Phoneme transcription of endangered languages: an evaluation of recent ASR architectures in the single speaker scenario

Gilles Boulianne
CRIM (Centre de recherche informatique de Montréal)
gilles.boulianne@crim.ca

Abstract

Transcription is often reported as the bottleneck in endangered language documentation, requiring large efforts from scarce speakers and transcribers. In general, automatic speech recognition (ASR) can be accurate enough to accelerate transcription only if trained on large amounts of transcribed data. However, when a single speaker is involved, several studies have reported encouraging results for phonetic transcription even with small amounts of training. Here we expand this body of work on speaker-dependent transcription by comparing four ASR approaches, notably recent transformer and pretrained multilingual models, on a common dataset of 11 languages. To automate data preparation, training and evaluation steps, we also developed a phoneme recognition setup which handles morphologically complex languages and writing systems for which no pronunciation dictionary exists. We find that fine-tuning a multilingual pretrained model yields an average phoneme error rate (PER) of 15% for 6 languages with 99 minutes or less of transcribed data for training. For the 5 languages with between 100 and 192 minutes of training, we achieved a PER of 8.4% or less. These results on a number of varied languages suggest that ASR can now significantly reduce transcription efforts in the speaker-dependent situation common in endangered language work.

1 Introduction

Recent progress in automatic speech recognition (ASR) was made by training neural networks on increasingly large amounts of annotated data. To significantly reduce the efforts needed to transcribe endangered languages, ASR must reach sufficient accuracy when trained on relatively much smaller amounts of transcribed data. Already several research efforts have been dedicated specifically to ASR for low-resource languages, such as the IARPA BABEL program[1] and the NIST OpenASR Challenge[2]. However, creating an ASR system for a task like speaker-independent phonetic transcription is still difficult and requires amounts of transcription that are very large in the context of endangered languages. For example, Shi et al. (2021) recently concluded that at least 50 hours of training data are needed for this task, comparing ESPnet and HMM-based models on two languages.

In language documentation, field recordings are seldom made with a large number of speakers, but rather with a few speakers and for long durations (Amith et al., 2021). In these conditions, small amounts of transcribed data from a single speaker might be enough to train a phoneme recognizer with sufficient accuracy to automatically transcribe the remaining recordings from the same speaker. Concentrating on the single speaker scenario, Adams et al. (2018) evaluated a CTC-based LSTM model on Na and Chatino, and showed encouraging results for automated phoneme transcription as well as the effectiveness of this approach for linguistic work on endangered languages; they also created the open-source phonemic transcription tool Persephone. Wisniewski et al. (2020) compared Persephone performance on several endangered languages, focussing on data preprocessing concerns. Gupta and Boulianne (2020) compared end-to-end Persephone and wav2letter++ with an HMM-BLSTM hybrid for single speaker phoneme transcription, but using only one language, Cree. More recently, Adams et al. (2021) evaluated ESPnet on Na, Chatino and Japhug and integrated it into Elpis to create a user friendly docker container.

Although these previous studies obtained promising results, they report on different systems and languages, making them difficult to compare. In

[1] https://www.iarpa.gov/index.php/research-programs/babel

[2] https://www.nist.gov/itl/iad/mig/openasr-challenge

Findings of the Association for Computational Linguistics: ACL 2022, pages 2301 - 2308

addition, none has yet evaluated fine-tuning recent large models pretrained on many languages, for example XLSR (Conneau et al., 2020)[3], which are particularly well suited for low-resource languages. We think it fair to include such models, as we aim at a practical solution for the transcription problem at hand, regardless of the underlying approach.

In this paper we extend the body of work on single speaker phonetic transcription for endangered or low-resource languages while introducing distinctive contributions. For a meaningful comparison, we evaluate 4 systems with different modeling approaches across a common set of 7 languages, and 3 of those systems across 11 languages, while previous work was limited to either a single system on many languages, or many systems on a single language. In addition to Persephone and HMM-GMM models, we compare two recent architectures that have never been evaluated for single-speaker phoneme recognition: a Conformer model with a LF-MMI criterion, and a large pretrained multilingual model that we fine-tune for this task. We more firmly establish feasibility of accurate phonemic transcription with 3 hours or less of transcribed data by reporting on 4 new languages, including Cree and highly polysynthetic Inuktitut, in addition to 7 other previously studied in the literature. Finally, for reproducibility we make publicly available the curated dataset of public languages and a platform-independent container which allow users to reproduce the experiments from this paper[4] or train their own phoneme recognizer for a new endangered language.

