Mohammad Ali JAUHAR

Robust Accented Speech Recognition

Master's Internship Report

to achieve the university degree of

Master of Science in Applied Mathematics

M2 MVA: Mathématiques, Vision, Apprentissage

submitted to

École normale supérieure Paris-Saclay

under the supervision of

Dr. Caroline Brun

Dr. Nikolaos Lagos

Dr. Salah Ait-Mokhtar

referent teacher

Dr. Chloe Clavel

Meylan, October 2024

NAVER LABS Europe

Abstract

The development of language has been crucial to human progress, enabling sophisticated communication and efficient knowledge sharing. Among communication methods, spoken language is one of the earliest and remains a core medium. Recent advancements in machine learning, particularly deep learning, have driven substantial progress in automatic speech recognition (ASR). However, despite the global reach of English, its many accents present significant challenges for ASR systems, which often struggle with accurate recognition across these variations. This limitation reduces the accessibility and inclusivity of ASR technology. In this work, we address the challenge of accented ASR through the lens of multilinguality and representation learning and introduce a novel contrastive learning technique via word-level offline triplet mining for accented ASR. We also identify a key limitation in current accented ASR methods related to shortcut learning and conduct a preliminary analysis to support further research in this area.

Acknowledgements

I would like to express my sincere gratitude to everyone who contributed to the successful completion of this research internship. This experience has been invaluable for my academic and professional growth, and it would not have been possible without the support, guidance, and encouragement of several individuals.

First and foremost, I am deeply thankful to my internship supervisors, *Dr. Caroline Brun*, *Dr. Nikolaos Lagos*, and *Dr. Salah Ait-Mokhtar*, whose expertise, insightful feedback, and continuous support guided me throughout this project. Their commitment to excellence and mentorship provided me with invaluable learning opportunities and inspired me to push the boundaries of my research capabilities.

I would also like to extend my appreciation to *STAG* group, especially *Dr. Marcely Zanon Boito* and *Dr. Ioan Calapodescu*, for creating a collaborative and intellectually stimulating environment. The discussions, feedback sessions, and collaborative efforts within the team greatly enriched my understanding and enhanced the quality of my research.

Special thanks to *Dr. Chloe Clavel* for agreeing to become my referent teacher and evaluate my work and provide valuable constructive feedback.

Special thanks also go to *Naver Labs Europe* for hosting me and *Agence Nationale de la recherche (ANR)* for providing the financial support needed to pursue this project.

Lastly, I am thankful to my family and friends, whose encouragement and understanding have been a constant source of motivation throughout my journey. Their unwavering belief in my abilities gave me the confidence to take on challenges and persevere.

Thank you all for your invaluable contributions to this internship experience.

Contents

1 Introduction

The development of language is a cornerstone in the evolution of mankind. It ushered into an era of more complex forms of communication, much more complex than other species. This enabled exchange of knowledge in a faster and more efficient way.

Among the different forms of human communication, speech developed first and much earlier than written language and is a fundamental forum of communication. In the current age, we use spoken and written communication almost equally. With the advent of machine learning and in particular deep learning, researchers and engineers have developed many technologies for automatic speech recognition (ASR). It is reflected in many modern digital systems such as voice assistants in motor vehicles, personal assistants such as Amazon's Alexa, Apple's Siri, and in many automated response centres.

English is a global language and the language for global trade and science. Its importance can't be understated. However, being a global language also entails considerable variance in the way the language is spoken. We generally refer to these variations as accents. More concretely, accents arise due to variations in pronunciation, intonation, and other speech characteristics caused by geographical, cultural, or linguistic differences [Lippi-Green, 2012]. However, the current automatic speech recognitions systems are not robust to these accent variations and show a degraded performance on accented speech which significantly decreases the usefulness of such systems. As we are moving towards more digitalised society every day, it is imperative that we design our technologies to be robust and non-biased against all kind of under-represented groups.

In this work, we look at robust accented speech recognition with the lenses of *multilinguality* and *representation learning*. Furthermore, during our research, we discover a *fundamental drawback* of current accented ASR techniques that renders them ineffective in real-world which we discuss extensively as well.

2 Background and Related Work

2.1 Automatic Speech Recognition

Automatic speech recognition, commonly referred to as ASR, is a speech-to-text technology. It converts human speech into text for downstream applications such as in voice assistants, search engines, emergency response centres, video transcriptions, etc. Implementation-wise, the technology has grown up from statistical models to hybrid statistical-neural network models and deep neural networks. Among neural network based implementations, it has evolved from convolutional neural networks (CNN) [Peddinti, Povey, and Khudanpur, 2015], recurrent neural networks (RNN), long-short term memory networks (LSTM) [H. Xu et al., 2018], and more recently, transformers [Kim, Hori, and S. Watanabe, 2017].

ASR is a supervised learning task where we have speech audios and corresponding transcripts in the dataset. However, the initial prediction is done on character level, upon which different decoding techniques such a greedy decoding, beam search, etc. are used for generating sentences. Therefore, we define ASR as a multi-class classification task where the task is to label each timestep in the input speech audio to one of the classest. The number of classes is a design choice and is called the vocabulary and is usually composed of the letters for the language(s) under consideration, numerics, and some punctuations. Formally,

Let $(x^i, Y^i)_{i=1}^N$ be the dataset with x^i as input speech and Y^i as corresponding transcription. Also, let K be the vocabulary size. Then each input speech consists of L time-steps as $x^i \in R^{LxD} = x_1^i, x_2^i, ..., x_L^i$ and the corresponding transcription of length T as $y^i \in R^{TxK} = y_1^i, y_2^i, ..., y_T^i$.

In practice, the input audio is much larger than the length of the transcription. Which results into $T << L$.

Finally, it should be noted that the length of speech audio and transcriptions vary for each sample.

As there is no strict alignment between the speech audio and the corresponding characters in the transcript. For example, in one audio, the speaker

may speak very slowly which would result into the same character taking more time-steps in the input audio than for a fast speaker. This makes common optimisation objectives such as cross entropy infeasible. Therefore, objectives such as connectionist temporal loss (CTC) [Graves et al., 2006] is often used.

As with all such classification problems, ASR technologies have been historically limited by the amount of available labeled data. Hence, many current state-of-the-art (SOTA) methods finetune on models pretrained on a large amount of unlabeled data. These include speech models of the Wav2Vec 2.0 family (Wav2Vec 2.0 small [Baevski et al., 2020], Wav2Vec2.0 large, XLSR-53 [Conneau et al., 2020], XLSR-128 [Babu et al., 2021], MMS-1407 [Pratap, Tjandra, et al., 2023]), HuBERT [Hsu et al., 2021] family (HuBERT base, HuBERT large ,mHuBERT-147 [Boito et al., 2024]), and Whisper [Radford et al., 2022] family.

2.2 Biases in ASR

ASR systems are deployed in households, industry, and critical emergency systems. Technology such as Apple's Siri, Amazon's Alexa, automotive vehicle's infotainment systems, etc. are all everyday use of ASR systems. There have been many anecdotal evidences that speak about the difficulties the voice assistants have in recognising the speech of certain demographies properly. [Koenecke et al., 2020, Qian et al., 2017, Tatman and Kasten, 2017, Alsharhan and Ramsay, 2020] This ranges from a bias against non-native speakers, children and elderly people, or even male population. Furthermore, automated (and real-time) transcriptions during video-conferencing is now a commonly provided service. However, as the underlying technology is ASR, these systems are also not robust against certain demography. It has been shown that ASR systems struggle specifically against certain accents, for example. While the two previously mentioned use cases could lead to real harm for many people, including problems of accessibility [Scharenborg, 2021], usage of non-robust ASR systems in emergency services (which has seen an increased adoptation in recent times) could put people in life-and-death situations [Feng et al., 2024, Markl, 2023].

There are many factors that can cause bias in ASR systems. Some of them are:

Under-representation of a speaker group in the training data An unbalanced dataset is a common cause of bias/degradation in performance for the underrepresented group in most machine learning models. The generally

extends to ASR systems as well where, for example, in English ASR datasets, there is often a stronger representation of US speakers compared to Scottish speakers which generally results into a degraded performance for Scottish accent.

Intra-group variability Even within a group, such as american speakers, there is often variability in the representation of different age groups, for example. This leads to an ASR system that doesn't perform equally well, for example, against the speech of elderly people, compared to teenagers.

Type of speech and recording conditions Many dataset have primarily read speech or recordings from news broadcasts. While the quality of such audio is better, they aren't representative of spontaneous speech spoken by most people. Therefore, ASR systems developed on such dataset show degraded performance in the real-world. Also, the audio recorded by a person from global south may not be of as good quality as ones recorded in a developed nation, which could be a source of bias against such demography.

Model Architecture/Algorithm Some research also look into specific model architecture as a potential cause for bias. However, this area is still highly unexplored.

2.3 Accented Speech Recognition

Accents in speech usually refers to the way certain speaker groups pronounce words. A large number of factors contribute to an accent. While primarily, it may be linked to the native tongues of the speaker, it be also be dependent on other factors such as age, educational background, and socio-economic status. Numerous work have tried tackling the issue and with varied success. Speech signal is assumed to be composed of two orthogonal components: content information and other speaker information. Under this framework, accent is a speaker information. Following this, we may divide current accented ASR work considering two complementary approaches, a) accent-agnostic approach, and b) accent-aware approach.

Accent-agnostic approach These approaches aim to do away with the accent information and focus solely on content information. Different techniques have been used for removing accent information. This includes using accent classifiers, adversarial training, and constrastive loss objectives. Concretely, some of the methods use an accent classifier as disciminator in an adversarial training setting or using a pre-trained accent classifier in other training regimes while other works use accent agnostic objectives such as cosine loss and contrastive loss to push models away from accents [Han et al., 2021].

Accent-aware approach These approaches add accent-specific information to the model in order to improve the ASR performance. The approaches includes: a) techniques for capturing a joint content and accent embedding in the same embedding space employed by methods such as multi-task learning [Zhang et al., 2021] and domain adversarial training [Das et al., 2021], and b) techniques for capturing accent encodings in separate units such as i-vectors, dialect symbols, accent-specific codebooks [Prabhu, Gupta, et al., 2024, Prabhu, Jyothi, et al., 2023], and LoRA adapters [Nassereldine et al., 2024].

Shortcomings of the current approaches:

- The current state-of-the-art approaches encode accent information in accent-specific units. This means if we have a new accent that we want to consider for training, the training has to be done again from scratch.
- Accent classification is very challenging for speaker disjoint dataset. However, none of the current state-of-the-art works consider this scenario.

2.4 Datasets

The pretrained speech models that we consider in this work has been trained on a variety of speech corpus. Here, we briefly present some of the major corpus that we encountered while exploring accented speech recognition. A detailed information is also available in Table 2.1 for the languages in each of the corpora.

1. MMS-Lab-U [Pratap, Tjandra, et al., 2023] : 1,362 languages with 55K hours

2. Multilingual Librispeech (MLS) [Pratap, Q. Xu, et al., 2020] : 8 languages, 50K hours of speech

3. Common Voice v9.0 (CV) [Ardila et al., 2020] : 89 languages, 8.8 hours of speech

4. VoxLingua-107 (VL) [Valk and Alumäe, 2020] : 107 languages with 5.3K hours

5. BABEL [Gales et al., 2014] : 17 languages with about 1K hours of speech

6. VoxPopuli (VP) [Wang et al., 2021] : 371K hours of unlabelled speech in 23 languages

7. MCV100 [Prabhu, Jyothi, et al., 2023] : 100 hours of labelled English speech created from CV specifically for accented ASR training and evaluation

8. Libri-light [Kahn et al., 2020] : 60K hours of English speech data

Table 2.1: A Overview of the Languages in the Corpora used by the Speech Models under Study

Dataset	No. of Languages	List of Languages
MLS [Pratap, Q. Xu, et al., 2020]	08	Dutch, English, French, German, Italian, Polish, Portuguese, Spanish
CV (3.6K) [Ardila et al., 2020]	36	Arabic, Basque, Breton, Chinese (CN), Chinese (HK), Chinese (TW), Chuvash, Dhivehi, Dutch, English, Esperanto, Estonian, French, German, Hakh-Chin, Indonesian, Interlingua, Irish, Italian, Japanese, Kabyle, Kinyarwanda, Kyrgyz, Latvian, Mongolian, Persian, Portuguese, Russian, Sakha, Slovenian, Spanish, Swedish, Tamil, Tatar, Turkish, Welsh
BABEL [Gales et al., 2014]	17	Assamese, Bengali, Cantonese, Bebuano, Georgian, Haitian, Kazakh, Kurmanji, Lao, Pashto, Swahili, Tagalog, Tamil, Tok, Turkish, Vietnamese, Zulu
VoxPopuli [Wang et al., 2021]	23	Bulgarian, Czech, Croatian, Danish, Dutch, English, Estonian, Finnish, French, German, Greek, Hungarian, Italian, Latvian, Lithuania, Maltese, Polish, Portuguese, Romanian, Slovak, Slovene, Spanish, Swedish
CV (7K) [Ardila et al., 2020]	60	CV (3.6K) + upper Serbian, Romanian, Frisian, Czech, Greek, Romansh Vallader, Polish, Assamese, Ukrainian, Maltese, Georgian, Punjabi, Odia, Vietnamese

