.8% and XLSR-53 4.3% and 13.5% respectively.
Fine-tuning on Luxembourgish (language) [73]	Large	x			Looking at WER, Wav2Vec2 was the best with 9.5%, where Whisper had 12.1%. However, **Whisper** was better than Wav2Vec2 due to restoration of capitalization; punctuation; better word recognition
Fine-tuning for African accent; AfriSpeech [75]	Medium		x		For all accents, **Whisper** outperforms XLSR. Fine-tuned Whisper performs best 13/20 times, unfine-tuned Whisper 7/20. For all datasets in AfriSpeech fine-tuned Whisper model outperforms fine-tuned XLSR-53.
System fine-tuning for low-resource speech translation [16]		x	x	x	Over many datasets, **Wav2Vec2** and **HuBERT** got very similar results. **XLSR-53** does not perform well for low resource task, but does well on dialectal transfer task.

TABLE 2.1: Part I (low-resource speech): Table of research on fine-tuning the pre-trained models Whisper [11], Wav2Vec2 [63], XLSR [61] and HuBERT [12].

Research papers	ASR Models (Types)				
Topic	Whisper	Wav2Vec2	XLS (R-53 & -R)	HuBERT	Conclusion
Fine-tuning for German speech recognition [74]	Small		x		Fine-tuning the full Whisper model, with the last six layers of encoder and decoder simultaneously, resulted in the lowest WER of 3.1%, although forgetting occurs. Experience Replay (counteracts forgetting) is a good asset for improving WER. **Whisper** outperforms XLS-R and XLSR-53 after 5 epochs.
Whisper Audio Tagging (identifying background noise simultaneously with spoken text) [79]	Tiny, Base, Small, Medium, Large	x		x	**Whisper-Large** and Whisper-Medium got similar results. The worst performing were Whisper -Tiny, HuBERT and Wav2Vec2. From signal-to-noise ratio 20 to 10, HuBERT had equal low (<3%) WER results to Whisper-Large. All models performed increasingly worse from signal-to-noise ratio 0 and up.
Robustness to distribution shifts and other disturbances[17]	Large	x			On average of 14 datasets the WER of wav2vec2 was 29.3% and for **Whisper** 12.8%. On LibriSpeech Clean, both got a WER of 2.7%.
A base-line for Code-switching speech (switching between Spanish, French and Chinese to English) [80]		x	x	x	For multilingual and monolingual speech encoders, the first heavily outperforms the latter, where **XLSR-53** or **XLS-R** perform the best, depending on the language of code-switching. The highest accuracy was at 75.16% for Spanish-English and an average of 59.21% over all languages by XLS-R.
Constructing (unordered) Acoustic Word Embeddings with E2E models, and their ability to convey the sentence meaning [76]		x	x	x	For Xitsonga, Mandarin, French and English, **HuBERT** (trained on English) outperforms XLSR-53 and Wav2Vec2, reaching the highest average precision of 6̃0% for the Buckeye (English) dataset. AWEs created by mean pooling, instead of subsampling, were used, where HuBERT also outperforms Wav2Vec2.
Fine-tuning for Automatic Speaker validation [77]		x	x		**XLSR** marginally but consistently performed better than HuBERT, as it had more diverse training data, e.g. the lowest EER of XLSR was 0.585 % and for HuBERT this was 0.590%

TABLE 2.2: Part II (miscellaneous tasks): Table of research on fine-tuning the pre-trained models Whisper [11], Wav2Vec2 [63], XLSR [61] and HuBERT [12].

Type	**Words**	**Occurrence**
Affect Bursts	Laughter	11 513
Vegetative sounds	Coughing	207
	Yawning	56
Filler Words	Uh	13 887
	Um	8 041
	Hmm	5 865
	Huh	826
	Total	*28 619*
Backchannel	Mm	4 618
	Yeah	15 671
	Right	5 482
	Total	*25 771*

TABLE 2.3: Counting occurrence of Non-vocalisations in the ICSI corpus [27].

Type	**Words**	**Occurrence**
Affect Bursts	Laughter	16 524
Vegetative Sounds	Coughing	1 116
Filler Words	Uh	25 764
	Um	12 394
	Hmm	1 446
	Huh	293
	Total	*39 897*
Backchannel	Mm-hmm	4 608
	Yeah	29 475
	Right	3 603
	Total	*37 686*

TABLE 2.4: Counting occurrence of Non-vocalisations in the AMI corpus [28].

Type	**Words**	**Occurrence**
Vocalisation	Laughter	35 713
Filler Words	Uh	86 528
	Um	37 319
	Hm	1 013
	Total	*124 860*
Backchannel	um-hum	16 016
	uh-huh	16 315
	Yeah	48 067
	Right	17 045
	Total	*97 443*

Table 2.5: Counting occurrence of Non-vocalisations in the Switchboard corpus [21].

Chapter 3

Methodology

This thesis aims to compare the optimal performance of fine-tuning two ASR models, HuBERT and Whisper, on speech and laughter. The evaluation is based on the WER for the ASR and on the F1-score for laughter detection, comparing the results for the two datasets AMI corpus and Switchboard. In this Chapter, the method explains: the experimental setup, the overview of the pipeline from the dataset divisions through to evaluation; what specific configurations were needed to optimise the two models for fine-tuning on laughter; and the specific software and hardware that were used.

3.1 Experimental Setup

This thesis consists of three main experiments. However, first, the correct parameters for HuBERT and Whisper have to be found. The main parameters investigated were the optimal learning rate, weight decay and number of epochs. Ideally, the parameters of both HuBERT and Whisper are fine-tuned for both datasets. However, due to computational and time constraints, only AMI (i.e. the smaller and noisier dataset) is used to find these parameters for HuBERT and Whisper. These initial tests used just 20 epochs. With these final parameters extrapolated from the initial tests with AMI on 20 epochs, the main experiments could be completed with more epochs on both datasets (i.e. AMI and Switchboard). It must be noted that Whisper and HuBERT could potentially perform better on Switchboard if the parameters were fine-tuned on the same dataset.

3.1.1 Types of Experiments

There are three kinds of experiments: the Zero Shot (i.e. no fine-tuning), the Fine-Tuned Without Laughter and the Fine-Tuned With Laughter. The first, requires no fine-tuning and considers the results based on evaluating the models on the test set (i.e. 20% of the full datasets). This shows the behaviour of the models as they are; without learning accents or dataset-specific noises yet. The second, fine-tuned the model but excluded the laughter labels from the training set. During evaluation (i.e. validation and test set), the laughter labels are included again. This shows the behaviour of the models when fine-tuned, so this could reflect a more nuanced comprehension of the underlying emotional or contextual signals in speech that are not directly tied to laughter. However, this may still be influenced by its presence or absence. Lastly, when fine-tuned with laughter, the potential performance of the models could be found. Evaluating these models demonstrates their ability to identify laughter, which could enhance the accuracy and relevance of the model's responses in real-world applications. For all experiments, the main dataset used was the AMI Corpus, later comparing the models' performances to the Switchboard dataset.

3.1.2 Model Configuration

Each model has to be configured to their best ability to optimise the fine-tuning, which specifics differ per model. This difference is due to the inherent nature of each model. The Whisper model is pre-trained on much more data and optimised for easy use. It is also the most popular and there are more possibilities to tweak it. The vocabulary consists of part-words that are co-mixed. The HuBERT model is more flexible with new sounds, due to the CTC head that uses a letter vocabulary: the focus is on phonetic sounds.

Whisper Model

The Whisper model is trained on a lot of (types of) audio, namely "680 000 hours of multilingual and multitask data" [11]. The audio used was both clean and noisy, extracted from various data types, like podcasts, lectures, conversations and interviews. Also, approximately 30% of the data was in another language than English. There are several types of Whisper models, but in this thesis, the largest model that still performed well was chosen, namely Large-V2. Large-V3 was also considered, but testing showed the model hallucinated more, e.g. showing "!!!" as output. This was in line with what other people have also observed [121]. In addition, due to computational constraints, the Distilled Whisper Large V2 version [122] was chosen, as this promises to be 6 times faster, half of the size, but performs within 1% WER of the normal Whisper Large V2. Whisper has its own compatible processor, which includes the feature extractor and tokenizer. Therefore, the WhisperFeatureExtractor and WhisperTokenizer are used. The laughter token is added to the vocabulary and model, with "<laughter>". The data collator, which is responsible for padding and batch creation, ensures that the "input_features" and "labels" are padded with the help of the processor.

HuBERT Model

HuBERT is a self-supervised model that was trained on LibriSpeech and Libri-Light. These datasets contain 61 000 hours of audio from audiobooks, with varying degrees of background noise. The model used in this thesis is hubert-large-ls960-ft, from Facebook. This is the second largest available, but due to computational constraints, the biggest was not possible to use in this research. HuBERT does not have its own processor, but the basis of this model is the Wav2Vec2 model, so the Wav2Vec2FeatureExtractor and Wav2Vec2Tokenizer can be used. After the "<" token is added to the vocabulary, a random and uncommon token chosen to represent laughter, the model embeddings again have to be resized. There is no function for this in HuBERT, so a new embedding layer is added to the LM head with the new size. The new embeddings are initialised with the existing ones and the new tokens are randomly initialised. This layer now replaces the old model's token embeddings. The data collator used is specifically for CTC, as it uses the appropriate processor, so this ensures the "input_values" and "labels" of the batch are padded. In addition, the feature extractor is frozen during fine-tuning, as this is redundant and saves computational capacity. This is recommended to do in a similar tutorial [123] and was done in similar research [124], [125].

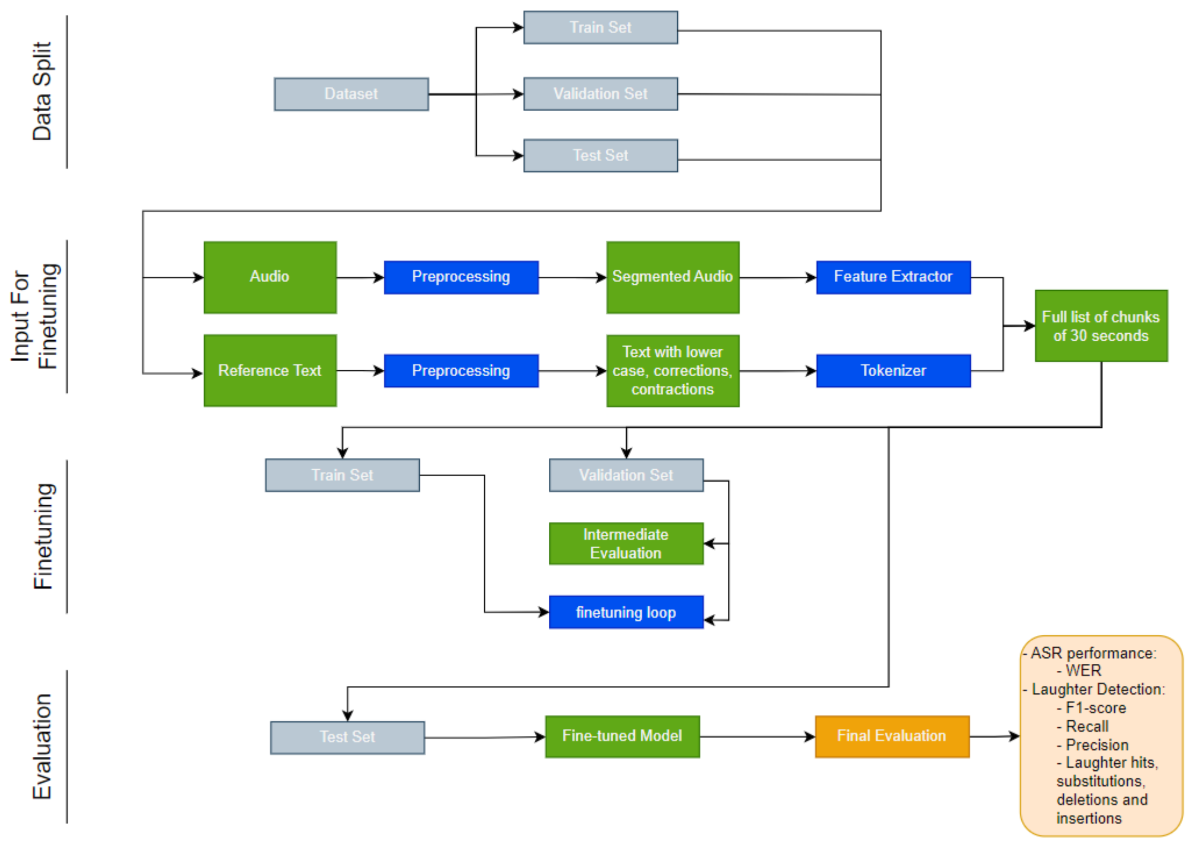


FIGURE 3.1: Illustration of the overall pipeline from data splitting, creating the input for finetuning to the finetuning process with evaluation, applicable to Whisper [11] and HuBERT [12].

3.2 Fine-tuning Pipeline Overview

The pipeline for both models is very similar, hence a general overview can be given that applies to both. First, the data is normalised and divided into three subsets: the training set, validation set, and test set. To create input for the models a feature extractor and processor are defined and configured. Then, the features are extracted from the audio and the preprocessed reference text is tokenised. It is then divided into chunks of 30 seconds and is used in batches as input for the fine-tuning process. During fine-tuning with 70% of the dataset (i.e. training set), the model weights are fine-tuned with 10% of the data (i.e. validation set) based on the validation loss. After fine-tuning but before evaluation, the output data and reference text are post-processed to calculate the metrics during the evaluation fairly and accurately. Lastly, the final evaluation of the model is based on the last 20% of the data (i.e. test set), where the ASR performance is assessed with the Word Error Rate (WER) and the laughter detection is assessed with the F1-score, recall and precision. This process is visualised in Figure 3.1.

3.3 Datasets

In this research, two datasets were used, namely Switchboard and AMI. Both contain spontaneous speech but differ in their structure and speakers. Switchboard [21] contains 260 hours of structured dyad telephone conversations, in which specific topics were given to discuss. There were 2400 conversations from 302 male and 241 female speakers from the United States (i.e. all had a US accent). In total, there were 11 513 laughter occurrences. The AMI corpus [28] contains 3 scenarios, Edinburgh, IDIAP, and TNO. These multi-party recordings contained English accent variations and a lot of overlapping speech, as there were four speakers. The audio files were mixed headsets with 4 respective reference texts (i.e. per individual speaker). These scenarios were meetings, which were free to discuss anything and less structured. In total, there were a 100 hours of audio recordings. The mixed headsets were chosen instead of individual headsets due to computational constraints, but also because there is less times it is quiet with soft background voices, which is quite ambiguous data. This could cause hallucinations, which Whisper already suffered from. There were 16 524 laughter occurrences in total in the dataset.

This data is split into a training set, validation set and test set, by using 70%, 10%, and 20% of the data respectively. This fits with the standard division strategy within the ASR and machine learning field of using 60 to 80%, 10 to 20 % and 10 to 20% respectively, and fits with similar fine-tuning research that also used Whisper and a derivative of HuBERT (i.e. Wav2vec2), namely XLS-R [74]. The latter used the same split as is used in this research. All data has been randomly divided over the three datasets within their divisions, by first shuffling all the data with Random [126] (i.e. for reproducibility random seed 42 was used) and then dividing it according to their split.

3.4 Creating Input For Fine-tuning

The input unit for the fine-tuning process is a 30-second chunk consisting of input features and processed reference text. As such, after the division of the datasets, the reference text files are pre-processed and chunked together with the audio files, which are then tokenised and features are extracted. After this, all chunks are normalised, ensuring all None or 0 values and small chunks are excluded.

3.4.1 Pre-processing

For Switchboard, there are separate reference text files per speaker, but one audio file was used for both, hence the reference files are combined based on the word time stamps. Also for AMI a combined audio file is used (i.e. the mixed headsets), so the individual reference texts are also combined. These text files are all pre-processed to ensure a fair comparison of all the different data inputs. They were pre-processed for: exclusion (i.e. extra notes like "silence" and exclude truncated words); lowercase; contractions (e.g. I don't know); corrections (e.g. dunno becomes don't know); and the inclusion of the laughter event as a new token (i.e. <laughter> for Whisper and < for HuBERT, which is based on their respective architecture and vocabulary type).

3.4.2 Feature Extraction

The audio and reference files are then separated into chunks of 30 seconds, as this is the optimal length for both Whisper and HuBERT to train on (i.e. to balance the amount of data and context needed). The first chunk starts at the beginning of the audio file from where there is actual talking. This is found by checking the timestamp of the first word in the matching reference file. The same is the case for finding the end of the last chunk in the audio file. The audio of these chunks are then put through the processor, using the feature extractor. Whisper converts the raw audio into a log-mel spectrogram, which is windowed and normalised. HuBERT does not do this but allows a CNN to directly learn from the data, which outputs features that are embeddings of acoustic patterns in context.

3.4.3 Laughter Token Insertion

To ensure that laughter is added during the training process, a new token has to be added to the vocabulary of the tokenizer. It should be clear that when the word "laughter" is spoken, this is different from the laughter event. It is irrelevant what token this is, as long as it is new in the vocabulary and not to be confused with another meaning, as this can be post-processed to any symbol that shows it means Laughter Event after fine-tuning. Therefore, for Whisper the <laughter> token is added. For HuBERT, due to the Connectionist Temporal Classification (CTC) head working with individual letters as a vocabulary, simply the symbol < is added as the laughter token in the training process. In this work, during post-processing after fine-tuning and evaluation, both of these tokens are converted to "<Laughter>" for the end users' interpretation and further consistency when discussing the laughter event in this thesis. However, the specific label does not affect the performance, so it can be changed into anything else, as relevant to its final application.

3.4.4 Chunk and Batch Creation

The last step is to create the input for the model. First, a full list of chunks is created. These combine the 30 seconds of input features with the corresponding processed labels that are tokenised. These chunks are normalised to exclude all empty reference text chunks or audio data chunks.

Based on the set batch size, as dependent on the hardware, an x number of chunks will go through a collate function. A custom collate function is used here, initialised as a Data Class. With this input, it will process the data and form a batch. In this function, the data is squeezed and padded with the use of the processor. Moreover, the padding is replaced with a special value (i.e. - 100), as designed by the model, so the loss ignores the padding. This is necessary to ensure the model does not take unimportant information into account. In addition, only for HuBERT, there is a mask for 15% of the input (i.e. as is the functionality for the *Hidden Unit* part of BERT) and in the collate function this is filled. This is the final input for the Trainer in the fine-tuning loop.

3.5 Fine-tuning loop

The fine-tuning loop is where the model is retaught on the new data, by using the training set. With the help of the Trainer class [127] and its Training Arguments functions from HuggingFace [1], the loop changes the weights of the model to learn from the new data, while still remembering the pre-trained data. The learning rate and weight decay play a big part here, as they decide how fast and well the model learns. In addition, every 1/5th epoch the training process is validated to see if the loss is still going down and the data is not over- or under-fitting, by using the validation set.

3.5.1 Trainer

The final batch is used as input for the Trainer class [127], which is a loop that does the fine-tuning. The Trainer can be configured with TrainingArguments, which gives many options to fully optimise the Trainer per model. These configurations are largely dependent on the hardware and model used.

The full configuration can be found in Appendix A, but the most important parameters are:

- The batch size is set between 2 and 4 (i.e. small, as the models are very big);
- The *gradient_ accumulation_ steps* and *eval_ accumulation_ steps* is set to 8 (i.e. to minimise memory usage for the GPU);
- The dataloader_pin_memory and fp16 are set to true (i.e. for optimised GPU usage);
- The evaluation strategy is set to steps, where the evaluation step frequency is approximately 5 times per epoch and the warm-up steps are set to 5% (i.e. as is standard) of total steps
- Early stopping is implemented, evaluated on the evaluation loss, with a patience of 20 epochs and a threshold of 0.001. The best model is then chosen based on the highest F1-score in those last 20 epochs.
- The learning rate, weight decay and number of epochs are also set here, the best values for which will be researched in this thesis

3.5.2 Validation

During initial testing, every 1/5th epoch the model is evaluated on the validation set. This is set to once per epoch when fine-tuning for more epochs. This does not necessarily benefit the model but it is to see how the fine-tuning is performing (e.g. overfitting). These models are evaluated on WER, F1-score and validation loss. The validation loss should be going down, otherwise too many epochs are being run. At the same time, it is interesting to see if the model is also actually learning more words (i.e. lower WER) or laughter events (i.e. higher F1-score), or if it is learning something else (e.g. background noises).

To make a fair comparison between HuBERT and Whisper on AMI and Switchboard, with a focus on identifying laughter events, the output of both models have been post-processed extensively just before evaluation. This will match the already pre-processed reference text, as mentioned in 3.4.1. This pre-processing was done by visually inspecting the reference and hypothesis texts and finding all the discrepancies, and by keeping in mind the standard steps in NLP [128].