2 Datasets

In this section we present the two sources of data used in the experiments. Although a number of low-resource language datasets are publicly available, very few provide enough data per speaker for speaker-dependent training. For example, the maximum duration from a single speaker in BABEL languages is limited to 20 minutes.

2.1 Public data

The Pangloss collection (Michailovsky et al., 2014) is an open archive of under-documented and mostly endangered languages. For our experiments we started from the single speaker subset[5] prepared by Wisniewski et al. (2020), which provides the audio file for each sentence and the corresponding sequence of labels, organized according to the format expected by Persephone.

Table 1 gives amounts of training and testing audio in minutes for each language in this dataset. The language code is ISO-639-3 (International Organization for Standardization, 2018). The number of phonemes depends on the particular rules for grapheme-to-phoneme conversion (more details in section 3.2). The IPA column says yes when the recording was transcribed in IPA phonemes, otherwise it was in orthographic text.

Language	code	train	test	IPA	phones
Yongning Na	nru	464	51	yes	68
Yongning Na	nru33	151	16	yes	68
Yongning Na	nru15	68	8.4	yes	68
Limbu	lif	99	11	yes	40
Dotyal	nep	95	10	no	58
Duoxo	ers	29	3.7	yes	33
Nahsta	mkd	23	2.9	yes	38
Mwotlap	mlv	20	2.5	no	26
Vatlongo	tvk	13	1.5	no	20

Table 1: Languages from the Pangloss collection. Train and test are amounts of speech in minutes. nru33 and nru15 are random subsets of nru, with respectively 33% and 15% of the original duration.

2.2 Private data

We also had access to transcribed Inuktitut, Cree and Tsuut'inai recordings collected and transcribed during the NRC Indigenous language project (Kuhn et al., 2020). We selected a single speaker subset from each language. Transcribed recordings from a single speaker of Kurmanji Kurdish were kindly shared with us by Translators without Borders. All private data was transcribed as text rather than phonetically, but writing systems for these four languages are sufficiently close to phonetic that it was not difficult to draw up their grapheme-to-phoneme table (section 3.2).

Language	code	train	test	IPA	phones
Cree	crl	192	18	no	24
Kurmanji	kmr	175	22	no	31
Inuktitut	iku	162	45	no	25
Tsuut'ina	srs	153	18	no	47

Table 2: Languages from private datasets. Train and test amount of speech recording in minutes.

[3]Note that this is different from "universal" multilingual systems which are not trained at all on the target language, such as the one from Li et al. (2020)

[4]Only the HMM-GMM baseline is already public at the time of this writing.

[5]Available at https://github.com/gw17/sltu_corpora.

3 STP test bed

In order to make a fair comparison, all models are evaluated through the same speech-to-phoneme recognition test bed. Called STP, it automates the steps required to train a phoneme recognizer from scratch i.e., with only a small number of audio files manually transcribed using a common transcription tool such as ELAN. Once trained, the recognizer can be applied to other audio files and yield the time-aligned phonetic transcription, in text or as ELAN annotations. The following sections detail the principles and design choices that were made to ensure STP could handle all the languages involved in the experiments, making it applicable to a wide range of features frequently encountered in endangered languages.

3.1 Training

Figure 1 illustrates the training process: it takes as input a set of ELAN transcription files in .eaf format, which point to audio files and contain their transcription in text or IPA phonemes. Then it: (1) prepares the input data as a Kaldi-compatible data directory, (2) splits data into train/validation sets, (3) converts the text transcript to IPA symbols using the user-supplied grapheme-to-phoneme table, (4) converts the IPA sequences to BPE (byte-pair encoding) sequences, (5) trains a BPE language model, (6) trains an acoustic model, and (7) applies the acoustic and language models to transcribe the test set in order to compute the phoneme error rate.