VoxLingua107 [Valk and Alumäe, 2020]	107	Abkhazian, Afrikaans, Amharic, Arabic, Assamese, Azerbaijani, Bashkir, Belarusian, Bulgarian, Bengali, Tibetan, Breton, Bosnian, Catalan, Cebuano, Czech, Welsh, Danish, German, Greek, English, Esperanto, Spanish, Estonian, Basque, Persian, Finnish, Faroese, French, Galician, Guarani, Gujarati, Manx, Hausa, Hawaiian, Hindi, Croatian, Haitian, Hungarian, Armenian, Interlingua, Indonesian, Icelandic, Italian, Hebrew, Japanese, Javanese, Georgian, Kazakh, Central Khmer, Kannada, Korean, Latin, Luxembourgish, Lingala, Lao, Lithuanian, Latvian, Malagasy, Maori, Macedonian, Malayalam, Mongolian, Marathi, Malay, Maltese, Burmese, Nepali, Dutch, Norwegian Nynorsk, Norwegian, Occitan, Panjabi, Polish, Pushto, Portuguese, Romanian, Russian, Sanskrit, Scots, Sindhi, Sinhala, Slovak, Slovenian, Shona, Somali, Albanian, Serbian, Sundanese, Swedish, Swahili, Tamil, Telugu, Tajik, Thai, Turkmen, Tagalog, Turkish, Tatar, Ukrainian, Urdu, Uzbek, Vietnamese, Waray, Yiddish, Yoruba, Mandarin Chinese
MMS-lab [Pratap, Tjandra, et al., 2023]	1107	**A large number of languages**
Aishell/Aishell-3 [Shi et al., 2015]	1	Mandarin
Bible-TTS [Meyer et al., 2022]	10	Ewe, Hausa, Kikuyu, Lingala, Luganda, Luo, Chichewa, Akuapem Twi, Asante Twi, Yoruba
ClovaCall [Ha et al., 2020]	1	Korean
GoogleTTS	10	Javanese, Khmer, Nepali, Sundanese, Afrikaans, Sesotho, Setswana, isiXhosa, Bangladeshi Bengali, Indian Bengali
IISc-MILE [A, Pilar, and G (2022b) and A, Pilar, and G (2022a)]	2	Tamil, Kannada
Japanese Versatile Speech (JVS) [Takamichi et al., 2019]	1	Japanese
Kokoro [Lida, 2022]	1	Japanese

Kosp2e [Cho et al., 2021]	1	Korean
MediaSpeech [Kolobov et al., 2021]	1	Turkish
Samromur [Hedström et al., 2022]	1	Icelandic
THCHS-30 [Dong Wang, 2015]	1	Chinese
THUYG-20	1	Uyghur

Representation Learning

Representation learning is the technique for learning feature embeddings that can reflect the intrinsic characteristics of underlying data through modeling [Bengio, Courville, and Vincent, 2013, Le-Khac, Healy, and Smeaton, 2020, Chen et al., 2020]. It could be thought of as an abstract representation of a signal, be it of any modality. The motivation behind representation learning is that if we could capture the essence of some input signal, we could use the learned embeddings for downstream tasks without requiring as much data as required otherwise. In the case of images, for example, the downstream task could be image classification or image retrieval. In the case of speech, for example, the downstream task could be speech recognition (ASR).

One of the most prominent use case of representation learning is that it enables transfer learning. A speech model, for example, although pre-trained on unlabeled data, captures the essence of speech to the extent that we are able to fine-tune it on specific small datasets and achieve state-of-the-art performances across tasks [Jiang et al., 2020]. However, speech signals face a peculiar issue. Compared to other signals such as image and natural language, speech signals do not have any explicit segmentation. It could be argued that individual phones could be considered as an atomic units for speech, but it's very tedious to label each speech audio into phones by human annotators and therefore, infeasible in practice. Therefore, to do any representation learning of speech signal, one of the primary tasks is to segment speech into a fixed-size speech units. In the following subsections, we take a look at some current state-of-the-art speech models that have been pretrained to learn speech representations by innovating around this fundamental limitation. In particular, we look at two closely related families of speech models, Wav2Vec 2.0 family and HuBERT family.

Wav2Vec 2.0 Family

Wav2Vec 2.0 is a family of models that follows same underlying architecture and training regime. It varies across sizes and number of languages used during pretraining. It includes monolingual models, Wav2Vec 2.0 small and large, and multilingual models, XLSR-53, XLSR-128, and MMS-1407. We take a brief look into them in the following sections.

Wav2Vec 2.0: A Framework for Self-Supervised Learning of Speech Representations

Wav2Vec 2.0 [Baevski et al., 2020] is an English language pre-trained speech model. As shown in Fig. 2.1, the Wav2Vec 2.0 proposes a CNN feature (waveform) encoder that encodes the raw audio input into discrete speech units and a transformer context encoder that encodes the discrete speech units to get speech embeddings. As mentioned previously, since there are no explicit segmentation available for speech signal, The feature encoder is trained to generate the discrete speech units. For this, it uses two codebooks with 320 possible codewords in each group. Furthermore, a quantisation network is learned to sample codewords from each codebook using a Gumbel Softmax. The quantised labels are then used as pseudo-labels for the training of the transformer encoder.

Formally, the Wav2Vec 2.0 architecture is as follows:

1. A convolutional feature encoder, $f : X \rightarrow Z$ that maps raw audio X to latent speech representation $z_1, z_2, ..., z_T$

2. A transformer context encoder, $g : Z \rightarrow C$ that encodes the latent representation to output context representations $c_1, c_2, ..., c_T$

3. During training, the latent feature encoder representations are discretised to $q_1, q_2, ..., q_T$ with a quantisation module $Z \rightarrow Q$ to form the pseudo-targets for self-supervised training.

The encoder training is inspired by BERT[Devlin et al., 2019] wherein it masks 50% of the discrete units. The training objective is however slightly different. Wav2Vec 2.0 employs contrastive loss where 100 negative distractors are uniformly sampled from other positions of the same audio for each masked position. The network then compares the cosine similarity of the projected embedding and positive target (sampled from the quantisation network) , along with all the negative distractors and penalises similarity between the projected embedding and negative distractor while encouraging similarity between the projected embedding and the positive target.

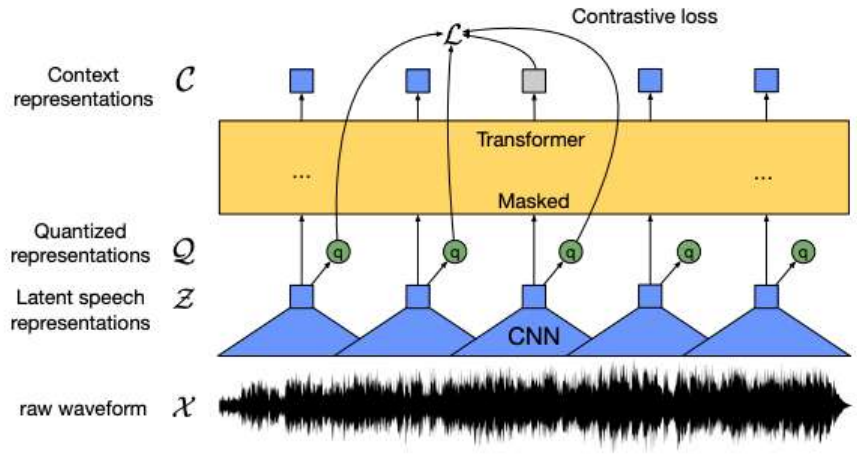

Figure 2.1: Wav2Vec 2.0 monolingual architecture (source: `https://arxiv.org/pdf/2006.11477`)

Wav2Vec 2.0 is available in small and large sizes, with the small version being 94 M parameters and the large version being 317 M parameters. Architecturally, the difference in size comes from the number of transformer layers in the respective context encoders.

Unsupervised Cross-Lingual Representation Learning for Speech Recognition

Referred to as XLSR-53 [Conneau et al., 2020], it is a multilingual model based on the Wav2Vec 2.0 architecture. Its version with highest number of languages is called XLSR-53 and has been trained on speech data of 53 different languages using the MLS, CV, and BABEL corpora. It has around 317M parameters. To represent multilinguality, it maps speech from different languages to discrete speech units in the same space as shown in Fig. 2.2.

XLS-R: Self-supervised Cross-lingual Speech Representation Learning at Scale

Referred to as XLSR 128 [Babu et al., 2021], it is also a multilingual speech model with an underlying Wav2Vec 2.0 architecture. As the name suggests, it has been trained on 128 different languages. It used the following corpora: VoxPopuli, LibriVox, Common Voice, VoxLingua, and BABEL. It comes in

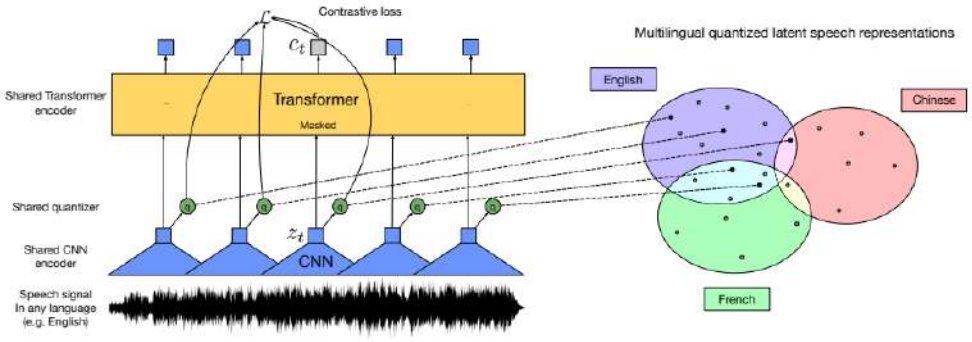

Figure 2.2: Wav2Vec 2.0 multilingual architecture (source: `https://arxiv.org/pdf/2006.13979`)

two different model sizes: a smaller 317M parameters and a larger 965M parameters model.

Scaling Speech Technology to 1,000+ Languages

Referred to as MMS [Pratap, Tjandra, et al., 2023] (massively multilingual speech), it is a sub-family of models that follows the Wav2Vec 2.0 architecture while being pretrained on a series of increasing number of languages. It presents a pretrained speech model on 1406 languages, one finetuned ASR model trained on 1107 languages, and other models trained on much larger number of languages with small quantity of data useful for relatively easier tasks such as language identification.

Its self-supervised pre-trained models come in two sizes, 317M parameters and 965M parameters. The pretraining data is 491K hours of speech in 1406 languages. More specifically, it uses the MMS-Lab-U, LibriVox, Common Voice, VoxLingua, BABEL, and VoxPopuli corpora.

HuBERT Family

HuBERT: Self-Supervised Speech Representation Learning by Masked Prediction of Hidden Units

HuBERT [Hsu et al., 2021](hidden-units BERT) follows the Wav2Vec 2.0 architecture. It differs however, in the way it finds discrete speech units which it refers to as *hidden units* and in the training loss objective.

Hidden units discovery in HuBERT follows a two step process. It's first iteration is offline and is done by performing k-means clustering with 100 cluters on 39-dimensional MFCC features over the training set. In the second iteration (for base variant) and further iterations, it refines the clusters by performing K-means with 500 clusters on the latent features extracted from the HuBERT model pre-trained on previous iterations. For the context encoder training, similar to wav2vec 2.0, it masks 50% units and computes softmax loss on the embeddings corresponding to masked and unmasked units separately and is combined together linearly with parameters α. The softmax loss corresponding to unmasked units mimic acoustic modeling while the softmax over masked units mimic language modeling. TThis makes it a combination of speech and language modeling, thus making it particularly suitable for finetuning for ASR.

Similar to Wav2Vec 2.0, HuBERT also comes in two sizes, base and large. HuBERT base model is trained on 960 hours of English speech data from Librispeech whereas HuBERT large has been trained on 60K hours of Libri-light (a subset of Librivox).

mHuBERT-147: A Compact Multilingual HuBERT Model

mHuBERT-147 [Boito et al., 2024] is a massive multilingual speech model that follows HuBERT architecture but improves upon various aspects of it. First of all, it present a multilingual model compared to HuBERT's monolingual model. Furthermore, mHuBERT uses a more efficient clustering algorithm to scale better over a much larger dataset. To achieve multilinguality, it employs a multi-lingual batching strategy during training and achieves state-of-the-art performances on various SUPERB benchmarks [Yang et al., 2021] and shows strong competition against much larger models.

It follows the same architecture as HuBERT-base and is 94 M parameters. Furthermore, it has been trained with 90K hours of clean multilingual speech data from various sources as detailed in Table 2.2.