All hypothesis text outputs have been post-processed with:

[1] All models and main fine-tune functions come from Huggingface, see [127]

- Removed Non-ASCII Characters, except for < > (i.e. to ensure correct "<laughter>" token in the case of Whisper and "<" for HuBERT)
- Convert to Lowercase
- Capitalise Pronoun "I" when on its own
- Correct Contractions: ensure I'm, I've, I'd are correct
- Convert Numbers to Words (e.g. 8 becomes eight)
- Remove Punctuation Marks
- Manual Text Corrections (i.e. to ensure both outputs align, e.g. "ok" to "okay")

3.6 Evaluation

After the model is fine-tuned, the model has to be evaluated with the post-processed hypothesis as explained in 3.5.2. This is done by using the final test set. This is divided into two parts, namely the lexical analysis meant for the ASR part of the research, and the laughter recognition analysis. The lexical analysis consists of the Word Error Rate, by using the alignment operations hits, insertions, deletions and substitutions as calculated by the Jiwer package [129]. The laughter recognition analysis uses the Jiwer package [129] to find the *laughter* alignment operations hits, insertions, deletions and substitutions. These alignments are used to calculate the F1-score, Recall and Precision. In addition, a laughter substitution vs deletion balance, substitution word content analysis and general laughter alignment analysis are conducted with these alignments.

3.6.1 Lexical (ASR) Analysis

The WER is the most common metric used in any kind of ASR research. It considers the error rate of all words in the text. The values range from 1 to 0, where the best is an error rate of 0 (i.e. no errors). All calculations of the WER are based on the word alignment, as calculated by Jiwer [129], which compares all words in the reference and hypothesis transcription. Based on the alignment of these transcriptions, each word will fall into a category: hits (i.e. True Positive), insertions (i.e. False Positive), deletions and substitutions (i.e. the two together would be False Negative).

To create a realistic example, all punctuation from the sentence is also removed, and the text is normalized (e.g., capital letters are removed). More examples and explanations about using Jiwer can be found at [130]. An example:

`Reference text: "i am writing a thesis about laughter recognition is this not fun <laughter>"`

`Hypothesis text: "i` *`will`* `writing a thesis` **`too`** `about *** recognition is this **` *`gun`* `<laughter>"`

Then the alignment per word is: hit (i), substitution (am -> will), hit (writing), hit (a), hit (thesis), insertion (-> too), hit (about), deletion (laughter ->), hit (recognition), hit (is), hit (this), deletion (not ->), substitution (fun -> gun), hit (<laughter>).

For this text, we can therefore conclude there is 1 insertion, 2 substitutions, 2 deletions and 9 hits. The number of True Positives is 9, False Positives is 1 and False Negatives is 4.

This can now be used to calculate the WER. There can be differentiated between two kinds of WER in this thesis: the WER including "laughter" as a word, and without. To distinguish the two during discussions, the WER including laughter will from here on be named WER_L, while the WER from just the ASR performance will remain the WER. The formula of the WER and WER_L is:

$$WER = \frac{number\_of\_errors}{total\_number\_of\_words} = \frac{insertions+deletions+substitutions}{total\_number\_of\_words\_in\_reference}$$

$$WER\_L = \frac{number\_of\_errors}{total\_number\_words\_including\_laughter} = \frac{insertions+deletions+substitutions}{total\_number\_words\_and\_laughter\_events\_in\_reference}$$

In the example sentence above, the WER and WER_L would be calculated as follows:

$$WER = \frac{1+2+2}{13} = 0.385$$

$$WER\_L = \frac{1+2+2}{14} = 0.357$$

3.6.2 Laughter Analysis

The laughter analysis consists of several parts, each evaluating how well the model detects and handles laughter events. First, the key performance metrics F1-score, Recall and Precision are considered, which provide a quantitative analysis of the model's accuracy in identifying laughter. The laughter alignment operations hits, insertions, deletions and substitutions used to calculate these metrics are also analysed both individually and in relation to one another to gain insight into the models' behaviour.

F1-score, Recall, Precision

The most optimal metric for this research to evaluate how well the model was able to identify correct laughter events is the F1-score. This is calculated by using recall and precision. Accuracy was not considered, because laughter events are relatively little present (i.e. 16 524 laughter events out of 11 147 783 total words for AMI, so 0.15%, and 35 771 laughter events out of 36 460 854 words for Switchboard, so 0.10%), which would result in a large number of True Negatives. This would create a false positive bias to its performance. Therefore, F1-score is better suited for these imbalanced classes. In addition, this was also used in relevant studies that fine-tune Out Of Vocabulary (OOV) words [131], [132]. This F1-score is calculated by using the Jiwer package [129] too, which finds the laughter alignments.

The F1-score is a combination of recall and precision, which are also interesting to consider individually. Both give their own insights. Recall measures the model's ability to find all relevant instances, whereas precision measures the model's accuracy in finding only the relevant instances. These are calculated with the formulas:

$$LaughterRecall = \frac{TP}{TP+FN} = \frac{hits}{hits+(deletions+substitutions)}$$

$$LaughterPrecision = \frac{TP}{TP+FP} = \frac{hit}{hit+insertions}$$

$$LaughterF1score = 2 * \frac{precision*recall}{precision+recall}$$

Keep in mind, that in this case the hits, deletions, substitutions and insertions are only relevant for the laughter instances. To give a more comprehensive understanding of this, a new example[2] is generated (i.e. excluding the normalisation and processing of the text, for clarity's sake):

`Reference text: "Hi! <laughter> I am not <laughter> here for a quick chat, but rather <laughter> a good cup of coffee <laughter>. How are you doing? <laughter> I hope the tea is nice? <laughter>"`

`Hypothesis text: "Hi! ** I am not <laughter> here for a quick chat, but rather` *okayay* `a good cup of coffee <laughter>. How are you doing? ** I hope the` *<laughter>* `tea is nice? <laughter>"`

Here the laughter alignment (i.e. ignoring all text besides laughter) for each laughter event in the text: deletion (laughter ->), hit (laughter), substitution (laughter -> okayay), hit (laughter), deletion (laughter ->), insertion (-> laughter), hit (laughter). Therefore, we can conclude there is 1 laughter insertion, 1 laughter substitution, 2 laughter deletions and 3 laughter hits for this text. This gives 3 True Positives, 1 False Positives and 3 False Negatives. When calculating the laughter recall, precision and F1-score of this text, this results in:

$LaughterRecall = \frac{3}{3+(2+1)} = 0.5$

$LaughterPrecision = \frac{3}{3+1} = 0.75$

$LaughterF1score = 2 * \frac{0.75*0.5}{0.75+0.5} = 0.60$

Further Laughter Analysis Insights

Not only the WER and F1-score can be calculated with the laughter alignment operations, but an extra insight into the laughter event hits, insertions, deletions and substitutions can also be helpful in further improving the model and understanding how it works. This quantitative analysis considers a balance between substitutions and deletions, and specifically the change in this balance when fine-tuned with or without laughter in the training set, can indicate if the model recognises the laughter as speech or ignores it.

In addition, a qualitative analysis can be done, by doing a laughter substitution word analysis can be conducted. The specific replacement (i.e. substitution) of the laughter event is considered and compared with zero-shot, fine-tuned without the laughter label in the training set and fine-tuned with laughter. This can indicate where or how the model makes the mistake and how it inherently handles laughter when it does not know the laughter event label.

Moreover, the balance between all four laughter operations can be observed throughout training, to analyse how the models behave. This can reveal the additional potential for further training, offering a more insightful assessment than only relying on the F1-score.

3.7 Implementation Details

To implement all the different steps, software was used for programming and particular hardware was used to enable the heavy models to run. Details on these specifics are described below.

[2]This example also clearly demonstrates the difference between WER and WER_L, as the WER would be 0 (i.e. no errors in the lexical part), but the WER_L (i.e. including errors in laughter events) would be WER_L = (1+2+1)/32 = 0.125.

3.7.1 Software

The programming language used in this thesis is Python, as this is the most accessible and common language in the field of Machine Learning. Several libraries and packages were used to accomplish the final program.

General Packages

Several packages were used throughout the program, that ensured the data was dealt with. To start with, the glob package [133] was used to easily find the reference files to relevant audio files. The os package [134] was used to manipulate paths and open files. Then, the random package [126] was used to ensure the files were first shuffled randomly but in a reproducible way (i.e. using seed 42), before dividing the files into their subsets. During the creation of chunks stage, where all data has already gone through the feature extractor and tokenisation process, the chunks are saved to a pickle file with the pickle package [135]. This way initial processing only has to be performed once.

Throughout the code, the NumPy package [136] was used several times. This consists of mathematical functions, arrays, and general computing tools. In addition, the gc package [137] (i.e. garbage collector) was also used throughout the program, to ensure the cache was deleted, as the model already takes up a lot of memory.

Training Models Packages

The main library used in this thesis is the Transformers from Huggingface [138]. For Whisper, the configuration class for this model is WhisperForConditionalGeneration [11], as this includes a Language Modeling (LM) head and is often used for ASR . The configuration class used for HuBERT is the HubertForCTC [12], as this can handle sequence-to-sequence issues of variable lengths (i.e. input vs output), as is the case for fine-tuning with ASR. For both, the *from_pre_ trained_ model* function is used so the model weights are loaded. The Trainer package [127], which also comes from the Transformers library, is responsible for the training loop. For fine-tuning with Huggingface models, this is the most common class to use [123], [139]. It is optimised for Transformer models, such as HuBERT and Whisper.

The underlying deep learning framework of Transformers is PyTorch [140], often called Torch. It provides all the computations, optimisations and manages the training loop (e.g. the dataloader and updating weights).

Evaluation

In the evaluation, the main package used is the Jiwer package [129]. With this package, not only the WER can be calculated, but its alignment function shows what happens to the words compared to the reference text (i.e. equal, insertion, substitution or deletion). In addition, it preprocesses the hypothesis and reference text, to ensure spacing and basic processing are the same for both texts. In this preprocessing step, the repackage [141] (i.e. regular expressions) is also used, to ensure words are spelled the same (e.g. "uhm" and "uh").

During validation and at the end of evaluation, many metrics and the hypothesis texts are saved. The metrics per epoch are saved to a comprehensive csv file, with the csv package [142]. The final hypothesis texts are saved to a json file together with the metrics, with the json package [143]. To analyse and read the json file, the pandas package [144] was used.

AI Assistance

All literature reviews, data processing and analyses were conducted independently from any AI assistance. In addition, all content and ideas in this thesis were constructed without the help of AI. However, during the programming of this thesis, the Python bot of ChatGPT 4.0 [145] was often consulted on errors and aided the programming process. In addition, Grammarly was used to check the grammar and spelling in this thesis.

3.7.2 Hardware

The models were run on the UT EEMSC-HPC Cluster [146]. Several nodes, which contain different types of GPUs, were used during testing and programming. However, only one kind has sufficient capacity to run the full models with all the data. This was the NVIDIA A40/48G GPU, with either 128 or 256 RAM capacity and 64 cores.

Chapter 4

Results

Many separate experiments have been conducted to comprehensively and fairly compare the performance of the fine-tuned models on the datasets that include laughter. The experiments could be divided into three categories, namely:

1. **Finding Appropriate Parameters**
2. **Model Baseline Evaluation**
   (a) Zero-shot
   (b) Fine-tuning without laughter
3. **Fine-tuning with laughter**
   (a) Training
   (b) Final results

First, the initial tests are completed, where the model parameters are optimised and analysed. The appropriate parameters investigated were the weight decay and learning rate for HuBERT and Whisper, which were found by fine-tuning using the AMI dataset for 20 epochs. To decide on the appropriate parameters, the WER_L and F1-score are used. These parameters are used as guidance to facilitate the correct parameter calibration of further experiments. After completing the initial stage, the second phase focused on exploring the models' baseline without laughter labels. Here the zero-shot model and the fine-tuned model without laughter labels are investigated. This baseline considers the laughter and ASR performance of the model. The last experiments consider the main fine-tuning experiments including the integration of laughter. The fully fine-tuned models are investigated throughout training, by considering the laughter alignment operations, F1-score and WER_L (i.e. WER including laughter). Here, the best-performing models are chosen. Finally, these best models are applied to the test set, where the full potential of both models in terms of ASR performance and laughter detection performance are evaluated.

All experiments are first conducted using the main AMI dataset. This dataset includes enough speech and laughter labels but is relatively small due to computational and time constraints. The second and third categories are then also compared to the performance of the models using the larger and cleaner Switchboard dataset.

4.1 Finding Appropriate Parameters

To find the appropriate parameters used for the two models in the main experiments, initial tests are conducted over 20 epochs. Both the WER_L (i.e. which indicates the performance of the ASR of the model including the laughter event as a "word") and the F1-score (i.e. which represents the performance of the laughter event recognition) were used to indicate the overall best-performing parameters. This research endeavours to incorporate laughter events while optimally maintaining a favourable WER_L. Therefore, the highest F1-score is chosen as the primary criterion on the most appropriate parameter, unless the WER_L is substantially greater ($<$ 0.01 relative difference) *and* the associated F1-score is not significantly smaller ($<$0.01 relative difference) for another parameter combination. These values are motivated by the high variability of the data and the impact WER also has on the F1-score when further fine-tuned (i.e. a higher WER will, most likely, also help with a better F1-score).

The HuBERT model is relatively small, but very prone to overfitting, due to the nature of its architecture (i.e. the CTC head with letter vocabulary). Therefore, a higher learning rate and weight decay were used initially. The other tests were decided based on these results, increasing or decreasing learning rate and weight decay where necessary. As shown in table 4.1, the highest F1-score for the HuBERT model on AMI with 20 epochs are for a learning rate of 1e-3 and weight decay of 0.3, at 0.234. While the WER_L was slightly lower in a few other parameter settings, the F1-score was zero in these experiments and therefore irrelevant in this research. Therefore, the final parameter selection tries to balance both metrics in the most optimal way possible. In addition, when comparing to the lowest WER_L of 0.309 (i.e. with an F1-score of 0), the relative WER_L difference was just 0.061, indicating that this variance is unlikely to have a significant impact. The second highest F1-score was at 0.088, with a WER_L of 0.313. Compared to the best parameter setting, it has a 0.004 lower WER_L, so a relative WER_L difference of 0.049 but a 0.146 lower F1-score, so a relative F1-score difference of 0.623. This relative difference of the F1-score significantly outweighs the relative difference in WER_L, hence the best parameter setting remains the one with the highest F1-score.

		Weight Decay			
		0.01	*0.1*	*0.3*	*1*
	3e-3	1.000 | 0.000	1.000 | 0.000	1.000 | 0.000	0.986 | 0.000
	1e-3	1.000 | 0.000	1.000 | 0.000	***0.329 | 0.234***	0.414 | 0.022
Learning	*3e-4*	1.000 | 0.000	0.309 | 0.000	0.313 | 0.088	0.3333 | 0.056
Rate	*1e-4*	0.319 | 0.000	0.318 | 0.000	0.314 | 0.001	0.317 | 0.000
	1e-5	0.363 | 0.000	0.356 | 0.000	0.348 | 0.000	0.356 | 0.000
	1e-6	0.400 | 0.000	0.410 | 0.000	0.400 | 0.000	0.400 | 0.000

TABLE 4.1: The WER_L (left) and F1-score (right) for HuBERT [12] fine-tuned on the AMI dataset [28] with 20 epochs for parameters weight decay and learning rate.

Whisper is quite a lot larger, but the Distilled Whisper model is much smaller (i.e. 6 times) than the standard Whisper Large V2. It is also more flexible and less prone to overfitting, hence a lower initial test weight decay and learning rate were chosen, as seen in table 4.2. The final best-performing parameters for Whisper on the AMI dataset with 20 epochs were at a learning rate of 1e-4 and a weight decay of 0.001. This time, several results were close to each other, where the highest F1-scores and lowest WER_L were present for learning rate 1e-4.

The highest F1-score was at 0.466, but with a WER_L of 0.325, at a weight decay of 0.003. However, at a weight decay of 0.001, the F1-score was 0.462 (i.e. the second highest) and the WER_L was 0.304. The relative F1-score difference is just 0.009, while the relative WER_L difference was 0.065. This relative difference in WER_L is significantly more than the relative F1-score difference, hence we neglect the difference in the F1-score and opt for the best-performing combination. There were no other F1-scores that came close to these F1 scores, so no other combinations were considered.

		Weight Decay			
		0.0001	*0.001*	*0.003*	*0.01*
	3e-4	0.365 | 0.430	0.372 | 0.417	0.371 | 0.428	0.363 | 0.438
Learning	*1e-4*	0.302 | 0.452	***0.304 | 0.462***	0.325 | 0.466	0.301 | 0.450
Rate	*1e-5*	0.319 | 0.420	0.321 | 0.433	0.324 | 0.425	0.337 | 0.400
	3e-6	0.356 | 0.370	0.356 | 0.343	0.345 | 0.350	0.360 | 0.334
	1e-6	0.356 | 0.310	0.356 | 0.310	0.365 | 0.290	0.360 | 0.314

Table 4.2: The WER_L (left) and F1-score (right) for Whisper [11] fine-tuned on the AMI dataset [28] with 20 epochs for parameters weight decay and learning rate.

4.2 Model Baseline Evaluation

To set a baseline for the models, they must be investigated without the integration of laughter. This is conducted in two parts. First, it is important to realise how fine-tuning impacts the models' behaviour. The models will not just learn this new "laughter" token, but also the accents, jargon and other sounds in the datasets. Therefore, it is interesting to see how well the pre-trained models perform without fine-tuning, showing their potential robustness in various environments. This is called a zero-shot setting. However, this specific research focuses on the impact of laughter on fine-tuning. Therefore, how the models' performance is affected will also be considered by fine-tuning the models, but by excluding laughter labels from the training data.

To assess the ASR performance of the models, the WER and WER_L are considered. Additionally, to understand how the model reacts to instances of laughter that actually occurred but were not labelled in the training data, we analyse the laughter alignment operations hits, substitutions, insertions, and deletions using a test set that does include laughter labels. These fine-tuning processes use the same parameters as investigated in 4.1.

4.2.1 Zero-Shot

First, the model is not fine-tuned at all. Therefore, the model does not know the "laughter" event label, as it has not been trained to know it. Table 4.3 shows the lexical performance of the models, in terms of WER_L and WER and laughter performance operations, so how the model handles the laughter events, by considering the laughter hits, substitutions and deletions. For the latter, a balance among the three laughter performance operations was also taken into account to assess how the models handled the laughter event without prior knowledge of it. Logically, there were always 0 hits, as it simply does not know the label. As can be seen in the table, the laughter events were substituted and deleted almost 50/50 on AMI for HuBERT, with a high WER_L of 0.450 and WER of 0.442. On Switchboard with HuBERT, this division was less equal, with approximately 1/3 deletions and 2/3 substitutions, with a lower WER_L of 0.325 and WER of 0.317. Whisper has many more substitutions, with approximately 1/3 deletions and 2/3 substitutions on AMI, and approximately 1/6 deletions and 5/6 substitutions on Switchboard. The WER_L and corresponding WER for AMI and Switchboard was the highest from Whisper, with 0.558 and 0.550 on AMI and 0.370 and 0.362 on Switchboard respectively.

Model	**Dataset**	**Fine-tuned**	**WER_L**	**WER**	**Hits**	**Substitutions**	**Deletions**	**Alignment Operations Balance**
HuBERT	AMI	no	0.450	0.442	0	1438	1509	0/49/51 %
		yes	0.317	0.308	0	1034	1914	0/35/65 %
	SWB	no	0.325	0.317	0	2065	1240	0/62/38 %
		yes	0.211	0.206	0	1768	1537	0/53/47 %
Whisper	AMI	no	0.558	0.550	0	2035	913	0/69/31 %
		yes	0.416	0.407	0	2320	628	0/79/21 %
	SWB	no	0.370	0.362	0	2743	562	0/83/17 %
		yes	0.161	0.151	0	3213	86	0/97/3 %

Table 4.3: Model metrics based on the performance of zero-shot and fine-tuning without laughter in the reference text.

