# Automated phonetic transcription for varieties of English: wav2vec 2.0 fine-tuned on the Buckeye Corpus

Virginia Partridge, Joe Pater, Parth Bhangla, Ali Nirheche & Brandon Prickett
*University of Massachusetts Amherst*

## 1 Introduction

Despite the impressive recent progress in speech recognition technology and its application to a range of domains, there has been little application to phonetic transcription in linguistics and adjacent fields. In phonology, reliable automated transcription would vastly increase the database for phonological analysis and theorizing. This dearth of application is particularly surprising since state-of-the-art results on phonetic transcription of English using the TIMIT corpus (Garofolo et al., 1993) are reported in the presentation of the landmark wav2vec 2.0 framework in Baevski et al. (2020). The current aim of our Wav2IPA project is to increase the usefulness of automated phonetic transcription for the study of varieties of English by developing new models, applying them to a range of dialects, and by making the models broadly accessible.

The primary focus of contemporary research on automated phonetic transcription is the development of universal models, trained on data from multiple languages, and applicable in principle to any language (Li et al., 2020; Xu et al., 2022; Taguchi et al., 2023; Zhu et al., 2025). The main applied goal of this research is to aid in documentation and analysis of under-resourced languages. These models are usually trained on transcriptions generated from grapheme-to-phoneme conversion, using existing orthographically transcribed speech corpora. Our approach differs by starting with a single variety of a single language, using high quality direct phonetic transcriptions of the speech itself. From there we are moving on to other varieties and other languages, taking into account differences in how phonetic symbols map to acoustics in different linguistic and transcription systems.

Here we present results from fine-tuning wav2vec 2.0 on the Buckeye Corpus of conversational English speech (Pitt et al., 2005). Buckeye differs from TIMIT in having speech from conversations rather than sentence reading, and in focusing on a single variety. The Buckeye speakers are white residents of the Columbus Ohio area, while the TIMIT speakers are from a number of dialect regions of the US.

wav2vec 2.0 is a foundation model, trained in a self-supervised fashion on a large corpus of unlabeled speech. It is adapted for use in a particular domain – fine-tuned – using labeled speech and supervised learning. We divide Buckeye into training, development and test sets, and experiment with different amounts of training data for fine-tuning, as well as different gender and age distributions in training data. We find that good results are achieved with about two hours of training data, and that performance is generally robust to skews in the makeup of the training data. These findings are encouraging for the project of extending these methods to languages and varieties that are less well resourced.

We know of no prior results on Buckeye to compare ours with[1]. One might ask whether it is necessary to create a custom model for a single variety or single language given the existence of models with broader scope. We therefore compare our model on the Buckeye test set to a group of universal models, and ones trained on TIMIT. These comparisons suggest that targeted fine-tuning is worthwhile where the data exist.

[1] Buckeye has been used to train phone alignment models (e.g. Kreuk et al. 2020) and to test a universal phone transcriber (Zhu et al., 2025), but we have not been able to find any prior work using it to train a phone transcriber.


Proceedings of *AMP 2025*

As a first step in extending our model to other varieties, we also used the TIMIT corpus test set (see Lee et al. (2025) for prior work using wav2vec 2.0 for dialectal analysis). Our Buckeye-tuned model continues to outperform the universal models on the TIMIT test set, but by a smaller margin. We expected our model to perform best on the variety on which it was trained (Northern Midland), and it did, but by a much smaller margin than expected. This is presumably because some dialect differences are captured by the Buckeye acoustics-to-symbol mapping (and many are absent in the TIMIT transcriptions). We provide some examples of where this is the case, and discuss some remaining challenges for automated transcription of other varieties.

To make our models broadly accessible, we have released them publicly along with a web-based interface which supports input and output in Praat TextGrid (Boersma & Weenink, 2026) format (see `https://websites.umass.edu/comphon/wav2ipa-automated-ipa-transcription/`).

## 2 Background and related work

The wav2vec 2.0 model architecture begins with a pre-training self-supervised learning stage during which only unlabeled audio data are required. The audio waveforms are passed through a multi-layer convolutional neural network which encodes continuous latent speech representations. Using a self-attention transformer layer to encode relative positions of the latent speech representations and quantization to limit the number of latent speech representations to a finite amount, the self-supervised learning proceeds by masking, or hiding, a time slice of audio representation. Minimizing the contrastive loss function requires the neural network to predict the correct quantized representation of the masked audio using the surrounding context, so the model jointly learns both the discrete speech representation for the masked slice and representations of the surrounding context (Baevski et al., 2020).

In the second phase, this self-supervised foundation model is fine-tuned with a corpus of transcribed audio by adding a final linear projection layer and *connectionist temporal classification* (CTC) loss function, where the model is updated to predict the character label that corresponds with each audio time step. CTC loss was introduced to avoid the need for pre-segmented and aligned audio transcriptions for training speech recognition models. Because sounds can stretch over multiple time steps in the audio, the CTC loss algorithm takes into account merging the same character label over adjacent time steps and maximizes the conditional probabilities of different labeling alignments that lead to the correct transcription (Graves et al., 2006). Thanks to the combination of self-supervised pre-training and the use of CTC loss to predict labels for time steps in the audio, a relatively small amount of transcribed audio is needed to fine-tune a wav2vec 2.0 model for automatic speech recognition. Baevski et al. (2020) estimated between 10 minutes and 1 hour of audio to be sufficient for fine-tuning. Furthermore, that audio does not need to be time-aligned at the segment (letter/phone) or word level, even for phone recognition tasks.

Transformer-based speech recognition models have frequently been employed for phone recognition tasks. With the introduction of wav2vec 2.0, Baevski et al. (2020) reported achieving an 8.3% phone error rate (PER) on the TIMIT test set when fine-tuning their models for phone recognition with the TIMIT corpus. Since then, focus has moved toward universal and multilingual phone recognition, where a single model can produce IPA transcriptions for many languages, even those never seen in training data. Conneau et al. (2021) found that a single wav2vec 2.0 model pre-trained on audio from 53 languages, XLSR-53[2], outperformed monolingual pre-trained wav2vec 2.0 models when fine-tuned for phone recognition on data in 10 languages, even when the monolingual pre-training language matched the target language. Building on this, Xu et al. (2022) confirmed the effectiveness of multilingual pre-training for phoneme recognition and extended XLSR-53's coverage to additional languages by incorporating grapheme-to-phoneme (G2P) tools and a scheme for mapping phoneme inventories between languages based on differences in articulatory features. Taguchi et al. (2023) also applied wav2vec 2.0 to multilingual transcription by using G2P techniques to prepare fine-tuning data, but selected data from languages with relatively transparent mapping between orthography and pronunciation. Taguchi et al. (2023) also established the phone feature error rate (PFER), a measure based on the Hamming distance between phones' articulatory features.