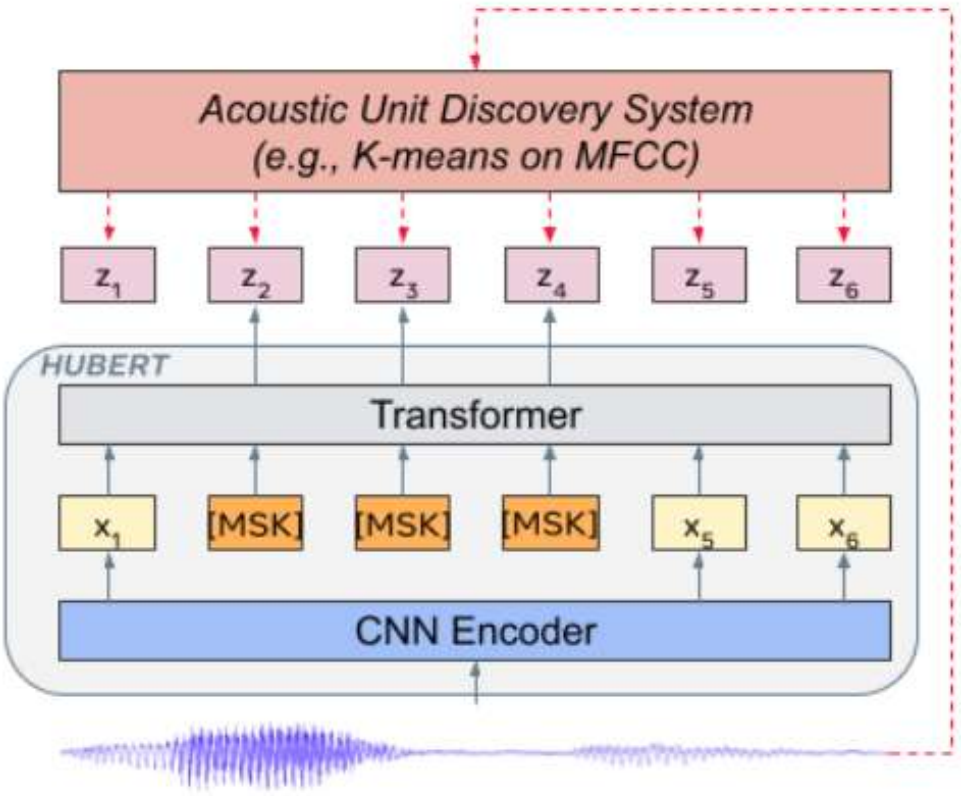

Figure 2.3: HuBERT architecture and pre-training overview (source: `https://arxiv.org/pdf/2106.07447`)

Model	Size	Pretraining datasets and hours
Wav2Vec 2.0 base	94 M	Librispeech (960h), LibriVox (53.2K)
HuBERT	94 M	Librispeech (960h)
mHuBERT-147	94 M	Aishell (178h), AISHELL-3 (85h), BibleTTS, ClovaCall (80h), CV, Google TTS, IISc-MILE, Japanese Versatile Speech (JVS), Kokoro, Kosp2e, Media Speech, MLS, Samromur (2.2K), THCHS-30, THUYG-20, VoxLingua107, VoxPopuli
XLSR-53	317 M	MLS (50.7K), Common Voice (3.6K), Babel (1.7K)
XLSR-128	317 M	VoxPopuli (372K), MLS (50.7K), CV (7K), VoxLingua107 (6.6K), Babel (1.7K)
MMS-1406	317 M	MMS-lab (49K), MLS (50.7K), Fleurs, CV (7K), VoxLingua107 (5.3K), Babel (1.7K), VoxPopuli (371K)

Table 2.2: A Summary of the speech models under consideration

3 Mozilla Common Voice MCV-100

Dataset Overview

MCV-100 [Prabhu, Jyothi, et al.] is an English language dataset with accent annotations derived from the Mozilla Common Voice corpus [Ardila et al., 2020]. The accent annotations are self-reported by the speakers. It has 05 accents in the training and dev split, hereafter referred to as *seen accents* and 09 additional accents in the test split, hereafter referred to as *unseen accents*. Two important characteristics of the dataset are:

- Train, Dev, and Test splits are speaker disjoint. This means there are no common speakers in any split.
- Train, Dev, and Test splits are not utterance disjoint. This means that some of the transcriptions/utterances are spoken by speakers across accents and across the splits.

Dataset	Train (h)	Dev (h)	Test (h)	
			Seen	Unseen
MCV-100	100	17.26	8.35	9.06

Table 3.1: Number of hours in each split

Training Split Characteristics

The training split of MCV-100 consists of 72,000 utterances. However, there is significant imbalance in terms of the duration of the utterances, speaker representation, accent representation, and transcriptions representation. In the following section, we take a look at these characteristics.

Seen Accents	Data (h)			Unseen Accents	Test (h)
	Train	Dev	Test		
Australia	6.95	4.33	0.46	Africa	1.71
Canada	6.79	1.16	1.21	New Zealand	2.11
England	19.51	3.22	1.65	The Philippines	0.90
Scotland	2.69	0.23	0.16	Singapore	0.64
US	64.12	8.32	4.87	Wales	0.27
				Malaysia	0.39
				Hong Kong	0.52
				India	0.58
				Ireland	1.94

Table 3.2: Number of hours per accent for each split

Metric	count	mean (s)	std (s)	min (s)	25% (s)	50% (s)	75% (s)	max (s)
Value	72500	4.97	1.66	0.76	3.74	4.87	6.04	20.78

Table 3.3: Durations of utterances in the training split.

Duration Statistics

We observe that the average utterance duration is just about 05 seconds with a standard deviation of 1.66 seconds. However, the shortest utterance is just 0.76 seconds long while the longest utterance lasts over 20 seconds. A detailed description is provided in Table 3.1.

Utterances Per Speaker

The 72K utterances in the training split are spoken by just over 5K different speakers. This means some of the speakers have significantly more representation than other speakers and if a model struggles against a particular speaker considerably, this may skew the average score significantly. Table 3.4 and Figure 3.2 presents a detailed picture.

As observed in Table 3.4, there are very few distinct speakers in the dataset and there are some speakers that together constitute a significant number of utterances. The top 25% most represented speakers constitute around 60% of the dataset. This poses significant challenges when learning accent

Metric	count	mean	std	min	25%	50%	75%	max
Value	5835	12.42	55.00	1	1	3	9	2693

Table 3.4: Number of utterances per speaker. We notice a highly imbalanced dataset.

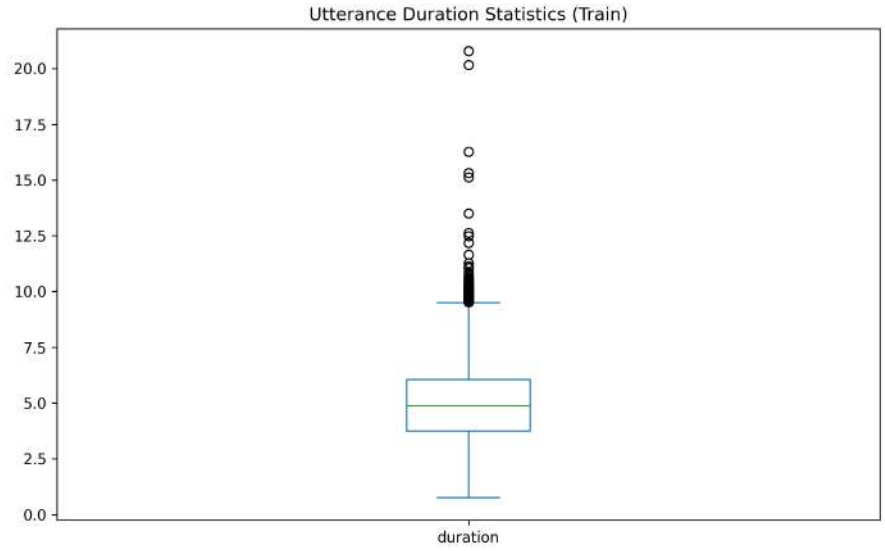

Figure 3.1: Utterance duration statistics for training split

Metric	count	mean (s)	std (s)	min (s)	25% (s)	50% (s)	75% (s)	max (s)
Value	14025	4.43	1.72	1.08	3.14	4.17	5.47	22.68

Table 3.5: Durations of utterances in the dev split

features as the model instead learns to overfit on speaker characteristics during training.

Dev Split Characteristics

The dev split of MCV-100 consists of 14,025 utterances. However, similar to the training split, there is a significant imbalance in terms of the duration of the utterances, speaker representation, accent representation, and transcriptions representation. In the following section, we take a look at these characteristics.

Duration Statistics

The average duration of utterances is 4.43 seconds, slightly lower compared to the training split, with a standard deviation of 1.72 seconds. The shortest utterance is just over 1 second while the longest utterance is around 22 seconds long. A details statistics is presented in Table 3.5.

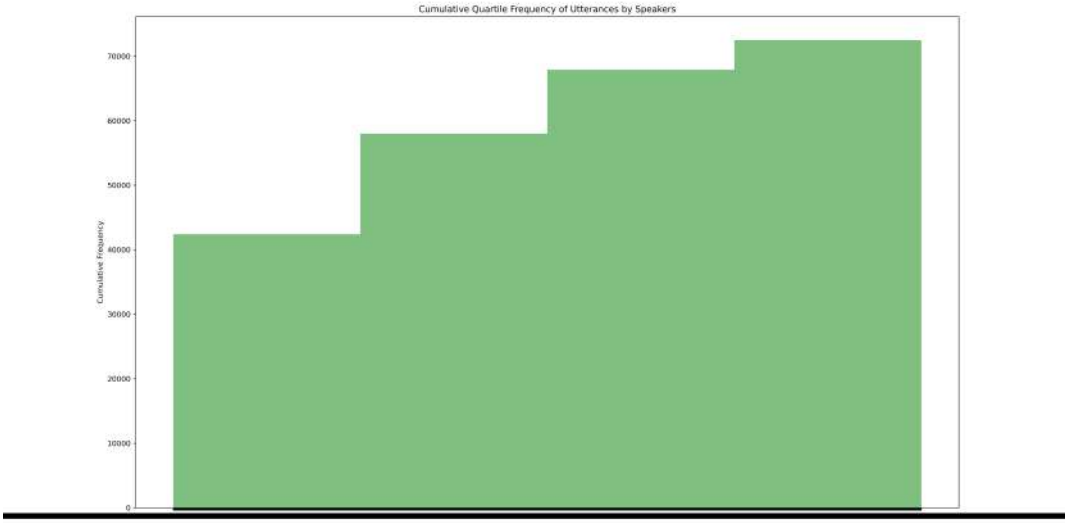

Figure 3.2: Cumulative Quartile frequencey of utterances by speakers in training split. We can observe that the top 25% of the speakers constitute more than half of the utterances.

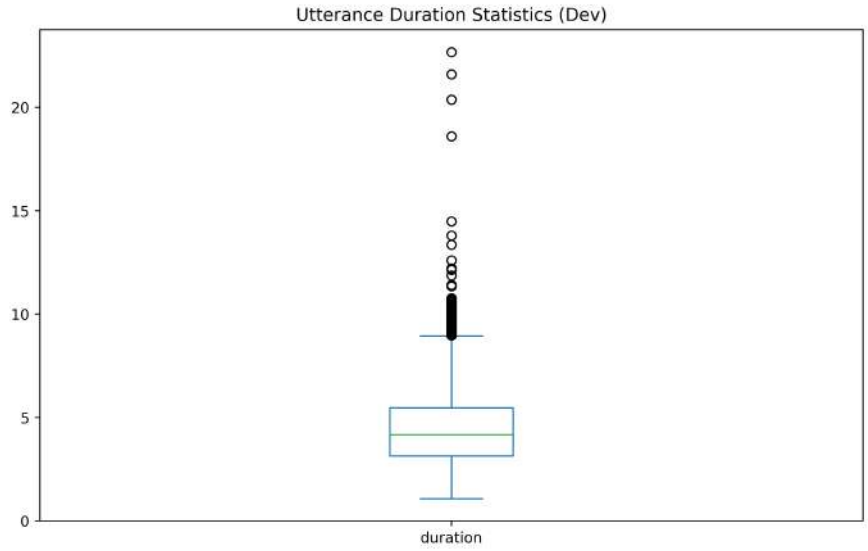

Figure 3.3: Utterance duration statistics for dev split

Metric	count	mean	std	min	25%	50%	75%	max
Value	990	14.16	80.77	1	3	5	13	2473

Table 3.6: Number of utterances per speaker. Similar to the training split, we notice a highly imbalanced dataset

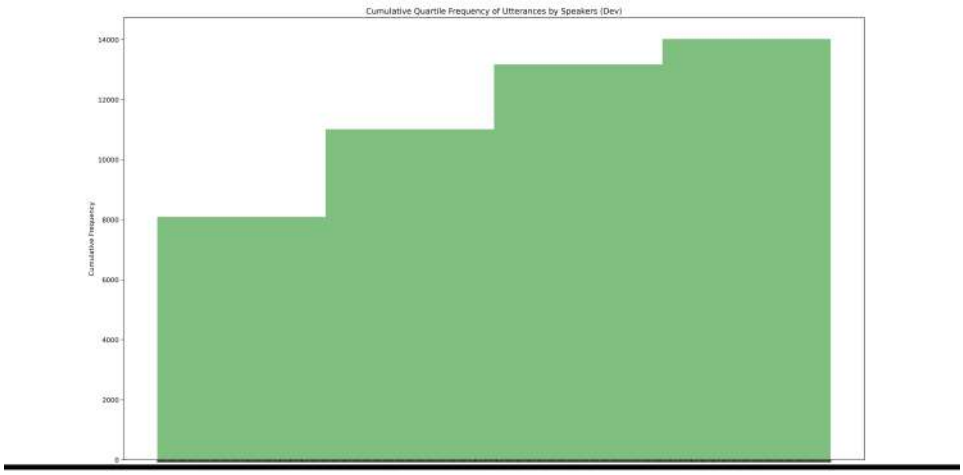

Figure 3.4: Cumulative Quartile frequencey of utterances by speakers in the dev split. We again observe that the top 25% of the speakers constitute more than half of the utterances.

Utterances Per Speaker

MCV-100 dev split has around 14k utterances spoken by just around 1K different speakers. This also means some of the speakers have significantly more representation than other speakers and if a model struggles against a particular speaker considerably, this may skew the average score significantly. In particular, one of the Australian speakers speaks 2473 utterances and has shown to adversely affect the average performance of Australian accent during evaluation. Table 3.1.2 and Figure 3.4 presents a detailed picture.

As observed in Table 3.1.2, there are very few distinct speakers in the dataset and there are some speakers that together constitute a significant number of utterances. We could also look at the cumulative quartile frequency in Figure 3.4 which shows that few speakers form the majority of the utterances in the dev split.

