Improving Dhivehi Automatic Speech Recognition (ASR) with Sub-word Modelling, Language Model Decoding and Automatic Spelling Correction

Arushad Ahmed
University of St. Andrews, St Andrews KY16 9AJ, United Kingdom
aa369@st-andrews.ac.uk

Abstract

Current state-of-the-art (SOTA) Automatic Speech Recognition (ASR) models are multilingual. While these models have greatly improved transcription accuracy for high-resourced languages, under-resourced languages still require further language specific optimizations and finetuning to achieve acceptable levels of accuracy. In this work we explore ways of improving ASR for Dhivehi, an under-resourced South Asian language, by finetuning pretrained multilingual ASR models, Sub-word Modelling, Language Model (LM) decoding and Automatic Spelling Correction. We finetune 5 Dhivehi ASR models and apply our accuracy boosting techniques, with one of our models achieving a new state-of-the-art Word Error Rate (WER) of 14.26% on the Dhivehi Common Voice ASR benchmark, which is a 31.93% relative WER improvement over the existing SOTA of 20.95%. We create a new Dhivehi text corpus, and train 2 new Dhivehi LMs to support our accuracy boosting techniques.

1 Introduction

Recent work on Automatic Speech Recognition (ASR) has focused on training multilingual models; (Zhang, et al., 2023; Hou, et al., 2020a; Pratap, et al., 2023; Conneau, et al., 2020; Radford, et al., 2022). While these multilingual models have produced state-of-the-art (SOTA) results for high-resource languages, results (Hou, et al., 2020a) show that, especially for low-resource languages, there is room for improving transcription accuracy using language specific optimizations and fine-tuning. In this work, we focus on improving ASR accuracy for Dhivehi (ދިވެހި), the native language of the Maldives.

Modern ASR models follow all-neural architectures that enable End-to-End (E2E) speech recognition by training directly on audio recordings and producing text transcriptions as output (Prabhavalkar, et al., 2023). E2E ASR models either explicitly or implicitly align the output text to the input audio.

Connectionist Temporal Classification (CTC) (Graves, et al., 2006) based E2E models use explicit alignment by assigning an output text token to each element of the input audio sequence (Hannun, 2017). CTC models simply learn a mapping from aspects of speech such as phonemes and diphones to output character sequences (Prabhavalkar, et al., 2023; Hannun, 2017). Therefore, CTC models can benefit from sub-word modelling of the output vocabulary, which better corresponds with the phonemes of the target language (Xu, et al., 2019; Zhou, et al., 2021). Moreover, CTC model accuracy can be further improved by incorporating the probabilities from a Language Model (LM) into the output decoding process, which can implicitly model the syntax and semantics of the language (Baevski, et al., 2020). The outputs generated by LM decoding can be further improved by Automatic Spelling Correction (Zhang, et al., 2019).

Attention-Based Encoder-Decoder (AED) ASR models contain an Encoder which produces context vectors using the input acoustic frames and a Decoder which uses the Attention Mechanism (Bahdanau, et al., 2016) to generate a text sequence from the context vectors, without explicitly aligning the text and audio (Prabhavalkar, et al., 2023). AED models implicitly learn a language model over the training outputs (Prabhavalkar, et al., 2023). As a result, they benefit less from external LMs and spelling correction.

E2E models pretrained on high-resourced languages can be fine-tuned on under-resourced languages like Dhivehi through transfer learning to give better results than training from scratch

(Baevski, et al., 2020; Conneau, et al., 2020; Radford, et al., 2022; Pratap, et al., 2023; Hou, et al., 2020b). In this work, we finetune pretrained XLSR (Conneau, et al., 2020), MMS (Pratap, et al., 2023) and Whisper (Radford, et al., 2022) models on Dhivehi speech data from the Mozilla Common Voice 13.0 (CV-13) (Ardila, et al., 2019) dataset and experiment with applying different accuracy boosting techniques.

2 Related Work

Tyers & Meyer (2021) used the Coqui STT [1] toolkit to train ASR models for several different under-resourced languages including Dhivehi. They used transfer learning by finetuning an English ASR model based on the Mozilla Deep Speech architecture, which is an open-source implementation of the Deep Speech ASR architecture from Baidu (Hannun, et al., 2014). They used hyperparameter search to optimize the model hyperparameters for specific languages (Tyers & Meyer, 2021). LM decoding was utilized to further boost the results. For Dhivehi, the authors used 3:56:12 of training data from Common Voice (CV) [2] for the ASR model and 6.8MB of text containing 419k tokens of 76k different types for LM training. For Dhivehi, the authors obtained a WER of 88.37% without LM and 66.49% with LM decoding (Tyers & Meyer, 2021).