Other deep learning approaches to phonetic speech recognition have had success recently as well. Zhu et al. (2025) present ZIPA, a model based on the Zipformer architecture, which they found to be more efficient than wav2vec 2.0 in terms of computational resources and the amount of data required for training (Zhu et al., 2025; Yao et al., 2024). Similar to Taguchi et al. (2023), training data for ZIPA was prepared through

[2] `https://huggingface.co/facebook/wav2vec2-large-xlsr-53`

careful selection, application of the GP2 tools Epitran (Mortensen et al., 2018) and CharsiuG2P (Zhu et al., 2022), and normalization to correct inconsistencies in Unicode encodings and to remove diacritics. The older Allosaurus architecture introduced a universal phone encoding layer that can be combined with a language specific, hand-built mapping between allophones and phonemes in the loss function, allowing the model to be used both in universal phone recognition or phone recognition for a single target language. The approach is flexible and modular – these can be the final layers in any neural network speech recognition model. Their implementation uses an LSTM model with a CTC loss function, with training data from general purpose speech recognition datasets preprocessed with Epitran (Li et al., 2020; Mortensen et al., 2018). A throughline through the non-wav2vec 2.0 approaches to automatic universal phonetic transcription is that they make careful, linguistically informed choices regarding selecting and pre-processing training data or the inclusion of language specific resources.

Our work seeks to combine careful attention to data selection and transcription quality with the success of the wav2vec 2.0 model architecture to create high quality phonetic transcription models for English. Through describing our considerations during the training process and publicly releasing code, models, and interactive tools, our hope is that this work will make wav2vec 2.0 and related models more accessible for linguistic research, whether that involves using pre-trained models directly or fine-tuning them on other datasets.

## 3 Methods

**3.1** *Corpus preparation and pre-processing* We divided the Buckeye Corpus into subsets for training, validation and a held-out test set by speaker, so that success on the test set requires generalizing to new speakers. Each audio file was split into sequences of speech separated by periods of silence or the interviewer talking. We subdivided samples into word sequences of 12 seconds or less. For training and validation, we removed samples with non-speech sound longer than 12 seconds. Any utterance samples shorter than 0.1 seconds were also removed from the training split. These length restrictions on utterance samples make fine-tuning wav2vec 2.0 tractable in terms of memory usage and reasonable minimum time step length. After the length restriction filter, the average duration of utterances in the training set is 2.74 seconds, with a standard deviation of 3.04. The upper-bound duration filter was not consistently applied to the held-out test set (i.e., some utterances longer than 12 seconds were split, while others were not), but this does not affect results since memory usage is not a concern during inference. No minimum length filter was imposed on the validation and held-out test splits. Our training set consists of 18,344 samples from 24 speakers, the validation set 5522 samples from 8 speakers, and the test set 5079 samples from 8 speakers. The ratio of speaker demographics for age and gender were maintained across each subset. Details on the speaker IDs and duration of audio included in each data split are available in the supplemental materials.

The Buckeye ARPABET-based transcription system straightforwardly maps to the IPA (this is not true of TIMIT; see 3.3). One aspect to note is that it uses a single symbol for stressed and unstressed mid central vowels since they are similar in quality, and transcribers had difficulty distinguishing them (Fosler-Lussier et al., 2007). We follow Buckeye in using [ah]/[ʌ], but schwa is likely more accurate (Lindsay, 2022), and in our symbol reduction (see 3.3), we map [ʌ] to [ə]. The full set of Buckeye symbols and our mapping to IPA can be found in the supplementary materials; how the transcribers were instructed to use the symbols is explained in the manual (Kiesling et al., 2006). We applied corrections of glottal transcription (Seyfarth & Garellek, 2020), and then converted the Buckeye transcriptions to IPA using our modified version[3] of the phonecodes Python library (Hasegawa-Johnson, 2019). Diphthongs are treated as a single symbol, and diacritics as part of the symbol they attach to (e.g. syllabic [ɪ̩] is a completely separate symbol from syllabic [l̩] and non-syllabic [ɹ]). We did not explore the alternative of using features rather than phones, nor did we examine models to see if they learned featural representations; both are excellent topics for further research.

**3.2** *Model fine-tuning* Our experiments with fine-tuning wav2vec 2.0 models on the Buckeye Corpus were designed with dual goals in mind: producing an effective English phonetic transcription model and understanding how the composition and quantity of training data affect model performance. By exploring these questions, we contribute not only useful models to the research community, but also establish some practical strategies for creating training corpora and fine-tuning models. Our model fine-tuning experiments can be characterized as groups designed to answer the following questions:

[3] `https://pypi.org/project/phonecodes/`

- **Can a strong baseline be achieved by fine-tuning XLSR-53 on 4000 Buckeye samples?** This initial set of experiments were targeted at determining the typical performance variation due to random variation in training data. We fine-tuned XLSR-53 five times using a different random set of 4000 utterance samples from the training split for each time. The number of utterance samples was balanced by speaker gender.
- **How does the total duration of training data available affect performance?** Here, we are interested in understanding the minimum amount of fine-tuning data required to produce a model that is adequate for phonetic transcription, as well as whether significantly increasing the amount of training data pays dividends in model performance improvements. We fine-tuned XSLR-53 using the follow number of utterance samples balanced by speaker gender: 100, 200, 400, 800, 1600, 3200, 6400, 12,800, and 18,252 (the maximum number of samples that can still be balanced for gender). Additionally, we fine-tuned XLSR-53 on the full training dataset of 18,433 samples. For each number of utterance samples, XLSR-53 was fine-tuned five times using a different randomly selected subset of the training data split with random orderings of samples in batches.
- **Does the share of data from speakers of different demographic groups in training data generalize to across the test set?** We fine-tuned XLSR-53 with 4000 randomly selected data samples, but chose samples according to a specific share of speaker gender or age. To check generalization across genders, a set percentage of samples (0%, 30%, 70%, or 100%) were chosen from female speakers with the remainder from male speakers. For age demographics, we kept an even share of samples by speaker gender, but in one experiment group we selected samples only from speakers over 40 years-old and in another only from speakers younger than 30.
- **How does the choice of base model being fine-tuned affect performance?** To examine the utility of transfer learning for phonetic transcription, we selected a different base model, Lee (2025a), which is a publicly available version of wav2vec 2.0 that has already been fine-tuned on the TIMIT dataset. Similar to other settings, we fine-tuned this model five times on 4000 randomly selected samples from the Buckeye training split, balancing for speaker gender. In this case, the model's symbol inventory was updated to match the phone symbols present in Buckeye.