Metric	count	mean (s)	std (s)	min (s)	25% (s)	50% (s)	75% (s)	max (s)
Value	15714	4.47	1.69	0.93	3.14	4.22	5.56	14.40

Table 3.7: Duration of utterances in the test split

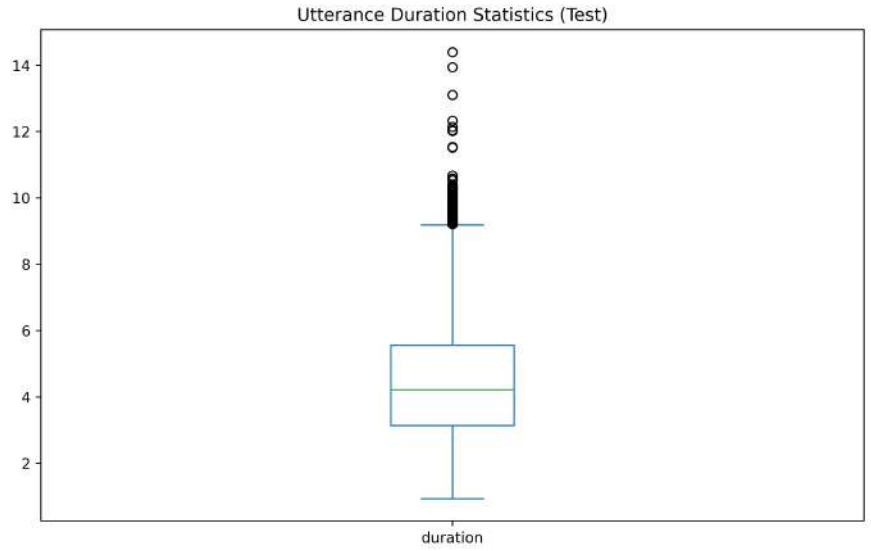

Figure 3.5: Utterance duration statistics for the test split

Test Split Characteristics

The test split of MCV-100 consists of 15,714 utterances. However, similar to the training and dev split, there is a significant imbalance in terms of the duration of the utterances, speaker representation, accent representation, and transcriptions representation. In the following section, we take a look at these characteristics.

Duration Statistics

The average duration of utterances is 4.47 seconds, slightly lower compared to the training split, with a standard deviation of 1.69 seconds. The shortest utterance is just a little less than 1 second while the longest utterance is around 14.40 seconds long. A details statistics is presented in Table 3.6.

Metric	count	mean	std	min	25%	50%	75%	max
Value	1890	8.31	21.13	1	2	4	8	476

Table 3.8: Number of utterances per speaker. Similar to the training and dev split, we notice a highly imbalanced dataset

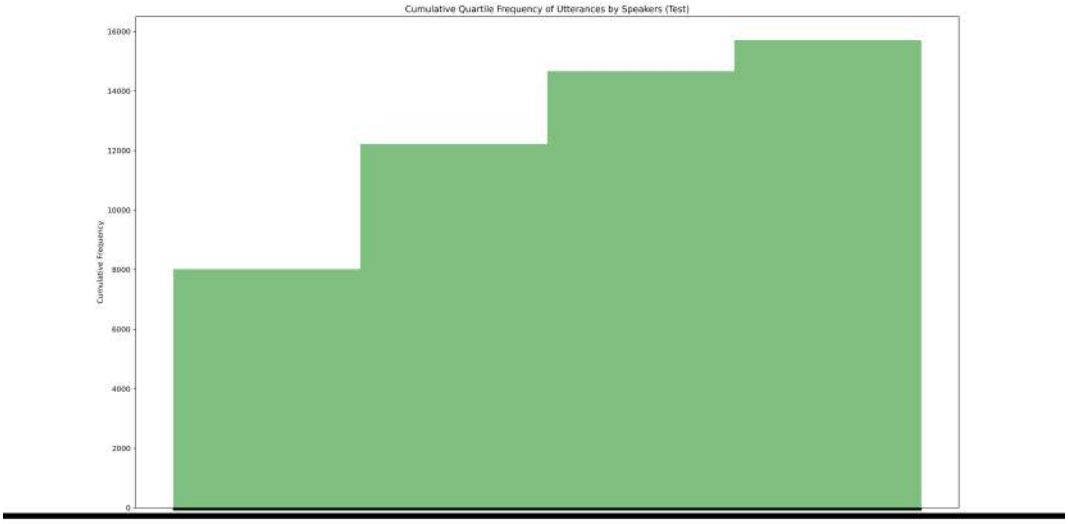

Figure 3.6: Cumulative Quartile frequencey of utterances by speakers in the test split. We again observe that the top 25% of the speakers constitute more than half of the utterances.

Utterances Per Speaker

MCV-100 test split has around 15K utterances spoken by around 1.9K different speakers. Some speakers have significantly more representation and if the model struggles against those speakers, it would adversely affect the performance. Table 3.7 and Figure 3.6 sheds more light on this.

Similar to train and dev splits, we observe that few speakers constitute the majority of the utterances in the test split. This further skews the test performance and hence, should be considered when analysing results.

Summary

We observe that MCV-100 dataset is a highly imbalanced dataset across accent and speaker representation. In the training split, American accent

has a significantly more representation, covering over 60% of the dataset. In terms of speakers, across the training, dev, and test split, we observe that just $\frac{1}{4}$ of the speakers contribute to around 60% of the utterances. However, the trend across the splits in terms of accent and speaker representation is similar and therefore, the dataset dev split serves as a good benchmark for predicting test performance.

4 Multilingual Speech Models and Accented Speech Recognition

Background

Accent in speech usually refers to the way certain speaker groups pronounce words. It is believed to be strongly influenced by the native tongues of the speaker. Multilingual speech models are trained on multiple different languages across language families and sub-families. Furthermore, multilingual speech models attempt to build a unified acoustic and language model. As the accent variation is represented through native tongues, it should be imperative that multilingual models are better equipped for accented speech.

Multilingual speech models have revolutionised speech processing tasks, particularly for low-resource languages [Conneau et al., 2020, Pratap, Tjandra, et al., 2023]. Prior work show strong cross-lingual transfer learning capabilities. Learned representations from multilingual models have also been shown to transfer well to unseen languages.

Usefulness of multilingual models for ASR tasks in different, particularly low resource languages has been explored in great detail before [Yadav and Sitaram, 2022]. [Vu, 2014] explored multilinguality for accented ASR and found that multilinguality helps in accented ASR performance significantly. However, the models that were considered in the work are not neural network based. In recent times, however, as per our best knowledge, there have been no work that explores multilinguality for accented ASR.

To address this gap in literature and in our attempt for developing robust accented speech recognition systems, we study the relationship between multilinguality and accent robustness in speech models. We vary two different parameters, a) size of the model, and b) the number of languages in pre-training. In the following sections, we look at our analysis setup, experiments, results, and conclusions.

Dataset

We use MCV-100 [Prabhu, Jyothi, et al., 2023] , which is an English language dataset derived from Common Voice Corpus [Ardila et al., 2020] and contains self-reported accent annotations for all our experiments. The dataset consists of three splits: *train*, *dev*, and *test*. We train on the *train* split and evaluate on the *dev* split. We only evaluate the performance on the *test* on the best checkpoints that we save based upon the best CER (character error rate).

Metrics

Word Error Rate (WER)

Word error rate is a widely used metric for comparing the performance in automatic speech recognition systems. It compares the predicted transcript against the reference transcript word-by-word. It is derived from Levenshtein distance. WER calculates three different errors when computing the score:

- Insertion (I): This calculates the number of words added by the ASR system into the predicted transcript
- Substitution (S): This calculates the number of words in the reference transcript that have been replaced by misinterpreted words in the predicted transcript
- Deletion (D): This calculates the number of words missing in the predicted transcript

The final WER is:

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

where N is the number of words in the reference transcript.

Character Error Rate (CER)

Character error rate is another widely used metric for comparing the performance of automatic speech recognition systems. While WER operates at word level, CER operates at a character level. While WER provides a good measure of language modelling, CER provides a good measure of acoustic modelling, surfacing mispronunciations and erroneous phonemes.

CER also calculates three different errors for computing the score:

- Insertion (I): This calculates the number of characters added by the ASR system into the predicted transcript
- Substitution (S): This calculates the number of characters in the reference transcript that have been replaced by other characters in the predicted transcript
- Deletion (D): This calculates the number of characters missing in the predicted transcript

The final CER is:

$$CER = \frac{I + S + D}{N}$$

where N is the number of characters in the reference transcript.

Mono and Multi-lingual Speech Models

In our analysis, we consider speech models from the Wav2Vec 2.0 and HuBERT families. We don't consider speech models from Whisper family, primarily because Whisper models have been pre-trained in a semi-supervised fashion which makes them incomparable with unsupervised models from the Wav2Vec 2.0 and HuBERT families as well as other accented ASR finetuning work in current literature.

Concretely, we analyse the following monolingual speech models:

- Wav2Vec 2.0 small: 94 M parameters
- HuBERT base: 94 M parameters
- Wav2Vec 2.0 large: 317 M parameters

Furthermore, we analyse the following multilingual speech models:

- mHuBERT-147: 94 M parameters
- XLSR-53: 317 M parameters
- XLSR-128: 317 M parameters
- MMS-1407: 317 M parameters

We evaluate accented ASR performance for the models across two dimensions, size (an alibi for capacity) and languages in pre-training (an alibi for speech diversity).

Experiment Set Up

We finetune each model under consideration on MCV-100 dataset in a supervised training fashion. Since the model complexity vary, we do a rigorous hyperparameter tuning to select the best hyperparameters in order to compare against the best performances. In the following sections, we shed more light on various aspects of finetuning.

Data Preprocessing

For finetuning, we preprocess both the input speech and corresponding transcriptions. We: 1. Normalise raw speech to zero mean and unit variance. 2. Normalise the transcriptions to remove many non-English characters and most punctuations,, following SpeechBrain convention for English language, and convert the transcriptions to lowercase.

Computing Environment

We use `PyTorch` [Paszke et al., 2019], *HuggingFace* `transformers`, `datasets`, and `trainer` libraries [Wolf et al., 2020] for finetuning.We perform our experiments on two *NVIDIA v100* on an private on-premise distributed computing platform.

Hyperparameter Tuning

For each model, we work with a fixed batch size of 32 and vary learning hyperparameters. We test with:

- **learning rates**: $1e^{-4}, 5e^{-5}, 1e^{-5}, 5e^{-6}, 1e^{-6}$
- **optimisers**: SGD [Ruder, 2016] with momentum and Adam [Kingma and Ba, 2017].
- **Adam beta values**: $(0.9, 0.999)$ and $(0.9, 0.990)$
- **Learning rate schedulers**: Two stage schedulers with 1000 warmup steps followed by linear decay, two stage scheduler with 1000 warmup steps followed by cosine decay, two stage scheduler with a fraction of total steps as warmup followed by linear and cosine decays, three stage scheduler with 10% warmup steps followed by constant learning rate for 40% of total steps followed by linear decay.

Model	learning rate	scheduler	batch size
Wav2Vec 2.0 small	1e-5	2 stage, 1000 steps warmup, linear decay afterwards	32
HuBERT base	1e-4	2 stage, 1000 steps warmup, linear decay afterwards	32
mHuBERT-147	5e-5	2 stage, 1000 steps warmup, linear decay afterwards	32
Wav2Vec 2.0 large	5e-5	2 stage, 1000 steps warmup, linear decay afterwards	32
XLSR-53	5e-5	2 stage, 1000 steps warmup, linear decay afterwards	32
XLSR-128	5e-5	2 stage, 1000 steps warmup, linear decay afterwards	32
MMS-1407	5e-5	2 stage, 1000 steps warmup, linear decay afterwards	32

Table 4.1: Best Set of Hyperparameters for fine-tuning on MCV-100

We found the training to be very sensitive to learning rate schedulers. Also, in all our experiments, we use an epsilon value of $1.e^{-08}$. Finally, all the experiments are run for a total of 226600 steps or 100 epochs for a batch size of 32.

Reproducibility

Modern deep learning is very sensitive to correct set of hyperparameters. For example, in our attempts to finetune the models using SGD, we failed to converge the model even after tweaking SGD associated parameters as well as learning rates, for example. The authors of XLSR-53 suggest a three stage scheduler and it has been reported to be a successful recipe by the authors of mHuBERT-147. However, in our finetuning attempts, we found that such a scheduler always performed worse than two stage schedulers. To make our results reproducible, we present the best set of hyperparameters in Table 4.1.

Results

- Among the smaller models, in both dev and test splits, we find that Wav2Vec 2.0 and HuBERT perform equivalently. However, mHuBERT-

147 shows significant degradation in performance.

- Among the larger models, in both dev and test splits, we find that the smallest multilingual models (in terms of the number of languages), XLSR-53 outperforms other models

Model	Size	Aggregated WER (%) ↓	US	England	Canada	Scotland	Australia	Aggregated CER (%) ↓
Wav2Vec 2.0 base	94 M	13.80	10.54	10.40	9.66	12.40	24.76	7.37
HuBERT base	94 M	13.74	10.54	10.5	10.08	11.50	24.35	7.30
mHuBERT-147	94 M	17.67	14.77	14.82	14.16	15.64	27.27	8.03
Wav2Vec 2.0 large	317 M	12.23	9.29	8.97	8.78	8.90	22.33	7.33
XLSR-53	317 M	10.32	7.63	7.49	7.33	6.79	19.43	6.27
XLSR-128	317 M	11.58	8.95	8.50	8.25	8.63	20.58	6.98
MMS-1407	317 M	12.81	9.95	9.77	8.99	11.01	22.55	7.22

Table 4.2: Results on Dev Split for Different Models

Model	Size	Aggregated (WER) ↓	Seen	Unseen
Wav2Vec 2.0 base	94 M	14.05	9.65	18.46
HuBERT	94 M	13.76	9.67	17.82
mHuBERT-147	94 M	15.87	11.70	20.04
Wav2Vec 2.0 large	317 M	12.45	8.34	16.17
XLSR-53	317 M	10.54	6.77	14.32
XLSR-128	317 M	11.64	7.83	15.46
MMS-1407	317 M	13.20	9.16	17.26

Table 4.3: Results on Test Split for Different models

Comments

- We observe that among the smaller models, HuBERT performs equivalent to Wav2Vec 2.0. However, mHuBERT-147, fares much worse than either of them. Even though mHuBERT-147 has already been pre-trained on Common Voice (CV) dataset of which MCV-100 is a subset, monolingual models - HuBERT and Wav2Vec 2.0 performs better.
- Among the larger models, we observe that compared to monolingual Wav2Vec 2.0 large, the multilingual models show improvement in performance with an introduction to multilinguality during pre-training. However, as the number of languages increase, the performance degrades with the performance of MMS-1407 being worse than Wav2Vec 2.0 large.