To understand how the model reacts in the instances where laughter was present in the test set, while it does not know the laughter label (i.e. as it was not fine-tuned with the laughter label), a pattern analysis was conducted on the substitution words per model, dataset and experiment type. A comprehensive and extensive overview of the most common word patterns and frequencies substituting the laughter events can be found in Appendix B, tables 7.1, 7.2, 7.3 and 7.4. In table 4.4, a simplified overview of Appendix B showing the most noteworthy patterns identified can be found. When investigating the substitution words, as shown in table 4.4, it can be seen that Whisper has a clear tendency to combine and repeat words. On the AMI dataset, where WER_L was high at 0.558, there was a repetition of part words, mainly smaller words. Most notably, the word "the" or variants were repeated very often. In tables 4.7 and 4.8 a clear example is shown of how these variations would be expressed. On the Switchboard dataset, with a lower WER_L of 0.37, more and longer words were formed and combined. Again, the most striking pattern was the high presence of words containing "th". For HuBERT, a very different pattern can be observed. For AMI, with a WER_L at 0.450, often individual letters were substituted instead of laughter. Sometimes words were created, but these were very often misspelled. For Switchboard, with a lower WER_L of 0.325, there were still many individual letters present. However, quite often specifically the letters "h" and "a" were present. It was sometimes even spelled "ha ha ha", as is how the English language types out the laughter sound. These letters, i.e. part "haha", were also often combined with another word, making one word in total, e.g. "hahahhadiferent". Moreover, quite often complete words were substituted instead of laughter, differing from one-syllable words e.g. "like", to longer words e.g. "sometimes".

4.2.2 Fine-tuning No Laughter

Secondly, the model is fine-tuned, but still without laughter present in the training set. As shown in table 4.3, both the WER_L and WER improved by at least 0.1, so 10%, each time over no fine-tuning. For HuBERT, the substitution/deletions balance leans more towards dele-

Models	Dataset	Zero Shot	Fine-tuned Without Laughter	Fine-tuned With Laughter
Whisper	AMI	the word "the"	the word "okayay" & words including "th" letters	words + "<laughter>" combined
	SWB	words including "th" letters	"you"+"t"-word (e.g. youto) & "yeah"+"t"-word (e.g. yeahtoo)	words + "<laughter>" combined
HuBERT	AMI	individual letters	words & some individual letters	words + "<" (i.e. laughter) combined
	SWB	variations on "haha"	words including "th" letters	words + "<" (i.e. laughter) combined

Table 4.4: Most noteworthy laughter substitution word patterns identified for HuBERT [12] and Whisper [11] on AMI [28] and Switchboard (i.e., SWB) [21], per fine-tune type.

tions, showing for AMI approximately 2/3ds are deletions and for Switchboard this is approximately 1/2. For Whisper, the substitution/deletion balance leaned even more towards substitutions, where for AMI the division was approximately 2/9 deletions and 7/9 substitutions and for Switchboard the deletions were approximately 1/38 and 37/38 substitutions. The lowest WER_L and WER were for Whisper on Switchboard at 0.161 and 0.151 respectively, improving approximately with 0.21 WER_L and WER over the non-fine-tuned WER_L and WER.

The substitution words for HuBERT when fine-tuned changed quite a lot for AMI and Switchboard, as shown in Table 4.4. The main substitution on AMI was now more fully formed words, with still some individual letters. On Switchboard, there was no specific pattern of "haha" present anymore. Instead, the main pattern identified was the presence of words including "th". For Whisper, the pattern of a high presence of "th" remained on both datasets. Additionally, on AMI an overwhelming number of occurrences of the word "okayay" could be found.

4.3 Fine-tuning With Laughter

Lastly, the models are fine-tuned with laughter based on the parameters earlier found in 4.1. This would mean that HuBERT had to be fine-tuned with a learning rate of 1e-3, but after fine-tuning for more than 20 epochs, this resulted in overfitting behaviour. Therefore, after examining the performance of HuBERT using different combinations of parameters close to those found, a learning rate of 1e-4 was chosen. As a result, all experiments are conducted with a learning rate of 1e-4. Whisper uses a weight decay of 0.001 and HuBERT a weight decay of 0.3.

First, the behaviour of Whisper and HuBERT fine-tuned on the AMI dataset was analysed by training for 200 epochs. After these two experiments, early stopping was implemented on the other experiments (i.e. for the previously mentioned experiments for AMI and all experiments on Switchboard), due to time constraints. Next, the behaviour of the fine-tuned models on Switchboard was investigated. Examining the behaviour of the models is beneficial for assessing the models' stability and reliability. It also provides insights into the requirements for deploying the models effectively on new datasets. Lastly, a final evaluation is done based on the best models per dataset. This shows the optimal performance the models can reach when fine-tuned with laughter. The models' behaviour during training is assessed using the validation set and the final evaluation of the best model is done with the test set. Both are based on the laughter event alignment operations (i.e. hits, insertions, substitutions, and deletions), WER_L and F1-score. During the final evaluation, there is a distinction between WER without laughter (i.e. WER), WER with laughter (i.e. WER_L),

4.3.1 Models' Behaviour With AMI (Training)

The behaviour of the models can be analysed over many epochs; here 200 epochs are chosen due to time limitations. As was the case when finding the parameters, this behaviour is analysed using the AMI dataset. The validation set is used for evaluation during training. Therefore, these are not the final results, but only an indication of the performance during training.

Whisper's behaviour is reasonably steady throughout. The graph showing the laughter alignment operations (i.e. substitutions, deletions, insertions, hits) per epoch can be found in Appendix C, table 7.1. All laughter event alignment operations of Whisper exhibit high volatility in the first 25 epochs but stabilise later. Overall, the model quickly reaches its best performance and does not improve significantly. Especially in the last 20 epochs, barely any fluctuations can be observed for any laughter detection operations. The number of substitutions is the most variable initially. These are also generally the highest present, at a count of around 1250. The hits start very low with many deletions, but this already steadies after 15 epochs. The least volatile are the insertions, which do not vary much.

The HuBERT model is very unstable throughout, on all fronts, but with few extreme outliers to the curves. The figure showing the laughter alignment operations per epoch for HuBERT over 200 epochs can be found in Appendix D, in table 7.2. In the first 27 epochs, there are 0 laughter event hits and insertions, with many deletions (i.e. at 1100) and many substitutions (i.e. at 420). In the first 80 epochs, the model improved the most, showing an upward curve for hits and a downward curve for deletions. The number of substitutions increases relatively little, with a count of 200, til epoch 80. The addition of insertions starts once the model also has some hits, increasing to a count of 180 insertions til epoch 80. At epoch 80, the curve flattens and only improves slowly. The model is very volatile throughout, often fluctuating with a count of 20 to 40, with outliers of up to 150, for all operations. The number of extreme outliers decreases nearing the end of the fine-tuning process, at around epoch 150. Notably, the insertions and hits graphs exhibit similar patterns (i.e. outliers and curves at similar positions), indicating that the addition of the total increase of laughter event outputs is correlated; specifically, a large number of total laughter insertions results in more hits but also more insertions.

The F1-score and WER_L per epoch for both Whisper and HuBERT fine-tuned on the AMI dataset is visualised in Figure 4.1. Whisper shows a very volatile WER_L score at the beginning, which settles down after 45 epochs, with a few significant peaks between epochs 55 and 65 and epochs 110 and 140. The F1-score has outliers throughout the training that go up and down, but 3 high peaks at 26, 71, and 140. However, the mean remains similar for both metrics throughout the linear graph. HuBERT demonstrates that the WER_L is extremely steady and exhibits significant exponential decay in the first 15 epochs by 12%. From there, the WER_L holds steady at approximately 0.315, fluctuating by 0.002 per epoch. However, it takes 27 epochs for HuBERT to recognise one laughter event, which means the F1-score is also 0 up to epoch 27. Up to epoch 75, the F1-score increases quickly with many outliers but steadies slightly from there on. Reaching the 200 epochs, the graph becomes less volatile and the number and severity of the outliers decrease.

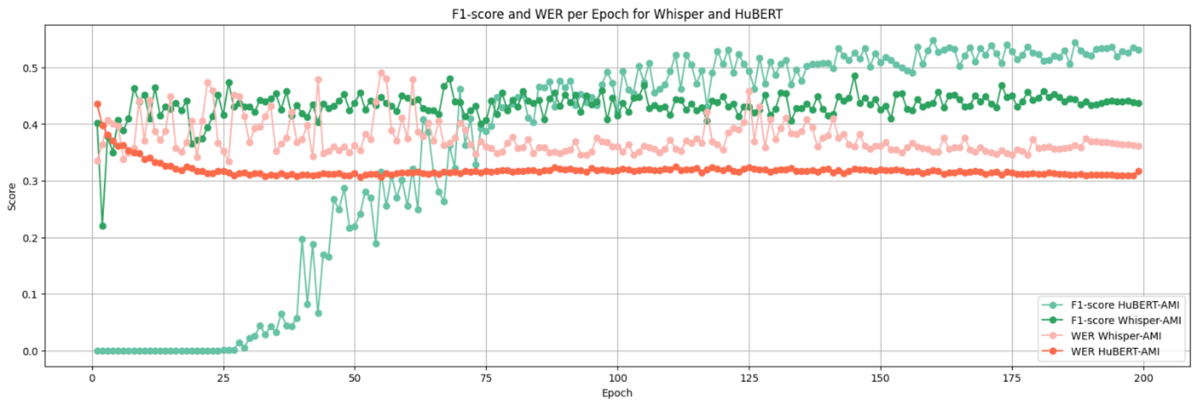


FIGURE 4.1: The WER_L and F1-score for Whisper [11] and HuBERT [12] per epoch fine-tuned on the AMI [28] dataset.

4.3.2 Models' Behaviour With SWB (Training)

The experiments to investigate the model's behaviour for 200 epochs were conducted on the AMI dataset. This dataset is quite noisy with a lot of overlapping speech. The Switchboard dataset is cleaner with just two speakers but has more data. Due to time and resource constraints, a soft early-stopping algorithm was integrated into all the experiments that followed after this one. This ensures it would not needlessly continue or start to overfit the data, yet still consider the local minima that might be reached instead of the global minima. For Whisper, the optimal performance is already reached after fine-tuning for one epoch. The model becomes more stable the longer it runs, but the performance is not improving significantly. However, at the beginning, there are a few local minima, which means the early stopping needs some patience before knowing when the model actually performs optimally. For HuBERT, while WER_L did not vary much, the F1-score increases quickly for the first 80 epochs. The curve flattens from there. However, the overall F1-score still increases slightly for another 30 epochs. Due to the varying nature of the F1-score and WER_L (i.e. a constant WER_L but with an F1-score that has many unexpected outliers), the validation loss is chosen as the criterion to see if the model is still learning. A high patience of 20 is used, to ensure no local maxima are reached. After stopping, the model with the highest F1-score in the last 20 epochs is chosen as the best model for the final evaluation.

Whisper on the Switchboard dataset already stopped after 39 epochs, showing the last epoch had the highest F1-score. This graph can be found in Appendix E, table 7.3. The number of laughter hits already started high at around 2800. After 10 epochs, it reaches its highest point at around 3100 hits. From here, the model stabilises and does not further improve significantly. The situation is similar for the number of substitutions, where they decrease in the first 10 epochs, but their performance does not improve. The deletions also go down the first epochs, but with even less, as there were little deletions initially. The insertions stay consistent throughout the fine-tuning, except for one peak at 330 from around the mean of 100 at epoch 3.

HuBERT on the Switchboard dataset stopped at 32 epochs, also with the highest F1-score at the last epoch. This fine-tuning process is visualised in Appendix F, table 7.4. In the first 5 epochs, no laughter events are recognised. From 5 to 10 epochs, all laughter event detection operations steadily improve. All operations continue to improve but much less extremely. All operations behave volatile and sporadic throughout fine-tuning, especially the number of hits and deletions. The number of hits still increases significantly, with an outlier at epoch 32 jumping to 1725 hits from 1423 at epoch 31. The number of deletions and substitutions both go down slightly from epoch 31 to epoch 32. Throughout the fine-tuning process, most times the number of hits has a peak that goes up, and the number of deletions has a peak that goes down. The number of substitutions is less volatile with fewer outliers.

The WER_L and F1-score for both models throughout fine-tuning on Switchboard are visualised in Figure 4.2. Whisper shows a very smooth and stable graph for both F1-score and WER. From the first epoch, the F1-score is at around 0.90 and the WER_L is around 0.17. The F1-score varies very little, fluctuating by approximately 0.001 on average. The WER_L graph is slightly more volatile, fluctuating by approximately 0.010 on average. No further outliers are seen during the rest of the fine-tuning of Whisper. However, HuBERT shows great improvement throughout. In the first 5 epochs, the F1-score is at 0 with no laughter event recognition. In the next 5 epochs, the F1-score increases in big steps. From epoch 10 to 15, the F1-score still increases slightly but very gradually. For the rest of the epochs, the performance is more volatile and performance goes up and down. In the first 5 epochs, the WER_L decreases with 0.8%, after which the WER_L performance stays steady at 0.20%, fluctuating less than 0.01%.

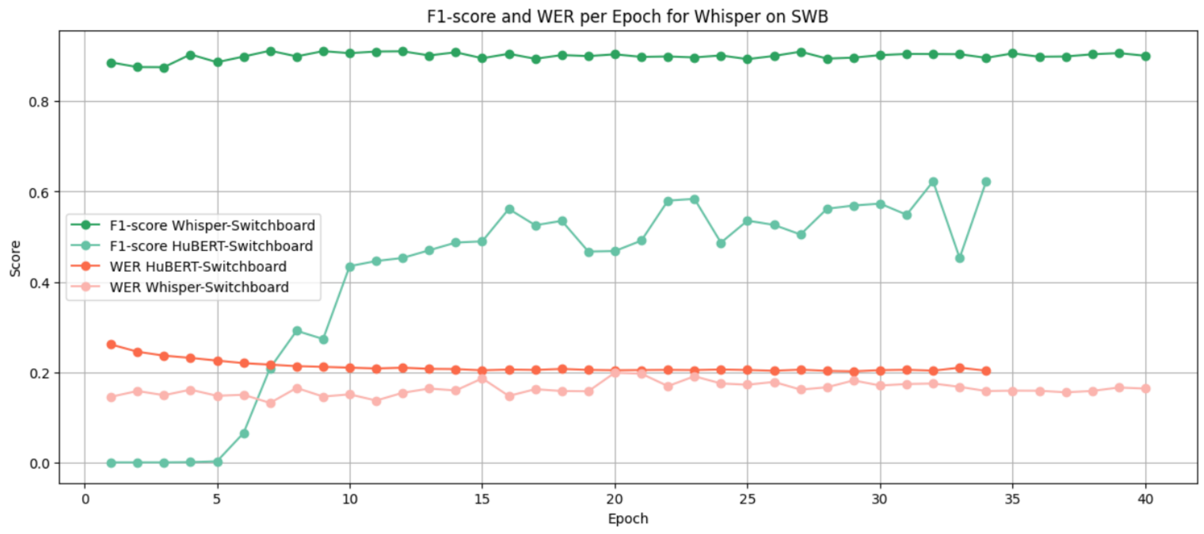


FIGURE 4.2: The WER_L and F1-score for Whisper [11] and HuBERT [12] per epoch fine-tuned on the Switchboard (SWB) [21] dataset.

4.3.3 Final Model Performances

The comparisons of the performance of the model's behaviour while fine-tuning were based on the validation dataset, which was just 10% of the full dataset. The final evaluation, with the highest-performing model, was done with the test dataset using 20% of the dataset. These results are visualised in table 4.5.

HuBERT has the most deletions, with approximately the same number for AMI and Switchboard, at around 830. As a result, the number of substitutions was also very similar, at around 1 160. The WER_L, WER and F1-scores were also quite far apart between AMI and Switchboard, with an approximate 0.10 improvement of Switchboard for all three metrics. This gives a relative WER_L difference of 0.562, a relative WER difference of 0.571 and a relative F1-score difference of 0.146. The number of hits was therefore also the highest on Switchboard.

Whisper's results varied more, where it had just 1 176 laughter hits on AMI, but 3012 laughter hits for Switchboard. Due to the high number of hits, there were few substitutions (i.e. 2 439 on AMI and just 564 on Switchboard) and even fewer deletions (i.e. 440 on AMI and 26 on Switchboard). The number of insertions did not follow this trend, where Whisper had 63 insertions on AMI and more insertions, namely 74, on Switchboard. The WER_L and F1-score were significantly better for Whisper on Switchboard than on AMI, with a difference of approximately 0.46 F1-score and 0.2 WER and WER_L.

For all metrics considered, the best-performing model on AMI is HuBERT, while the best-performing model on Switchboard is Whisper. Overall, the very best performance was reached on Switchboard with Whisper. All results on the test data matched the results based on the validation set but scaled higher.

The most noteworthy substitution words for when the fully fine-tuned model did not register the laughter event as laughter, but did identify it as speech, is visualised in table 4.4. This showed that the most prominent pattern for all models was the combined new word of a laughter event together with an existing word or a letter.

Model	**Dataset**	**Epochs**	**Hits**	**Substitutions**	**Deletions**	**Insertions**	**F1**	**WER_L**	**WER**
HuBERT	AMI	200	1365	1180	857	374	0.531	0.317	0.311
	SWB	32	1725	1147	811	135	0.622	0.203	0.198
Whisper	AMI	200	1176	2439	440	63	0.444	0.357	0.348
	SWB	39	3012	564	26	74	0.901	0.163	0.161

TABLE 4.5: Performance metrics F1-score, WER, WER_L and laughter alignment operations for fine-tuning on HuBERT [12] and Whisper [11] .

4.4 Experiment Overview and Breakdown

In total, there were 12 main experiments. These were divided into initial testing, baseline testing and full fine-tuning including laughter. The first revealed some behavioural aspects of the models, the second showed the state of the art applied to the datasets as processed for the final experiments, and the last gave the final results of this research regarding the lexical and laughter detection performance of the models. From all of these results, an overview is created, as shown in 4.6. Here, the best metrical results over all models and datasets are summarised. In addition, tables 4.7 and 4.8 take a deeper look into what these metrical numbers are based on, by visualising an example sentence from the reference text including a laughter event, together with all hypothesis text outputs of the models per experiment type.

4.4.1 Best Models

The metric performance results of these experiments are summarised in table 4.6, including the *best* results of the parameter tuning on AMI with 20 epochs (i.e. note: for HuBERT on AMI this is with weight decay 1e-3, while the rest of experiments use weight decay 1e-4). This table is divided per dataset, then per model and then per experiment. This visualises an overview of what model performed best in each scenario in terms of lexical and laughter integration performance.

As is shown, for both HuBERT and Whisper, all results are best on Switchboard. The overall best-performing model when considering its performance for laughter event recognition was Whisper fine-tuned on Switchboard, with an F1-score of 0.901, recall of 0.836 and precision of 0.976. The WER_L was at 0.163 and WER at 0.161, which was the second lowest WER_L out of all experiments, the lowest being at 0.162 WER_L and WER at 0.151 for Whisper fine-tuned on Switchboard without laughter. This is also the highest relative WER difference between WER_L and WER, at 0.073.

The best-performing model when considering laughter event recognition on AMI was HuBERT. When just considering F1-score the best model was when HuBERT was fine-tuned with laughter on AMI. It also had the highest recall, at 0.401. However, the precision (i.e. 0.785 for HuBERT on AMI with laughter with 200 epochs) for all other experiments, was much higher, at 0.910 or more. This includes all experiments of Whisper on AMI, but also the precision of HuBERT fine-tuned on AMI with laughter for just 20 epochs. The lowest WER_L for AMI was with Whisper fine-tuned on AMI with laughter for just 20 epochs, at 0.304.

Dataset	*Model*	*Fine-tuned*	*Epochs*	*F1-score*	*Recall*	*Precision*	*WER_L*	*WER*	*Relative WER diff*
AMI	HuBERT	Yes + L	20	0.234	0.134	0.910	0.329	-	-
		Yes + L	**200**	**0.531**	**0.401**	0.785	0.317	0.311	0.019
		Yes	45	-	-	-	0.317	**0.308**	0.029
		No	-	-	-	-	0.450	0.442	0.018
	Whisper	Yes + L	20	0.462	0.305	0.948	**0.304**	-	-
		Yes + L	200	0.444	0.290	**0.949**	0.357	0.348	0.026
		Yes	57	-	-	-	0.416	0.407	0.022
		No	-	-	-	-	0.558	0.550	0.015
SWB	HuBERT	Yes + L	32	0.622	0.468	0.927	0.203	0.198	0.025
		Yes	32	-	-	-	0.217	0.206	0.053
		No	-	-	-	-	0.325	0.317	0.025
	Whisper	**Yes + L**	**39**	**0.901**	**0.836**	**0.976**	0.163	0.161	0.012
		Yes	50	-	-	-	**0.162**	**0.151**	0.073
		No	-	-	-	-	0.370	0.362	0.022

TABLE 4.6: All experiments summarised: the performance of fine-tuned with laughter (Yes + L), fine-tuned without laughter (Yes), and non-fine-tuned (No) for HuBERT [12] and Whisper [11] fine-tuned on AMI and Switchboard (SWB), comparing laughter detection metrics and WER.