Before fine-tuning the models in each experiment group, we performed basic hyperparameter grid search to select a reasonable learning rate and batch size based on validation set performance before proceeding. Models within each experiment group use the same hyperparameters, so variation in performance is attributed to variation in training sample selection and randomization of the samples' order during training, similar to cross validation. All experiments used 10 training epochs with 500 warm-up steps and a fixed mask time length of 4, reduced from 10 in Baevski et al. (2020), as our utterance samples can be short, and the mask length must be smaller than the duration of the audio samples to avoid errors during training. Our fine-tuning and evaluation is based on the code released by Taguchi et al. (2023), although we have made significant modifications to data pre- and post-processing. It is publicly available with more details on grid search parameter choices at `https://github.com/UMassCDS/wav2ipa`.

Models were trained with compute resources from the Unity High Performance Cluster,[4] where we requested one compute node with at least 40GB of available VRAM and were typically allocated a NVIDIA A40 or Ada Lovelace L40S GPU. With 10 epochs, fine-tuning usually completed in less than 3 hours, or 5 hours when using the entire Buckeye train split. By modifying gradient accumulation and batch size settings, it is possible to fine-tune models on a GPU with less VRAM or a CPU, with some trade-offs in performance or longer train time.

**3.3** *Comparison models and symbol normalization* To contextualize our models within the current landscape of automatic phonetic transcription, we benchmarked the following publicly available models on the test splits of both Buckeye and TIMIT corpora:[5]

- Allosaurus[6] fine-tuned for English phoneme recognition (Li et al., 2020)

[4] `https://unity.rc.umass.edu`

[5] We were unable despite some effort and access to computational resources to implement ZIPA (Zhu et al., 2025).

[6] `https://github.com/xinjli/allosaurus`

- facebook/wav2vec2-lv-60-espeak-cv-ft[7] based on Wav2Vec2-Large-LV60[8], a wav2vec 2.0 model pre-trained on English data from the LibriVox corpus, then further fine-tuned to multi-lingual phoneme recognition in conjunction with using Espeak[9] G2P on training data (Baevski et al., 2020; Conneau et al., 2021). We also evaluated facebook/wav2vec2-xlsr-53-espeak-cv-ft,[10] which was pre-trained on multilingual data, but have excluded results here, as the English pre-trained version performed better on Buckeye and TIMIT.

- A pipeline using Whisper English-Medium[11] (Radford et al., 2023) to predict orthographic transcriptions, followed by Epitran[12] G2P post-processing (Mortensen et al., 2018). We also tested the Turbo and English-Large variants of Whisper, but they are excluded for brevity, as English-Medium performed best in all instances.

- Multipa, XLSR-53 fine-tuned for phonetic transcription on 7000 total samples from 7 languages in the Common Voice corpus, with training transcriptions obtained using Epitran (Taguchi et al., 2023)

- Wav2Vec2-Large-LV60 fine-tuned on the train split of the TIMIT corpus. There are two versions of this model, one that is trained using the original TIMIT transcriptions (Lee, 2025a), and one trained on transcriptions that use a "simplified" version of the TIMIT phone inventory (Lee, 2025b), as described in Lee & Hon (1989) which we discuss further in the next paragraph. These models output transcriptions in the TIMIT ARPABET, which we convert to IPA using the phonecodes library (Hasegawa-Johnson, 2019); we deal with closure and release as discussed in the next paragraph.

Since each model and its training data use different phone symbol inventories, we further normalized symbols to a shared symbol set for fair comparison of phone error rates. First, phone tokenization, including grouping symbols in diphthongs, and Unicode symbol normalization are completed using the ipatok library[13] (Sofroniev & Martinović, 2024), which merges [g] and [g], for example. Next, we defined a mapping to a reduced symbol set for each model and test corpus. We started by creating a shared symbol set for Buckeye and TIMIT. The main challenge here is that TIMIT transcribes obstruent closure and release as separate symbols, with, for example, [TCL] representing the closure of [t], and [T] its release, so that [TCLT] is a released [t], [TCL] is unreleased [t], and [BCLT] is unreleased [b] followed by [t] release (as in "obtain"). We decided to reduce all combinations of closure and release for the same phone to a single phone: [T], [TCL], [TCLT] and [TCL T] all become [t]. It is possible that in so doing we lost some information about consonant length or release, but our impression is that these were not transcribed consistently. The standard approach in prior literature is to map all closures to silence (Lee & Hon, 1989; Baevski et al., 2020). This would result in losing distinctions between unreleased consonants, which we judged to be a worse loss than for our approach.[14] Other reductions included the elimination of TIMIT's voiceless vowels and Buckeye's nasalized ones by mapping to the corresponding plain vowels, and mapping TIMIT [ʉ] to [u] and [ɨ] to [ɪ].[15] We mapped [ʌ] to [ə] (see 3.1 above), and the vocalic-R symbols ([ɝ], [ɚ]) were mapped to syllabic [ɻ̩]. Other models had the same mappings applied, along with other symbol normalizations (e.g. [g] and [g] to [g]) and length diacritic removals. The full set of reductions can be found in the supplemental materials. Although this does not exhaustively cover the symbol inventories for all models – Multipa in particular has a large one with many symbols not typically used for English – this manual remapping sets realistic expectations for practical off-the-shelf performance, where investing significant effort in post-processing is undesirable.