Conclusions

- For smaller models, even though mHuBERT-147 has been exposed to the superset of the MCV-100 dataset, i.e., Common Voice corpus, it performs worse than monolingual models. This could indicate that smaller size models lack enough capacity to model the diversity in speech across languages well while maintaining their performance on the tasks for each language.
- For the larger models, even though multilinguality may suggest an improvement in performance at a first glance, we find a confounder in that the multilingual models have already been exposed to the evaluation dataset. Therefore, the performance gain could be totally due to the evaluation being on an in-domain dataset.

Future Work

- Analysis on a dataset that hasn't been used during the pre-training for any of the model would provide a better picture on the performance tendencies measures across differences in size and multilinguality. For accented English language, *L2 Arctic* [Zhao et al., 2018] is one such corpus.

5 Robust Accented Speech Recognition through Representation Learning

Introduction

One of the key challenges to robust accented speech recognition learning a good representation of speech conditioned on accent variation. This has led to two primary approaches in literature: a) accent-agnostic, and b) accent-aware. Accent-agnostic approaches aims to optimise for speech recognition while actively removing accent information. In contrast, accent-aware approaches optimise for speech recognition while considering accent information. More formally, we could look at *accent-agnostic* approach as a dual optimisation task where we maximise ASR accuracy while minimising accent information. In comparison, *accent-aware* approach is also a dual optimisation task where we maximise ASR accuracy as well as accent information.

Current Approaches to Accented Speech Recognition

For both accent-agnostic and accent-aware approaches, the key to good ASR performance lies in how well accent information is encoded. Current accent-agnostic approaches use data augmentation techniques to employ contrastive loss objective. However, we believe such augmentation techniques to be insufficient for capturing good accent encoding. On the other hand, accent-aware approaches encode accent information in accent-specific units such as codebooks, i-vectors, or LoRA adapter [Prabhu, Jyothi, et al., 2023, Prabhu, Gupta, et al., 2024, Das et al., 2021, Zhang et al., 2021, Nassereldine et al., 2024]. While this approach ensures good learned accent encodings, since the accent encodings are specific to a single accent, an introduction of new accent necessitates training from ground-up.

Our Proposal

Representation learning is a powerful technique for encoding the intrinsic characteristics of signals that makes it suitable for downstream tasks. We propose a general representation learning technique that employs contrastive loss objective anchored on accent annotations present in the MCV-100 dataset. This technique could be employed towards learning an accent-agnostic representation as well as an accent representation.

More concretely, we propose a self-supervised representation learning regime trained with a triplet contrastive objective. Furthermore, we present an offline hard triplet mining technique based on accent annotations in the dataset. We design the triplets with `anchors`, `positives`, and `negatives` in such way that with respect to accent-information, they are complementary in the way that flipping the positives and negatives either lead to an accent-agnostic representation or just accent representation.

Triplet Mining for Self-supervised Pre-finetuning

Forming accent-based triplets from MCV-100 dataset is not trivial. There are very few utterances that are spoken by speakers with different accent. For reference, only 4 utterances in the train split are being spoken by all 5 accents present in the training split. We work around this limitation by forming triplets based on words instead. We present the algorithm in 1.

A natural question at this point would rise regarding our consideration of triplets based on word level instead of for example, character level or sentence level. As mentioned previously, we were constrained by the dataset where we don't have enough samples for forming triplets at sentence level. The reason why we did not go for forming triplets based on characters is:

1. Aligners such as Montreal Forced Aligner (MFA) [McAuliffe et al., 2017] are not perfect. For free flowing speech, slicing at finer granuality such as character would incorporate significant amount of noise which we think would make training difficult and performance suboptimal.

Input : An accent-annoted dataset D,
Output: Triplets Z based on words and anchored on accents

Step 1: Get audio-to-transcript word alignment (using aligners such as Montreal Forced Aligner (MFA)
Step 2: Extract the alignment output to form samples with audio and transcriptions offsets as S
Step 3:

for *sample* $s \in S$ **do**
    anchor $\leftarrow s$
    /* Find a sample with same word as anchor but with a different accent. We call it positive */
    for *sample* $t \in S$ **do**
        if $Word_{anchor} = Word_t$ **and** $Accent_{anchor} \neq Accent_t$ **then**
            positive $\leftarrow t$
        end
    end
    /* Find a sample with same accent as anchor and the most dissimilar word. We call it negative */
    leastSimilarity $\leftarrow$ +Inf
    for *sample* $u \in S$ **do**
        similarity $\leftarrow$ $\frac{\texttt{Character Error Rate}(Word_u, Word_{anchor}) + \texttt{Phonetic Distance}(Word_u, Word_{anchor})}{2}$
        if $similarity < leastSimilarity$ **and** $Accent_{anchor} = Accent_u$ **then**
            leastSimilarity $\leftarrow$ similarity
            negative $\leftarrow$ u
        end
        z $\leftarrow$ {anchor, positive, negative}
    end
    $Z \leftarrow$ z $\cup$ Z
end

Algorithm 1: Offline Hard Triplet Mining

The resulting triplets from Algorithm 1 follow the following schema and are stored as a `jsonl` file.

```
{
anchor = {
                "word": dtype=str,
                "audio_start":  dtype=float,
                "audio_end": dtype=float,
                "text_start": dtype=int,
                "text_end": dtype=int,
                "accent": dtype=str,
                "transcription": dtype=str,
                "audio_file": "full_path_to_audio", dtype=str,
                },

positive = {

                "word": dtype=str,
                "audio_start":  dtype=float,
                "audio_end": dtype=float,
                "text_start": dtype=int,
                "text_end": dtype=int,
                "accent": dtype=str,
                "transcription": dtype=str,
                "audio_file": "full_path_to_audio",  dtype=str,
                },

negative = {
                "word": dtype=str,
                "audio_start":  dtype=float,
                "audio_end": dtype=float,
                "text_start": dtype=int,
                "text_end": dtype=int,
                "accent": dtype=str,
                "transcription": dtype=str,
                "audio_file": "full_path_to_audio",  dtype=str,
                },
}
```

Contrastive Pre-training

We perform contrastive pre-training to get two different embedding, a) content embedding (accent-agnostic), and b) accent embedding, using our offline hard mined triplets. While the triplets are at the word-level, we use the full utterance as input. We do this because we believe that full utterance context is important and a differentiator.

Triplet Loss Triplet loss applied on triplets with *anchor*, *positive*, and *negative* samples. Triplet loss is a type of contrastive loss where the objective is to push positive closer to anchor while pushing negative away from the anchor.

More formally, for a triplet dataset Z with triplets of the form $\{anc_i, pos_i, neg_i\}$,

$$L_{Triplet}(anc_i, pos_i, neg_i) = max\{d(anc_i, pos_i) - d(anc_i, neg_i) + margin, 0\}$$

where

$$d(x, y) = ||x - y||_2$$

is a distance metric, and in our case, an L2 norm.

Data Preprocessing

The only preprocessing that we do for contrastive pre-training is that we normalise raw speech to zero mean and unit variance.

Content Embedding Contrastive Pre-training

To get content (accent-agnostic) embeddings, we train an encoder with a contrastive triplet loss objective. As shown in Fig. 5.1, we use a HuBERT feature encoder and HuBERT context encoder network, hereafter referred to as *content encoder*. The feature encoder is taken from a pre-trained HuBERT and is kept frozen during the training. The context encoder is trained with a triplet loss objective. The encoder receives full utterances corresponding to the triplets as input. In the forward pass, the utterances are processed to get full embeddings. The embeddings are then sliced at word-level to get embeddings for *anchors*, *positives*, and *negatives*, respectively. The embeddings

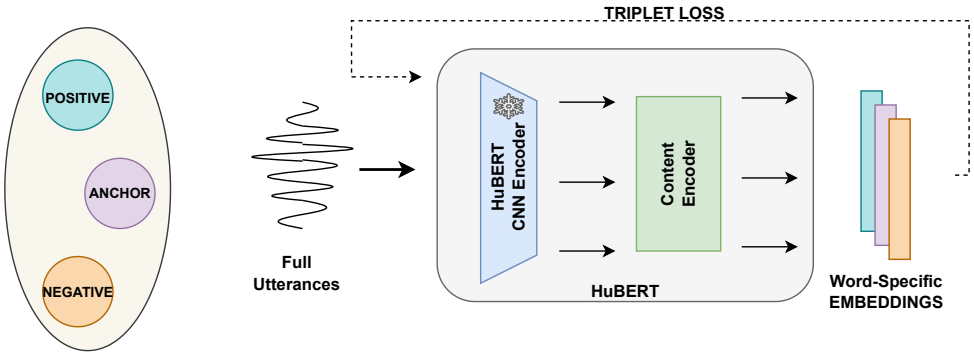

Figure 5.1: Architecture of Content Encoder

thus received are mean pooled across time-steps. The pooled embeddings are then used to calculate the triplet loss.

Formally, the content encoder architecture is as follows:

1. A feature encoder, $f : X \rightarrow Z$ that maps *full* utterances X to *full* latent speech representation $Z^{T \times D}$, where D is the hidden embedding dimension, and T is the number of time-steps.

2. A context encoder, $g : Z^{T \times D} \rightarrow C^{T \times D}$ that encodes the *full* latent representation to *full* output context embeddings.

3. A slicing operator: $h : C^{T \times D} \rightarrow C_{sliced}^{N \times D}$ that slices the *full* context embeddings, where $N = \Delta(offsets)$

4. A mean pooling operator: $p : C_{sliced}^{N \times D} \rightarrow P^{1 \times D}$

We would like to highlight that the decision for using mean pooling isn't arbitrary. We pool the embeddings for two reasons:

1. The utterances are of differing lengths and for calculating the loss, they ought to be of the same dimension. 2. We hypothesise that accent information is order-invariant. Averaging is also an order invariant operation.

Accent Embedding Contrastive Pre-training

Similar to the content encoder, we train an encoder (different than the content encoder) for learning accent embeddings. We again use a contrastive triplet loss objective. As shown in Fig. 5.2, we use a HuBERT feature encoder followed by a 3-layer transformer encoder [Vaswani et al., 2023], hereafter referred to as *accent encoder*. The feature encoder is again from a pre-trained HuBERT and it kept frozen during the training. The embedding network

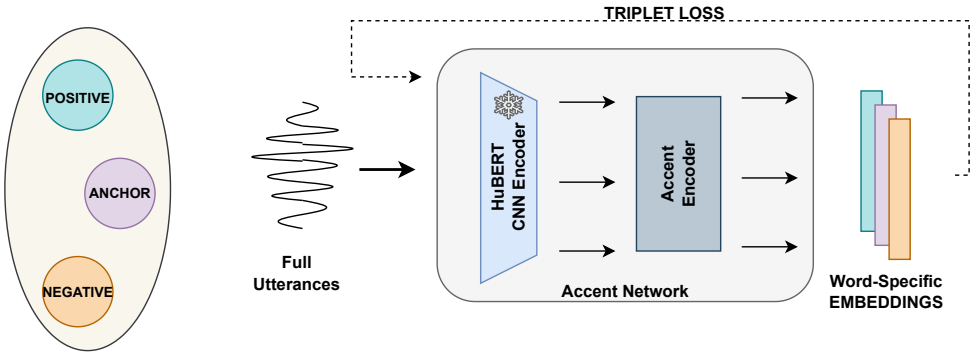

Figure 5.2: Architecture of Accent Encoder

is trained with a triplet loss objective. Similar to the content encoder, the encoder receives full utterances corresponding to the triplets as input. In the forward pass, the utterances are processed to extract full embeddings. The embeddings are then sliced at work-level to get embeddings for *anchors*, *positives*, and *negatives*, respectively. The embeddings thus received are mean pooled across time-steps. The pooled embeddings are then used to calculate the triplet loss.

Apart from the accent encoder being smaller with fewer number of transformer layers (3 vs 12) and with a smaller embedding dimension (256 vs 768), the training process is the same as the content encoder. However, there is a major difference in the dataset. The dataset that we get from Algorithm 1 finds triplets for accent-agnostic training. As our triplets are complimentary by design, to get accent embeddings, we flip the *positives* and *negatives*. We use this flipped version for the accent encoder traininng.

The reasoning behind slicing and mean pooling remain the same as before.

Finetuning

In this section, we discuss the set up for ASR finetuning on the encoders trained with a contrastive objective. We look into two configuration, a) accent-agnostic finetuning, b) accent-aware finetuning.

Data Preprocessing

For finetuning, we preprocess both the input speech and corresponding transcriptions. We: 1. Normalise raw speech to zero mean and unit variance.

2. Normalise the transcriptions to remove many non-English characters and most punctuations, following SpeechBrain [Ravanelli, Parcollet, Plantinga, et al., 2021, Ravanelli, Parcollet, Moumen, et al., 2024] convention for English language, and convert the transcriptions to lowercase.

Baseline

MCV-100 is a relatively new dataset and benchmark and therefore, we have little reference in the literature to compare fairly. Therefore, we develop our own baseline to compare against our proposed method. The baseline is a HuBERT(librispeech 960h) pre-trained encoder which we finetune on MCV-100.

Accent-agnostic Finetuning

For accent-agnostic finetuning, we take the accent encoder pre-trained with contrastive objective and finetune on MCV-100 dataset in a supervised fashion. Architecturally, we add a single fully connected neural network on top of the content encoder as a classifier.

During the training, the content encoder processes the speech input into content emeddings which is then fed as an input to the classifier. The classifier outputs logits for each time-step upon which the classification loss is calculated using a CTC criterion.

During inference, we do an *argmax* operation on the logits (which constitutes a greedy approach) and then perform a CTC decoding for final transcription generation.

Formally, the accent-agnostic network is as follows:

1. A feature encoder, $f : X \rightarrow Z$ that maps raw audio X to hidden speech units $z_1, z_2, ..., z_T$

2. A content encoder, $g : Z \rightarrow C$ that maps hidden speech units X to content representation $c_1, c_2, c_3, ..., c_T$

3. A fully connected neural classifier, $g : C \rightarrow L$ that outputs logits $l_1, l_2, l_3, ..., l_T$ for each time-step.