4.4.2 Sentence Level Elucidation

Per experiment, there was a metrical output (i.e. table 4.6), showing the performance, but this came from the calculation based on comparing a lengthy output hypothesis text and reference text. In table 4.7 one example sentence taken from the reference text of the AMI dataset and in table 4.8 one example sentence is taken from the Switchboard dataset. is used to illustrate how some of the models' output and previously established patterns (i.e. Appendix B, tables 7.1, 7.2, 7.3 and 7.4) manifest within the sentence per type of experiment and model. This provides a more detailed view of what occurs at the sentence level behind the calculations.

The sentence from AMI in table 4.7 has uncommon words (i.e. ninja and Japan) in it and two people laughing. The "um" word also complicates matters, as this was said together with laughing. As a result, there are many mistakes in this sentence and only laughter or similar was registered once for HuBERT.

Type	*Model*	*Sentence*	*Laughter Operation*
Reference	-	"if you want to present your prototype go ahead uhoh this is it ninja home made in japan <**laughter**> <**laughter**> um there are a few changes weve made"	-
Zero-shot	Whisper	"if you want to present your proot go o this is mademade in japanenes**therethereothethein therethere** are a few weved wellto"	Sub
	HuBERT	"if hu want to pres an your prototyp go homing jap ******* um there are a few jane made"	Del
Fine-tuned Without Laughter	Whisper	"if you want to present your prototype with with wellhoh this it o made um edge japan **um um um um okayay** while wow a lookay ofve made"	Sub
	HuBERT	"if you want to present your prototype go ahead this is it ninjo homade joa pen ******* um there are a few changes weve made"	Del
Fine-tuned With Laughter	Whisper	"if you want to present your prototype ahead hoh this is it fromo cat in your peg **um um um um um** um therethere are a few changes ve made"	Sub
	HuBERT	"if you want to present your prototype go ahead thats it nino hmk ma your pen <**laughter**> um there are a few changes wive made"	Hit

Table 4.7: Reference sentence from the AMI dataset including laughter per dataset together with the Whisper [11] and HuBERT [12] output (i.e. hypothesis text) per type of experiment and laughter alignment type (i.e. hit, substitution, or deletion).

The sentence from Switchboard in table 4.8 has two people speaking relatively quickly and over each other, but with simple vocabulary and little background noise. The word "yes" is also said while there was partial laughter present, making it harder to transcribe for the models. However, the laughter itself was clearer (e.g. louder and not two people laughing overlapping), which is also shown by both Whisper and HuBERT correctly recognising laughter here.

Type	*Model*	*Sentence*	*Laughter Operation*
Reference	-	"oh i love that show yeah do you <**laughter**> yes that is great yeah its fun and then theres a new one that started out that ive caught occasionally a couple of times called good and evil i think"	-
Zero-shot	Whisper	"i i love that show yeah do you **yesyes** yes that is great yeah it fun and then theres a new one that started up that ithat i a i couple of timestimes called good and evil i think"	Sub
	HuBERT	"i love that show ya **yo yaya**ya great its fun and then theres a new won that started out that it clout occasion a couple of times called good and evil i think"	Sub
Fine-tuned Without Laughter	Whisper	"oh i love that show yeah do you **yes** yes yeah that is great yeah its fun and then theres a new one that started up that ive caught a a couple of times called good and evil i think"	Sub
	HuBERT	"oh i love that show yeah do you ******* yes that is great yeah its fun and then theres a new one thats started up that ive caught a cancel a couple of times caled god and evil i think"	Del
Fine-tuned With Laughter	Whisper	"oh i love that show yeah do you yes <**laughter**> yes that is great yeah its fun and then theres a new one that started up that ive caught a a couple of times called good and evil i think"	Hit
	HuBERT	"oh o i love that show yeah do you <**laughter**> yes that is great yeah its fun and then theres a new one that started up that ive caught a caucel a couple of times caled god and evil i think"	Hit

TABLE 4.8: Reference sentence from Switchboard including laughter per dataset together with the Whisper [11] and HuBERT [12] output (i.e. hypothesis text) per type of experiment and laughter alignment type (i.e. hit, substitution or deletion).

Chapter 5

Discussion

This research aimed to integrate laughter event transcription into pre-trained E2E models by fine-tuning Whisper and HuBERT. To explore this performance based on the results and facilitate drawing comprehensive conclusions later, this section is organised around the research questions. This thesis had one main objective, summarised by the research question: *How effectively can laughter transcription be integrated into established high-performing ASR systems through the fine-tuning of pre-trained End-To-End models?* This question was divided into three distinct sub-questions to comprehensively evaluate the integration of laughter into the systems, while also taking into account their baseline lexical performance. The sub-questions address: the annotation of laughter by the state-of-the-art, adding laughter transcriptions to the fine-tuning pipeline, and the impact of said fine-tuning on the linguistic component.

First, the focus is on examining how the models currently transcribe laughter. This is based on which words, letters or phrases are present in the hypothesis text (i.e. models output) at the location of laughter in the reference text. This is considered for the non-fine-tuned models and the fine-tuned (i.e. zero-shot, so without laughter annotations in the labelled training data) models. Next, the performance of the fully fine-tuned models is considered, based on the WER without laughter, WER including laughter (i.e. WER_L), F1-score, and qualitative analysis of the laughter event operations hits, substitutions, deletions, and insertions. Lastly, the impact of fine-tuning the models on the ASR performance is investigated by comparing the behaviour of the non-fine-tuned, fine-tuned without laughter, and fully fine-tuned models. Due to time limitations, the AMI dataset was chosen as the main dataset in this research. Switchboard was used to compare the performance of the models, as this was a bigger and cleaner dataset.

5.1 Baseline Omitting Laughter Transcription

The first sub-question is: *To what extent, if any, do the E2E ASR models Whisper and HuBERT currently transcribe laughter in conversational language?* The focus of this section is therefore on finding how the models behave without having learned the "laughter" label during training. When examining the inherent nature of the models, two types of "model baseline" can be considered when evaluating the laughter performance. The first considers how the models behave in terms of laughter recognition in a zero-shot setting, so without any fine-tuning. Secondly, The models will be fine-tuned on AMI and Switchboard, but without laughter present in the labels of the training dataset. Therefore, it cannot learn "laughter" exactly, but the ASR performance should improve (i.e. by learning accents and noises present in the dataset). The evaluation is done with laughter present in the test set, and this is compared to the models' output from the training set. This is done to identify how the models behave by themselves without specifically learning laughter while being aware of the location of the laughter events.

It is challenging to investigate how the models register laughter events without an exact label, as it does not yet know the "laughter event" label. To explore how the models handle laughter events without fine-tuning for laughter, it is considered whether the model can:

1. Identify the laughter event as part of speech.
2. Recognise the laughter event as something humans can interpret as laughter.

To achieve this, the substitution and deletion balance is analysed for the first point. This can show if the model interprets laughter as part of speech (i.e. inserts a word, so substituting laughter) or not (i.e. removes it). For the second, a laughter substitution word analysis is conducted, showing what that word is and if it can be interpreted as laughter or if there are other patterns present. These analyses are first conducted on the main dataset (i.e. AMI), after which the results on the Switchboard dataset are also discussed and compared.

5.1.1 Zero-Shot

The non-fine-tuned model, also called zero-shot learning, best shows how the model would behave on new data. This is when the models are used as-is and show the flexibility and robustness of the models. It has not learned the concept of "laughter" yet and does not know anything about the data. However, the question is, what does the model output at the places of laughter? Therefore, zero-shot learning can show what happens to these laughter events that are present in the validation dataset. HuBERT and Whisper responded very differently to the laughter event on the AMI dataset and Switchboard, each showing their inherent nature through their output.

HuBERT

First, the HuBERT model is considered with the AMI dataset. To find how well the model recognises laughter as part of speech, the laughter substitution and deletion balance is considered. HuBERT has an approximately 50/50 division of substitutions and deletions. This is a decent proportion of substitutions, but not high. This shows it does not identify a laughter event as something which is part of speech very easily without any fine-tuning. This is probably due to two parts. The CTC head and letter vocabulary HuBERT works with ensures the sounds do not need to recognise part of words, which would be a more complicated sound pattern. It will easily output a guess of some letters in the spaces. However, this also shows the inflexible nature of HuBERT. It has been trained on audiobooks and has not encountered any spontaneous speech including laughter in its training data before.

From here, the specific laughter substitution words can be considered, to show if the model has any kind of interpretation of those events that lie close to laughter. The laughter substitution word analysis showed that the main output was individual letters or miss-spelled words. This is due to this same letter vocabulary HuBERT operates on. No further patterns could be identified which could be recognised as laughter. Overall, this shows that zero-shot HuBERT did not do well in transcribing laughter instances on the AMI dataset.

Whisper

Also for Whisper, first, how well the model recognises laughter as speech is considered. Whisper has an approximate split of 1/3 deletions and 2/3 substitutions. This shows that it often does recognise the laughter event as text that should be transcribed; so as a potential word. This is most likely due to its flexibility, where Whisper has been pre-trained on a lot of data, including spontaneous speech with laughter, so it can recognise the laughter as some kind of sound from the speaker.

The laughter substitution word analysis shows that Whisper fills it in with some standard small words, e.g. "okayayay" or even puts a small word next to the word that was said before "andthatthat". This is the case due to the vocabulary of Whisper; which consists of part- or co-words (i.e. partial words that often belong together). It will recognise a part-word (e.g. "and") and adds it together with a similar sounding word (e.g. "thatthat"). Alternatively, sometimes there was laughter present while a word was spoken (i.e. speech-laughter), often during smaller words like "okay" or "yeah", from which the model creates a combination with laughter like "yeahyeah".

This suggests that Whisper has a high potential to recognise laughter events, as it often identifies them as speech. This will increase the chance of successful fine-tuning, as it already registers these events. However, as it currently is not fine-tuned for laughter specifically, it struggles to generate a coherent output that can be interpreted as laughter on the AMI dataset.

Comparison to Switchboard

HuBERT and Whisper performed similarly on the Switchboard dataset as on the AMI dataset. The substitution/deletions balance leaned more towards substitutions for both models, showing that on this dataset it could recognise a laughter event more easily as part of speech.

Interestingly, the substitution words were quite different in this dataset. For HuBERT, most words included "haha" or at least started with "a" or "h". This could be because the dataset is cleaner, so it can more easily identify these sounds. It just does not know what laughter is, so the closest sounds are the "haha". This is also how it is written in many languages, showing these are the letters of the alphabet most associated with these sounds. This demonstrates that the cleaner dataset improves HuBERT's ability to both recognise laughter events as part of speech and more frequently interpret them as something recognisable as laughter.

The Whisper model on Switchboard was more similar to the results on AMI, but with an increase of words starting with "th". This could be due to, or a combination of, its similar spectral characteristics (e.g. noisy profile or "h" sound), or with the better ASR performance on Switchboard it has learned there is a high probability of a "th" word there, or due to the soft sound of "th" it might be a default guess. The reason that Whisper did not output any letters or phrases that could be recognised as laughter, is probably due to its specific vocabulary. It is word-based, not letter-based, so it does not have any labels representing laughter sounds in it. This shows that on Switchboard, an increase in recognisable patterns was found compared to AMI, but still, nothing concretely related to laughter could be identified.

5.1.2 Fine-tuned Without Laughter

Another type of baseline for the model is where it learns all ASR elements of the data by fine-tuning on the lexical (i.e. words) parts of the dataset, but while excluding laughter, omitted these labels from the training dataset. This approach assesses the model's ability to incorporate the laughter token without the possibility of confusing it with other elements or variables that are new to the model, like noise. However, it must be noted that the model will also learn from the training data that there is no label every time a laughter event occurs, hence, this will result in the model recognising laughter as speech less in the first place. To comprehensively assess the performance of the fine-tuned models baseline, the models' performance is immediately described relative to the zero-shot model. First, the models' performance is assessed when fine-tuned on AMI, then they are compared to fine-tuned on Switchboard.

First, the impact of fine-tuning is assessed. When considering the AMI and Switchboard datasets, it can be seen that indeed both Whisper and HuBERT have learned from the data, as the validation loss kept going down and the WER_L and WER for both models improved. Secondly, how well the models recognise a laughter event as speech is considered. For AMI, the effect on the substitution and deletion balance compared to the zero-shot setting was minimal, with a slight increase in deletions for HuBERT and an increase in substitutions for Whisper. A possible explanation for this could be that HuBERT has learned that this sound does not correspond to one of its letters in the vocabulary (i.e. deletes more); while Whisper becomes even more flexible and more easily recognises the pattern or accent of the laughter as part of the person speaking. This results in a slight improvement in Whisper's ability to recognise a laughter event as part of speech, but a slight deterioration for HuBERT. Thirdly, how well the laughter event can be recognised as laughter is examined. The substitution word analysis for HuBERT reveals that it responds very similarly to that of the zero-shot; with mainly individual letters and almost words. However, some small and correct words were added, showing it has learned these words during its fine-tuning and HuBERT tries to apply them. For Whisper a similar trend to zero-shot also persists, namely the presence of words with "th". However, the word "okayay" was also present extremely often. This could be due to a high presence of "okay" in the dataset, and this is its backup word that sounds similar to laughing. Alternatively, the acoustic laughter sound does not follow the same linguistic structure Whisper is trained on. However, "okay" could be a word that it has learned is sometimes input more randomly in a sentence than the average word (i.e. as people sometimes use this word while thinking or interrupting). Therefore, due to the longer sound than the word "okay", the longer version of okay (i.e. okayay) could be input, but with linguistically correct placement. This shows that neither model improved much in interpreting a laughter event as laughter when fine-tuned on the AMI dataset.

On the Switchboard dataset, again a similar trend can be seen as in the zero-shot setting for the substitution/deletion balance. There are even more substitutions relatively, the highest being for Whisper on Switchboard at the lowest WER_L of 0.161. This shows that it does recognise these sounds as something the speaker said. The substitution words for Whisper are also very similar, showing an increase in "th" words or "t" words. However, the substitution words are very different for HuBERT. HuBERT no longer outputs "haha", but also seems to recognise the spectral characteristic patterns of the "th" words, similar to Whisper.

5.1.3 Overall Baseline

Two baselines were analysed to form an overall baseline that considers two aspects. Zero-shot focuses on the importance of flexibility and robustness, while fine-tuning without laughter increases overall performance on the data, showing its potential without factors like noise. For HuBERT, the ability to recognise laughter as part of speech went down for both datasets when fine-tuned, while for Whisper this went up (i.e. fewer substitutions). Therefore, it shows that by fine-tuning, HuBERT learns that these individual sounds are not part of the data, while Whisper's flexibility increases and by learning the individual sound patterns of the speaker, it recognises laughter as part of that.

For both baselines and datasets, Whisper had the highest relative number of substitutions. The highest recognition rate was for Whisper on Switchboard, at 97% laughter substitutions and 3% deletions. This also corresponded to the lowest WER of 0.151 and WER_L of 0.161. This shows the flexibility of Whisper and its potential to recognise laughter as part of speech. However, while some patterns were identified in the substitution word analysis (i.e. words with "th"), there were no laughter substitutions that could be identified as something resembling laughter for Whisper. The only time laughter was substituted with anything that could be identified as a laughter event, was for HuBERT on the Switchboard dataset. This was with "haha", or a variant of this. Still, the fact that patterns were identified in the substitution words for Whisper is a positive sign, suggesting the model does relate a laughter event to a specific type of substitution word. Therefore, together with the large number of laughter substitutions and low number of deletions (i.e. best at recognising laughter as part of speech), the Whisper model shows the largest potential to learn laughter events when fine-tuned with laughter labels in the dataset.

However, when considering the direct implementation of the models without fine-tuning with laughter, it will depend on the environment which model performs best. While Whisper might be better at recognising the laughter event as a sound belonging to the speaker, especially useful on a noisy dataset like AMI, HuBERT has the most comprehensive output that could be recognised as laughter on the cleaner dataset. Therefore, when applying a model to a new (i.e. zero shot) but clean (e.g. little noise and overlapping speech) environment without fine-tuning with laughter, HuBERT is the best model to apply, if the focus is on recognising laughter. A person using the system would recognise "haha" or similar to laughter. However, if the model is applied to a noisy environment, as is often the case in real life, neither model could comprehensively show that it interpreted a laughter event as anything recognisable as laughter.

5.2 Fine-tuning With Laughter

The main part of this research focuses on the integration of laughter transcription into the standard ASR. This considers the second sub-question, which is: *To what extent, if any, can the E2E ASR models Whisper and HuBERT be fine-tuned on conversational speech including laughter annotations to improve the accuracy of laughter transcription?* The pre-trained models are fine-tuned on data that includes lexical and laughter annotations. To evaluate these models and their implementation feasibility, it is crucial to not just investigate the laughter detection elements, but also the model efficiency. These are considered and discussed per model, in this order:

1. Foundational model behaviour
2. Training resource demands
3. Final performance on laughter event detection

The first is evaluated during initial testing (i.e. finding weight decay and learning rate), by doing a general analysis of the model based on its behaviour. The second is done during training, by considering the four laughter alignment operations hits, deletions, substitutions and insertions, F1-score and WER with WER_L. Lastly, the final performance is considered after training, so when evaluating on the test set. Here, the F1-score, recall and precision are examined. The AMI dataset is chosen as the main dataset to fine-tune on and discussed first, but the results of the models are later also compared to each other and that of the Switchboard dataset.

HuBERT

During the initial testing, the general model performance could be abstracted. The initial testing phase on the AMI dataset allowed not only for the investigation of the weight decay and learning rate, but also for an overview of the model's general performance to be gathered. It showed that the model quickly climbed to a WER_L of 1.0 and an F1-score of 0. The output showed that the model was very prone to overfitting and catastrophic forgetting, as seen by the decreasing training loss, quickly increasing WER and end output of just padding (i.e. indicating silence, which was often present). This is due to the letter-based vocabulary that HuBERT functions on, unlike a partial word vocabulary like Whisper has. These letters are short sounds and therefore very accent-specific. It needs to learn the speech patterns of a person before it can accurately recognise the speech of various speakers. Otherwise, it will apply the specific vocabulary it has learned to another speaker and will not recognise what other speakers are saying anymore. For example, one person's pronunciation of the letter "G" might sound like another person's "J", confusing the model.

To explore how the model behaves while fine-tuning and how well it learns over many epochs (i.e. 200) on AMI, the laughter alignment operations are considered together with the F1-score during training. For HuBERT, it takes 27 epochs before the first laughter event is identified correctly (i.e. hit), and most of the further learning occurs in the first 100 epochs. Interestingly, what can be seen when comparing the hits and deletions graph of HuBERT during those 100 epochs, is that the model learns quite conservatively. While insertions show a slight increase and substitutions fluctuate too, the number of deletions more significantly decreases and the number of hits correspondingly rises with equal intensity during the full training process. This indicates that the model does not necessarily make more guesses when it encounters laughter more frequently during continued training (i.e. which would lead to an increase in insertions). Instead, it learns to place these guesses more accurately, resulting in an increase in hits without a corresponding rise in substitutions. This means that the precision of HuBERT is high throughout training. All laughter alignment operations show the curves flatten after about 100 epochs, but are still volatile. Fine-tuning for 100 epochs is quite a lot, showing the model needs a lot of data and iterations before it learns how to appropriately apply the new token. The volatile graph also indicates that the model is still learning with each new iteration. This is logical when considering the letter vocabulary again, as laughter can sound like several letters and HuBERT has to learn that this combination of sounds is the new laughter token. After the 100 epochs, the number of hits still slightly increases, showing it might further improve with more epochs. The F1-score over the last 100 epochs also supports this, as it was still increasing slightly. This shows that while this model is conservative in training, resulting in high precision, it takes a long time to train on new data and, hence is expensive to train. However, it also shows that it still has the potential to grow. Due to time limitations, this was not further investigated beyond the 200 epochs in this research.

However, to properly examine the efficiency of training to achieve high laughter recognition, it is also necessary to examine how HuBERT learned the new laughter token in comparison to the rest of the alphabet tokens (i.e. ASR performance). The WER_L was already very stable after 27 epochs and did not improve or fluctuate with further fine-tuning. This would suggest that the model has learned the ASR features of the dataset quickly, but it takes longer to introduce a new token to HuBERT. This is because the model is already pre-trained on the alphabet tokens. Fine-tuning these tokens takes minimal effort, as it only needs to learn the details, while the laughter token has to be learned from scratch. In addition, the laughter events are present relatively little compared to most of the rest of the alphabet. This demonstrates that the model is much more efficient in improving the ASR performance, but takes much longer specifically for laughter. This is likely the case for any new token or acoustic non-lexical sound added to the vocabulary.