[7] https://huggingface.co/facebook/wav2vec2-lv-60-espeak-cv-ft
[8] https://huggingface.co/facebook/wav2vec2-large-lv60
[9] https://github.com/espeak-ng/espeak-ng
[10] https://huggingface.co/facebook/wav2vec2-xlsr-53-espeak-cv-ft
[11] https://huggingface.co/openai/whisper-medium
[12] https://github.com/dmort27/epitran
[13] https://github.com/pavelsof/ipatok
[14] The standard TIMIT reduction (Lee & Hon, 1989) also merges a number of other distinctions, some allophonic, but also some phonemes like [ɑ] and [ɔ], which seem to be well distinguished acoustically in the TIMIT (and Buckeye) data, and are reasonably accurately transcribed by our model
[15] We tried mapping [ɨ] to [ə], but that gave worse results.

This reduction did not lead to much, if any loss of information for Buckeye itself, since the distinctions that were lost within it were infrequent and not reliably transcribed. The reduced symbol set might in fact be better than the full Buckeye set for applications to new data.

**3.4** *Evaluation* To compare models, we follow Taguchi et al. (2023) by calculating phone error rate (PER) and phone feature error rate (PFER), both implemented in the PanPhon Python library[16] (Mortensen et al., 2016). PER is computed by tokenizing Unicode strings into phones, calculating the minimum edit distance between the prediction and the reference phones, then normalizing by the number of the reference phones. Substitution, deletion and insertion errors are given an equal penalty of 1. A minimum PER of 0 is achieved when the predicted transcription perfectly matches the reference, and it can be greater than 1 if the prediction is significantly longer than the reference. PFER is the Hamming Edit Distance between vectors of articulatory features for each pair of prediction and reference phones. In PFER feature substitutions are penalized with a $\frac{1}{24}$ weight, as PanPhon supports 24 articulatory features, while insertions and deletions of an entire phone receive a penalty of 1. PFER is not normalized by length of the reference, but does satisfy the triangle equality, constituting a metric space. We average PER and PFER across utterance samples in the test set as high-level aggregate metrics that can be used to compare models at a glance. For ease of use, we provide a HuggingFace-friendly evaluation tool as a wrapper around the PanPhon implementation at `https://huggingface.co/spaces/ginic/phone_errors`.

## 4 Results and Discussion

**4.1** *Comparison between Wav2IPA experiment groups on the Buckeye test split* Overall, we were able to fine-tune models that achieved good results on the Buckeye test split for nearly every experimental setting described in 3.2, although there are some caveats which reveal aspects of training data selection that are important for effective fine-tuning. Figure 1 presents the average PER on Buckeye test samples for all models in each experiment group compared to average test set PER for the single best model in that experiment group. This figure shows that the majority of models we fine-tuned on the Buckeye training split achieve a PER better than 0.3, which we believe meets the bar for practical usage in many settings.

Most strikingly, we only needed a relatively small portion of our training data to achieve good fine-tuning performance. Our models were better than publicly available off-the-shelf alternatives starting with just 1600 samples for fine-tuning. To be more explicit, Figure 2 plots the duration of fine-tuning data seen by each model compared with its average PER on the Buckeye test set. Models trained on 800 samples or less are completely unusable, with average PER values often exceeding 1. Those models tend to output long transcriptions that repeat the same few characters over and over. With 1600 samples for fine-tuning, an average of about 73 minutes of transcribed data, models have an average PER of 0.4376. The average PER continues to improve with more data – models with 3200 samples see about 147 minutes of data and have an average PER of 0.3375. After a point there are diminishing returns for increasing the amount of training data, and performance levels off at about 0.26 PER with 6400 training samples.

Training data demographics and the choice of foundation model may have some effect on model performance, though the evidence is not particularly strong. From Figure 1, we see two experiment groups where the best model's average PER is much better than the average PER across its experiment group: 1) Lee (2025a) fine-tuned on 4k Buckeye samples and 2) fine-tuning on 4k samples from only male speakers. Unlike our other experiment groups, results after fine-tuning are inconsistent in these cases. It's possible we did not select good hyperparameters for these experiments, or there could be something unique about these experimental settings that makes them particularly challenging. We opted not to use k-fold cross validation due to how compute intensive it is, but it likely would have been beneficial in these instances. In other experiment groups, performance variation within the experiment group is low. Biases toward particular demographic groups in the training data do not appear to impact model performance at the aggregate level, but an investigation of performance between groups in the test set may reveal differences that we have thus far missed.

**4.2** *Comparison with other models on the Buckeye test split* The following table breaks down the average PER and PFER of various models on the Buckeye test samples. Performance without phone

[16] `https://pypi.org/project/panphon/`

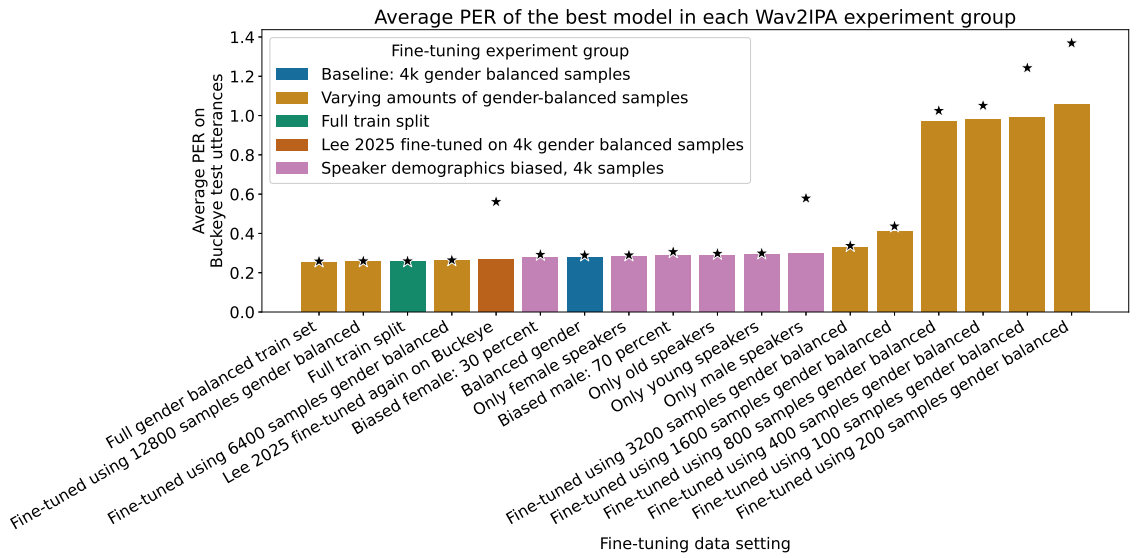


**Figure 1:** The bars in this plot show the average PER of the *best model* in each experiment group across utterance samples in the Buckeye test set. The stars indicate the PER averaged across all utterance sample predictions for *all five models* trained for each experiment group to demonstrate the performance variation within each experiment group. These results were computed without mapping to the shared symbol set.