The Kaldi-compatible data directory is a simple format supported by several speech recognition toolkits and represents basically the same information as the ELAN file i.e., segments, features and time-aligned text transcriptions. The pipeline partitions the audio files at random, in separate train and test sets, in a 9:1 ratio. When training is complete, this held-out test set is used to measure the phoneme error rate as a diagnostic (section 3.5).

3.2 Grapheme-to-phoneme conversion

Some speech recognition models requires a pronunciation lexicon to convert provided transcriptions to IPA symbols, if they are written in text rather than IPA. Frequently such a lexicon does not already exist and would require effort and expertise to create. In STP we replace this requirement by a G2P (grapheme-to-phoneme) table. The table format is simple and can be quickly created manually from a description of the writing system. Each line has two fields: a sequence of UTF-8 text characters representing a grapheme from the writing system, and a sequence of IPA symbols for the corresponding pronunciation. An empty IPA symbol can be specified for graphemes that are to be ignored. The input text transcription is parsed, matching first the longest grapheme, to yield an IPA symbol sequence. This simple scheme is enough for languages which have a writing system close to phonetic. If the transcript is already in IPA, the table can be used to map several distinct IPA symbols to a single one, to remove tonal markers, for example. The main limitation of such a table is that each grapheme can only have a single IPA mapping, so no variant or alternative pronunciations are allowed for a given grapheme.

Figure 2 gives as an example the G2P table for Inuktitut (iku). All graphemes that appear in the text transcription must be listed in the table (or they will be ignored). For this study stress markers and tone markers were ignored when mapping to IPA symbols, but other markers (such as palatalization) were kept. The actual tables used for the public dataset in this paper are publicly available as well as the rest of the STP setup, as described in section 3.6.

3.3 Subword units: byte-pair encoding

Word units are not suitable for agglutinative or polysynthetic languages, since even impractically large vocabularies cover only a fraction of all possible words in those languages. The coverage problem could be solved with subword units such as morphemes or syllables, but BPE units (byte pair encoding) (Sennrich et al., 2015) are more commonly used and require no extra linguistic knowledge. We use BPE to encode commonly co-occurring groups of phonemes as single character. We capture phonotactic constraints with a N-gram language model of BPE units, which allows the N-gram model to capture contexts larger than the preceding N-1 phonemes.

To easily map between BPE units in language modeling and IPA symbols in acoustic modeling, we use an intermediate code (that we call "nxsampa") which unambiguously represents any IPA symbol with a single character symbol. With nxsampa, mapping from BPE to IPA is simple and invertible. BPE sequences are created by encoding nxsampa sequences with a BPE encoder, which is estimated on the training nxsampa sequences.

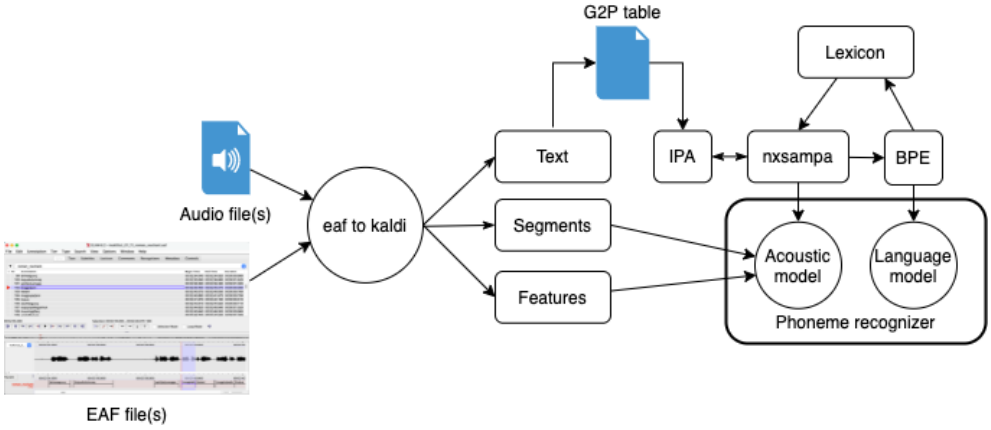

Figure 1: STP training pipeline.