Pham, et al. (2021) trained multi-lingual ASR models for 27 languages (including Dhivehi) based on the Transformer architecture (Vaswani, et al., 2023) and LSTM architectures (Hochreiter & Schmidhuber, 1997) using a novel weight factorization scheme for efficient multi-lingual training. The models contained weights shared across all the languages as well as language specific adapter layers (Pham, et al., 2021). Under this work, the best result obtained for Dhivehi was a WER of 63.72% using the weight factorized version of the Transformer based model (Pham, et al., 2021).

Hou, et al. (2020a), trained multilingual ASR models for 42 languages based on a hybrid CTC/attention architecture using 5,000 hours of speech. One of these models were trained using a character level output vocabulary while the other was trained using a sub-word vocabulary (Hou, et al., 2020a). It was shown that, generally the larger sub-word vocabulary produced better results across all 42 languages. After training these multilingual models, transfer learning was used to finetune ASR models for 14 different low-resource languages including Dhivehi. For each of these 14 languages, a language specific model was finetuned as well as a joint 14-language multilingual model. The authors experimented with finetuning these models using the pretrained models as well as training the models from scratch. For Dhivehi, the authors had used 6 hours of training data from CV. The best result obtained was a WER of 54.7% using the Dhivehi specific model with pretraining (Hou, et al., 2020a).

Hassaan, et al. (2018) trained Dhivehi ASR models using CMUSphinx speech recognition toolkit. These models were Hidden Mark Model (HMM) based acoustic models which were boosted using N-gram LMs (Hassaan, et al., 2018). The authors had initially trained a model to recognize spoken numerals in Dhivehi which had a reported accuracy of 75%. They also trained another model to recognize general Dhivehi speech, which had a reported accuracy of 42.5%. The LM used by the authors was trained on 600MB of Dhivehi text scraped from the web and the acoustic model was trained on 48:33 of speech data collected through a web interface, mobile app, and Telegram bot that the authors created, which presented users with samples from the text corpus that were read and recorded.

Apart from formal work conducted on Dhivehi ASR, there has been some personal work done on the topic by some individuals as well. The most notable of these works are the Dhivehi ASR models trained by Shahuza Abdul Kareem [3] and published on Hugging Face. Specifically, her Wav2Vec2-XLS-R-1B-dv model [4] has the best reported WER of any Dhivehi ASR model known publicly so far. This model is a finetuned version of Facebook's Wav2Vec2-XLS-R-1B checkpoint (Conneau, et al., 2020) which follows the hybrid CNN/Transformer/CTC architecture introduced in Baevski, et al. (2020). The model was trained using

[1] https://github.com/coqui-ai/STT

[2] https://commonvoice.mozilla.org/

[3] https://huggingface.co/shahukareem?sort_models=downloads#models

[4] https://huggingface.co/shahukareem/wav2vec2-xls-r-1b-dv

around 25 hours of speech from Common Voice (Ardila, et al., 2019) version 8.0 with a character-based output vocabulary. The reported WER was 21.23% on Common Voice 8.0 evaluation set, which can be considered as the state-of-the-art for Dhivehi ASR.

3 Design

3.1 Pretrained ASR Models

Based on the results from previous works, a transfer learning approach of finetuning pretrained models was chosen instead of training models from scratch. The models chosen for finetuning were pretrained multilingual models from recent works that had claimed state-of-the-art performance results on common English ASR benchmarks. Here we will list and discuss the architectures of these models.

XLSR: XLSR (Conneau, et al., 2020) is a model pretrained on 53 languages following the Hybrid CTC architecture of (Baevski, et al., 2020). This architecture consists of a Convolution Neural Network (CNN) based feature extractor which extracts log-mel features from the input audio which are quantized through product quantization (Baevski, et al., 2020). The feature vectors are then passed into a Transformer based encoder network, which learns context vectors from these feature vectors using Contrastive Loss (Baevski, et al., 2020). For fine tuning of the model, the CNN layers can be frozen and a linear output layer corresponding to the desired output vocabulary can be initialized on top of the encoder network and trained using CTC loss (Baevski, et al., 2020). For the XLSR model, the authors were able to demonstrate that pretraining the encoder on a large number of languages produced improved performance when finetuning on low-resource languages (Conneau, et al., 2020). The authors had publicly released different sizes of the pretrained model checkpoints, out of which the Wav2Vec2-XLS-R-1B (XLSR-1B) [5] model checkpoint with 1B parameters was chosen for finetuning.