The final laughter performance output of HuBERT on the test set (i.e. while the performance analysis above was based on the smaller validation set) showed a final F1-score of 0.531. This F1-score is not high at all, where about half of the laughter events are incorrect or missed. This represents a moderate performance of correct laughter recognition, but with room for improvement. The F1-score is a balance of recall and precision, where recall was 0.401 and precision was 0.785. The higher precision shows it was better at identifying correct positive identifications (i.e. quality), than the recall, which indicates the relevant laughter instances the model has found (i.e. quantity). Therefore, HuBERT found it hard to identify all the laughter instances, but the ones it did find, were often correct. This shows that HuBERT is not good at detecting laughter events for the AMI dataset, but when it does, its precision is reasonably high.

Whisper

The initial testing was also completed for Whisper on the AMI dataset, revealing the behaviour of Whisper in various scenarios. Many of the parameter combinations that were considered performed close together, where the Whisper model consistently avoided overfitting or catastrophic forgetting as with HuBERT, only performing less well after those 20 epochs. This indicates and supports the known flexibility of Whisper. It has been pre-trained on data that includes a lot of accents, languages, noise levels and even laughter. In addition, the ability of Whisper to remain stable across various parameter fine-tuning runs, shows that its flexibility might also reduce the need for extensive parameter tuning to achieve optimal result for a specific dataset. This is particularly useful when applied to new scenarios or settings.

When investigating the behaviour of Whisper during training, again 200 epochs were chosen. Most notably, the F1-score shows a stable graph very quickly. It is still volatile throughout the 200 epochs, but at 180 epochs this also disappears. This shows that while it is still learning the specifics for the laughter event recognition, it immediately has a lot of hits. This is due to the flexible nature of Whisper, as established during initial testing. Therefore, it can immediately recognise the sound but needs a corresponding label, which it finally gets during training. The most volatile graph is the number of substitutions, which has many fluctuations and outliers in the first 35 epochs. This is probably due to this same flexibility, where the model has so many sounds it can relate it to in its multi-lingual database, that it is able to make a guess but it takes longer to find the correct label. This indicates that Whisper is quick to learn the laughter token and is high in efficiency.

When considering the laughter detection performance compared to the ASR performance, the WER_L performance with the laughter F1-score is compared. Here, a similar trend can be seen. The model takes just as long to steadily learn the ASR performance as it does the laughter event (i.e. less outliers or improvements). However, after the 75 epochs, the WER_L graph is already less volatile than the laughter F1-score. This would suggest that the model is still learning more about the laughter event than the ASR performance. This could be due to its vocabulary, which consists of part- and co-words. It only has to tweak what it knows about the ASR to fit with a new scenario, while laughter is a new token completely. Moreover, Whisper is primarily optimised for linguistic content and will need additional data and training to accurately handle acoustic and non-verbal sounds like laughter. The model would need a distinct laughter pattern, but there are many types of laughter that differ per person and scenario, so this is easily confused with background noise. Also, one of the strengths of Whisper lies in its ability to use context from past and future samples. However, these linguistic rules that Whisper has learned during pre-training do not apply in the same way to acoustic features like laughter. This will confuse the model, making it hard to learn acoustic features and hinder its progress. Overall, this establishes that Whisper is a very appropriate model to apply in this situation, as it reaches its full potential for both ASR performance and laughter detection performance quickly.

The final results on the test set showed an F1-score of 0.444 after 200 epochs. This is not very high, as it shows moderately low accuracy for laughter even recognition. However, strikingly, the precision was very high at 0.949. The recall was very low, at 0.290. This shows that Whisper identified almost 95% of the total instances as positive correctly. However, as shown by the low recall, it also misses a lot of laughter events. This did not improve with more epochs. This is probably due to the noisy data and spontaneous speech, where there was a lot of difficult-to-recognise and overlapping laughter, with lots of types of laughter (e.g. speech-laughter and overlapping laughter). The high precision shows the potential of integrating Whisper into systems, where it will miss many laughter instances (i.e. low F1-score) but will not identify them incorrectly.

Notably, the F1-score for Whisper when fine-tuned with just 20 epochs was higher, at 0.462. This is due to the outliers it experiences, so it was "lucky" to stop at 20. This does show that the best Whisper model, so when to stop training, should be carefully selected. It does not need the full 200 epochs, but an early stopping algorithm should be integrated with a formula to select the best-performing model when fine-tuning this model on additional data.

Comparison

HuBERT and Whisper behave very differently while fine-tuning on AMI. Where Whisper does not improve much from epoch one with many laughter events, although volatile still; HuBERT needed 35 epochs to learn just one laughter token. After 100 epochs, this graph was finally steady, but the laughter hits and laughter F1-score still showed an increase until 200 epochs. This indicates that Whisper achieves its optimal performance quite rapidly, occasionally producing an outlier with a higher F1-score, whereas HuBERT requires at least 100 epochs, and potentially more, to realise its full potential. Interestingly, the F1-score for HuBERT (i.e. 0.531) was almost 10% higher than Whisper (i.e. 0.444). This shows that while HuBERT needs longer to fine-tune and is less efficient, it is better at identifying the laughter event in this noisy dataset. This is also due to its architecture, where HuBERT is better at identifying acoustic sounds than Whisper. However, for both models the F1-score was low. There are many reasons for this. For instance, laughter is quite a different sound (i.e. at a different pitch) from normal conversations. Moreover, laughter differs a lot per person and scenario; even more than normal speech. It is a very different sound (i.e. at a different pitch) that is very personal. The laugh per person is also variable depending on the context, think for instance of a polite chuckle or belly laugh. In addition, the laughter is not clean, as there is often laughed during speaking, while someone else is speaking, or two laugh at the same time. Diving deeper into the F1-score, there can be seen that both models had high precision and low recall, but especially Whisper. Whisper has a precision of 95%, showing that it only outputs laughter if its very sure.

Both models have also been fine-tuned on Switchboard, using the weight decay and learning rate found on the AMI dataset. The behaviour of HuBERT on Switchboard was quite similar, except for the fast increase in how quickly the performance improved. For instance, HuBERT fine-tuned on AMI took 27 epochs before just one laughter event was hit, while on Switchboard this was 5 epochs. In addition, the performance was also much better, with a final F1-score of 0.622 instead of 0.531. Most significantly, Whisper also performed much better on Switchboard, from the first epoch. The F1-score improved from 0.444 to 0.901. This improvement in both HuBERT and Whisper is due to the amount and quality of data, where Switchboard has 2.5 times as many hours of data (i.e. per epoch it already trains on more data). In addition, the dataset is cleaner. In this case, that means there is less complex speech (e.g. less overlapping speech and just two people talking), less background noise and a more structured setting (i.e. a telephone conversation with set topics, instead of a meeting). In such a setting, the laughter events are also cleaner, as they happen less "spontaneously" or during speech. They are often more of a polite chuckle or softer laugh.

Interestingly, a major trend across all datasets for all models could be seen that showed that after fine-tuning the main confusion point was a combination of laughter and a word. This was seen in the substitution word analysis. The main words left over as laughter substitutions were a combination of laughter together with a word or letter. This shows that while the model does not fully recognise it as a separate laughter event, it does still recognise it as laughter. This is the downfall of how laughter detection was registered in this research, as the alignment metric does not show the full picture. A human might even register this combination of laughter and speech the same way, but this metric does not register this as a correct hit. The reason that this happens is because there was laughed while a word was spoken. Also, especially for HuBERT, the model might misinterpret the laughter sounds as certain individual letters, thereby putting those letters in front or after the event. Moreover, especially for Whisper, the model may struggle to correctly align non-verbal sounds with the timing of words. To fill in those gaps where it expects a word, this could result in the combination of laughter with words. Lastly, due to this ambiguity of distinguishing between acoustic sounds and lexical patterns, the model might opt to output both words and laughter, to minimise errors. This means that the model has more potential to recognise laughter correctly when not just looked at the alignment, which should be investigated in future work.

5.3 Lexical Impact

After examining the behaviour of the models for the laughter event, it is also vital to investigate the impact they have on the lexical performance. This considers the last sub-question, which is: *How does fine-tuning the E2E ASR models Whisper and HuBERT on conversational speech with laughter annotations affect their overall performance in transcribing lexical elements?* This is done by considering the WER and WER_L to each other and the two datasets AMI and Switchboard.

First, the difference in performance between the two lexical metrics, namely WER and WER_L, are discussed. Then, the baselines (i.e. zero-shot and fine-tuning without laughter) are compared to each other and the final fine-tuning with laughter. The next focus is on analysing the key lexical differences of fine-tuning with and without laughter labels. Thirdly, a broader interpretation of the implications of WER is discussed. Lastly, the WER is compared to the state-of-the-art.

5.3.1 WER vs WER_L

For all experiments, the difference between the WER and WER_L was minimal. The highest relative WER_L to WER difference was at 0.73, with a difference of 0.10, between 0.161 and 0.151 respectively. This was for the fine-tuned excluding laughter Whisper model on Switchboard. The second highest relative WER_L to WER was 0.053, for the fine-tuned excluding laughter huBERT model on Switchboard. This is logical, as this is one of the lowest WERs with the cleanest dataset (i.e. easiest to transcribe) combination. In addition, there are no "<laughter>" word hits when fine-tuned without laughter. Therefore, the difference of WER_L, so with laughter words, and WER will be the highest. However, most relative WER_L to WER differences varied between 0.012 and 0.029. This is quite low, and given the consistency in the differences between WER and WER_L, similar conclusions can be drawn when comparing the lexical performance across all models using either metric. However, this only works for the relative difference in this specific work and as long as the same metric is consistently used in the comparison.

5.3.2 Baseline Comparison

The WER_L and WER were the highest for the fine-tuned models, showing a 0.1 or more, so 10% +, difference between all zero-shot and fine-tuned models. Both HuBERT and Whisper show that general (i.e. with and without laughter) fine-tuning improve the ASR performance, where the models learn the details of the specific dataset. This impact is likely so large due to the noisy nature of this dataset. As a result, the models need to be fine-tuned to learn this noise and transfer their pre-trained knowledge to this dataset.

However, the difference in performance between the fine-tuned models with and without laughter in the training data labels differed per model. Most notably, on AMI the best initial test had the even lower WER_L, at 0.304, for Whisper fine-tuned with laughter on just 20 epochs. This seems to be an outlier, as this is not reflected in any further testing. However, this does suggest some potential for improvement in Whisper. Perhaps with further fine-tuning of parameters and then strategically picking the best model Whisper could outperform HuBERT in the main experiments. For the main experiments on the AMI dataset, the lowest WER was by HuBERT at 0.308, with a corresponding WER_L of 0.317. This was for the fine-tuned model excluding laughter. However, the WER for the fine-tuned model including laughter was just 0.003 higher and the WER_L of the two were the same. This is so small that the difference in WER is neglectable and the laughter integration for HuBERT did not detriment the ASR performance.

On Switchboard, the lowest WER and WER_L were found for Whisper fine-tuned without laughter. Here, the WER_L was at 0.162 and the WER was at 0.151. The second lowest WER and WER_L were found for Whisper fine-tuned including laughter, at a WER of 0.161 and WER_L of 0.163. The difference in WER_L is just 0.001, which is neglectable. However, the fairer comparative metric here is the WER, which fairly shows the performance of the models. For these, there was a difference of 0.01 WER, so 10%. This difference is more significant, showing a significant increase in WER when fine-tuned with laughter. This can especially be seen when considering the relative WER difference, because the WER is quite low, at 0.066. When applying this model, it must be noted that while laughter recognition was the highest (with an F1-score of 0.901), this is detrimental to the integral ASR performance.

Notably, no further pattern could be found in whether fine-tuning with laughter had an impact on the WER. For 2 out of 4 experiments (i.e. Whisper with Switchboard and HuBERT with AMI), the WER increased and for the other half (i.e. Whisper with AMI and HuBERT with Switchboard), it decreased. The amount with which it decreased and increased also differed for every model and experiment. One study in state-of-the-art shows that the ASR performance was not detrimented when laughter detection was integrated [116], but also did not improve. This does not match the results in this thesis, although the difference in WER was minimal and in that study, the CER was used as the lexical performance metric.

5.3.3 WER Significance

When considering the WER of all models, including the best-performing ones, it is clear that the ASR performance is not very good. Both WER_L and WER are high across all models. Seeing as the WER was close to but lower than the WER_L every time, mainly the WER is considered here, to sketch a good picture.

For AMI, the lowest WER from the main experiments was at 0.308, which still means that only 2/3ds of the words are correct. This was the case for the HuBERT model when fine-tuned without laughter. For HuBERT this is less surprising, as it is known to be less accurate in terms of lexical performance when compared to other models, as it is not as flexible as Whisper and needs a lot of data.

For Switchboard, the WER improved significantly. For HuBERT, the WER improved from 0.308 to 0.198. Whisper performed the best, with the lowest WER established at 0.151. This is much better, due to the clean data, i.e. less complex ASR data, from Switchboard. It is generally agreed that a WER for machine learning in ASR of 0.2 or less is acceptable, but a WER of 0.1 to 0.05 is considered good quality [147]. To give a further impression, manual annotation by humans has a WER of approximately 5% or lower [148]. Therefore, while the WERs are acceptable to deploy and get comprehensible text output, they would still benefit from some further training and parameter fine-tuning.

5.3.4 Comparison WER to State-of-the-art

While the F1-score on laughter recognition was not studied before, the WER of HuBERT and Whisper has been researched previously on both AMI and Switchboard. In this thesis, the WER was notably high for both models when fine-tuned on AMI when using the generally accepted WER indications for ML in ASR [147]. However, it is also insightful to compare these results with the state-of-the-art. This will show if the WER is high due to aspects related to the dataset, due to the model, or potentially due to an oversight in this thesis.

HuBERT

The current research on HuBERT mainly explores variations of the model, which was improved in some way by including more languages or categorising (e.g., emotions). In addition, often different metrics were used that cannot immediately be compared (i.e. CER or EER).

However, when considering the AMI dataset, two studies tested their baseline with the WER. One study tested the HuBERT-Base on the AMI dataset with the Single Distant Microphone (SDM), resulting in a WER of 33.1% [149]. Another study differentiated between the Individual Headsets (IH) and SDM, where the baseline HuBERT-conformer-L model reached a WER of 13.52% and 32.01% respectively [150]. In this research, the WER reached a score of 0.442, so 44%, with zero-shot settings on the AMI dataset with HuBERT. This is higher than the HuBERT-Base model and the HuBERT-Conformer-L for either of the AMI dataset types. The HuBERT model used is the large-ls960-ft, which is better than the Base, but potentially less good than the inclusion of the Conformer, as it enhances the local acoustic features in speech. The AMI corpus in this research uses the Mixed Headset (MH), which is a mixture of the quality of the IH but includes more complex audio aspects like the overlapping speech of the SDM. Therefore, a direct comparison cannot be made, but taking the different models and data types into account, this thesis should have better results than the SDM (i.e. MH is cleaner) and HuBERT-Base (i.e. smaller). However, this is not the case for either.

For Switchboard, a WER of 9.8% was found for the fine-tuned HuBERT-large-ll60k in one study [151]. The WER of HuBERT fine-tuned on Switchboard in this thesis is 0.198, so 19.8%, which is significantly higher (i.e. 10%). This model is quite similar to the HuBERT-large-ls960 in this thesis, with the main difference being in what data it is trained on. The ll60k was pre-trained on 60 000 hours of unlabelled data from Libri-Light, while the ls960 was pre-trained on 960 hours of labelled LibriSpeech data. As a result, the ll60k is better at generalising and acoustics, while the ls960 is linguistically better with cleaner audio (i.e. read speech). This could be the reason that the ll60k performed better for Switchboard, as it is a spontaneous speech dataset.

Whisper

The original Whisper paper [10] also studied the ASR performance of the model on both AMI and Switchboard. The zero-shot settings on Whisper-large-v2 resulted in a WER of 16.7% on AMI IH and a 13.8% WER on Switchboard. This is much lower than what was found in this thesis, where Whisper zero-shot on AMI was at 0.550, so 55.0%, and on Switchboard at 0.362, so 36.2%. The model used in this thesis was Distilled Whisper, which should only increase the WER with up to 1% [152]. However, one study included evaluations by comparing the WER for Whisper-large-v2 and the Distilled-Whisper-large-v2 on both AMI and Switchboard [153]. This respectively showed a WER of 16.9% and 14.7% for AMI SDM, a WER of 36.5% and 33.9% for AMI IH, and a WER of 14.2% and 11.2%. This even showed an improvement for all datasets used in this research.

Substantiating Performance Shortcomings

Seeing as the WER is higher for both models on both datasets, it seems that the model configurations are not the reason for this. In addition, the training behaviour showed that the models were learning and the validation loss was decreasing. The main reason that the WER is higher in this thesis is most likely due to the processing steps of the data. In this thesis, all the data have been pre- and post-processed, but this could be done more thoroughly by investigating more carefully how other papers with better results have done this. Processing some frequent words could have been missed in this process (e.g. "you are" vs you're), which could significantly increase WER. As far as the author is aware, the data was processed as effectively as possible, having checked the texts manually and by following all the standard processing steps in NLP. Due to time constraints, this was not further investigated. Also, it must be noted that many papers do not transparently show their processing steps, which makes replication difficult. Moreover, after investigating the differences between this thesis and that of the models, the biggest difference that was found is how the data was chunked. All studies used a chunk size of 30 seconds, as was done in this research, but with a focus on chunking more optimally. There was ensured that the data was either chunked using a sliding window (i.e. Whisper) or ensuring the batch ended at the end of a sentence (i.e. HuBERT). This could influence how well the model can learn from the context and ensure there were no confusing partial words in a batch. This was not done in this thesis and could influence the performance.

5.4 Limitations & Recommendations

After summarising the key findings in this thesis, it is important to acknowledge its limitations to ensure a fair consideration of its results. Additionally, several recommendations for future work will be outlined. These include limitations of the model, the process in this thesis and further improvements that could be made.

Firstly, the models that were used were HuBERT-large and the Distilled-Whisper-Large-V2. For both models there are several similar alternatives, that could potentially outperform the ones used here. For instance, HuBERT has the versions large-ll60k (i.e. potentially more flexible in acoustic sounds) and xlarge (i.e. trained on even more data). Moreover, the Distilled Whisper was used instead of the Whisper. While research shows that Distilled Whisper even outperformed on the datasets used in this thesis, looking at the standard ASR performance, perhaps the original Whisper could be more flexible when considering such a specific acoustic token. The V2 was also used, due to tests with Whisper-V2 showing fewer hallucinations than V3, but it could be that the *Distilled*-Whisper-V3 did not have these hallucinations. Therefore, for future work, I recommend investigating the various models more before continuing with advancements in this research. This was not done in this research due to time and computational limitations.

Secondly, the state-of-the-art clearly shows the benchmark WER was much lower than in this thesis. Therefore, only a relative conclusion (i.e. comparing my own baseline results to improved WER and F1-score performance when fine-tuned) could properly be made in this thesis and not an improvement to the state-of-the-art. Some possible explanations for this have been identified in the Discussion, but this must further be tested and applied in continuing research. This may have also significantly affected the laughter event recognition process, as a better-performing model allows the models to focus on its errors (i.e. focus on laughter integration).

Moreover, some parameter fine-tuning was conducted. However, this was only done on AMI and subsequently applied to Switchboard. It is recommended that future work finds the correct parameters for the models on Switchboard too. Additionally, while the three main parameters (i.e. number of epochs, learning rate and weight decay) were investigated and some additional strategies were applied (e.g. early stopping and learning rate warm up time), there are many more that can affect performance. For instance, there could be looked into other regularisation techniques, freezing layers, dropout rate, loss function and optimiser selection. Due to time limitations, these were not investigated, but previous research has shown its potential in this field (e.g. as in [154]–[156]).

Furthermore, the metrics used in this thesis were the WER (i.e. ASR performance) and F1-score (i.e. laughter identification performance). However, the CER metric is often used for a model with a CTC head, as does HuBERT (i.e., which has a letter-based vocabulary). This makes it (1) difficult to compare to state-of-the-art and (2) WER might not represent the performance of the model very well. This will depend a lot on the aim of your research, where WER accurately represent what full words are correct, but as soon as a word is miss-spelled, it is miss-classified.

Also, the F1-score and WER are calculated using the Jiwer package, which has a built-in way of finding the alignment. This could impact the results negatively, as the alignment method may not always perfectly capture certain nuances of spontaneous or overlapping speech, leading to potential misinterpretation of substitutions, deletions, and insertions. As a result, this can skew the precision and recall, particularly in more complex speech environments like the AMI dataset contains. Additionally, Jiwer operates at the word level and does not account for phonetic similarities between words. This means that even if the model is close to recognising laughter but predicts a similar yet incorrect word, it will still be marked as an error, leading to an inaccurate evaluation. Especially when adding an acoustic sound like laughter, this could be very relevant. It would be valuable to investigate another metric that better assesses how close the model is to accurately recognising laughter, e.g. Levenshtein Distance, Perceptual Evaluation of Speech Quality or Phoneme Error Rate (PER).