<aa>	[a:]	<a>	[a]	<ii>	[i:]	<i>	[i]	<uu>	[u:]
<u>	[u]	<h>	[h]	<p>	[p]	<t>	[t]	<k>	[k]
<g>	[ɣ]	<m>	[m]	<n>	[n]	<s>	[s]	<l>	[l]
<jj>	[jj]	<j>	[j]	<v>	[v]	<r>	[ʁ]	<qk>	[qq]
<q>	[q]	<nng>	[ŋ:]	<ng>	[ŋ]	<ɬ>	[ɬ]	<b>	[b]

Figure 2: Grapheme-to-phoneme table for Inuktitut (iku) roman writing. Graphemes are enclosed in < >, phonemes in [ ]. This format is for illustration and differs from the actual format.

In preliminary experiments with Inuktitut (iku), we compared character-based perplexity[6] for language models based on BPE-encoded IPA sequences rather than roman character sequences. We found that perplexity was smaller (better) for IPA symbols, and was relatively independent of the BPE vocabulary size; we selected a value of 160 that we kept for all the following experiments. BPE training and extraction are implemented with SentencePiece (Kudo and Richardson, 2018).

Looking at the 160 BPE units extracted for Inuktitut, we find that they partially capture morphological information. 15% of the BPE units are single IPA symbols, 41% are syllables with 2 phonemes, and the remaining 44% of length 3 or more are morphemes[7] at least 76% of the time.

3.4 Transcription

Figure 3 details the transcription process, which takes an untranscribed audio file as input and returns an ELAN file containing a transcription tier with time-aligned IPA phonemes. The transcription steps are: (1) apply voice-activity detection (VAD) and group together adjacent voice segments that belong to the same speaker to define speech segments to be processed (diarization), (2) apply the trained phoneme recognizer to produce BPE sequences, (3) convert BPE sequences to IPA, (4) produce an ELAN file containing an annotation tier of time-aligned IPA phonemes. Note that the first step of segmenting the raw audio into short segments of speech can by itself significantly reduce transcription efforts, as it automates the first step of manual transcription.

3.5 Error rate computation

The training pipeline includes a diagnostic measurement of phoneme error rate on the held-out test set. It follows the transcription process of Figure 3 except that segments are defined by the reference transcription rather than VAD output. The recognizer output sequences are compared to the reference sequences obtained by applying the G2P table to the EAF transcription. The phoneme error rate is computed as usual as the ratio of the total number of insertions, deletions and substitutions over the number of phonemes in the reference.

3.6 Reproducibility

We make STP publicly available for research purposes[8], as a docker container which can be run on many operating systems. Already prepared datasets in ELAN format and their G2P tables for the 7 PanGloss languages are also made available in a github repository[9]. HMM-GMM baseline results found in this paper can be easily reproduced by running the container on the provided datasets.

[6] Counting roman characters rather than words, as it is directly related to the bits-per-character measure and is less dependent on the subword inventory (Cotterell et al., 2018).

[7] More exactly, are in the set of morphemes produced by the Uqailaut analyzer (Farley, 2012) from the Nunavut Hansards.

[8] https://hub.docker.com/r/crimca/speech_to_phonemes

[9] https://github.com/crim-ca/speech_to_phonemes

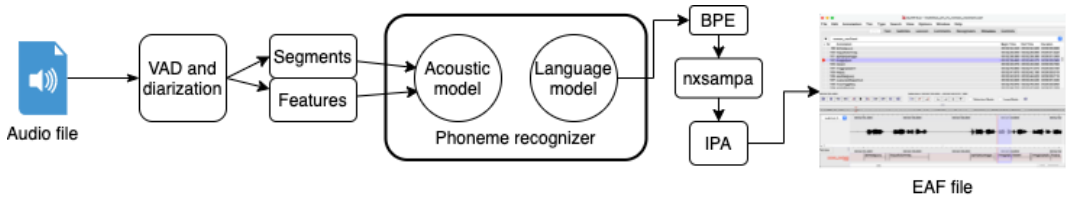

Figure 3: STP transcription pipeline.