Whisper: Unlike Baevski, et al. (2020) which has relied on unsupervised pretraining on large amounts of raw speech, the authors of Whisper (Radford, et al., 2022) took the approach of pretraining a Transformer (Vaswani, et al., 2023) model with semi-supervised learning. The authors collected 680,000 hours of speech audio and corresponding text transcription data of different qualities from web sources for training (Radford, et al., 2022). Apart from just speech transcription, the authors trained the model in a multitask training format to perform a variety of speech processing tasks including, speech translation, language identification and voice activity detection (Radford, et al., 2022). Whisper uses a byte-level Byte Pair Encoding (BPE) text tokenizer and as opposed to typical ASR models, does not normalize the transcription text during training (Radford, et al., 2022). This results in more natural transcription that doesn't require further processing such as punctuation restoration (Radford, et al., 2022). The authors demonstrated that Whisper is able to achieve good performance on its supported languages in a zero-shot setting without any finetuning (Radford, et al., 2022). However, finetuning has been shown to further improve this performance. More interestingly, it has also been shown that Whisper could be finetuned on a new language that was not included in the original training data, by setting the target language to the phonetically closest language among the supported languages. Dhivehi is not officially supported by Whisper, but its closest neighbour Sinhalese is supported, which was used as a target language to finetune the model for Dhivehi. Whisper authors had also released pretrained checkpoints of different sizes, out of which, the Whisper Small [6] checkpoint containing 244M model parameters was chosen for this work.

Massively Multi-lingual Speech (MMS): Whereas XLSR was pretrained on 53 languages, the authors of MMS (Pratap, et al., 2023) scaled this to 1,107 languages. Their primary data source consisted of recordings of people reading translations of the New Testament in different languages (Pratap, et al., 2023). The authors created a GPU accelerated version of the Viterbi algorithm for computing the forced alignment of these recordings to the corresponding texts (Pratap, et al., 2023). Using this forced alignment method, the training dataset was constructed by chunking the recordings and texts into samples of short durations (Pratap, et al., 2023). The authors had used the same model architecture as XLSR

[5] https://huggingface.co/facebook/wav2vec2-xls-r-1b

[6] https://huggingface.co/openai/whisper-small

(Radford, et al., 2022) for MMS (Pratap, et al., 2023). Some of the models they had trained were fully multilingual with all the weights shared across all the training languages, while other models had used language specific adapter layers added to the encoder Transformer blocks (Pratap, et al., 2023). These language adapters constitute an additional 2M parameters which can be swapped out on the fly depending on the language being transcribed. They can also be finetuned separately without finetuning the whole model (Pratap, et al., 2023). The authors recommend finetuning only the language adapters for low-resource languages, but do suggest that full model finetuning is beneficial when more training data is available (Pratap, et al., 2023). The authors had released pretrained model checkpoints of different sizes, out of which MMS-1B-ALL (MMS-1B) [7] model checkpoint with 1B parameters was chosen for finetuning.

3.2 CTC Output Vocabularies

For finetuning the CTC based XLSR and MMS models, an output vocabulary had to be modelled. Previous works (Wav2Vec2-XLS-R-1B-dv) had used a character-based vocabulary of all 49 Thaana symbols and the Arabic ligatures Allah ﷲ (U+FDF2) and *Sallallahou Alayhe Wasallam* (Peace be Upon Him) ﷺ (U+FDFA) which commonly appear in Dhivehi text. However, Xu, et al. (2019) and Zhou, et al. (2021) had shown that training on sub-word based vocabularies where the tokens better correspond to phonemes can produce better results than simple character-based vocabularies. Similarly, Hou, et al. (2020a)'s results also show sub-word vocabularies generally giving better performance. Therefore, it was decided to also train using a more acoustically relevant sub-word vocabulary modelled with all consonants, all consonant-vowel pairs and aforementioned Arabic ligatures. Vowel diacritics were not included separately in this vocabulary as phonetically in Dhivehi, the vowels by themselves do not make any sound. In both vocabularies, additional special tokens were included, which were: the word delimiter token for spaces, the "[UNK]" token for unknown tokens, and the "[PAD]" token for the CTC blank token ϵ. The character-based vocabulary had 54 tokens while the sub-word vocabulary had 461 tokens.