Additionally, this work was only tested on a subset of one dataset per experiment. It would be intriguing to investigate what the performance is using other and more (types of) input data. Fine-tuning on more data will, most likely, make it more robust in various settings, ensuring it can be applied in real life (i.e. where many different scenarios will come up). However, this will be costly time-wise and computationally.

Lastly, in this thesis, only laughter was investigated. However, these datasets, and others, also included other sounds like coughing, sneezing and yawning. Investigating how the models react to integrating these types of sounds could be beneficial in understanding and further developing these models. Exploring this topic would be intriguing, as it could further enhance the robustness of ASR models and increase the accuracy of the representation of real-life scenarios more thoroughly than they currently do.

Chapter 6

Conclusion

This research aimed to integrate laughter into pre-trained end-to-end (E2E) models. This study focused on two particular models, namely Whisper and HuBERT. They were each fine-tuned on two datasets separately, namely the AMI Corpus and Switchboard. In addition, two baselines were set (i.e. zero-shot and fine-tuning without laughter) and the results were compared. Overall, we can now draw a conclusion to the main research question: *How effectively can laughter transcription be integrated into established high-performing ASR systems through the fine-tuning of pre-trained End-To-End models?* To address this question, the laughter detection performance of the fine-tuned models is evaluated. Additionally, the impact of this fine-tuning on lexical performance is analysed. Next, a general conclusion and recommendation are provided regarding the models' practical application. Finally, the main research question is addressed with a concluding statement.

6.1 Laughter Performance

Before fine-tuning on the datasets including laughter, two baselines were set. These consisted of the zero-shot setting and the fine-tuned without laughter. Here it was concluded that Whisper was better than HuBERT at identifying a laughter event as part of speech and also patterns could be found in how it deals with these events. However, HuBERT was the only model that substituted these laughter events with something recognisable as laughter, i.e. "haha" or a variant. This was only with the zero-shot on AMI though, so it seems that in this case, the noisy dataset was beneficial for HuBERT.

After fully fine-tuning the noisy *AMI* dataset including laughter, Whisper had a F1-score of 0.444 and **HuBERT** had a **F1-score of 0.531**. HuBERT performed slightly better than Whisper, but still this F1-score is low, where both models would recognise about half of the laughter instances correctly. This by itself would suggest that neither model is suitable to apply to such an environment to integrate laughter recognition into ASR. A deeper analysis of laughter alignment operations together with the recall and precision, showed that for both models the precision was high with few insertions. Both models had a low recall, but, the HuBERT model had a precision of 0.785, while **Whisper** had a **precision of 0.949**. This shows that while the F1-score for HuBERT was better, and Whisper might miss more instances, Whisper barely made any incorrect predictions. Therefore, in this situation, Whisper has some potential to be applied in scenarios where it does not matter as much if laughter is sometimes missed, as long as it is not wrong.

When fine-tuned on the cleaner *Switchboard* dataset, **Whisper** had the **highest F1-score at 0.901**, while HuBERT had a F1-score of 0.622. While HuBERT performs quite badly, Whisper performed very well. At 90%, this shows very good potential for the integration of laughter. A deeper look into the behaviour of the models showed that for **both precision** was **high** (above 0.9). The recall for HuBERT was still quite low, at 0.468, but **Whisper's recall** went up to **0.836**. Based on these results, we can conclude that while HuBERT shows potential in similar scenarios, Whisper consistently delivers superior performance for Switchboard.

6.2 Lexical Performance Impact

The lexical performance is usually the most important in research, where the integration of laughter should be an extra, non-detrimental faction. In this research, the effect of fine-tuning with laughter gave varying results. In every experiment, fine-tuning with or without laughter significantly decreased the WER. However, for 2/4 experiments the fine-tuning without laughter outperformed fine-tuning with laughter, and vice versa. The amount with which this differed was low but variable, ranging from 0.010 to 0.003. Therefore, no definite conclusion could be made as to the impact on the lexical performance. However, we can say that it does have some kind of impact, as every time there was some variation.

Notably, throughout this research, the WER was significantly high. Generally agreed, a WER of 0.05 to 0.10 has good quality and 0.10 to 0.20 is decent. If the models with the highest F1-scores are taken, as these are also the ones with the lowest WER, these are still in this just 0.10 to 0.20 range, but mostly higher. For instance, for **AMI, HuBERT** had the best laughter performance with a **WER of 0.311**, while with **Switchboard, Whisper** had the best laughter performance with a **WER of 0.161**.

However, this research used models which have been applied to the same or similar datasets before. For instance, the Distilled-Whisper model from state-of-the-art reached a WER of 0.169 for the AMI Single Distant Microphone setup and 0.365 WER for the AMI Individual Headset setup with zero-shot. The Distilled-Whisper model in this research had a zero-shot WER of 0.550 on the AMI Mixed Headset (i.e. similar to a mixture of the other types of AMI audio). This illustrates the potential of the model, as problems may lie in pre- or post-processing and not with the model. Therefore, this could be further investigated. The exact reasons and issues with this are discussed in Chapter 5.3.4.

6.3 Application Implications

During this research, several other conclusions were drawn from the research. Throughout training and fine-tuning, several notable elements of the two models came up that are important when applying the models. In addition, the two quite different datasets were used to shed some light on where and how each model could be applied.

6.3.1 Model Behaviour

In this thesis, two models are compared. They are each known for their strengths and weaknesses, which have also been explored in this paper. Whisper is more flexible and robust, but focuses on lexical word and sentence patterns. It also converges quicker, making fine-tuning more cost-efficient. This is because it was pre-trained on many types of audio data including laughter. The HuBERT model is better at recognising acoustic patterns, so phonemes, especially amid noisy data. This is because its pre-training was unsupervised, it does not focus on lexical performance. However, fully fine-tuning HuBERT takes many epochs and demands high resources.

6.3.2 Data Implications

As established, the only result of the experiment that resulted in a usable model, was when Whisper was fine-tuned on Switchboard including laughter. However, it was the case for all experiments (i.e. baselines and fine-tuning) that the F1-score was higher and WER lower on Switchboard than on AMI. While both models have been fine-tuned to two datasets containing spontaneous speech, the two datasets represent very different environments. The AMI dataset is "noisier" and Switchboard "cleaner" and bigger. In this case, this is expressed in AMI having more speakers, less guided topics, more overlapping speech and more background noise in the meeting room. Switchboard only has two speakers, which was a telephone conversation, with clear seed topics. The AMI dataset has around 11 million words, out of which 0.15% are laughter events (i.e. 16 500), while the Switchboard dataset has around 36 million words, out of which 0.10% are laughter events (i.e. 36 000). This also triples the ASR data and doubles the laughter events.

As concluded, the performance of both models is highly dependent on the amount and quality of the data. This has implications for when the models are to be applied to real life or tested on other datasets. The Whisper model can easily be applied to controlled spontaneous speech scenarios like telephone conversations, e.g. customer service or call centres, or one- or two-person podcasts. This model is preferably fine-tuned to this specific scenario, but this is quick to do (i.e. little epochs) and more important is the pre-processing. HuBERT is better applied to situations where a lot of people are talking and the focus is not on the lexical performance, but on getting an impression of the atmosphere, e.g. large group meetings, social gatherings, or background noise analysis in public spaces. It will need fine-tuning with a significant amount of data.

6.4 Final Conclusion

After considering all the partial conclusions from the experiments, we can finally conclude the main research question: *How effectively can laughter transcription be integrated into established high-performing ASR systems through the fine-tuning of pre-trained End-To-End models?* When applied to clean data, in this case Switchboard, the Whisper model is effective. It has a good laughter detection score and lexical performance, with a high F1-score (0.901), high precision (0.948) and relatively high WER (0.161). HuBERT's performance in both laughter detection and lexical tasks was significantly lower, never reaching satisfactory levels. On noisier data, in this case the AMI dataset, both models underperformed, with neither achieving results that could be considered adequate. In addition, no conclusion could be made on the lexical impact of integrating laughter, as this differed too much per experiment. Lastly, it seems that specific ASR systems perform very differently when fine-tuned, where surprisingly a flexible yet lexical-based model outperforms the acoustic-based model on an acoustic sound like laughter. Therefore, laughter can effectively be integrated into ASR systems, but elements like the specific model type, amount and quality of data and its application have to be kept in mind.

References

[1] D. S. Pallett, "A look at NIST's benchmark ASR tests: Past, present, and future," *2003 IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU 2003*, pp. 483–488, 2003. DOI: 10.1109/ASRU.2003.1318488.

[2] J. Li, "Recent advances in end-to-end automatic speech recognition," *eess*, 2020, ISSN: 01651684. DOI: https://doi.org/10.48550/arXiv.2111.01690.

[3] L. Lazli and M. Sellami, "Connectionist probability estimators in HMM arabic speech recognition using fuzzy logic," *Lecture Notes in Artificial Intelligence (Subseries of Lecture Notes in Computer Science)*, vol. 2734, pp. 379–388, 2003, ISSN: 03029743. DOI: 10.1007/3-540-45065-3{_}33/COVER. [Online]. Available: https://link.springer.com/chapter/10.1007/3-540-45065-3_33.

[4] D. A. Moses, N. Mesgarani, M. K. Leonard, *et al.*, "Overview of end-to-end speech recognition You may also like Out Domain Data Augmentation on Punjabi Children Speech Recognition using Tacotron Taniya Hasija, Virender Kadyan and Kalpna Guleria-Neural speech recognition: continuous phoneme decoding using spatiotemporal representations of human cortical activity Improving Tibetan End-To-End Speech Recognition with Transfer Learning Overview of end-to-end speech recognition," *J. Phys.: Conf. Ser*, vol. 1187, p. 52 068, 2019. DOI: 10.1088/1742-6596/1187/5/052068.

[5] B. Schuller, S. Steidl, A. Batliner, *et al.*, "Paralinguistics in speech and language—State-of-the-art and the challenge," *Computer Speech & Language*, vol. 27, no. 1, pp. 4–39, Jan. 2013, ISSN: 0885-2308. DOI: 10.1016/J.CSL.2012.02.005.

[6] H. Lnaguma, M. Mimura, K. Inoue, K. Yoshii, and T. Kawahara, "An end-to-end approach to joint social signal detection and automatic speech recognition," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 2018-April, pp. 6214–6218, 2018, ISSN: 15206149. DOI: 10.1109/ICASSP.2018.8462578.

[7] J. Trouvain and K. P. Truong, *Comparing non-verbal vocalisations in conversational speech corpora*, 2012. [Online]. Available: https://research.utwente.nl/en/publications/comparing-non-verbal-vocalisations-in-conversational-speech-corpo.

[8] V. Mendelev, T. Raissi, G. Camporese, and M. Giollo, "IMPROVED ROBUSTNESS TO DISFLUENCIES IN RNN-TRANSDUCER BASED SPEECH RECOGNITION," *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021. DOI: 10.1109/ICASSP39728.2021.9413618.

[9] K. Horii, M. Fukuda, K. Ohta, R. Nishimura, A. Ogawa, and N. Kitaoka, "End-to-End Spontaneous Speech Recognition Using Hesitation Labeling," *APSIPA Annual Summit and Conference*, no. December, pp. 14–17, 2021, ISSN: 19909772. DOI: 10.21437/Interspeech.2022-281.

[10] *GitHub - openai/whisper: Robust Speech Recognition via Large-Scale Weak Supervision.* [Online]. Available: https://github.com/openai/whisper.

[11] *Whisper.* [Online]. Available: https://huggingface.co/docs/transformers/en/model_doc/whisper#transformers.WhisperForConditionalGeneration.

[12] *Hubert.* [Online]. Available: https://huggingface.co/docs/transformers/model_doc/hubert.

[13] *HuBERT: Speech representations for recognition & generation.* [Online]. Available: https://ai.meta.com/blog/hubert-self-supervised-representation-learning-for-speech-recognition-generation-and-compression/.

[14] A. Hernandez, P. Andrea Pérez-Toro, E. Nöth, J. R. Orozco-Arroyave, A. Maier, and S. H. Yang, "Cross-lingual Self-Supervised Speech Representations for Improved Dysarthric Speech Recognition,"

[15] P. Wang and H. Van hamme, "Benefits of pre-trained mono- and cross-lingual speech representations for spoken language understanding of Dutch dysarthric speech," *Eurasip Journal on Audio, Speech, and Music Processing*, vol. 2023, no. 1, pp. 1–25, Dec. 2023, ISSN: 16874722. DOI: 10.1186/S13636-023-00280-Z/FIGURES/14. [Online]. Available: https://asmp-eurasipjournals.springeropen.com/articles/10.1186/s13636-023-00280-z.

[16] J. K. Mbuya and A. Anastasopoulos, "GMU Systems for the IWSLT 2023 Dialect and Low-resource Speech Translation Tasks," pp. 269–276,

[17] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. Mcleavey, and I. Sutskever, "Robust Speech Recognition via Large-Scale Weak Supervision," [Online]. Available: https://github.com/openai/.

[18] J. Gillick, W. Deng, K. Ryokai, and D. Bamman, "Robust Laughter Detection in Noisy Environments," *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, vol. 1, pp. 2481–2485, 2021, ISSN: 19909772. DOI: 10.21437/INTERSPEECH.2021-353. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2021-353.

[19] M. Zeineldeen, A. Zeyer, R. Schl, and A. Gmbh, "LAYER-NORMALIZED LSTM FOR HYBRID-HMM AND END-TO-END ASR Human Language Technology and Pattern Recognition Group , Computer Science Department RWTH Aachen University , 52074 Aachen , Germany," pp. 7674–7678, 2020.

[20] M. Malik, M. Kamran Malik, K. Mehmood, and I. Makhdoom, "Automatic speech recognition: a survey," DOI: 10.1007/s11042-020-10073-7. [Online]. Available: https://doi.org/10.1007/s11042-020-10073-7.

[21] *Switchboard-1 Release 2 - Linguistic Data Consortium.* [Online]. Available: https://catalog.ldc.upenn.edu/LDC97S62.

[22] R. Dufour, V. Jousse, Y. Estève, F. B. ´. Echet, and G. Linarès, "SPONTANEOUS SPEECH CHARACTERIZATION AND DETECTION IN LARGE AUDIO DATABASE,"

[23] S. A. Sheikh, M. Sahidullah, F. Hirsch, and S. Ouni, "Machine Learning for Stuttering Identification: Review, Challenges and Future Directions," [Online]. Available: https://www.healthline.com/health/speech-disorders.

[24] *openslr.org.* [Online]. Available: https://www.openslr.org/12.

[25] *Common Voice.* [Online]. Available: https://commonvoice.mozilla.org/en.

[26] C. Wang, M. Rivière, A. Lee, *et al.*, "VoxPopuli: A large-scale multilingual speech corpus for representation learning, semi-supervised learning and interpretation," *ACL-IJCNLP 2021 - 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, Proceed-*

ings of the Conference, pp. 993–1003, 2021. DOI: 10.18653/V1/2021.ACL-LONG.80. [Online]. Available: https://huggingface.co/datasets/facebook/voxpopuli.

[27] *ICSI Corpus*. [Online]. Available: https://groups.inf.ed.ac.uk/ami/icsi/.

[28] *AMI Corpus*. [Online]. Available: https://groups.inf.ed.ac.uk/ami/corpus/.

[29] A. Polychroniou, H. Salamin, and A. Vinciarelli, "The SSPNet Mobile Corpus: Social Signal Processing Over Mobile Phones,"

[30] V. Zayats, M. Ostendorf, and H. Hajishirzi, "Disfluency Detection using a Bidirectional LSTM," [Online]. Available: http://ssli.ee..

[31] J. Ferguson, G. Durrett, and D. Klein, "Disfluency Detection with a Semi-Markov Model and Prosodic Features,"

[32] X. Qian and Y. Liu, "Disfluency Detection Using Multi-step Stacked Learning," pp. 9–14, 2013. [Online]. Available: http://code.google.com/p/disfluency-detection/downloads/list.

[33] Y. Liu, E. Shriberg, A. Stolcke, *et al.*, "Enriching Speech Recognition With Automatic Detection of Sentence Boundaries and Disfluencies," *IEEE TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING*, vol. 14, no. 5, 2006. DOI: 10.1109/TASL.2006.878255. [Online]. Available: http://www.nist.gov/speech/tests/rt/rt2004/fall/.

[34] *UASpeech Database*. [Online]. Available: http://www.isle.illinois.edu/sst/data/UASpeech/.

[35] *GitHub - Tatsu1020/self-supervised-dutch-dysarthria-asr: This is a repository presenting the outcome of my thesis for MSc. Voice Technology at the University of Groningen. The research developed the Dutch dysarthric speech recognition with self-supervised learning (SSL) models, wav2vec 2.0 and XLSR-53. The repo contains the fine-tuned models and the evaluation dataset.* [Online]. Available: https://github.com/Tatsu1020/self-supervised-dutch-dysarthria-asr.

[36] J. Frederick, "Continuous Speech Recognition by Statistical Methods," *IEEE Transactions on Neural Networks*, vol. 64, no. 4, 1976, ISSN: 19410093. DOI: 10.1109/72.286885.

[37] S. E. Levinson, L. R. Rabiner, and M. M. Sondhi, "An Introduction to the Application of the Theory of Probabilistic Functions of a Markov Process to Automatic Speech Recognition," *Bell System Technical Journal*, vol. 62, no. 4, pp. 1035–1074, 1983, ISSN: 15387305. DOI: 10.1002/j.1538-7305.1983.tb03114.x.

[38] B. .-. Juang, "On the Hidden Markov Model and Dynamic Time Warping for Speech Recognition—A Unified View," *AT&T Bell Laboratories Technical Journal*, vol. 63, no. 7, pp. 1213–1243, 1984, ISSN: 15387305. DOI: 10.1002/j.1538-7305.1984.tb00034.x.

[39] M. Gales and S. Young, "The Application of Hidden Markov Models in Speech Recognition," *Foundations and Trends R in Signal Processing*, vol. 1, no. 3, pp. 195–304, 2007. DOI: 10.1561/2000000004.

[40] N. Najkar, F. Razzazi, and H. Sameti, "A novel approach to HMM-based speech recognition system using particle swarm optimization," *BIC-TA 2009 - Proceedings, 2009 4th International Conference on Bio-Inspired Computing: Theories and Applications*, pp. 296–301, 2009. DOI: 10.1109/BICTA.2009.5338098.

[41] H. Sameti, H. Sheikhzadeh, L. Deng, and R. L. Brennan, "HMM-based strategies for enhancement of speech signals embedded in nonstationary noise," *IEEE Transactions on Speech and Audio Processing*, vol. 6, no. 5, pp. 445–454, 1998, ISSN: 10636676. DOI: 10.1109/89.709670.

[42] S. Renals, N. Morgan, S. Member, H. Bourlard, M. Cohen, and H. Franco, "Connectionist Probability Estimators in HMM Speech Recognition," *IEEE TRANSACTIONS ON SPEECH AND AUDIO PROCESSING*, vol. 2, no. 1, 1994.

[43] P. Pujol, S. Pol, C. Nadeu, A. Hagen, and H. Bourlard, "Comparison and combination of features in a hybrid HMM/MLP and a HMM/GMM speech recognition system," *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 1, pp. 14–22, 2005, ISSN: 10636676. DOI: 10.1109/TSA.2004.834466.

[44] A. Ganapathiraju, "SUPPORT VECTOR MACHINES FOR SPEECH RECOGNITION," 2002.

[45] J. E. Hamaker, "MLLR: A SPEAKER ADAPTATION TECHNIQUE FOR LVCSR,"

[46] E. Trentin and M. Gori, "Robust combination of neural networks and hidden Markov models for speech recognition," *IEEE Transactions on Neural Networks*, vol. 14, no. 6, pp. 1519–1531, 2003, ISSN: 10459227. DOI: 10.1109/TNN.2003.820838.

[47] X. Tang, "Hybrid Hidden Markov Model and Artificial Neural Network for Automatic Speech Recognition," 2009. DOI: 10.1109/PACCS.2009.138.

[48] S. Newatia and R. K. Aggarwal, "Convolutional Neural Network for ASR," *Proceedings of the 2nd International Conference on Electronics, Communication and Aerospace Technology, ICECA 2018*, no. Iceca, pp. 638–642, 2018. DOI: 10.1109/ICECA.2018.8474688.