4 Experiments

We evaluated models from four main classes: a conventional hidden Markov models with Gaussian mixture models (HMM-GMM), an end-to-end recurrent neural network, a convolutional/transformer-based neural network, and a large pretrained transformer neural network. We compare time required for training, hardware and software requirements, and accuracy of transcription. For a fair comparison, all models are trained and evaluated using the same STP test bed and languages. Only the training pipeline needs to be run since it includes computation of phoneme error rate on the held-out part of the dataset. For a given model, the same hyperparameters were used across all languages, and are taken from the reference published paper (except where differences are noted in following sections). The test set is used only for measuring phoneme error rate and is not involved in any tuning.

4.1 Baseline (HMM-GMM)

Good results were previously obtained with HMM-GMM for single speaker phoneme recognition, in low-resource conditions for Cree (Gupta and Boulianne, 2020). To extend those results to other languages, we implemented a general HMM-GMM baseline with the Kaldi toolkit (Povey et al., 2011), modified for phoneme recognition with BPE units. The HMM-GMM acoustic model training follows the usual steps of the Kaldi "wsj" recipe[10], starting with monophone models (larger than usual beamwidth) and building up to LDA+MLLT+SAT triphone models (tri4), with 1000 model states and a total of 20,000 Gaussian means, amounting to about 800K free parameters. Input features are MFCC "hires" features with 40 coefficients computed from audio sampled at 16 kHz. The language model is a 4-gram backoff trained using srilm (Stolcke, 2002) with Witten-Bell discounting (Witten and Bell, 1991).

[10] https://github.com/kaldi-asr/kaldi/tree/master/egs/wsj/s5

4.2 Persephone (Wisn20)

For reference we also include results published by Wisniewski et al. (2020). This end-to-end system is a long short-term memory neural recurrent network (LSTM) trained using the Persephone toolkit, with a connectionist temporal classification (CTC) loss criterion. It has no explicit language model, relying only on the implicit modeling of the LSTM. The dataset on which Wisniewski et al. (2020) reported their results was the same as described here in Section 2.1, except that due to limitations of Persephone, they had to exclude audio chunks longer than 10 seconds. This only made a significant difference for Dotyal (nep), which was limited to 44 minutes in Wisniewski et al. (2020), while here we are able to use 95 minutes.

4.3 Pretrained multilingual model (XLSR-53)

XLSR-53[11] is a large version of the wav2vec2.0 model (Conneau et al., 2020), pretrained on 56,000 hours from 53 languages from Multilingual LibriSpeech, CommonVoice and BABEL datasets. The encoder is transformer-based with a convolutional front-end and is shared across languages, similar to the approach of Dalmia et al. (2018).

We fine-tune XLSR-53 on each language using the audio segments from the STP prepared data. The feature extraction layers are frozen and only decoder layers are trained, using nxsampa labels with a CTC loss. We use nxsampa rather than BPE since XLSR-53 model words as sequences of single characters. We rely on decoder attention heads for the language model and do not use an external one.

SpecAugment (Park et al., 2019) was applied with the default parameters. Batch size and learning rate are optimized separately for each language to obtain stable learning on the training set. For all languages, training is stopped after a fixed number of epochs that represents approximately 16,000

[11] https://github.com/pytorch/fairseq/tree/main/examples/wav2vec, model revision 38ae400ce326d5c29d1c66ec6140e4b50a9b34dd from March 10, 2021.

steps; warmup is set at 10% of total steps. The total number of parameters in the model is 315M, but fine-tuning updates only the language model head layers, which amount to 76K trainable parameters.

4.4 Conformer with LF-MMI (k2-conf)

The Conformer model (Gulati et al., 2020) is a transformer-based architecture augmented with convolutional input layers. We based our implementation on the snowfall k2-fsa[12] version. As for HMM-GMM, we trained the model with the same audio segments and BPE labels prepared by the STP test bed. The training criterion was LF-MMI (Povey et al., 2016). All languages were trained for 160 epochs. The language model is the same 4-gram model used by the HMM-GMM baseline. Data augmentation was performed using speed perturbation with five values $[0.8, 0.9, 1.0, 1.1, 1.2]$. Other data augmentation like SpecAugment and noise/reverberation were not used. The number of trainable parameters in this model is 32M.