remapping means that only Unicode normalization is applied to transcriptions before computing metrics, whereas performance with phone remapping means that both the test data transcriptions and the models' predictions undergo the appropriate reduction to the shared TIMIT and Buckeye symbol set described in 3.3. The first rows in the table show the performance of the best Wav2IPA model for a selection of the experiments described in 3.2. These are followed by results for the models described in section 3.3. Top PER and PFER values are marked in bold.

| | **Without phone remapping** | | **With phone remapping** | |
|---|---|---|---|---|
| | PER | PFER | PER | PFER |
| Wav2IPA fine-tuned on full Buckeye train split | 0.2563 | 3.90 | 0.2479 | 3.83 |
| Wav2IPA fine-tuned on full gender-balanced train split | **0.2527** | 3.87 | **0.2447** | 3.80 |
| Wav2IPA fine-tuned on 4k gender-balanced samples | 0.2798 | 4.26 | 0.2715 | 4.18 |
| Lee 2025a fine-tuned again on 4k Buckeye samples | 0.2672 | **3.86** | 0.2588 | **3.78** |
| Lee 2025a: Wav2Vec2-Large-LV60, TIMIT | 0.5306 | 5.71 | 0.4641 | 5.47 |
| Lee 2025b: Wav2Vec2-Large-LV60, TIMIT (simplified) | 0.4900 | 5.67 | 0.4501 | 5.43 |
| Whisper English Medium + Epitran | 0.5533 | 7.35 | 0.5232 | 7.24 |
| facebook/wav2vec2-lv-60-espeak-cv-ft | 0.6124 | 6.12 | 0.5531 | 5.92 |
| Allosaurus English | 0.6604 | 7.61 | 0.6360 | 7.50 |
| Multipa | 0.7903 | 8.20 | 0.7583 | 8.06 |

Our Wav2IPA model fine-tuned on the largest possible gender-balanced subset of the Buckeye train split achieved the best PER, but all the models which were fine-tuned on Buckeye data perform similarly, with PER of less than 0.3 and a PFER of less than 4.3. The next best third-party models are those of Lee 2025, fine-tuned only on TIMIT data, but they never achieve a PER lower than 0.4, or a PFER lower than 5. Among models that utilize a G2P component, the Whisper English Medium model followed by Epitran G2P has the best PER. Despite being created for orthographic transcription, this Whisper model combined with Epitran performs better in terms of PER and PFER than both Allosaurus and Multipa models, which are specifically designed for phone recognition, but use G2P for preprocessing training data. However, facebook/wav2vec2-

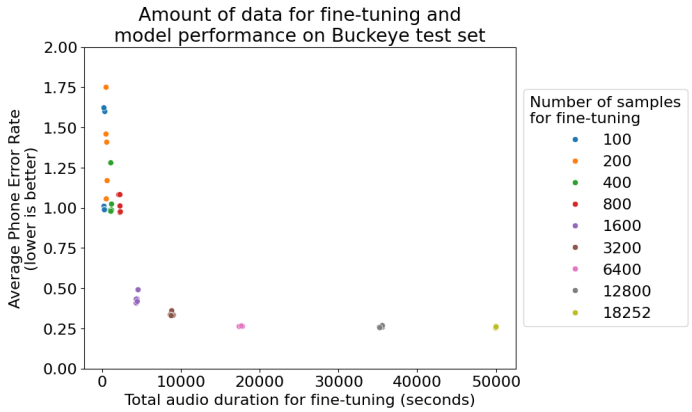


**Figure 2:** Average Phone Error Rate and duration of training data for each model in experiment groups that vary the amount of training data used for fine-tuning.

lv-60-espeak-cv-ft does have the best PFER of the G2P-reliant approaches, meaning it's doing a better job at capturing sub-phone articulatory features. This difference is likely accounted for by differences between Espeak G2P, used by the Facebook model, and Epitran, which is used by the other G2P models. The Multipa model in particular seems to struggle with the Buckeye data, which may be a reflection of the lack of English data in its fine-tuning corpus and its large phone inventory.

Remapping phones from the test data and model output to a shared symbol set improves both PER and PFER for all models and has a bigger impact for the models which did not see Buckeye data during training. For the models fine-tuned on Buckeye, remapping phones leads to just an 0.07-0.08 improvement in PFER. This is likely due to the low frequency and inconsistent use in human transcription of the symbols removed in the remapping (e.g. nasalized vowels). Other models benefited from phone remapping by an improvement in PFER ranging from 0.11 (Allosuarus, Whisper pipeline) to 0.2 (facebook/wav2vec2-lv-60-espeak-cv-ft). Despite this, the relative performance of models on the Buckeye test split remained the same. Phone remapping is a naive, but effective, way of dealing with differences in symbol inventories between a model and the target corpus, making it an appropriate technique when using an off-the-shelf model for phonetic transcription. However, it does not completely close the performance advantage afforded to models fine-tuned on the target corpus.