3.3 Speech Dataset

The Dhivehi speech dataset chosen for training the ASR models were taken from Mozilla Common Voice (Ardila, et al., 2019). Common Voice (CV) is a crowd-sourced dataset where volunteers contribute recordings by reading text samples through a web interface (Ardila, et al., 2019). For version 13.0 of Common Voice that was used for this work, there were in total 64 hours of Dhivehi recordings from 331 different speakers. Out of these, only 38 hours were validated by contributors to be correct. The publishers further split these validated hours into different splits, out of which the "train" and "other" splits were selected for model training, "validation" split for evaluation during training and the "test" split for final model evaluation. The average length of samples in the dataset is 4.9 seconds.

Split	Duration	Samples
Train*	25:19:33	19,072
Validation	3:18:39	2,227
Test	3:21:26	2,212
Total	31:59:38	23,511

Table 1: Dataset splits. *For the train split, the original "train" and "other" splits have been combined.

3.4 Text Corpus

To train the language models for CTC LM decoding and for building synthetic spelling correction datasets, a Dhivehi text corpus was needed. For this, a text corpus was built using 4.2GB of text extracted from 1,197,470 news articles provided by ArchiveMV [8], a Maldives online news archive. To build the text corpus, the raw texts were tokenized into sentences using NLTK (Bird, et al., 2009), and the sentences were normalized by removing punctuation and replacing multiple whitespaces with single whitespaces. The text corpus contained 18,117,809 sentences and 258,345,316 tokens of 3,744,110 types.

3.5 Pretrained Text-to-Text Models

For the spelling correction task, a suitable Text-to-Text model architecture had to be chosen. As with

[7] https://huggingface.co/facebook/mms-1b-all

[8] https://archive.mv/

the ASR models, it was decided to use a pretrained model for this task to take advantage of the benefits of transfer learning. The model chosen was the UniMax (Chung, et al., 2023) model by Google Research. This is a multilingual version of the T5 (Raffel, et al., 2020) Text-to-Text Transformer model trained using a novel fairer language sampling method. The authors had released different sized pretrained checkpoints of this model, and the specific checkpoint chosen was the umT5-Small [9] checkpoint containing 300M model parameters.

4 Implementation

4.1 Data Processing

For preprocessing of the speech dataset for training, the text samples were normalized to remove all punctuation marks, which included removing the following characters; ‹ʕ,?.!-;:"‟%ʻ‟—ʻ…– . Furthermore, any newline characters or multiple spaces were replaced with single spaces. For training the Whisper model, the text was not normalized, but the generated output text and the reference text were normalized when calculating the WER. For the audio data, the audio files were resampled to 16Khz as this was the sampling rate the pretrained models were trained on.

4.2 ASR Model Training

The ASR models were trained using Hugging Face Transformers package (Wolf, et al., 2020), using PyTorch (Paszke, et al., 2019). All the models were trained using the AdamW (Loshchilov & Hutter, 2019) optimizer using a linear learning rate schedule with a warmup of 500 steps. For the MMS and XLSR models, a learning rate of 4.5e-05 was used, while a learning rate of 1e-5 was used for Whisper models. MMS and XLSR models were trained for 30 epochs (except for the MMS-1B-VL-DL-dv which was trained for 40 epochs) while Whisper models were trained for 15 epochs. During training, the models were evaluated every 400 steps using the validation set and a checkpoint saved. At the end of training, the checkpoint with the lowest WER was loaded and saved.

It was noted for MMS and XLSR models using the character-based vocabulary, the training reached the best performance around epoch 2, and beyond this the loss and the WER goes back up. This behaviour was not observed for the models using the sub-word vocabulary, which had a smooth decline of WER until the end of the training.

For both the MMS and XLSR models, the CNN feature extractor layers (Baevski, et al., 2020) were frozen during training. For the MMS models, initial experiments were conducted to train only the language adapter layers (Pratap, et al., 2023), however this resulted in relatively poor performance. So, for the final model training, full model training was performed for the MMS models.

The training was conducted on a AMD Ryzen 9 7950X 16-Core Processor machine with 128GB of RAM and dual RTX 3090 Ti 24GB GPUs running Ubuntu 23.04. Even though the machine had 2 GPUs, training was conducted using single GPUs as there were issues with parallel training the MMS and XLSR models. However, training sessions were conducted two at a time, with sessions assigned to either of the GPUs to take advantage of them both. The XLSR and Whisper models on average took around 11.5 hours to train, while the MMS model trained for 30 epochs took around 13 hours and the model trained for 40 epochs took around 17 hours.