5 Results

The four architectures are compared in terms of phoneme error rate, and elapsed time for training, in Table 3 for the public dataset and Table 4 for the private dataset. HMM-GMM refers to the baseline HMM-GMM from section 4.1, Wisn20 to Persephone from section 4.2, XLSR-53 to the pretrained multilingual model of section 4.3, and k2-conf to the Conformer model of section 4.4.

In each table, languages appear in descending order of total audio duration available for training. Note that the nru33 subset is used here rather than the full nru, to make it more comparable with other languages. True in the IPA column indicates that transcriptions are IPA symbols, false means that transcriptions are orthographic.

Phoneme error rates (PER) reported are obtained using the speaker turn segmentation from the transcript. In an actual transcription pipeline, VAD would be used and might introduce errors that could slightly degrade the actual PER. Also note that the reference is the phoneme string generated by the G2P table, so tone or stress errors are not counted if tone or stress is not represented by distinct phonemes in the table.

[12] https://github.com/k2-fsa/snowfall

5.1 Discussion

In (Gupta and Boulianne, 2020) we observed that pretraining an HMM-BLSTM on several languages, rather than Cree only, did not help. Here, phoneme error rate (PER) columns in Table 3 show that pretrained XLSR-53 outperforms other models for all languages in public datasets. In one case (mlv), it obtains 8.6% PER with only 20 minutes of training. Similarly in the private dataset, Table 4 shows XLSR-53 outperforming the other models for all languages. Note that the HMM-GMM result for Cree (crl) is 13.0% PER, slightly better than for the HMM-BLSTM model without LM result from (Gupta and Boulianne, 2020).

It was feasible to train HMM-GMM with 10 different random train/test partitions[13] and compute the Student's t 95% uncertainty intervals shown in the PER column. The uncertainty remains relatively small even for the smallest datasets which contain only a few minutes of test speech.

We find a significant degradation of performance for all models when audio training duration drops to 99 minutes or less. This can be seen in Table 5, where we summarized results from Tables 3 and 4 by grouping languages in two classes based on amounts of audio available for training. Languages with more than 99 minutes are nru33, crl, kmr, iku, srs, and those with 99 minutes or less are lif, nep, ers, mkd, mlv and tvk. The average weights each language equally.

Group	%PER HMM-GMM	%PER XLSR-53	%PER k2-conf
> 99min	13.8 ±1.2	5.9	11.0
<= 99min	46.0 ±3.5	15.3	53.5

Table 5: Average phoneme error rate when public and private datasets are grouped by audio training duration.

Table 5 shows that with over 99 minutes, HMM-GMM, XLSR and k2-conf have a PER of 13.8% or less. When training falls to 99 minutes or less, PER increases considerably for k2-conf, moderately for HMM-GMM and less dramatically for XLSR-53. To confirm this, in Table 6 we compare various amounts of training for the *same* language, Yongning Na (nru). From 464 to 151 minutes, error rates increase much less for HMM-GMM and XLSR, than from 151 to 68 minutes, so there seems to be a divide around 90 minutes, or 1.5 hours. The result for the full nru set from Wisniewski et al. (2020) is included for completeness.

[13] Except 6 for iku, which had only 6 different recordings.

Language code	IPA	Audio (minutes)	%PER HMM-GMM	%PER Wisn20	%PER XLSR-53	%PER k2-conf	Time (h) HMM-GMM	Time (h) XLSR-53	Time (h) k2-conf
nru33	True	151	19.3 ±1.1	-	**7.1**	11.4	0.43	23.2	4.4
lif	True	99	30.2 ±0.9	36.8	**14.0**	30.4	0.72	13.5	2.60
nep	False	95	62.0 ±1.7	96.5	**22.3**	66.0	0.68	16.3	2.86
ers	True	29	45.8 ±1.7	38.3	**14.5**	69.6	0.27	10.9	0.92
mkd	True	23	53.1 ±3.0	92.6	**17.3**	27.3	0.35	10.1	0.84
mlv	False	20	28.8 ±2.6	93.2	**8.6**	69.1	0.25	10.5	1.00
tvk	False	13	57.2 ±3.6	81.8	**15.0**	58.7	0.17	9.1	0.35
Average		61.4	42.1 ±3.7	73.2	**13.6**	47.5	0.4	13.4	1.9

Table 3: Percent phoneme error rate (%PER) for languages in the public dataset, ordered by decreasing amount of audio used in training (Audio). Elapsed hours for training are in the Time columns. Average gives equal weight to every language.