**4.3** *Comparison with other models on the TIMIT test split* Similar trends for model performance hold for the TIMIT corpus (with the closure and release treatment discussed in 3.3). As with Buckeye data, models that are fine-tuned on data from the training split of the same corpus perform best here as well. Both Lee 2025 models have better PER and PFER scores on the TIMIT test data than our Wav2IPA models. Yet the Wav2IPA models are noticeably better on TIMIT than any of the off-the-shelf models with G2P components, especially when phones are mapped to the shared symbol inventory. The fine-tuned wav2vec 2.0 models also have strong performance across language varieties in TIMIT, despite Wav2IPA being trained only on the North Midland variety in Buckeye. As seen in Figure 3, the Wav2IPA models' average PER by dialect group is between 0.14 and 0.18 (with phone remapping). Fine-tuning wav2vec 2.0 models on high quality phonetically transcribed data yields benefits, even when those models are applied to other corpora or dialects.

| | **Without phone remapping** | | **With phone remapping** | |
|---|---|---|---|---|
| | PER | PFER | PER | PFER |
| Wav2IPA fine-tuned on full Buckeye train split | 0.2566 | 2.66 | 0.1590 | 2.30 |
| Wav2IPA fine-tune on full gender-balanced train split | 0.2589 | 2.66 | 0.1599 | 2.30 |
| Wav2IPA fine-tuned on 4k gender-balanced samples | 0.2631 | 2.80 | 0.1671 | 2.44 |
| Lee 2025a fine-tuned again on 4k Buckeye samples | 0.2383 | 2.44 | 0.1412 | 2.08 |
| Lee 2025a: Wav2Vec2-Large-LV60, TIMIT | **0.1300** | **2.22** | **0.0794** | **1.12** |
| Lee 2025b: Wav2Vec2-Large-LV60, TIMIT (simplified) | 0.1847 | 2.40 | 0.0830 | 1.13 |
| Whisper English Medium + Epitran | 0.3532 | 3.71 | 0.2885 | 3.41 |
| facebook/wav2vec2-lv-60-espeak-cv-ft | 0.3177 | 3.25 | 0.2390 | 3.08 |
| Allosaurus English | 0.3689 | 3.79 | 0.3035 | 3.50 |
| Multipa | 0.6288 | 6.11 | 0.5904 | 5.99 |

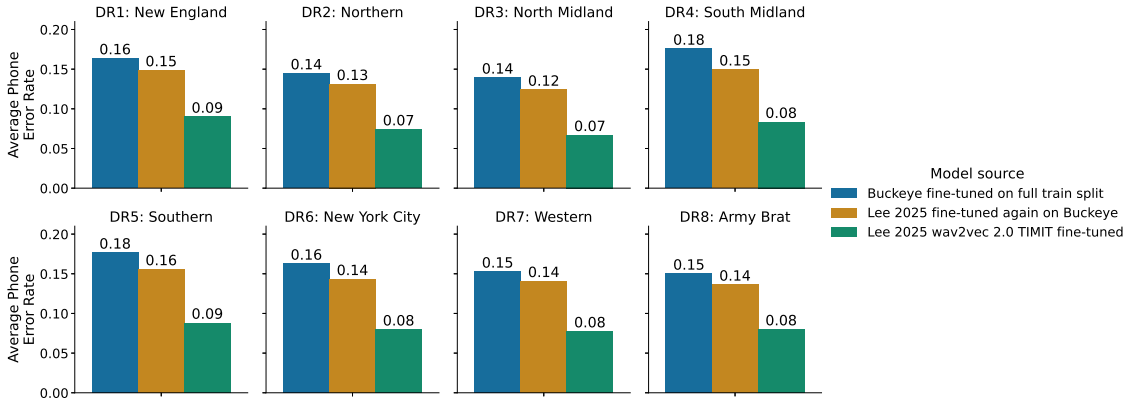


**Figure 3:** Average PER with phone remapping on the TIMIT test split broken down by dialect region.

It is also worth noting that with phone remapping, Wav2IPA performs better on the TIMIT test set than on our Buckeye test set. This is likely because the speech from sentence reading in TIMIT is easier to transcribe than the Buckeye conversational speech, for both computers and humans. Further research with other models should reveal whether the 0.25 PER that we have achieved on Buckeye is a performance ceiling that results from properties of the data. If this is the case, one way of moving the state of the art ahead might be to improve the Buckeye data by using models to identify potential instances of human error for retranscription.

## 5 Example cross-dialect transcriptions and next steps

In this section we provide some examples of Wav2IPA being applied to varieties of English, and discuss potential directions for further research. Our first set of examples comes from TIMIT, with three speakers' pronunciation of "She had your dark suit in greasy wash water all year". We chose speakers who had regionally distinctive speech. The first is noted in TIMIT's SPKRTINFO.txt file as having the "best New England accent so far", while the second "has [a] good NY pronunciation of 'saw'". The third was chosen to represent the Buckeye North Midland region. For each one, the speaker ID, sex, and birthdate are provided in the first line, followed by the human TIMIT and automated Wav2IPA transcriptions (spaces between words inserted for readability).

(1) New England, VMH0, F, b. 1960
TIMIT: [ʃi fiæd jɨ dɑk sʉt ɳ ɡɹisi wɑʃ wɑɾə ʔɔl jɪə]
Wav2IPA: [ʃi hæd jɪ dɑk suʔ ɳ ɡɹisi wɑʃ wɑɾʌ oʊ jɛ]

(2) New York, HXS0, F, b. 1941
TIMIT: [ʃi fiæd jʊ dɑk sʉt ʔɨn ɡɹisi wɑʃ wɔɾʌ ʔɔl jɪɚ]
Wav2IPA: [ʃi hæd juʌ dɔk sut ɪn ɡɹisi wɑʃ wɔɾɻ oʊl jiɹ]

(3) North Midland, DSS1, M, b. 1955

TIMIT: [ʃi fiæd jəˠ dɑɪk sʉr ɪn gɹisi wɔɪʃ wɔrəˠ ɔl jɪəˠ]

Wav2IPA: [ʃi hæd jɻ dɑɪk sur ɛn gɹeɪsi wɔɹʃ fʌɑrɻ ɔl jɪɹ]

Wav2IPA is able to capture some dialectal variations seen here. The New England and New York speaker are both mostly non-rhotic, while the North Midland speaker is rhotic; the difference can be most clearly seen in the transcriptions of "dark". In addition, the New York speaker has a LOT/THOUGHT contrast, seen in "wash water", which the New England speaker lacks, as illustrated by the identical stressed vowels in that phrase. And the third speaker's Midland "warsh" is also accurately transcribed. These examples also serve to illustrate the general degree of fidelity of the automated transcriptions to the human ones – close, but not identical. Some of the mismatch is due to limitations in the Buckeye symbol inventory (e.g. it has no [ɦ], [ɨ], [ʉ], or [ə]), but some is due to the human being more faithful to the speech (e.g. North Midland "water"), and some is due to Wav2IPA being more accurate (the glottal stop in New England "suit"). All of the Buckeye and TIMIT test set transcriptions can be found at our GitHub site (`https://github.com/ginic/wav2ipa/tree/main/data/evaluation_results/detailed_predictions/`). The corresponding audio can be obtained from the maintainers of the Buckeye and TIMIT corpora.