We should mention that during finetuning, only the feature encoder of the content encoder stays frozen. The context network undergoes finetuning along with the classifier.

Accent-aware Finetuning

For accent-aware finetuning, we take the two encoders pre-trained using a contrastive objective, content encoder and accent encoder, and concatenate their embeddings to feed to the fully connected classifier network for a supervised finetuning on MCV-100 dataset. Since the feature encoders in both the content and accent encoders remained frozen during pre-training, we use a common feature encoder for both the encoders. This set up is further shows in Fig. 5.3.

Formally, the accent-aware network is as follows:

1. A feature encoder, $f : X \rightarrow Z$ that maps raw audio X to hidden speech units $z_1, z_2, ..., z_T$

2. A content encoder, $g : Z \rightarrow C^{T \times d1}$ that maps hidden speech units X to content representation, where $d1$ is the content embedding dimension.

3. An accent encoder, $h : Z \rightarrow A^{T \times d2}$ that maps hidden speech units X to accent representation, where $d2$ is the accent embedding dimension.

4. A fully connected neural classifier, $t : CA^{T \times (d1+d2)} \rightarrow L$ that outputs logits $l_1, l_2, l_3, ..., l_T$ for each time-step.

We should again mention that during finetuning, only the feature encoder remains frozen. The content and accent encoders as well as the classifier is finetuned.

Accent-agnostic Training with Combined Supervised and Contrastive Loss

This training regime is different than both the previously discussed accent-agnostic finetuning and accent-aware finetuning in the way that instead of performing a two-stage training (contrastive pretraining followed by supervised finetuning), we jointly optimise for classification loss and contrastive loss by penalising the network for learning accent information. As shown in Fig. 5.4, this set up uses a HuBERT feature encoder and context encoder. Formally,

1. A feature encoder, $f : X \rightarrow Z$ that maps *full* utterances X to *full* latent speech representation $Z^{T \times D}$, where D is the hidden embedding dimension, and T is the number of time-steps.

2. A context encoder, $g : Z^{T \times D} \rightarrow C^{T \times D}$ that encodes the *full* latent representation to *full* output context embeddings.

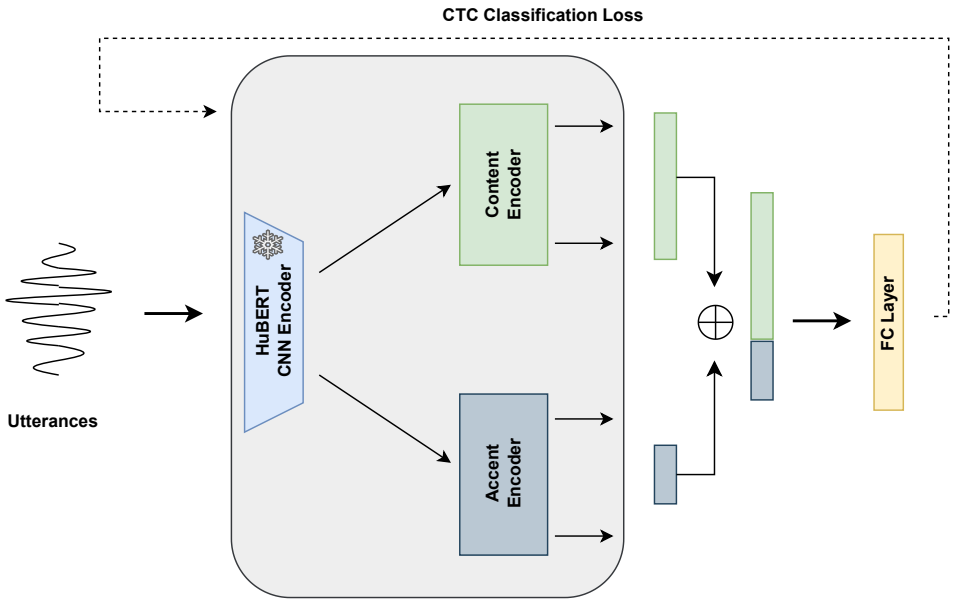

Figure 5.3: Architecture of Accent-aware Finetuning

3. A fully connected neural classifier, $t : C^{T \times D} \rightarrow L$ that outputs logits $l_1, l_2, l_3, ..., l_T$ for each time-step.

3. A slicing operator: $h : C^{T \times D} \rightarrow C^{N \times D}_{sliced}$ that slices the *full* context embeddings, where $N = \Delta(offsets)$

4. A mean pooling operator: $p : C^{N \times D}_{sliced} \rightarrow P^{1 \times D}$

The network is trained on the triplets dataset with a classification loss objective on the full utterances while a triplet loss objective on the sliced utterances. This decision is again motivated by the order invariance of accent information and the limitation posed by the dataset under consideration.

We formulate the joint loss as following:

$$loss = \alpha * L_{CTC} + (1 - \alpha) * L_{Triplet}$$

where α is a hyperparameter.

Preliminary Results

We present our preliminary results in Table 5.1. We observe that finetuning on accent-agnostic pretrained checkpoint doesn't perform worse than our

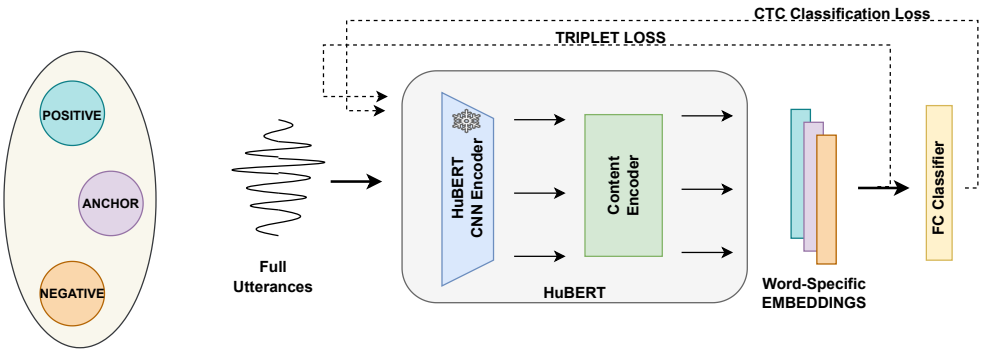

Figure 5.4: Architecture of Combined Objective Training

baseline. In comparison, however, accent-aware finetuning performs worse than the baseline. We hypothesise some issues with accent encoding which we investigate in further sections.

Model	Size	Average WER ↓	US	England	Canada	Scotland	Australia
Baseline	94M	13.72	10.05	10.14	9.49	14.02	25.57
Accent-agnostic	94M	12.99	9.36	9.28	9.05	12.27	24.88
Accent-agnostic	94M	14.43	10.72	11.29	9.84	14.34	26.14
Accent-aware	96M	14.01	10.16	10.73	9.55	13.70	26.08

Table 5.1: Results on Dev Split for Different Models

Method	Size	Average (WER) ↓	Seen	Unseen
Our Approach (MCV100)	94M	12.89	8.57	17.21
HuBERT (MCV600)	104M	13.1	9.1	17.1
HuBERT (pretrained + MCV600)	104M	9.3	6.0	12.5
MTL (MCV600)	104M	9.4	6.0	12.5
DAT (MCV600)	104M	9.3	6.0	12.5
AccentCodebook (MCV600)	104M	8.9	5.9	11.9

Table 5.2: Results on Test Split for SOTA Methods. MCV600 is a superset of MCV100 with 600h of labelled data.

Analysis of Accent and Content Representations

Evaluation is an important aspect of any learning task, especially to assess the performance on unseen data. Across many tasks, this is done using techniques such as clustering analysis, mutual information, etc. One can also train a neural network classifier for a definitive embedding performance result.

However, for our embeddings, evaluation is not straightforward. In particular, we can't train a classifier for predicting the transcripts. Therefore, we evaluate embedding quality of both content and accent encoders on a small curated dataset. In the following sections, we discuss the methodology, results, and observations.

Embedding Quality on Samples from the Training Dataset

For any properly implemented task, it would come as a surprise if the evaluation on samples from the training dataset doesn't come up to be good. Therefore, the main objective of doing analysis on the samples from the training split of MCV-100 is to provide a sanity check and establish a baseline for comparing on the dev and test splits.

Method K-Mean clustering over the embeddings

Dataset 20 unique transcriptions covered by at least 4 different accents from the train split

Metric F1-micro score

Results As seen in Table 5.3 For both the content encoder and accent encoder, we observe a significant improvement in the embedding quality when compared to the HuBERT baseline. A visualisation is presented in Fig. 5.5.

Encoder	Content Embeddings (f1) ↑	Accent Embeddings (f1) ↑
Baseline	0.8980	0.4286
Accent Encoder	0.2380	0.7415
Content Encoder	0.9728	0.4285

Table 5.3: Clustering Results of Accent and Content Encoder

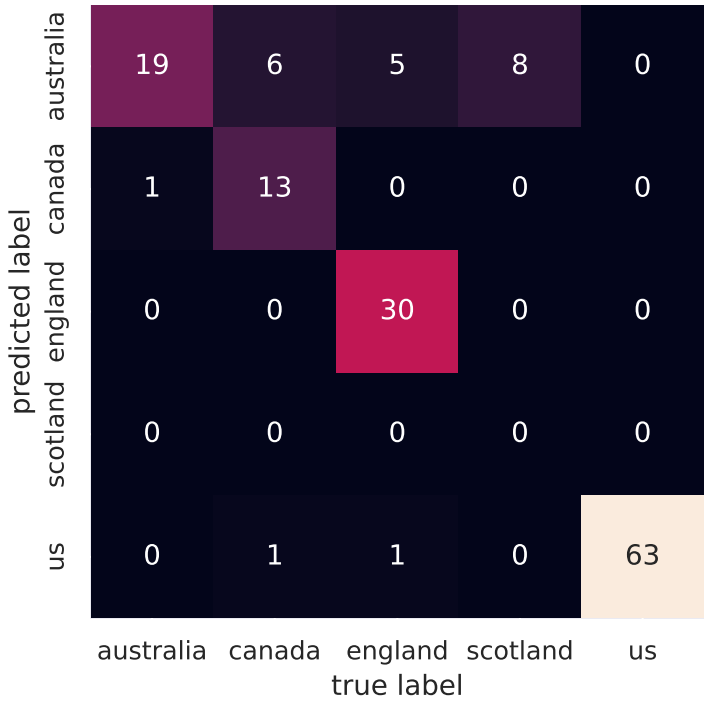

Figure 5.5: Heatmap of classification per accent for samples from training split

Encoder	Accent Embeddings (f1-micro) ↑
Baseline	0.344
Accent Encoder	0.344
Content Encoder	0.304

Table 5.4: Clustering Results of Accent Encoder on the Dev Split

Embedding Quality on Samples from the Evaluation Dataset

Generalisation is the ultimate metric of success of any learning algorithm and accent encoding is nothing different. Therefore, we now evaluate accent embedding performance on samples from the dev split.

Method K-Mean clustering over the embeddings

Dataset 20 unique transcriptions covered by at least 4 different accents from the dev split

Metric F1-micro score

Results The results, in Table 5.4 are very disappointing. The performance of the accent encoder is significantly worse compared to the performance on samples from the training split. This raises many questions and we dive deeper into the issues in the next sections. A visualisation is presented in Fig. 5.6 for another perspective.

Possible Cause: Initial Hypotheses

One common reason for having a good performance on training data while a poor performance on the evaluation data is model overfitting. It means that the model is memorising the training dataset and therefore is struggling to generalise to unseen evaluation dataset.

Another reason could be that the network is learning to encode accents from irrelevant features such as individual speaker identity, which may explain the significant degradation in evaluation performance of the network.

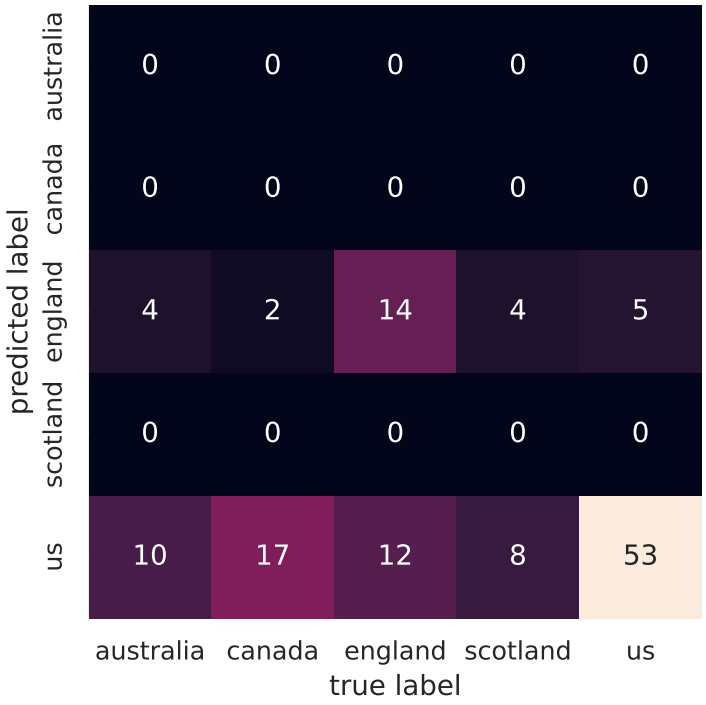

Figure 5.6: Heatmap of classification per accent for samples from dev split

Accent Classification

Background

Accent classification is a task of predicting the accent type of an speech input. A good accent classifier has the potential to shape tasks such as accented speech recognition which we explored in the previous sections. Prior works [Purwar et al., 2022 ,Song, Nguyen, and Ta, 2023] have shown that accent classification is an achievable with high accuracy. However, some other works show [C. Watanabe and Kameoka, 2024] that accent classifiers actually fails on spontaneous speech. Motivated by these conflicting results and the alarming performance of our own accent encoder as shown in Table 5.4, we take a deeper look into accent classification. In short, we find:

1. There is a significant degradation of performance on speaker disjoint datasets.

2. The classifier tend to learn shortcuts where it learns to classify accent using speaker identity instead of accent information

Accent Classification Benchmarks

Accent classification is considered a solved task with close to 99% accuracy on corpus such as Speech Accent Archive dataset by George Mason University [Weinberger and Kunath, 2011] and Common Accent [Weinberger and Kunath, 2011] and some other works that work with a subset of Common Voice dataset. However, we find that none of the current benchmarks are speaker disjoint in their training and evaluation splits. [C. Watanabe and Kameoka, 2024] have explored speaker disjoint dataset, CSTR VCTK [Oord et al., 2016], for classification and have reported just around 50% accuracy on the test split compared to around 99% on the training split. This trend is visible across languages as well. [Matos et al., 2024] report that while working with Portuguese accents, they find that accent classification fails on cross-dataset evaluation.