[49] O. Vinyals, S. V. Ravuri, and D. Povey, "REVISITING RECURRENT NEURAL NETWORKS FOR ROBUST ASR,"

[50] T. Yoshioka and M. J. F. Gales, "ScienceDirect Environmentally robust ASR front-end for deep neural network acoustic models," 2014. DOI: 10.1016/j.csl.2014.11.008. [Online]. Available: www.sciencedirect.com.

[51] V. Kadyan, A. Mantri, ·. R. K. Aggarwal, and A. Singh, "A comparative study of deep neural network based Punjabi-ASR system," *International Journal of Speech Technology*, vol. 22, no. 3, pp. 111–119, 2019. DOI: 10.1007/s10772-018-09577-3. [Online]. Available: https://doi.org/10.1007/s10772-018-09577-3.

[52] A. Ganapathiraju, J. E. Hamaker, and J. Picone, "Applications of support vector machines to speech recognition," *IEEE Transactions on Signal Processing*, vol. 52, no. 8, pp. 2348–2355, 2004, ISSN: 1053587X. DOI: 10.1109/TSP.2004.831018.

[53] R. Solera-Ureña, A. I. García-Moral, C. Peláez-Moreno, M. Martínez-Ramon, and F. Díaz-De-María, "Real-time robust automatic speech recognition using compact support vector machines," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 20, no. 4, pp. 1347–1361, 2012, ISSN: 15587916. DOI: 10.1109/TASL.2011.2178597.

[54] G. C. Batista, D. L. Oliveira, O. Saotome, and T. S. Curtinhas, "A new asynchronous pipeline architecture of support vector machine classifier for ASR system," *Proceedings of the 2019 IEEE 26th International Conference on Electronics, Electrical Engineering and Computing, INTERCON 2019*, pp. 1–4, 2019. DOI: 10.1109/INTERCON.2019.8853577.

[55] D. Martín-Iglesias, J. Bernal-Chaves, C. Peláez-Moreno, A. Gallardo-Antolín, and F. Díaz-de-María, "A speech recognizer based on multiclass SVMs with HMM-guided segmentation," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 3817 LNAI, pp. 257–266, 2005, ISSN: 03029743. DOI: 10.1007/11613107{_}22/COVER. [Online]. Available: https://link.springer.com/chapter/10.1007/11613107_22.

[56] M. J. Gales, A. Ragni, H. AlDamarki, and C. Gautier, "Support vector machines for noise robust ASR," *Proceedings of the 2009 IEEE Workshop on Automatic Speech Recog-*

nition and Understanding, ASRU 2009, pp. 205–210, 2009. DOI: 10.1109/ASRU.2009.5372913.

[57] J. Padrell-Sendral, D. Martfn-Ig, /. Esias2, and F. Dlaz-De-Marfa2, "SUPPORT VECTOR MACHINES FOR CONTINUOUS SPEECH RECOGNITION," 2006.

[58] R. Prabhavalkar, T. Hori, S. Member, T. N. Sainath, R. Schlüter, and S. Watanabe, "End-to-End Speech Recognition: A Survey,"

[59] *An Overview of Transducer Models for ASR.* [Online]. Available: https://www.assemblyai.com/blog/an-overview-of-transducer-models-for-asr/.

[60] *Introducing Whisper.* [Online]. Available: https://openai.com/research/whisper.

[61] A. Conneau, A. Baevski, R. Collobert, A. Mohamed, and M. A. Facebook, "UNSUPERVISED CROSS-LINGUAL REPRESENTATION LEARNING FOR SPEECH RECOGNITION," [Online]. Available: https://github.com/pytorch/fairseq/tree/master/examples/wav2vec.

[62] A. Babu, C. Wang, A. Tjandra, *et al.*, "XLS-R: Self-supervised Cross-lingual Speech Representation Learning at Scale," *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, vol. 2022-September, pp. 2278–2282, Nov. 2021, ISSN: 19909772. DOI: 10.21437/Interspeech.2022-143. [Online]. Available: https://arxiv.org/abs/2111.09296v3.

[63] A. Baevski, H. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations," [Online]. Available: https://github.com/pytorch/fairseq.

[64] *Pre-trained models for automatic speech recognition - Hugging Face Audio Course.* [Online]. Available: https://huggingface.co/learn/audio-course/chapter5/asr_models.

[65] W.-N. Hsu, B. Bolte, Y.-H. H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed, "HuBERT: Self-Supervised Speech Representation Learning by Masked Prediction of Hidden Units,"

[66] *CTC architectures - Hugging Face Audio Course.* [Online]. Available: https://huggingface.co/learn/audio-course/chapter3/ctc.

[67] Y. Zhang, D. S. Park, W. Han, *et al.*, "BigSSL: Exploring the Frontier of Large-Scale Semi-Supervised Learning for Automatic Speech Recognition,"

[68] *Automatic Speech Recognition With Whisper | by Shiry Yonash | Better Programming.* [Online]. Available: https://betterprogramming.pub/automatic-speech-recognition-with-whisper-84ad29d3b0bd.

[69] M. Rivì Ere †, A. Joulin, P.-E. Mazaré, and E. Dupoux, "UNSUPERVISED PRETRAINING TRANSFERS WELL ACROSS LANGUAGES," [Online]. Available: https://github.com/facebookresearch/CPC_audio.

[70] H. Inaguma, J. Cho, M. K. Baskar, T. Kawahara, and S. Watanabe, "Transfer Learning of Language-independent End-to-end ASR with Language Model Fusion," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 2019-May, pp. 6096–6100, May 2019, ISSN: 15206149. DOI: 10.1109/ICASSP.2019.8682918. [Online]. Available: https://waseda.elsevierpure.com/en/publications/transfer-learning-of-language-independent-end-to-end-asr-with-lan.

[71] *BERT Explained: State of the art language model for NLP | by Rani Horev | Towards Data Science.* [Online]. Available: https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270.

[72] Z. Kozhirbayev, "Kazakh Speech Recognition: Wav2vec2.0 vs. Whisper," DOI: 10.12720/jait.14.6.1382-1389. [Online]. Available: https://www.ethnologue.com/language/kaz.

[73] P. Gilles, L. Hillah, and N. Hosseini-Kivanani, "ASRLUX: AUTOMATIC SPEECH RECOGNITION FOR THE LOW-RESOURCE LANGUAGE LUXEMBOURGISH,"

[74] T. Pekarek and S. Wermter, "Replay to Remember: Continual Layer-Specific Fine-Tuning for German Speech Recognition," 2023. DOI: 10.1007/978-3-031-44195-0{_}40. [Online]. Available: https://doi.org/10.1007/978-3-031-44195-0_40.

[75] T. Olatunji, T. Afonja, A. Yadavalli, *et al.*, "AfriSpeech-200: Pan-African Accented Speech Dataset for Clinical and General Domain ASR," *Transactions of the Association for Computational Linguistics*, vol. 11, pp. 1669–1685, Dec. 2023, ISSN: 2307-387X. DOI: 10.1162/TACL{_}A{_}00627. [Online]. Available: https://dx.doi.org/10.1162/tacl_a_00627.

[76] R. Sanabria, H. Tang, and S. Goldwater, "Analyzing Acoustic Word Embeddings from Pre-Trained Self-Supervised Speech Models," *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, no. 1, pp. 1–5, 2023, ISSN: 15206149. DOI: 10.1109/icassp49357.2023.10096099.

[77] Z. Chen, S. Chen, Y. Wu, *et al.*, "LARGE-SCALE SELF-SUPERVISED SPEECH REPRESENTATION LEARNING FOR AUTOMATIC SPEAKER VERIFICATION," DOI: 10.1109/ICASSP43922.2022.9747814. [Online]. Available: https://github.com/pytorch/fairseq.

[78] A. Williams, A. Demarco, and C. Borg, "The Applicability of Wav2Vec2 and Whisper for Low-Resource Maltese ASR," [Online]. Available: https://www.um.edu.mt/projects/masri/.

[79] Y. Gong, S. Khurana, L. Karlinsky, and J. Glass, "Whisper-AT: Noise-Robust Automatic Speech Recognizers are Also Strong General Audio Event Taggers,"

[80] K.-P. Huang, C.-K. Yang, Y.-K. Fu, E. Dunbar, and H.-Y. Lee, "ZERO RESOURCE CODE-SWITCHED SPEECH BENCHMARK USING SPEECH UTTERANCE PAIRS FOR MULTIPLE SPOKEN LANGUAGES," [Online]. Available: https://github.com/nobel861017/cs_zs_baseline..

[81] A. Lopez, A. Liesenfeld, and M. Dingemanse, "Evaluation of Automatic Speech Recognition for Conversational Speech in Dutch, English, and German: What Goes Missing?," [Online]. Available: https://clarin.phonetik.uni-muenchen..

[82] S. Johar, "Paralinguistic profiling using speech recognition," *Int J Speech Technol*, vol. 17, pp. 205–209, 2014. DOI: 10.1007/s10772-013-9222-4.

[83] M. E. Wingate, "Fluency and disfluency; Illusion and identification," *Journal of Fluency Disorders*, vol. 12, no. 2, pp. 79–101, 1987, ISSN: 0094730X. DOI: 10.1016/0094-730X(87)90015-5.

[84] S. Wang, W. Che, and T. Liu, "A Neural Attention Model for Disfluency Detection," pp. 278–287,

[85] S. Wang, W. Che, Y. Zhang, M. Zhang, and T. Liu, "Transition-Based Disfluency Detection using LSTMs," pp. 2785–2794, [Online]. Available: https://github.com/hitwsl/transition.

[86] Q. Dong, F. Wang, Z. Yang, W. Chen, S. Xu, and B. Xu, "Adapting Translation Models for Transcript Disfluency Detection," p. 19, [Online]. Available: www.aaai.org.

[87] V. Zayats, M. Ostendorf, and H. Hajishirzi, "Multi-domain disfluency and repair detection," *Proceedings of the Annual Conference of the International Speech Communication*

Association, INTERSPEECH, pp. 2907–2911, 2014, ISSN: 19909772. DOI: 10.21437/INTERSPEECH.2014-603.

[88] R. L. Rose, "THE COMMUNICATIVE VALUE OF FILLED PAUSES IN SPONTANEOUS SPEECH," 1998.

[89] T. Schultz and A. Waibel, "Language Independent and Language Adaptive Large Vocabulary Speech Recognition,"

[90] H. H. Clark and J. E. Fox Tree, "Using uh and um in spontaneous speaking," [Online]. Available: www.elsevier.com/locate/cognit.

[91] M. Harper, B. Dorr, J. Hale, *et al.*, "Parsing and Spoken Structural Event Detection,"

[92] M. Lease and M. Johnson, "Early Deletion of Fillers In Processing Conversational Speech," 2006. [Online]. Available: http://www.cpan.org.

[93] *Interspeech 2023*. [Online]. Available: https://interspeech2023.org/.

[94] B. Schuller, F. Weninger, Y. Zhang, *et al.*, "Affective and behavioural computing: Lessons learnt from the First Computational Paralinguistics Challenge," *Computer Speech & Language*, vol. 53, pp. 156–180, Jan. 2019, ISSN: 0885-2308. DOI: 10.1016/J.CSL.2018.02.004.

[95] R. Gupta, K. Audhkhasi, S. Lee, and S. Narayanan, "Paralinguistic event detection from speech using probabilistic time-series smoothing and masking," 2013.

[96] L. . Kaushik, A. Sangwan, and J. H. L. Hansen, *Laughter and Filler Detection in Naturalistic Audio*, 2016. [Online]. Available: http://hdl.handle.net/10735.1/5058.

[97] R. Ruede, M. Müller, S. Stüker, and A. Waibel, "Yeah, right, uh-huh: A deep learning backchannel predictor," *Lecture Notes in Electrical Engineering*, vol. 510, pp. 247–258, 2019, ISSN: 18761119. DOI: 10.1007/978-3-319-92108-2{_}25/TABLES/2. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-92108-2_25.

[98] B. Heinz, "Backchannel responses as strategic responses in bilingual speakers' conversations," *Journal of Pragmatics*, vol. 35, no. 7, pp. 1113–1142, Jul. 2003, ISSN: 0378-2166. DOI: 10.1016/S0378-2166(02)00190-X.

[99] *Backchannels across Cultures: A Study of Americans and Japanese on JSTOR*. [Online]. Available: https://www.jstor.org/stable/4168001.

[100] V. Jain, M. Leekha, and J. Shukla, "Exploring Semi-Supervised Learning for Predicting Listener Backchannels," 2021. DOI: 10.1145/3411764.3445449. [Online]. Available: https://doi.org/10.1145/3411764.3445449.

[101] L. P. Morency, I. De Kok, and J. Gratch, "Predicting listener backchannels: A probabilistic multimodal approach," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 5208 LNAI, pp. 176–190, 2008, ISSN: 03029743. DOI: 10.1007/978-3-540-85483-8{_}18/COVER. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-540-85483-8_18.

[102] M. Mueller, D. Leuschner, L. Briem, *et al.*, "Using neural networks for data-driven backchannel prediction: A survey on input features and training techniques," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9170, pp. 329–340, 2015, ISSN: 16113349. DOI: 10.1007/978-3-319-20916-6{_}31/TABLES/11. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-20916-6_31.

[103] D. Ortega, C. Y. Li, and N. T. Vu, "OH, JEEZ! or UH-HUH? A Listener-Aware Backchannel Predictor on ASR Transcriptions," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 2020-May, pp. 8064–8068, 2020, ISSN: 15206149. DOI: 10.1109/ICASSP40776.2020.9054223.

[104] G. Deshpande and B. W. Schuller, "Highlights Audio, Speech, Language, & Signal Processing for COVID-19: A Comprehensive Overview Audio, Speech, Language, & Signal Processing for COVID-19: A Comprehensive Overview," 2020. [Online]. Available: www.who.int.

[105] A. Sharma and D. Malhotra, "Speech recognition based IICC - Intelligent infant cry classifier," *Proceedings of the 3rd International Conference on Smart Systems and Inventive Technology, ICSSIT 2020*, pp. 992–998, Aug. 2020. DOI: 10.1109/ICSSIT48917.2020.9214193.

[106] S. Ntalampiras, "Audio Pattern Recognition of Baby Crying Sound Events," *Journal of the Audio Engineering Society*, vol. 63, no. 5, pp. 358–369, May 2015, ISSN: 15494950. DOI: 10.17743/JAES.2015.0025.

[107] J. Yadav, N. Satrughan Kumar Singh, and M. Sundararajan, "AN AUTOMATIC INFANT CRY SPEECH RECOGNITION USING ARTIFICIAL NEURAL NETWORK AN AUTOMATIC INFANT CRY SPEECH RECOGNITION USING ARTIFICIAL NEURAL," *HYPOTHESIS-National Journal of Research in Higher Studies*, no. 2, 2022. [Online]. Available: https://www.researchgate.net/publication/368308131.

[108] M. Ochocki and D. Sawicki, "Yawning Recognition based on Dynamic Analysis and Simple Measure," DOI: 10.5220/0006497701110117.

[109] W. Zhang and J. Su, "Driver yawning detection based on long short term memory networks," *2017 IEEE Symposium Series on Computational Intelligence, SSCI 2017 - Proceedings*, vol. 2018-January, pp. 1–5, Feb. 2018. DOI: 10.1109/SSCI.2017.8285343.

[110] H. Yang, L. Liu, W. Min, X. Yang, and X. Xiong, "Driver Yawning Detection Based on Subtle Facial Action Recognition," *IEEE Transactions on Multimedia*, vol. 23, pp. 572–583, 2021, ISSN: 19410077. DOI: 10.1109/TMM.2020.2985536.

[111] T. Wang and P. Shi, "Yawning detection for determining driver drowsiness," *Proceedings of the 2005 IEEE International Workshop on VLSI Design and Video Technology, IWVDVT 2005*, pp. 385–388, 2005. DOI: 10.1109/IWVDVT.2005.1504628.

[112] K. P. Truong and D. A. van Leeuwen, "Automatic discrimination between laughter and speech," *Speech Communication*, vol. 49, no. 2, pp. 144–158, Feb. 2007, ISSN: 0167-6393. DOI: 10.1016/J.SPECOM.2007.01.001.

[113] M. T. Knox and N. Mirghafori, "Automatic Laughter Detection Using Neural Networks,"

[114] G. Gosztolya, A. Beke, T. Neuberger, and L. Tóth, "Laughter Classification Using Deep Rectifier Neural Networks with a Minimal Feature Subset," *Archives of Acoustics*, vol. 41, no. 4, pp. 669–682, Dec. 2016, ISSN: 2300-262X. DOI: 10.1515/AOA-2016-0064. [Online]. Available: https://acoustics.ippt.pan.pl/index.php/aa/article/view/1700.

[115] G. Rennie, O. Perepelkina, and A. Vinciarelli, "Which Model is Best: Comparing Methods and Metrics for Automatic Laughter Detection in a Naturalistic Conversational Dataset," 2022. DOI: 10.21437/Interspeech.2022-11379.

[116] N. Li, N. Sepúlveda, and N. Li, *AN END-TO-END APPROACH TO JOINT SOCIAL SIGNAL DETECTION AND AUTOMATIC SPEECH RECOGNITION*, 2011. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7781053%0Ahttps://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5485276%0Ahttps://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9511469%0Ahttps://ieeexplore-ieee-org.uproxy.library.dc-uoit.ca/.

[117] P. J. Lou and M. Johnson, "End-to-End Speech Recognition and Disfluency Removal," *Findings of the Association for Computational Linguistics Findings of ACL: EMNLP*

2020, pp. 2051–2061, 2020. DOI: 10.18653/V1/2020.FINDINGS-EMNLP.186. [Online]. Available: https://aclanthology.org/2020.findings-emnlp.186.

[118] K. Horii, M. Fukuda, K. Ohta, R. Nishimura, A. Ogawa, and N. Kitaoka, "End-to-End Spontaneous Speech Recognition Based on Disfluency Labeling," DOI: 10.2139/SSRN.4550755. [Online]. Available: https://papers.ssrn.com/abstract=4550755.

[119] T. Matsuda and Y. Arimoto, "Detection of Laughter and Screaming Using the Attention and CTC Models," *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, vol. 2023-August, pp. 1025–1029, 2023, ISSN: 19909772. DOI: 10.21437/INTERSPEECH.2023-1412.

[120] S. Condron, G. Clarke, A. Klementiev, D. Morse-Kopp, J. Parry, and D. Palaz, "Non-Verbal Vocalisation and Laughter Detection Using Sequence-to-Sequence Models and Multi-Label Training," *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, vol. 2, pp. 2506–2510, 2021, ISSN: 19909772. DOI: 10.21437/INTERSPEECH.2021-1159. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2021-1159.

[121] *Whisper-v3 Hallucinations on Real World Data | Deepgram*. [Online]. Available: https://deepgram.com/learn/whisper-v3-results.

[122] *Paper page - Distil-Whisper: Robust Knowledge Distillation via Large-Scale Pseudo Labelling*. [Online]. Available: https://huggingface.co/papers/2311.00430.

[123] *Fine-Tune Wav2Vec2 for English ASR in Hugging Face with Transformers*. [Online]. Available: https://huggingface.co/blog/fine-tune-wav2vec2-english.

[124] Y. Wang, A. Boumadane, and A. Heba, "A FINE-TUNED WAV2VEC 2.0/HUBERT BENCHMARK FOR SPEECH EMOTION RECOGNITION, SPEAKER VERIFICATION AND SPOKEN LANGUAGE UNDERSTANDING," [Online]. Available: https://github.com/speechbrain/speechbrain/tree/develop/recipes.

[125] S. Amiriparian, F. Packa´npacka´n, M. Gerczuk, and B. W. Schuller, "ExHuBERT: Enhancing HuBERT Through Block Extension and Fine-Tuning on 37 Emotion Datasets," [Online]. Available: https://huggingface.co/openai/whisper-medium.

[126] *random — Generate pseudo-random numbers — Python 3.12.5 documentation*. [Online]. Available: https://docs.python.org/3/library/random.html.

[127] *Trainer*. [Online]. Available: https://huggingface.co/docs/transformers/en/main_classes/trainer.

[128] *Preprocessing Steps for Natural Language Processing (NLP): A Beginner's Guide | by Maleesha De Silva | Medium*. [Online]. Available: https://medium.com/@maleeshadesilva21/preprocessing-steps-for-natural-language-processing-nlp-a-beginners-guide-d6d9bf7689c9.

[129] *jiwer · PyPI*. [Online]. Available: https://pypi.org/project/jiwer/.

[130] *alignment - jiwer*. [Online]. Available: https://jitsi.github.io/jiwer/reference/alignment/.

[131] L. Qu, C. Weber, and S. Wermter, "Emphasizing Unseen Words: New Vocabulary Acquisition for End-to-End Speech Recognition," 2023.