The transcription in (4) further illustrates the ability of Wav2IPA to represent some dialectal variations, and also highlights some challenges. This speaker is also mostly non-rhotic. There is no [ɹ] in the transcriptions of "over", "car", and "where", though there is for "door" (another version of Wav2IPA transcribed no [ɹ] in all four). Two human judges heard no "r" in the first three words and disagreed on "door", with just one judging there to be no "r".

(4) Warren "Bun" Doubleday, b. 1910, Dana Massachusetts (recording courtesy UMass Amherst Archives).

Orthography: over the side of the car where the door should have been

Wav2IPA: [oʊvʌ ðʌ saɪd dʌ ðʌ kɑ wɛ ðʌ doʊɹ ʃʊd ɛv bɛn]

However, his pronunciation of the vowel in "side" and the one in "car" are not well represented by the Buckeye symbol set; these would be better transcribed as a higher [əɪ] and fronter [a] respectively. Both of these are socially/regionally distinctive pronunciations of the vowels in New England (Roberts, 2007; Stanford, 2020), and for many purposes the current Wav2IPA transcription would be inadequate.

To broaden the coverage of Wav2IPA, we plan to fine-tune it on other varieties, including the variety of New England English spoken by Doubleday and other former residents of the lost towns of the Quabbin Valley. We plan an iterative process in working with untranscribed data, getting initial Wav2IPA transcriptions and then correcting them to serve as data for further fine-tuning. Interesting challenges, parallel to those seen in a multi-lingual context, will arise when we want to use a single model for multiple varieties, as we will want to in a dialect contact situation, as existed in the Quabbin. For example, once a model has been given the option of a third low vowel [a] between front [æ] and back [ɑ], would we want it to be used for all varieties?

The feasability and utility of dialect study using automated transcription with a fine-tuned wav2vec 2.0 model has already been demonstrated by Lee et al. (2025), who studied the realization of consonants across Korean dialects. Consonant realization is also the target of the study by Tanner et al. (2025), who use wav2vec 2.0 to classify stops for the presence of burst in English and Japanese. Future work on segment classification, especially on English, might benefit from adopting Wav2IPA as a foundation model for further fine-tuning. Finally, a different direction for further research is to better understand how Wav2IPA is choosing its symbolic representations of the speech signal; see de Heer Kloots & Zuidema (2024) for related work.

## 6 Accessibility to models and computation

Access to compute resources can be a tremendous barrier to fine-tuning foundation models, and even publicly available fine-tuned models can be challenging to use off-the-shelf due to complicated dependencies and installation processes. Python is currently the language of choice for working with transformer models, many of which are shared freely on the HuggingFace model hub. Yet, of the models we tested, only Allosaurus and HuggingFace-compatible models from Taguchi et al. (2023) and Lee (2025a,b) were available as standard Python packages without additional external software dependencies. The XLSR-53 phonemic transcription models released by Facebook depend on Espeak, and Epitran depends on Flite. Both these standalone programs must be installed individually with operating system specific instructions, which can pose a challenge to cross-platform reproducibility.

To enable more user-friendly access to HuggingFace-compatible transcription models, including the ones

we've trained on the Buckeye Corpus, we have released an open-source Python package at https://pypi.org/project/autoipaalign/, which lets the user produce transcriptions with Praat TextGrid input and output using the Python library or a command line interface. An interactive, web-based version is available at https://huggingface.co/spaces/ginic/wav2ipa, where audio to transcribe can be recorded or uploaded. Furthermore, all models and evaluation code produced during this work are collected publicly for reuse at https://huggingface.co/collections/ginic/wav2ipa.

## Acknowledgments

We gratefully acknowledge the support of the Center for Data Science and Artificial Intelligence, the HFA/CICS Collaborative Seed Fund, and the Public Interest Technology Initiative, all at UMass Amherst, and NSF grant BCS-2140826 to UMass Amherst. The computational resources for this work are provided by the Unity Research Computing Platform, a multi-institutional cluster lead by UMass Amherst, the University of Rhode Island, and UMass Dartmouth. We thank AMP 2025 participants, UMass Amherst PIT fellows, the UMass Amherst Sound Workshop, and Chihiro Taguchi for discussion.

## References

Baevski, Alexei, Yuhao Zhou, Abdelrahman Mohamed & Michael Auli (2020). wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations. *Advances in Neural Information Processing Systems*, vol. 33, 12449–12460, URL https://proceedings.neurips.cc/paper/2020/file/92d1e1eb1cd6f9fba3227870bb6d7f07-Paper.pdf.

Boersma, Paul & David Weenink (2026). Praat: doing phonetics by computer [Computer program]. URL https://www.praat.org.

Conneau, Alexis, Alexei Baevski, Ronan Collobert, Abdelrahman Mohamed & Michael Auli (2021). Unsupervised Cross-Lingual Representation Learning for Speech Recognition. *Interspeech 2021*, ISCA, 2426–2430, URL https://www.isca-archive.org/interspeech_2021/conneau21_interspeech.html.

Fosler-Lussier, Eric, Laura Dilley, Na'im Tyson & Mark Pitt (2007). The Buckeye Corpus of Speech: Updates and Enhancements. *Interspeech 2007*, ISCA, 934–937, URL https://www.isca-archive.org/interspeech_2007/foslerlussier07_interspeech.html.

Garofolo, John S., Lori F. Lamel, William M. Fisher, David S. Pallett, Nancy L. Dahlgren, Victor Zue & Jonathan G. Fiscus (1993). TIMIT Acoustic-Phonetic Continuous Speech Corpus. URL https://catalog.ldc.upenn.edu/LDC93S1.

Graves, Alex, Santiago Fernández, Faustino J. Gomez & Jürgen Schmidhuber (2006). Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. *ICML 2026*, vol. 148, 369–376, URL https://doi.org/10.1145/1143844.1143891.

Hasegawa-Johnson, Mark (2019). Phonecodes. URL https://github.com/jhasegaw/phonecodes.

de Heer Kloots, Marianne & Willem Zuidema (2024). Human-like linguistic biases in neural speech models: Phonetic categorization and phonotactic constraints in wav2vec2.0. *Interspeech 2024*, ISCA, p. 4593–4597, URL http://dx.doi.org/10.21437/Interspeech.2024-2490.