Model Name	Base Model	Vocab.
XLSR-1B-VS-DL-dv	XLSR-1B	character
XLSR-1B-VL-DL-dv	XLSR-1B	sub-word
MMS-1B-VS-DL-dv	MMS-1B	character
MMS-1B-VL-DL-dv	MMS-1B	sub-word
Whisper-Small-DL-dv	Whisper-Small	-

Table 2: ASR models trained.

4.3 Language Models

The language models were trained using KenLM (Heafield, 2011) as 5-gram word-based models. KenLM uses modified Kneser-Ney smoothing [10] and produces efficient inference once trained. The language models were trained using the ArchiveMV text corpus. The first LM was the ArchiveMV-5gram (Amv-5g) model, trained

[9] https://huggingface.co/google/umt5-small

[10] http://www.foldl.me/2014/kneser-ney-smoothing/

without any restrictions. The second LM was the ArchiveMV-5gram-500k (Amv-5g-500k) model which was trained with the vocabulary limited to 500k common Dhivehi words to see whether this had any effect on the performance. For decoding of the CTC ASR model outputs using these LMs, pyctcdecode [11] Python package was used.

4.4 Spelling Correction Models

Model Name	Base Model	Training Samples
umT5-S-1M-dv	umT5-Small	1,043,532
umT5-S-10M-dv	umT5-Small	10,351,970

Table 3: Spelling Correction Models

To train the spelling correction models, first 2 different spelling correction datasets were created which were the 1M and 10M datasets. To create the datasets, random text samples from the ArchiveMV corpus were taken, which were then normalized, and synthetically spelling mistakes introduced into them using a series of random transformations. These transformations were designed to mimic the specific types of errors observed in the ASR model outputs. These include; randomly repeating characters at the end of the string, randomly inserting spaces within words, randomly removing spaces, interchanging phonetically similar letters and vowels (such as ﷺ *sheenu* and ﷻ *shaviyani*), and random insertions, deletions and modifications of characters. These transformed texts were used as the inputs for the 1M and 10M datasets, and the corresponding original normalized texts were used as the labels. The 1M dataset contained around 1 million samples while the 10M dataset contained around 10.3 million samples. For evaluation, a third dataset was created using the MMS-1B-VS-DL-dv model outputs decoded using the ArchiveMV-5gram LM on the validation set of the speech dataset.

The models were trained using the Hugging Face Transformers package, using PyTorch (Paszke, et al., 2019). All the models were trained for only 1 epoch as Transformer models are shown to overfit quickly to training data. AdamW optimizer was used with a learning rate of 4e-5. During training, the model was evaluated using the evaluation dataset and the relative WER improvement was recorded. At the end of the training the checkpoint with the highest relative WER improvement was loaded and saved. The spelling models were also trained using the same machine as the ASR models. The 1M model took around 5 hours to train, while the 10M model took around 56 hours.

[11] https://github.com/kensho-technologies/pyctcdecode

5 Evaluation

5.1 Experimental Setup

All the experiments were conducted on the same machine as used for training. Each experiment was conducted on a single GPU using an evaluation script that ran the ASR inference using all the trained ASR models, language models and spelling correction models and recorded the results into CSV files. For the spelling correction evaluations, for the CTC-based ASR models, the outputs from the ArchiveMV-5gram LM decoding was used as a baseline. For the Whisper model, the normalized text outputs were used as inputs to the spelling correction models.

5.2 Baseline ASR Model Results

First, we evaluate the baseline WER of all the trained models on the test sets of all the speech datasets without using any LM decoding or spelling correction and compare the results with the state-of-the-art Dhivehi ASR model Wav2Vec2-XLS-R-1B-dv. Here, for the CTC based XLSR and MMS models, Greedy Search decoding was used. And for the Whisper models, the generated texts were normalized as described in 4.1 before calculating the WER.

Model	Vocabulary	WER
Wav2Vec2-XLS-R-1B-dv	character	20.95
This work		
XLSR-1B-VS-DL-dv	character	56.29
XLSR-1B-VL-DL-dv	sub-word	19.94
MMS-1B-VS-DL-dv	character	38.26
MMS-1B-VL-DL-dv	sub-word	**16.19**
Whisper-Small-DL-dv	-	43.13

Table 4: Baseline ASR Model Results

As can be seen from the results, our XLSR and MMS based XLSR-1B-VL-DL-dv and MMS-1B-VL-DL-dv models were able to beat the state-of-the-art model on the CV-13 test benchmark. MMS-1B-VL-DL-dv had the best result with a relative improvement of 22.72% as compared to the state-of-the-art. It must be noted that MMS-1B-VL-DL-dv was trained for 10 additional epochs as compared to the other XLSR and MMS models. Therefore, this suggests that the other models also could potentially benefit from longer training.