[132] L. Qin, F. Metze, and C. Mark Dredze, "Learning Out-of-Vocabulary Words in Automatic Speech Recognition," [Online]. Available: www.lti.cs.cmu.edu.

[133] *glob — Unix style pathname pattern expansion — Python 3.12.5 documentation*. [Online]. Available: https://docs.python.org/3/library/glob.html.

[134] *os — Miscellaneous operating system interfaces — Python 3.12.5 documentation*. [Online]. Available: https://docs.python.org/3/library/os.html.

[135] *pickle — Python object serialization — Python 3.12.5 documentation.* [Online]. Available: https://docs.python.org/3/library/pickle.html.

[136] *NumPy* -. [Online]. Available: https://numpy.org/.

[137] *gc — Garbage Collector interface — Python 3.12.5 documentation.* [Online]. Available: https://docs.python.org/3/library/gc.html.

[138] *Transformers.* [Online]. Available: https://huggingface.co/docs/transformers/en/index.

[139] *Fine-Tune Wav2Vec2 for English ASR in Hugging Face with Transformers.* [Online]. Available: https://huggingface.co/blog/fine-tune-wav2vec2-english.

[140] *PyTorch.* [Online]. Available: https://pytorch.org/.

[141] *re — Regular expression operations — Python 3.12.5 documentation.* [Online]. Available: https://docs.python.org/3/library/re.html.

[142] *csv — CSV File Reading and Writing — Python 3.12.5 documentation.* [Online]. Available: https://docs.python.org/3/library/csv.html.

[143] *json — JSON encoder and decoder — Python 3.12.5 documentation.* [Online]. Available: https://docs.python.org/3/library/json.html.

[144] *pandas · PyPI.* [Online]. Available: https://pypi.org/project/pandas/.

[145] *ChatGPT.* [Online]. Available: https://chatgpt.com/.

[146] *EEMCS-HPC Hardware [University of Twente, HPC Wiki].* [Online]. Available: https://hpc.wiki.utwente.nl/eemcs-hpc:hardware.

[147] Microsoft, *Test accuracy of a custom speech model.* [Online]. Available: https://learn.microsoft.com/en-us/azure/ai-services/speech-service/how-to-custom-speech-evaluate-data?pivots=speech-studio.

[148] R. Lippmann, "Speech perception by machines and humans," *Speech communication*, vol. 22, pp. 1–16, 1997.

[149] "mHuBERT-147: A Compact Multilingual HuBERT Model," 2024. [Online]. Available: https://github.com/utter-project/mHuBERT-147-scripts.

[150] M. Karthick Baskar, A. Rosenberg, B. Ramabhadran, and Y. Zhang, "Reducing Domain mismatch in Self-supervised speech pre-training," 2022. DOI: 10.21437/Interspeech.2022-736.

[151] P. Kumar, V. N. Sukhadia, and S. Umesh, "INVESTIGATION OF ROBUSTNESS OF HUBERT FEATURES FROM DIFFERENT LAYERS TO DOMAIN, ACCENT AND LANGUAGE VARIATIONS," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 2022-May, pp. 6887–6891, 2022, ISSN: 15206149. DOI: 10.1109/ICASSP43922.2022.9746250. [Online]. Available: https://www.researchgate.net/publication/360793169_Investigation_of_Robustness_of_Hubert_Features_from_Different_Layers_to_Domain_Accent_and_Language_Variations.

[152] *distil-whisper/distil-large-v2 · Hugging Face.* [Online]. Available: https://huggingface.co/distil-whisper/distil-large-v2.

[153] S. Gandhi, P. von Platen, and A. M. Rush, "Distil-Whisper: Robust Knowledge Distillation via Large-Scale Pseudo Labelling," Nov. 2023. [Online]. Available: https://arxiv.org/abs/2311.00430v1.

[154] X. Zheng, Y. Liu, D. Gunceler, and D. Willett, "Using synthetic audio to improve the recognition of Out-of-vocabulary words in end-to-end asr systems," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 2021-June, pp. 5674–5678, 2021, ISSN: 15206149. DOI: 10.1109/ICASSP39728.2021.9414778.

[155] J. Cui, C. Weng, G. Wang, *et al.*, “IMPROVING ATTENTION-BASED END-TO-END ASR SYSTEMS WITH SEQUENCE-BASED LOSS FUNCTIONS The Ohio State University , USA,” *2018 IEEE Spoken Language Technology Workshop (SLT)*, pp. 353–360, 2018.

[156] T. Moon, H. Choi, H. Lee, and I. Song, “RNNDROP: A novel dropout for RNNS in ASR,” *2015 IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU 2015 - Proceedings*, pp. 65–70, 2016. DOI: 10.1109/ASRU.2015.7404775.

Acronyms

ANN Artificial Neural Networks.

ASR Automatic Speech Recognition.

AUC Area Under Curve.

BLSTM Bidirectional Long-Short Term Memory Neural Network.

CER Character Error Rate.

CNN Convolutional Neural Network.

CRF Conditional Random Field.

CSJ Corpus of Spontaneous Japanese.

CTC Connectionist Temporal classification.

DNN Deep Neural Network.

E2E End-To-End.

EER Equal Error Rate.

GMM Gaussian Mixture Model.

HMM Hidden Markov Model.

LM Language Models.

MLP multilayer perceptron.

NMT neural machine translation.

NN Neural Network.

PER Phoneme Error Rate.

REF reference transcript.

ReLU	Rectified Linear Unit.
RNA	Recurrent Neural Aligner.
RNN	Recurrent Neural Network.
RNN-T	Recurrent Neural Network Transducer.
Seq2Seq	Sequence-to-Sequence.
SER	Sentence Error Rate.
SP	Speed Perturbation.
STT	Speech-Recognition Output.
SVM	Support Vector Machines.
WER	Word Error Rate.

Chapter 7

Appendices

7.1 Appendix A: Trainer and TrainingArguments

```
if device == "cpu":
    extra_args = {
        "use_cpu": True,
        "fp16": False,
    }
else:
    extra_args = {
        "fp16": True,
    }
early_stopping_callback = EarlyStoppingCallback(
    early_stopping_patience=25,
    early_stopping_threshold=0.0001,
)

# Variables:
learning_rate = 3e-4 # HuBERT
learning_rate = 5e-5 # Whisper

weight_decay = 0.1
weight_decay = 0.01
weight_decay = 0.001
weight_decay = 0.0001

num_training_epochs = 300

# Training Arguments
training_args = TrainingArguments(
    save_total_limit=3,
    gradient_checkpointing=True,
    do_train=True,
    do_eval=True,
    save_steps=400,
    eval_steps=40,
    num_train_epochs=num_training_epochs,
    per_device_train_batch_size=4,
    per_device_eval_batch_size=4,
    learning_rate=learning_rate,
    warmup_steps=500,
    eval_strategy="steps",
    output_dir="./output",
    overwrite_output_dir=True,
    weight_decay=weight_decay,
    eval_accumulation_steps=4,
    gradient_accumulation_steps=4,
    **extra_args
)

# Trainer
trainer = Trainer(
    model=model,
    data_collator=data_collator,
    args=training_args,
    compute_metrics=compute_metrics,
    train_dataset=train_dataset,
    eval_dataset=valid_dataset,
    tokenizer=processor,
    callbacks=[CSVLoggerCallback('./H_validation_results_AMI.csv'),
    early_stopping_callback]
)
```

7.2 Appendix B: substitution word list with pattern frequency

7.3 Appendix C: laughter alignment operations for Whisper fine-tuned on AMI

7.4 Appendix D: laughter alignment operations for HuBERT fine-tuned on AMI

7.5 Appendix E: laughter alignment operations for Whisper fine-tuned on Switchboard

7.6 Appendix F: laughter alignment operations for HuBERT fine-tuned on Switchboard

Not Fine-tuned		
Substitution Words	**Example**	**Amount (% of Total)**
small words	'you', 'with', 'like'	80
repetition of small words	'yeahyeah', 'andthatthat', "sososoi'	13
"the" words	'thethe'	4
thinking / acknowledgement words	'uh', 'oh', 'um'	2
"okay"	'okayaya yokayay okayay'	1
Fine-tuned without laughter		
Substitution Words	**Example**	**Amount (% of Total)**
"okay"	'okayayay'	23
words with "th"	'thankth', 'whatthe', 'wasthink'	22
'yeah'	'yeah'	21
individual words	'mute', 'just', 'but'	19
thinking / acknowledgement words	'uh', 'um', 'mm'	16
Fine-tuned with laughter		
Substitution Words	**Example**	**Amount (% of Total)**
words with "<"	'a<laughter>', 'do<laughter>', 'see<laughter>'	62
small words	'got', 'and', 'just'	25
"th" words	'thankthank', 'that', 'the'	8
"yeah"	'yeah'	3
"okay"	'okayayay'	2

Table 7.1: Patterns of substitution word types for Whisper on AMI dataset

Not Fine-tuned		
Substitution Words	**Example**	**Amount (% of Total)**
combined words	'cometo', 'somethingthe', 'apartmentthe'	36
small "th" words	'the', 'they', 'that'	34
(almost) words	'if', 'her', 'yout'	21
double words	'yearyear', 'yeahyeah', 'yesyes'	9
Fine-tuned without laughter		
Substitution Words	**Example**	**Amount (% of Total)**
word + "th" word	'reallythe', 'hadthink', 'ithink'	66
small words	'we', 'and', 'see'	15
"yeah" + "t" word	'yeahtoo', 'yeahthere', 'yeahthe'	14
"you" + "t" word	'youto', 'youthe', 'youth'	5
Fine-tuned with laughter		
Substitution Words	**Example**	**Amount (% of Total)**
word with "<laughter>"	'the<laughter', 'a<laughter>', 'with<laughter>'	53
combined words	'canthat', 'youthey', 'acrossthe'	29
small "th" words	'that', 'than', 'them'	15
"yeah"	'yeah'	3

TABLE 7.2: Patterns of substitution word types for Whisper on the Switchboard dataset

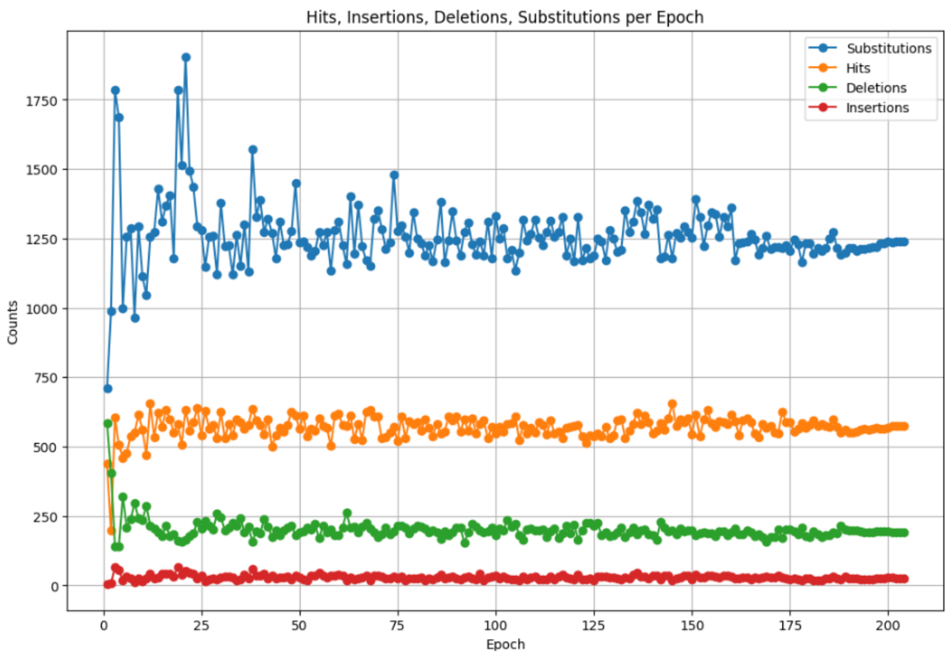


FIGURE 7.1: The counts of substitutions, hits, deletions and insertions of laughter events for Whisper fine-tuned on AMI for 200 epochs

Not Fine-tuned		
Substitution Words	**Example**	**Amount (% of Total)**
almost words	'bsurd', 'acepted', 'spece'	80
individual letters	'a', 'o', 'i'	20
Fine-tuned without laughter		
Substitution Words	**Example**	**Amount (% of Total)**
(almost) words	'grey', 'fine', 'time'	80
individual letters	'i', 'o', 'a'	11
thinking / acknowledgement words	'uhu', 'mhm'	3
'wel'	'wel'	3
"yeah"	'yeah'	3
Fine-tuned with laughter		
Substitution Words	**Example**	**Amount (% of Total)**
individual words	'volunters', 'agred', 'discusion'	50
words with "<" (laughter)	'o<', 'the<', 'a<'	40
individual letters	'i', 'a', 'm'	6
"yeah"	'yeah', 'yea'	2
"wel"	'wel'	2

TABLE 7.3: Patterns of substitution word types for HuBERT on the AMI dataset

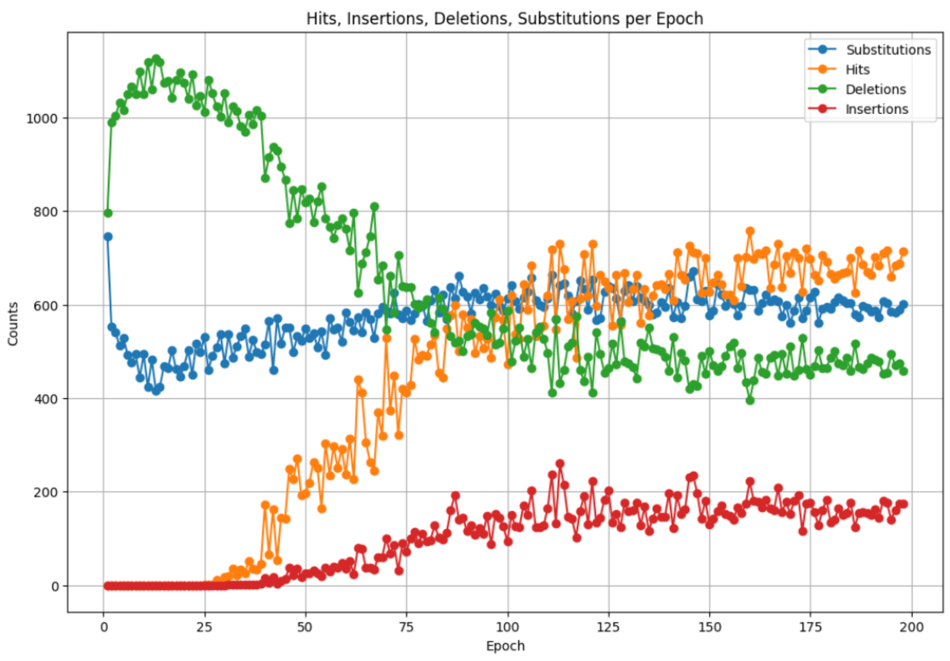


FIGURE 7.2: The counts of substitutions, hits, deletions and insertions of laughter events for HuBERT fine-tuned on AMI for 200 epochs

Not Fine-tuned		
Substitution Words	**Example**	**Amount (% of Total)**
words	'sometimes', 'caus', 'like'	52
individual letters	'i', 'm', 's'	24
letter a or h combination	'ha ha ha', 'aha', 'h', 'a'	10
filler words	'uh', 'um', 'ya'	8
words + "a" + "h"	'hahahh adiferent', 'hahuanow', 'hamerican'	6
Fine-tuned without laughter		
Substitution Words	**Example**	**Amount (% of Total)**
'th' words	"there", "thing", "the"	26
individual words	'hunderd', 'everybody', 'if'	20
single letters	's', 'a', 'm'	20
thinking / acknowledgement words	'umh', 'uhuh', 'uh'	17
"yeah"	'yeah', 'ityeah', 'yeahyeah'	10
"you"	'you', 'youl', 'youre'	8
Fine-tuned with laughter		
Substitution Words	**Example**	**Amount (% of Total)**
individual words	'something', 'than', 'thing'	54
words with laughter, or "<"	'<ay', '<rety', '<n'	39
sounds	'uhm', 'oh', 'uhu'	5
'yeah'	'yeah'	2

TABLE 7.4: Patterns of substitution word types of laughter for HuBERT on the Switchboard dataset

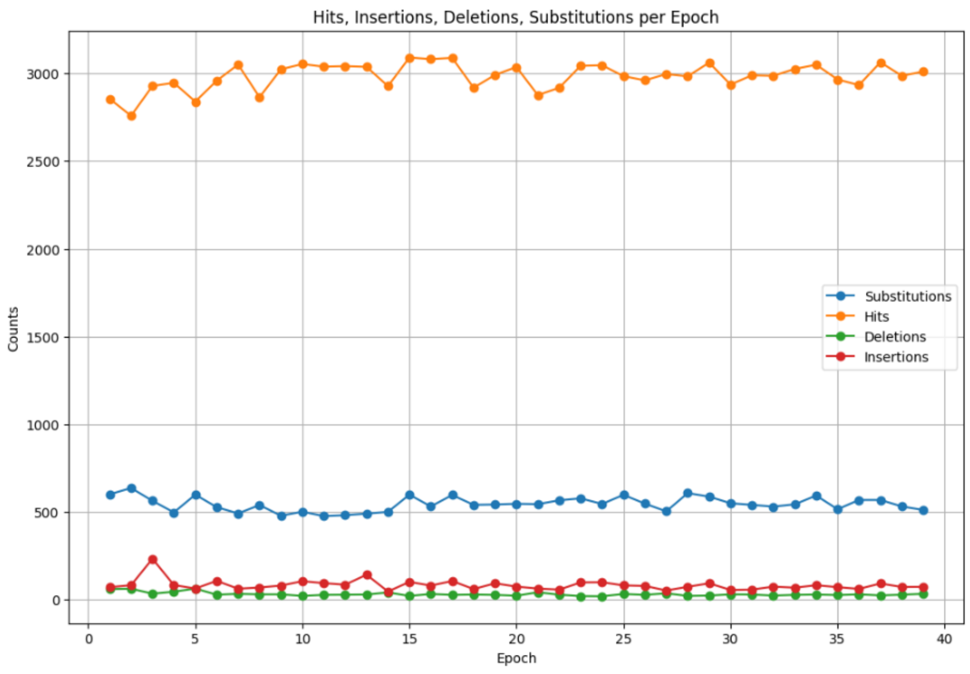


FIGURE 7.3: The counts of substitutions, hits, deletions and insertions for Whisper fine-tuned on SWB for 39 epochs

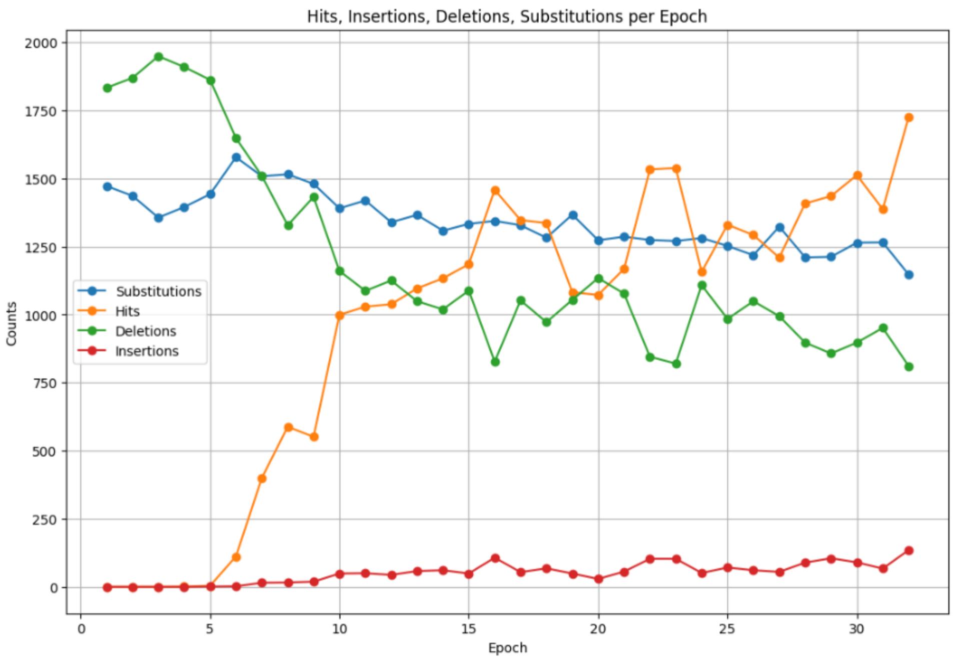


FIGURE 7.4: The counts of substitutions, hits, deletions and insertions for HuBERT fine-tuned on SWB for 32 epochs