Language code	IPA	Audio (minutes)	%PER HMM-GMM	%PER XLSR-53	%PER k2-conf	Time (h) HMM-GMM	Time (h) XLSR-53	Time (h) k2-conf
crl	False	192	13.0 ±0.7	**6.6**	10.4	0.82	22.4	5.37
kmr	False	175	14.4 ±0.8	**4.4**	15.9	0.85	15.4	4.52
iku	False	162	13.8 ±3.3	**8.4**	12.2	0.65	21.2	4.15
srs	False	153	8.4 ±0.3	**3.1**	5.1	0.48	14.7	3.89
Average		170.5	12.3 ±1.0	**5.6**	10.9	0.7	18.4	4.5

Table 4: Percent phoneme error rate (%PER) for languages in the private dataset, ordered by decreasing amount of audio used in training (Audio). Elapsed hours of training time are in the Time columns. Average gives equal weight to every language.

Code	Audio (minutes)	%PER HMM-GMM	%PER XLSR-53	%PER Wisn20
nru	464	13.1	6.5	18.6
nru33	151	17.0	7.1	-
nru15	68	25.6	13.6	-

Table 6: Percent phoneme error rate (%PER) for Yongning Na (nru) when random subsets of various duration are used in training. nru=full set, nru33 = 33% of full set, nru15 = 15% of full set.

Are these error rates low enough to facilitate language documentation? Amith et al. (2021) found that character error rates around 6 to 10% could reduce the effort of accurate transcription by 75%. Here a PER below 9% was obtained for all the languages in Tables 3 and 4 which had more than 99 minutes for training, so it looks like useful error rates are feasible with 1.7 hours of transcribed data.

Regarding the elapsed time required for training, the last three columns in Tables 3 and 4 show major differences between the models[14]. The HMM-GMM system is not only much faster, but is also the only one which does not use a GPU. So although it does not yield the best PER, it could still be a useful model for field work, since it can run on limited hardware, and makes it possible to test many different hypothesis in a short time, for example about the phoneme inventory.

[14]Times measured on a single Intel (R) Core(TM) i5-7500 CPU running at 3.40 GHz, or equivalent, and a single GTX1080Ti GPU, or equivalent, when applicable

These results are obtained with only one speaker per language. While generalization is possible when looking at several languages, interpretation for one language in particular must be done carefully. This is a true limitation but also reflects the challenge of working with endangered languages.

6 Conclusion

Fine-tuning a large pretrained multilingual model clearly outperformed the other approaches. For the 6 languages with 99 minutes or less of training data, the pretrained model was able to average a phoneme error rate of 15.3%. We obtained 8.4% or less PER for the 5 languages which had between 100 and 192 minutes. At this level of performance, we expect ASR to significantly reduce the effort required for transcription of endangered languages. Further work is needed to explore handling of tone and stress markers, and enlarge the curated speaker-dependent dataset with other publicly available languages.

Acknowledgements

This work would not have been possible without collaborators from the NRC project and all other contributors to the datasets. The authors would also like to thank the Ministry of Economy and Innovation (MEI) of the Government of Quebec for its continued support.

References

Oliver Adams, Trevor Cohn, Graham Neubig, Steven Bird, and Alexis Michaud. 2018. Evaluating Phonemic Transcription of Low-Resource Tonal Languages for Language Documentation. In *Proc. LREC*, pages 3356–3365.