Interestingly, the best performing CTC based XLSR and MMS models had used the sub-word vocabulary. This is consistent with the results observed by Xu, et al. (2019) and Zhou, et al. (2021) as the sub-words better correspond to phonetic characteristics of Dhivehi, making it easier for the models to learn a relationship between them and the audio. The sub-word vocabulary was able to improve the average WER for both MMS and XLSR models, with an average relative WER improvement of 61.79%. This goes to show that better sub-word modelling using just a little domain knowledge of the target language can go a long way in improving ASR model performance. The current sub-word vocabulary doesn't include all possible Dhivehi phonemes, such as consonant-vowel pairs followed by *sukun* (eg. مُنْ *un*) or diphones like مَ◌ِ *a-i.* Further investigations need to be done to see whether expanding the sub-word vocabulary to include these phonemes would improve performance.

5.3 Language Model Decoding Results

Model	Amv-5g-500k	Amv-5g
Wav2Vec2-XLS-R-1B-dv	20.03	19.85
This work		
XLSR-1B-VS-DL-dv	52.63	52.42
XLSR-1B-VL-DL-dv	18.66	18.44
MMS-1B-VS-DL-dv	39.50	38.99
MMS-1B-VL-DL-dv	**14.69**	**14.49**

Table 5**:** LM decoding results; WER for each of the trained XLSR and MMS models and the state-of-the-art Dhivehi ASR model when decoded using the 2 different LMs. Marked in bold is the best WER obtained for each LM.

The results show a further improvement of WER when LM decoding is used. Using the ArchiveMV-5gram LM, the WER of the MMS-1B-VL-DL-dv model is reduced to 14.49%, which is a 10.50% relative improvement over baseline. The ArchiveMV-5gram LM showed the overall best performance, followed by the ArchiveMV-5gram-500k LM which had the vocabulary limited to 500k words. This indicates that, for this particular case, limiting the LM vocabulary is worse for performance. Table 6 shows that the ArchiveMV-5gram LM improves the WER by 5.65% on average across all models.

Model	Amv-5g-500k	Amv-5g
Wav2Vec2-XLS-R-1B-dv	4.39%	5.25%
This work		
XLSR-1B-VS-DL-dv	6.50%	6.88%
XLSR-1B-VL-DL-dv	6.42%	7.52%
MMS-1B-VS-DL-dv	-3.24%	-1.91%
MMS-1B-VL-DL-dv	**9.26%**	**10.50%**
Average	4.67%	**5.65%**

Table 6: Relative WER improvement for LMs; Showing the relative WER improvement for all XLSR and MMS models using the 2 different LMs as compared to their baseline overall WER without LM decoding.

5.4 Spelling Correction Results

Model	umT5-S-10M-dv	umT5-S-1M-dv
Wav2Vec2-XLS-R-1B-dv	20.83	21.28
This work		
XLSR-1B-VS-DL-dv	50.14	50.73
XLSR-1B-VL-DL-dv	18.17	18.95
MMS-1B-VS-DL-dv	38.52	39.01
MMS-1B-VL-DL-dv	**14.26**	**15.27**
Whisper-Small-DL-dv	42.77	43.54

Table 7: Spelling correction results; WER for each of the trained models & the SOTA Dhivehi ASR model when spelling correction applied. For the MMS and XLSR models, spelling correction was applied after decoding with the ArchiveMV-5gram LM. For the Whisper model, spelling correction was applied after normalizing the output text.

The results show even more WER reduction with spelling correction. Using the umT5-S-10M-dv spelling model, WER for the MMS-1B-VL-DL-dv model is now reduced to 14.26%, which is the new state-of-the-art WER for CV-13 Dhivehi benchmark [12] [13] as of August 2023. This is a 31.93% relative WER improvement as compared to the 20.95% baseline WER of the SOTA Dhivehi ASR model on CV-13.

As can be seen from Table 8, only the umT5-S-10M-dv spelling correction model trained on 10M samples produced any overall WER improvement on average across all the models. This indicates the importance of training on more data. Interestingly, the Whisper model seem to benefit less from spelling correction, which is to be expected as the AED architecture of Whisper essentially learns a language model on the training transcriptions, negating the need for further text processing.