What the Classifier Actually Learns

Previous works such as [C. Watanabe and Kameoka, 2024] hypothesise that the failure of classifier could be linked to the classifier learning irrelevant features such as speaker related information instead of accent information. MCV-100 dataset is derived from the Common Voice and therefore has

Classifier	Accuracy (Train (%)) ↑	Accuracy (Dev (%) ↑
Accent	96.86	49.00
Speaker	91.90	18.97

Table 5.5: Classification performance on MCV-100 training and dev splits

Classifier	Accuracy (Train (%)) ↑	Accuracy (Dev (%)) ↑
Accent	94.72	85.09

Table 5.6: Classification performance on MCV-100 speaker leakage dataset

unique identification number for every speaker. Therefore, we test the pretrained accent encoder for its ability to classify accents on the dev split. For this, we finetune a frozen accent encoder with just one classification layer for both accents and speakers. On the training split, we find that the classification accuracy to be very good for both task, with around 96% accuracy against 5 accent classes and 91% accuracy against 5832 unique speakers as shown in Table 5.5. However, on the dev split, we observe a sharp degradation in performance with an accent accuracy of just around 49% and 18%, respectively as seen in Table 5.5. The initial results seem to validate the hypothesis of [C. Watanabe and Kameoka, 2024].

A control test: Effect of speaker leakage

To verify classifier results and reports of failure of spontaneous speech, we perform an experiment where we mix the train and dev split and randomly sample from the mixed dataset to form new train and dev split while maintaining the original split ratio. This achieves speaker leakage which we use as a control test to determine if the degraded performance is indeed due to the classifier learning speaker related information. We train a classifier on this new train split and evaluate on the new dev split. We achieve around 94% training accuracy compared to around 85% evaluation accuracy. This preliminary result show that speaker leakage seem to hide the bigger issue of accent classifier learning speaker information instead of accent embeddings.

Shortcut Learning: Simple Vs. Complex Features

Studies [Geirhos et al., 2020, Fel et al., 2024, McCoy, Pavlick, and Linzen, 2019] reveal that deep learning models have an inductive bias towards favouring simpler features. This has been widely reported in other fields

No. of Layers	Accuracy (Train (%)) ↑	Accuracy (Dev (%)) ↑
1	98.83	53.51
2	95.43	50.62
3	99.26	55.55
12	83.42	51.56

Table 5.7: Classification performance on MCV-100 with a varying number of encoder layers

Objective	Accuracy (Train (%)) ↑	Accuracy (Dev (%)) ↑
Cross Entropy	98.83	53.51
Contrastive Loss	99.30	55.55

Table 5.8: Classification performance on MCV-100 across different loss functions

such as object classification where a model may learn to classify an object based on the prevalence of its location in the image [Fig. 5.7]. Other reports show that some models tend to fail if a picture has a cow near an ocean, for example, as the model has actually learned to identify cows based on it's common environment like a field of grass. Similarly, for accent classification, our initial results show that the model is learning simpler features for classifying accents such as speaker information instead of complex actual accent information. This represents a problems referred to as *shortcut learning* in the literature. Shortcuts, as defined by [Geirhos et al., 2020] are decision rules that perform well on standard benchmarks but fail to transfer to more challenging real-world scenario.

[Fel et al., 2024] show that earlier layers of a neural network tend to learn simpler features whereas later layer encode complex features. Motivated from these findings, we perform experiments where we train classifiers with increasing number of layers in the encoder. In particular, we test with {1, 2, 3, 12} layers of transformer layers. Tables 5.7, 5.8 presents the results. We observe that the evaluation performance remains consistently bad.

Furthermore, [C. Watanabe and Kameoka, 2024] report the bias of cross-entropy loss towards simpler predictor. To evaluate if using other metric could prove helpful, we also compare accent classification results across cross entropy loss and a metric (contrastive) loss. We again observe that the evaluation performance remains consistently bad.

Conclusions

- In our preliminary research, we find that accent classifiers fail on speaker disjoint evaluation split of MCV-100 across different model complexity and objectives.

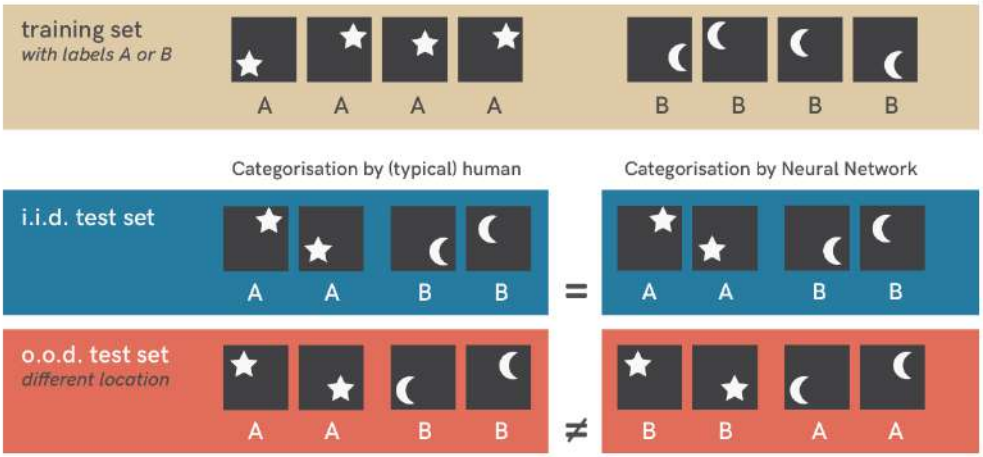

Figure 5.7: An example of shortcut learning from computer vision (source: `https://arxiv.org/pdf/2004.07780`)

- We find that speaker leakage masks this issue and current benchmarks are insufficient.

Future Works

- Works on shortcut learning [Geirhos et al., 2020] suggest that failure on evaluation dataset might mean that being speaker disjoint constitutes *out-of-distribution* (OOD) task for accent classification. One might think that this issue of generalising could be countered with more data. However, we might have to first test if being speaker disjoint mean OOD since if it were to be true, increasing the amount of data does not show improvement [Jo and Bengio, 2017, Ilyas et al., 2019, Lehman et al., 2019].
- MCV-100 could be proposed as a new benchmark for accent classification for further work by the research community in hopes of improving on the task.
- Since accent classification/encoding is the key component in all state-of-the-art accented ASR methods, evaluating the encodings and classifiers in these work would be necessary in order to test if the learned accent encodings are actually useful for inference.

Summary

In this chapter, we presented a representation learning technique for learning accent-agnostic content representation. We also presented a novel technique for generating triplets for contrastive training based on accent information. We then evaluated the representations, both qualitatively and quantitatively. Furthermore, we compared our technique against the baseline.

Furthermore, we presented our findings on a challenging problem of accent classification on speaker disjoint datasets, which we believe are more faithful representation of real-world scenario. We find that accent classifier learn simpler and irrelevant features which helps them achieve very good accuracy during training. Meanwhile, they fail to generalise. We evaluate preliminary solutions but find no success. Finally, we propose further investigation into speaker disjoint datasets to check if being speaker disjoint constitutes out-of-distribution problem.

Finally, we believe that generalisable accent-classifier would be key for robust accented speech recognition systems as the principles of learning generalised classifier and accent embeddings are transferable. We hope that our finding would guide the research in robust accented ASR systems in the future.

6 Conclusion

Below, we present a summary of our key conclusions derived from our analysis of multilinguality and representation learning, highlighting insights and implications for future research.

- Our preliminary research shows that accent classifiers do not perform well on a speaker-disjoint evaluation split of the MCV-100 dataset, regardless of model complexity or objective.

- Additionally, we observe that the phenomenon of speaker leakage has obscured this limitation, revealing inadequacies in current benchmarks for detecting and addressing it. This lack of effective benchmarks poses a significant challenge for evaluating the true generalisability of accent classification models and necessitates further refinement in evaluation standards.

- For smaller models, despite mHuBERT-147 being trained on a broader dataset that includes the Common Voice corpus (a superset of MCV-100), it underperforms relative to monolingual models. This may indicate that smaller models lack the capacity to effectively capture cross-linguistic diversity while maintaining performance across individual languages.

- For larger models, while multilinguality may initially appear to enhance performance, we find a confounding factor: the multilingual models have prior exposure to the evaluation dataset. Consequently, the performance gain may be due solely to the evaluation occurring on an in-domain dataset.

7 Future Work

The challenge of achieving robust accented speech recognition presents several fundamental issues. However, we view this as an opportunity to outline future directions for improvement.

- Analysing a dataset that has not been used in any model's pre-training phase would offer a clearer perspective on performance tendencies across variations in model size and multilinguality. For accented English, the *L2 Arctic corpus* is one such dataset.

- Research on shortcut learning [Geirhos et al., 2020] suggests that poor performance on an evaluation dataset may indicate that being speaker-disjoint constitutes an *out-of-distribution* (OOD) scenario for accent classification. While it might seem that increasing data volume could address this generalisation challenge, it is essential to first assess if speaker disjointness indeed creates an OOD condition. If it does, adding more data may not lead to improved performance [Jo and Bengio, 2017, Ilyas et al., 2019, Lehman et al., 2019].

- The MCV-100 dataset could also be proposed as a new benchmark for accent classification, encouraging further research from the community to enhance performance on this task.

- As accent classification and encoding are core components in all state-of-the-art accented ASR systems, evaluating the encodings and classifiers used in these models would be important to determine whether the learned accent encodings truly contribute to inference quality.

Bibliography

A, Madhavaraj, Bharathi Pilar, and Ramakrishnan A G (2022a). *Knowledge-driven Subword Grammar Modeling for Automatic Speech Recognition in Tamil and Kannada*. DOI: 10.48550/ARXIV.2207.13333. URL: https://arxiv.org/abs/2207.13333 (cit. on p. 7).

– (2022b). *Subword Dictionary Learning and Segmentation Techniques for Automatic Speech Recognition in Tamil and Kannada*. DOI: 10.48550/ARXIV.2207.13331. URL: https://arxiv.org/abs/2207.13331 (cit. on p. 7).

Alsharhan, Eiman and Allan Ramsay (2020). "Investigating the effects of gender, dialect, and training size on the performance of Arabic speech recognition." In: *Language Resources and Evaluation* 54.4, pp. 975–998 (cit. on p. 3).

Ardila, Rosana et al. (2020). *Common Voice: A Massively-Multilingual Speech Corpus*. arXiv: 1912.06670 [cs.CL]. URL: https://arxiv.org/abs/1912.06670 (cit. on pp. 6, 15, 24).

Babu, Arun et al. (2021). *XLS-R: Self-supervised Cross-lingual Speech Representation Learning at Scale*. arXiv: 2111.09296 [cs.CL]. URL: https://arxiv.org/abs/2111.09296 (cit. on pp. 3, 10).

Baevski, Alexei et al. (2020). *wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations*. arXiv: 2006.11477 [cs.CL]. URL: https://arxiv.org/abs/2006.11477 (cit. on pp. 3, 9).

Bengio, Yoshua, Aaron Courville, and Pascal Vincent (2013). "Representation learning: A review and new perspectives." In: *IEEE transactions on pattern analysis and machine intelligence* 35.8, pp. 1798–1828 (cit. on p. 8).

Boito, Marcely Zanon et al. (2024). *mHuBERT-147: A Compact Multilingual HuBERT Model*. arXiv: 2406.06371 [cs.CL]. URL: https://arxiv.org/abs/2406.06371 (cit. on pp. 3, 12).

Chen, Ting et al. (2020). "A simple framework for contrastive learning of visual representations." In: *International conference on machine learning*. PMLR, pp. 1597–1607 (cit. on p. 8).

Cho, Won Ik et al. (2021). "kosp2e: Korean Speech to English Translation Corpus." In: *Proc. Interspeech 2021*, pp. 3705–3709. DOI: 10.21437/Interspeech.2021-1040 (cit. on p. 8).

Conneau, Alexis et al. (2020). *Unsupervised Cross-lingual Representation Learning for Speech Recognition*. arXiv: 2006.13979 [cs.CL]. URL: https://arxiv.org/abs/2006.13979 (cit. on pp. 3, 10, 23).

Das, Nilaksh et al. (2021). *Best of Both Worlds: Robust Accented Speech Recognition with Adversarial Transfer Learning*. arXiv: 2103.05834 [eess.AS]. URL: https://arxiv.org/abs/2103.05834 (cit. on pp. 5, 31).

Devlin, Jacob et al. (2019). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. arXiv: 1810.04805 [cs.CL]. URL: https://arxiv.org/abs/1810.04805 (cit. on p. 9).

Dong Wang Xuewei Zhang, Zhiyong Zhang (2015). *THCHS-30 : A Free Chinese Speech Corpus*. URL: http://arxiv.org/abs/1512.01882 (cit. on p. 8).

Fel, Thomas et al. (2024). *Understanding Visual Feature Reliance through the Lens of Complexity*. arXiv: 2407.06076 [cs.CV]. URL: https://arxiv.org/abs/2407.06076 (cit. on pp. 48, 49).