Kiesling, Scott, Laura Dilley & William Raymond (2006). The Variation in Conversation (ViC) Project: Creation of the Buckeye Corpus of Conversational Speech. URL https://buckeyecorpus.osu.edu/BuckeyeCorpusmanual.pdf.

Kreuk, Felix, Yaniv Sheena, Joseph Keshet & Yossi Adi (2020). Phoneme Boundary Detection using Learnable Segmental Features. URL https://arxiv.org/abs/2002.04992. _eprint: 2002.04992.

Lee, Jooyoung (2025a). Wav2Vec2 Large LV-60 English-TIMIT phoneme recognition model. URL https://huggingface.co/excalibur12/wav2vec2-large-lv60_phoneme-timit_english_timit-4k.

Lee, Jooyoung (2025b). Wav2Vec2 Large LV-60 English-TIMIT simplified phoneme recognition model. URL https://huggingface.co/excalibur12/wav2vec2-large-lv60_phoneme-timit_english_timit-4k_simplified.

Lee, Jooyoung, Sunhee Kim & Minhwa Chung (2025). Analysis of Korean Dialect Obstruents in a Large Corpus Using Speech Recognition Technology. *Japanese/Korean Linguistics* 31:1, URL https://escholarship.org/uc/item/8s67h3rh.

Lee, K.-F. & H.-W. Hon (1989). Speaker-independent phone recognition using hidden Markov models. *IEEE Transactions on Acoustics, Speech, and Signal Processing* 37:11, 1641–1648.

Li, Xinjian, Siddharth Dalmia, Juncheng Li, Matthew Lee, Patrick Littell, Jiali Yao, Antonios Anastasopoulos, David R. Mortensen, Graham Neubig, Alan W. Black & Florian Metze (2020). Universal Phone Recognition with a Multilingual Allophone System. *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, arXiv, 8248–8253, URL http://arxiv.org/abs/2002.11800. ArXiv:2002.11800 [cs].

Lindsay, Geoff (2022). STRUT ʌ, schwa ə and American English. URL https://www.englishspeechservices.com/blog/strut-%CA%8C-schwa-%C9%99-and-american-english/.

Mortensen, David R., Patrick Littell, Akash Bharadwaj, Kartik Goyal, Chris Dyer & Lori S. Levin (2016). PanPhon: A Resource for Mapping IPA Segments to Articulatory Feature Vectors. *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, ACL, 3475–3484.

Mortensen, David R., Siddharth Dalmia & Patrick Littell (2018). Epitran: Precision G2P for Many Languages. *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, European Language Resources Association (ELRA), Miyazaki, Japan, URL https://aclanthology.org/L18-1429/.

Pitt, Mark A., Keith Johnson, Elizabeth Hume, Scott Kiesling & William Raymond (2005). The Buckeye corpus of conversational speech: labeling conventions and a test of transcriber reliability. *Speech Communication* 45:1, 89–95, URL https://linkinghub.elsevier.com/retrieve/pii/S0167639304000974.

Radford, Alec, Jong Wook Kim, Tao Xu, Greg Brockman, Christine Mcleavey & Ilya Sutskever (2023). Robust speech recognition via large-scale weak supervision. *Proceedings of the 40th International Conference on Machine Learning*, PMLR, vol. 202 of *Proceedings of Machine Learning Research*, 28492–28518, URL https://proceedings.mlr.press/v202/radford23a.html.

Roberts, Julie (2007). Vermont lowering? Raising some questions about /ai/ and /au/ south of the Canadian border. *Language Variation and Change* 19:2, p. 181–197.

Seyfarth, Scott & Marc Garellek (2020). Physical and phonological causes of coda /t/ glottalization in the mainstream American English of central Ohio. *Laboratory Phonology: Journal of the Association for Laboratory Phonology* 11:1, p. 24, URL https://www.journal-labphon.org/article/10.5334/labphon.213/.

Sofroniev, Pavel & Viktor Martinović (2024). ipatok: A simple IPA tokeniser. URL https://github.com/pavelsof/ipatok/tree/master.

Stanford, James N. (2020). *New England English: large-scale acoustic sociophonetics and dialectology*. Oxford scholarship online, Oxford University Press, New York, NY.

Taguchi, Chihiro, Yusuke Sakai, Parisa Haghani & David Chiang (2023). Universal Automatic Phonetic Transcription into the International Phonetic Alphabet. *INTERSPEECH 2023*, ISCA, 2548–2552, URL https://www.isca-archive.org/interspeech_2023/taguchi23_interspeech.html.

Tanner, James, Morgan Sonderegger, Jane Stuart-Smith, Jeff Mielke & Tyler Kendall (2025). Automatic classification of stop realisation with wav2vec2.0. URL https://arxiv.org/abs/2505.23688. _eprint: 2505.23688.

Xu, Qiantong, Alexei Baevski & Michael Auli (2022). Simple and Effective Zero-shot Cross-lingual Phoneme Recognition. URL https://www.isca-archive.org/interspeech_2022/xu22b_interspeech.html.

Yao, Zengwei, Liyong Guo, Xiaoyu Yang, Wei Kang, Fangjun Kuang, Yifan Yang, Zengrui Jin, Long Lin & Daniel Povey (2024). Zipformer: A faster and better encoder for automatic speech recognition. *International Conference on Representation Learning*, vol. 2024, 44440–44455, URL https://proceedings.iclr.cc/paper_files/paper/2024/file/c1bb0e3b062f0a443f2cc8a4ec4bb30d-Paper-Conference.pdf.

Zhu, Jian, Cong Zhang & David Jurgens (2022). ByT5 model for massively multilingual grapheme-to-phoneme conversion. *Interspeech 2022*, ISCA, 446–450, URL https://www.isca-archive.org/interspeech_2022/zhu22_interspeech.html.

Zhu, Jian, Farhan Samir, Eleanor Chodroff & David R. Mortensen (2025). ZIPA: A family of efficient models for multilingual phone recognition. *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Association for Computational Linguistics, Vienna, Austria, 19568–19585, URL https://aclanthology.org/2025.acl-long.961/.