Model	**umT5-S-10M-dv**	**umT5-S-1M-dv**
Wav2Vec2-XLS-R-1B-dv	-4.94%	-7.20%
This work		
XLSR-1B-VS-DL-dv	**4.35%**	**3.22%**
XLSR-1B-VL-DL-dv	1.46%	-2.77%
MMS-1B-VS-DL-dv	1.21%	-0.05%
MMS-1B-VL-DL-dv	1.59%	-5.38%
Whisper-Small-DL-dv	0.83%	-0.95%
Average	**0.75%**	-2.19%

Table 8: Relative WER improvement for spelling correction; Showing the relative WER improvement for all the ASR models using the 2 different spelling correction models as compared to their baseline overall WER without spelling correction.

While spelling correction in addition to LM decoding seems to be a viable technique to boost the accuracy of CTC models, the relative improvement going from LM decoding to spelling correction is markedly less compared to the relative improvement going from baseline to LM decoding. Moreover, doing spelling correction on top of LM decoding adds additional memory and processing time overhead. The memory overhead could be mitigated by loading the spelling model after ASR inference is complete, however, the processing time can still be up to 1.6 times slower than LM decoding alone.

6 Conclusion

Our results show that pretrained multilingual ASR models can greatly benefit from language specific finetuning and optimizations. Specifically for CTC based models, proper sub-word modelling and language model decoding seems crucial. Multilingual models could also benefit from these techniques, as the language optimized sub-word vocabularies can be incorporated back into multilingual models and language specific LMs can be swapped out on the fly when decoding outputs for a specific language. While AED models seem to benefit less from these accuracy boosting techniques, they also seem to benefit from language specific finetuning. As for further accuracy improvement of Dhivehi ASR, more labelled speech data is needed for training, which could be generated using forced alignment as was done by Pratap, et al., 2023. Moreover, further expansion of the CTC sub-word vocabulary can be explored to see if it yields any improvement. Furthermore, for production systems, punctuation restoration Alam, et al., 2020 will need to be performed on the generated text to produce more readable transcripts, especially for the CTC based models.

Acknowledgments

I would like to thank my supervisor Dr. Mark-Jan Nederhof for his invaluable insight and guiding me throughout the project. A special thanks to the Commonwealth Scholarship Commission (CSC) and the UK Foreign, Commonwealth & Development Office (FCDO) and the British people for giving me the opportunity to study at the University of St. Andrews. Special thanks to Naail Abdul Rahman (Kudanai) for his valuable insight into Dhivehi ASR and sharing his past results with me. A similar thanks to Shahuza Abdul Kareem (Shahu) for her ground-breaking works on Dhivehi ASR and sharing her results with me. Thanks to Afrah Ismail for giving access to the ArchiveMV data. A very special thanks to CEO of Javaabu

12 https://paperswithcode.com/sota/speech-recognition-on-common-voice-dhivehi

13 https://huggingface.co/shahukareem/wav2vec2-xls-r-1b-dv

(where I'm a Co-founder) Mohamed Jailam for providing the hardware for training and evaluation without which this work would not have been possible. Finally, a special thanks to my lovely wife, Iju for always being there for me.

References

Tanvirul Alam, Akib Khan, and Firoj Alam. 2020. Punctuation Restoration using Transformer Models for High-and Low-Resource Languages. In *Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020)*, pages 132–142, Online. Association for Computational Linguistics.

Rosana Ardila, Megan Branson, Kelly Davis, Michael Henretty, Michael Kohler, Josh Meyer, Reuben Morais, Lindsay Saunders, Francis M. Tyers, and Gregor Weber. 2019. Common voice: A massively-multilingual speech corpus. *arXiv preprint arXiv:1912.06670. Verson 2*.

Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations. *arXiv preprint arXiv:2006.11477. Version 3.*

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2016. Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv preprint arXiv:1409.0473. Version 7.*

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit.* O'Reilly Media, Inc.

Hyung Won Chung, Noah Constant, Xavier Garcia, Adam Roberts, Yi Tay, Sharan Narang, and Orhan Firat. 2023. UniMax: Fairer and more Effective Language Sampling for Large-Scale Multilingual Pretraining. *arXiv preprint arXiv:2304.09151. Version 1*

Alexis Conneau, Alexei Baevski, Ronan Collobert, Abdelrahman Mohamed, and Michael Auli. 2020. Unsupervised Cross-lingual Representation Learning for Speech Recognition. *arXiv preprint arXiv:2006.13979. Version 2*.

Alex Graves, Santiago Fernández, Faustino John Gomez, and Jürgen A. Schmidhuber. 2006. Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 369–376. New York, NY, USA. Association for Computing Machinery.

Awni Hannun. 2017. Sequence Modeling with CTC. *Distill*.

Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, and Andrew Y. Ng. 2014. Deep Speech: Scaling up end-to-end speech recognition. *arXiv preprint arXiv:1412.5567. Version 2.*

Ibrahim Hassaan, Ifham Mohamed, Raaid Adam Rasheed, and Yameen Mohamed. 2018. Dhivehi automatic speech recognition system. *Project, Faculty of Engineering Science and Technology, Maldives National University.*

Kenneth Heafield. 2011. KenLM: Faster and Smaller Language Model Queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197. Edinburgh. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-term Memory. *Neural computation, Volume 9, Issue 8*, pages 1735-1780.

Wenxin Hou, Yue Dong, Bairong Zhuang, Longfei Yang, Jiatong Shi, and Takahiro Shinozaki. 2020a. Large-scale end-to-end multilingual speech recognition and language identification with multi-task learning. In *Proceedings of Interspeech 2020,* pages 1037-1041. Shanghai, China.

Wenxin Hou, Yue Dong, Bairong Zhuang, Longfei Yang, Jiatong Shi, and Takahiro Shinozaki. 2020b. Super Multi-lingual End-to-End Speech Recognition and Its Transfer learning to Low Resource Languages. *IPSJ SIG Technical Report, Vol.2020-SLP-133 No.8*

Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. *arXiv preprint arXiv:1711.05101. Version 3*

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32,* pages 8024–8035. Curran Associates, Inc.

Ngoc-Quan Pham, Tuan-Nam Nguyen, Sebastian Stueker, and Alexander Waibel. 2021. Efficient weight factorization for multilingual speech recognition. *arXiv preprint arXiv:2105.03010. Version 1*.

Rohit Prabhavalkar, Takaaki Hori, Tara N. Sainath, Ralf Schlüter, and Shinji Watanabe. 2023. End-to-End Speech Recognition: A Survey. *arXiv preprint arXiv:2303.03329. Version 1*.

Vineel Pratap, Andros Tjandra, Bowen Shi, Paden Tomasello, Arun Babu, Sayani Kundu, Ali Elkahky, Zhaoheng Ni, Apoorv Vyas, Maryam Fazel-Zarandi, Alexei Baevski, Yossi Adi, Xiaohui Zhang, Wei-Ning Hsu, Alexis Conneau, and Michael Auli. 2023. Scaling Speech Technology to 1,000+ Languages. *arXiv preprint arXiv:2305.13516. Version 1*.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever 2022. Robust Speech Recognition via Large-Scale Weak Supervision. *arXiv preprint arXiv:2212.04356. Version 1*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *arXiv preprint arXiv:1910.10683. Version 4*.

Francis M. Tyers and Josh Meyer. 2021. What shall we do with an hour of data? Speech recognition for the un-and under-served languages of Common Voice. *arXiv preprint arXiv:2105.04674. Version 1*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention Is All You Need. *arXiv preprint arXiv:1706.03762. Version 7*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations,* pages 38-45, Online. Association for Computational Linguistics.

Hainan Xu, Shuoyang Ding, and Shinji Watanabe. 2019. Improving End-to-end Speech Recognition with Pronunciation-assisted Sub-word Modeling. *arXiv preprint arXiv:1811.04284. Version 2*.

Shiliang Zhang, Ming Lei, and Zhijie Yan. 2019. Automatic Spelling Correction with Transformer for CTC-based End-to-End Speech Recognition. *arXiv preprint arXiv:1904.10045. Version 1*.

Yu Zhang, Wei Han, James Qin, Yongqiang Wang, Ankur Bapna, Zhehuai Chen, Nanxin Chen, Bo Li, Vera Axelrod, Gary Wang, Zhong Meng, Ke Hu, Andrew Rosenberg, Rohit Prabhavalkar, Daniel S. Park, Parisa Haghani, Jason Riesa, Ginger Perng, Hagen Soltau, Trevor Strohman, Bhuvana Ramabhadran, Tara Sainath, Pedro Moreno, Chung-Cheng Chiu, Johan Schalkwyk, Françoise Beaufays, and Yonghui Wu. 2023. Google USM: Scaling Automatic Speech Recognition Beyond 100 Languages. *arXiv preprint arXiv:2303.01037. Version 3*.

Wei Zhou, Mohammad Zeineldeen, Zuoyun Zheng, Ralf Schlüter, and Hermann Ney. 2021. Acoustic Data-Driven Subword Modeling for End-to-End Speech Recognition. In *Proceedings of Interspeech 2021*, pages 2886-2890. Brno, Czech Republic.