Feng, Siyuan et al. (2024). "Towards inclusive automatic speech recognition." In: *Computer Speech Language* 84, p. 101567. ISSN: 0885-2308. DOI: https://doi.org/10.1016/j.csl.2023.101567. URL: https://www.sciencedirect.com/science/article/pii/S0885230823000864 (cit. on p. 3).

Gales, Mark John Francis et al. (2014). "Speech recognition and keyword spotting for low-resource languages: Babel project research at CUED." In: *Workshop on Spoken Language Technologies for Under-resourced Languages*. URL: https://api.semanticscholar.org/CorpusID:7439227 (cit. on p. 6).

Geirhos, Robert et al. (Nov. 2020). "Shortcut learning in deep neural networks." In: *Nature Machine Intelligence* 2.11, pp. 665–673. ISSN: 2522-5839. DOI: 10.1038/s42256-020-00257-z. URL: http://dx.doi.org/10.1038/s42256-020-00257-z (cit. on pp. 48–50, 53).

Graves, Alex et al. (2006). "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks." In: *Proceedings of the 23rd international conference on Machine learning*, pp. 369–376 (cit. on p. 3).

Ha, Jung-Woo et al. (2020). "ClovaCall: Korean Goal-Oriented Dialog Speech Corpus for Automatic Speech Recognition of Contact Centers." In: *arXiv preprint arXiv:2004.09367* (cit. on p. 7).

Han, Tao et al. (2021). "Supervised Contrastive Learning for Accented Speech Recognition." In: *ArXiv* abs/2107.00921. URL: https://api.semanticscholar.org/CorpusID:235727605 (cit. on p. 5).

Hedström, Staffan et al. (2022). "Samrómur Unverified 22.07." In: Reykjavik University: Language and Voice Lab (cit. on p. 8).

Hsu, Wei-Ning et al. (2021). *HuBERT: Self-Supervised Speech Representation Learning by Masked Prediction of Hidden Units*. arXiv: 2106.07447 [cs.CL]. URL: https://arxiv.org/abs/2106.07447 (cit. on pp. 3, 11).

Ilyas, Andrew et al. (2019). *Adversarial Examples Are Not Bugs, They Are Features*. arXiv: 1905.02175 [stat.ML]. URL: https://arxiv.org/abs/1905.02175 (cit. on pp. 50, 53).

Jiang, Dongwei et al. (2020). "Speech simclr: Combining contrastive and reconstruction objective for self-supervised speech representation learning." In: *arXiv preprint arXiv:2010.13991* (cit. on p. 8).

Jo, Jason and Yoshua Bengio (2017). *Measuring the tendency of CNNs to Learn Surface Statistical Regularities*. arXiv: 1711.11561 [cs.LG]. URL: https://arxiv.org/abs/1711.11561 (cit. on pp. 50, 53).

Kahn, J. et al. (2020). "Libri-Light: A Benchmark for ASR with Limited or No Supervision." In: *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. https://github.com/facebookresearch/libri-light, pp. 7669–7673 (cit. on p. 6).

Le-Khac, Phuc H, Graham Healy, and Alan F Smeaton (2020). "Contrastive representation learning: A framework and review." In: *Ieee Access* 8, pp. 193907–193934 (cit. on p. 8).

Kim, Suyoun, Takaaki Hori, and Shinji Watanabe (2017). "Joint CTC-attention based end-to-end speech recognition using multi-task learning." In: *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, pp. 4835–4839 (cit. on p. 2).

Kingma, Diederik P. and Jimmy Ba (2017). *Adam: A Method for Stochastic Optimization*. arXiv: 1412.6980 [cs.LG]. URL: https://arxiv.org/abs/1412.6980 (cit. on p. 26).

Koenecke, Allison et al. (2020). "Racial disparities in automated speech recognition." In: *Proceedings of the national academy of sciences* 117.14, pp. 7684–7689 (cit. on p. 3).

Kolobov, Rostislav et al. (2021). *MediaSpeech: Multilanguage ASR Benchmark and Dataset*. arXiv: 2103.16193 [eess.AS] (cit. on p. 8).

Lehman, Joel et al. (2019). *The Surprising Creativity of Digital Evolution: A Collection of Anecdotes from the Evolutionary Computation and Artificial Life Research Communities*. arXiv: 1803.03453 [cs.NE]. URL: https://arxiv.org/abs/1803.03453 (cit. on pp. 50, 53).

Lida, Katsuya (2022). URL: https://github.com/kaiidams/Kokoro-Speech-Dataset (cit. on p. 7).

Lippi-Green, Rosina (2012). *English with an accent: language, ideology and discrimination in the United States*. 2nd ed. OCLC: ocn731009712. London ; New York: Routledge. ISBN: 9780415559102 9780415559119 9780203348802 (cit. on p. 1).

Markl, Nina (2023). "Language variation, automatic speech recognition and algorithmic bias." In: (cit. on p. 3).

Matos, Ariadne et al. (Mar. 2024). "Accent Classification is Challenging but Pre-training Helps: a case study with novel Brazilian Portuguese datasets." In: *Proceedings of the 16th International Conference on Computational Processing*

of Portuguese - Vol. 1. Ed. by Pablo Gamallo et al. Santiago de Compostela, Galicia/Spain: Association for Computational Lingustics, pp. 364–373. URL: https://aclanthology.org/2024.propor-1.37 (cit. on p. 47).

McAuliffe, Michael et al. (2017). "Montreal Forced Aligner: Trainable Text-Speech Alignment Using Kaldi." In: *Interspeech*. URL: https://api.semanticscholar.org/CorpusID:12418404 (cit. on p. 32).

McCoy, Tom, Ellie Pavlick, and Tal Linzen (July 2019). "Right for the Wrong Reasons: Diagnosing Syntactic Heuristics in Natural Language Inference." In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Ed. by Anna Korhonen, David Traum, and Lluís Màrquez. Florence, Italy: Association for Computational Linguistics, pp. 3428–3448. DOI: 10.18653/v1/P19-1334. URL: https://aclanthology.org/P19-1334 (cit. on p. 48).

Meyer, Josh et al. (2022). "BibleTTS: a large, high-fidelity, multilingual, and uniquely African speech corpus." In: *Interspeech*. ISCA. URL: https://arxiv.org/pdf/2207.03546.pdf (cit. on p. 7).

Nassereldine, Amir et al. (2024). *PI-Whisper: An Adaptive and Incremental ASR Framework for Diverse and Evolving Speaker Characteristics*. arXiv: 2406.15668 [cs.CL]. URL: https://arxiv.org/abs/2406.15668 (cit. on pp. 5, 31).

Oord, Aaron van den et al. (2016). *WaveNet: A Generative Model for Raw Audio*. arXiv: 1609.03499 [cs.SD]. URL: https://arxiv.org/abs/1609.03499 (cit. on p. 47).

Paszke, Adam et al. (2019). "PyTorch: An Imperative Style, High-Performance Deep Learning Library." In: *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf (cit. on p. 26).

Peddinti, Vijayaditya, Daniel Povey, and Sanjeev Khudanpur (2015). "A time delay neural network architecture for efficient modeling of long temporal contexts." In: *Interspeech*. URL: https://api.semanticscholar.org/CorpusID:8536162 (cit. on p. 2).

Prabhu, Darshan, Abhishek Gupta, et al. (2024). *Improving Self-supervised Pre-training using Accent-Specific Codebooks*. arXiv: 2407.03734 [cs.CL]. URL: https://arxiv.org/abs/2407.03734 (cit. on pp. 5, 31).

Prabhu, Darshan, Preethi Jyothi, et al. (Dec. 2023). "Accented Speech Recognition With Accent-specific Codebooks." In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Ed. by Houda Bouamor, Juan Pino, and Kalika Bali. Singapore: Association for Computational Linguistics, pp. 7175–7188. DOI: 10.18653/v1/2023.emnlp-main.444. URL: https://aclanthology.org/2023.emnlp-main.444 (cit. on pp. 5, 6, 15, 24, 31).

Pratap, Vineel, Andros Tjandra, et al. (2023). *Scaling Speech Technology to 1,000+ Languages*. arXiv: 2305.13516 [cs.CL]. URL: https://arxiv.org/abs/2305.13516 (cit. on pp. 3, 5, 7, 11, 23).

Pratap, Vineel, Qiantong Xu, et al. (Oct. 2020). "MLS: A Large-Scale Multilingual Dataset for Speech Research." In: *Interspeech 2020*. interspeech$_2$020. ISCA. DOI: 10.21437/interspeech.2020-2826. URL: http://dx.doi.org/10.21437/Interspeech.2020-2826 (cit. on pp. 5, 6).

Purwar, Archana et al. (2022). "Accent classification using Machine learning and Deep Learning Models." In: *2022 1st International Conference on Informatics (ICI)*, pp. 13–18. URL: https://api.semanticscholar.org/CorpusID:249474183 (cit. on p. 47).

Qian, Yao et al. (2017). "Bidirectional LSTM-RNN for Improving Automated Assessment of Non-Native Children's Speech." In: *INTERSPEECH*, pp. 1417–1421 (cit. on p. 3).

Radford, Alec et al. (2022). *Robust Speech Recognition via Large-Scale Weak Supervision*. arXiv: 2212.04356 [eess.AS]. URL: https://arxiv.org/abs/2212.04356 (cit. on p. 3).

Ravanelli, Mirco, Titouan Parcollet, Adel Moumen, et al. (2024). *Open-Source Conversational AI with SpeechBrain 1.0*. arXiv: 2407.00463 [cs.LG]. URL: https://arxiv.org/abs/2407.00463 (cit. on p. 38).

Ravanelli, Mirco, Titouan Parcollet, Peter Plantinga, et al. (2021). *SpeechBrain: A General-Purpose Speech Toolkit*. arXiv:2106.04624. arXiv: 2106.04624 [eess.AS] (cit. on p. 38).

Ruder, Sebastian (2016). "An overview of gradient descent optimization algorithms." In: *arXiv preprint arXiv:1609.04747* (cit. on p. 26).

Scharenborg, O (2021). "Inclusive speech technology: Developing automatic speech recognition for everyone." In: *Webinar delivered to the TU Delft Safety and Security Institute and Campus, The Hague, the Netherlands* (cit. on p. 3).

Shi, Yao et al. (2015). "AISHELL-3: A Multi-speaker Mandarin TTS Corpus and the Baselines." In: URL: https://arxiv.org/abs/2010.11567 (cit. on p. 7).

Song, Tianyu, Linh Thi Hoai Nguyen, and Ton Viet Ta (2023). *MPSA-DenseNet: A novel deep learning model for English accent classification*. arXiv: 2306.08798 [cs.CL]. URL: https://arxiv.org/abs/2306.08798 (cit. on p. 47).

Takamichi, Shinnosuke et al. (2019). *JVS corpus: free Japanese multi-speaker voice corpus*. arXiv: 1908.06248 [cs.SD]. URL: https://arxiv.org/abs/1908.06248 (cit. on p. 7).

Tatman, Rachael and Conner Kasten (2017). "Effects of Talker Dialect, Gender & Race on Accuracy of Bing Speech and YouTube Automatic Captions." In: *Interspeech*, pp. 934–938 (cit. on p. 3).

Valk, Jörgen and Tanel Alumäe (2020). *VoxLingua107: a Dataset for Spoken Language Recognition*. arXiv: 2011.12998 [eess.AS]. URL: https://arxiv.org/abs/2011.12998 (cit. on pp. 6, 7).

Vaswani, Ashish et al. (2023). *Attention Is All You Need*. arXiv: 1706.03762 [cs.CL]. URL: https://arxiv.org/abs/1706.03762 (cit. on p. 36).

Vu, Ngoc Thang (2014). "Automatic speech recognition for low-resource languages and accents using multilingual and crosslingual information." Doctoral dissertation. Karlsruhe, Karlsruher Institut für Technologie (KIT), Diss., 2014 (cit. on p. 23).

Wang, Changhan et al. (2021). *VoxPopuli: A Large-Scale Multilingual Speech Corpus for Representation Learning, Semi-Supervised Learning and Interpretation*. arXiv: 2101.00390 [cs.CL]. URL: https://arxiv.org/abs/2101.00390 (cit. on p. 6).

Watanabe, Chihiro and Hirokazu Kameoka (2024). *GE2E-AC: Generalized End-to-End Loss Training for Accent Classification*. arXiv: 2407.14021 [eess.AS]. URL: https://arxiv.org/abs/2407.14021 (cit. on pp. 47–49).

Weinberger, Steven and Stephen Kunath (Dec. 2011). "The Speech Accent Archive: Towards a typology of English accents." In: *Language and Computers* 73 (cit. on p. 47).

Wolf, Thomas et al. (2020). *HuggingFace's Transformers: State-of-the-art Natural Language Processing*. arXiv: 1910.03771 [cs.CL]. URL: https://arxiv.org/abs/1910.03771 (cit. on p. 26).

Xu, Hainan et al. (2018). "Neural network language modeling with letter-based features and importance sampling." In: *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, pp. 6109–6113 (cit. on p. 2).

Yadav, Hemant and Sunayana Sitaram (2022). *A Survey of Multilingual Models for Automatic Speech Recognition*. arXiv: 2202.12576 [cs.CL]. URL: https://arxiv.org/abs/2202.12576 (cit. on p. 23).

Yang, Shu-wen et al. (2021). *SUPERB: Speech processing Universal PERformance Benchmark*. arXiv: 2105.01051 [cs.CL]. URL: https://arxiv.org/abs/2105.01051 (cit. on p. 12).

Zhang, Jicheng et al. (2021). "E2E-Based Multi-Task Learning Approach to Joint Speech and Accent Recognition." In: *Interspeech*. URL: https://api.semanticscholar.org/CorpusID:235435686 (cit. on pp. 5, 31).

Zhao, Guanlong et al. (2018). "L2-ARCTIC: A Non-native English Speech Corpus." In: *Proc. Interspeech*, pp. 2783–2787. DOI: 10.21437/Interspeech.2018-1110. URL: http://dx.doi.org/10.21437/Interspeech.2018-1110 (cit. on p. 30).