Oliver Adams, Benjamin Galliot, Guillaume Wisniewski, Nicholas Lambourne, Ben Foley, et al. 2021. User-friendly automatic transcription of low-resource languages: Plugging ESPnet into Elpis. In *Proc. 4th Workshop on the Use of Computational Methods in the Study of Endangered Languages*, pages 1–12.

Jonathan D. Amith, Jiatong Shi, and Rey Castillo García. 2021. End-to-End Automatic Speech Recognition: Its Impact on the Workflowin Documenting Yoloxóchitl Mixtec. In *Proc. 1st Workshop on Natural Language Processing for Indigenous Languages of the Americas*, pages 64–80.

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

Ryan Cotterell, Sebastian J. Mielke, Jason Eisner, and Brian Roark. 2018. Are All Languages Equally Hard to Language-Model? *Computing Research Repository*, arXiv:1806.03743.

Siddharth Dalmia, Ramon Sanabria, Florian Metze, and Alan W Black. 2018. Sequence-based Multilingual Low Ressource Speech Recognition. In *Proc. ICASSP*, pages 4909–4913.

Benoît Farley. The Uqailaut project [online]. 2012. Accessed 2021-11-02.

Anmol Gulati, James Qin, Chung Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang. 2020. Conformer: Convolution-augmented transformer for speech recognition. In *Proc. Interspeech*, pages 5036–5040.

Vishwa Gupta and Gilles Boulianne. 2020. Speech Transcription Challenges for Resource Constrained Indigenous Language Cree. In *Proc. 1st Joint SLTU CCURL*, pages 362–367.

International Organization for Standardization. Codes for the representation of names of languages - Part 3: Alpha-3 code for comprehensive coverage of languages [online]. 2018. Accessed 2021-11-13.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing. In *Proc. EMNLP*, pages 66–71.

Roland Kuhn, Fineen Davis, Alain Désilets, et al. 2020. The Indigenous Languages Technology project at NRC Canada : An empowerment-oriented approach to developing language software. In *Proc. COLING*, pages 5866–5878.

Xinjian Li, Siddharth Dalmia, Juncheng Li, Matthew Lee, Patrick Littell, Jiali Yao, Antonios Anastasopoulos, David R. Mortensen, Graham Neubig, Alan W. Black, and Florian Metze. 2020. Universal Phone Recognition with a Multilingual Allophone System. In *Proc. ICASSP*, pages 8249–8253.

Boyd Michailovsky, Martine Mazaudon, Alexis Michaud, Séverine Guillaume, Alexandre François, and Evangelia Adamou. 2014. Documenting and researching endangered languages: the Pangloss Collection. *Conservation*, 8:119–135.

Daniel S. Park, William Chan, Yu Zhang, Chung Cheng Chiu, Barret Zoph, Ekin D. Cubuk, and Quoc V. Le. 2019. SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition. In *Proc. Interspeech*, pages 2613–2617.

Daniel Povey, Arnab Ghoshal, et al. 2011. The Kaldi speech recognition toolkit. In *Proc. ASRU*.

Daniel Povey, Vijayaditya Peddinti, Daniel Galvez, et al. 2016. Purely sequence-trained neural networks for ASR based on lattice-free MMI. In *Proc. Interspeech*, pages 2751–2755.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural Machine Translation of Rare Words with Subword Units. *Computing Research Repository*, arXiv:1508.07909.

Jiatong Shi, Jonathan D. Amith, Rey Castillo García, Esteban Guadalupe Sierra, Kevin Duh, and Shinji Watanabe. 2021. Leveraging End-to-End ASR for Endangered Language Documentation: An Empirical Study on Yoloxóchitl Mixtec. *Computing Research Repository*, arXiv:2101.10877.

Andreas Stolcke. 2002. SRILM - An extensible language modeling toolkit. *Proc. ICSLP*, pages 901–904.

Guillaume Wisniewski, Séverine Guillaume, and Alexis Michaud. 2020. Phonemic Transcription of Low-Resource Languages: To What Extent can Preprocessing be Automated? In *Proc. 1st Joint SLTU CCURL Workshop*, pages 306–315.

Ian H. Witten and Thimoty C. Bell. 1991. The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression. *IEEE Transactions on Information Theory*, 37(4):1085–1094.