Graph Connectionist Temporal Classification for Phoneme Recognition

Henry Grafé Hugo Van hamme
Department of Electrical Engineering-ESAT, KU Leuven, Belgium
henry.grafe@kuleuven.be, hugo.vanhamme@kuleuven.be

***Abstract*—Automatic Phoneme Recognition (APR) systems are often trained using pseudo phoneme-level annotations generated from text through Grapheme-to-Phoneme (G2P) systems. These G2P systems frequently output multiple possible pronunciations per word, but the standard Connectionist Temporal Classification (CTC) loss cannot account for such ambiguity during training. In this work, we adapt Graph Temporal Classification (GTC) to the APR setting. GTC enables training from a graph of alternative phoneme sequences, allowing the model to consider multiple pronunciations per word as valid supervision. Our experiments on English and Dutch data sets show that incorporating multiple pronunciations per word into the training loss consistently improves phoneme error rates compared to a baseline trained with CTC. These results suggest that integrating pronunciation variation into the loss function is a promising strategy for training APR systems from noisy G2P-based supervision.**

***Index Terms*—Phoneme, Automatic Phoneme Recognition, Connectionist Temporal Classification**

I. Introduction

Automatic Phoneme/Phone Recognition (APR) is the task of transcribing the phonemes or phones that were pronounced in an utterance. Phoneme-level transcriptions are useful for multiple applications, including Automatic Speech Recognition (ASR) for low-resource languages [21], dialect analysis and dialectal speech recognition [3], Computer Assisted Language Learning [2], or speech pathology [4]. A long-standing problem to train models for APR is the lack of manually annotated data sets for supervised training. Manual annotation requires careful examination of the recordings by individuals having knowledge of phonology, which makes it very costly [1]. Another approach is to use tools such as the Montreal Forced Aligner (MFA) [5] to obtain automatic phoneme annotations in large quantities, which then can be used as pseudo-labels for training. However, these annotations contain errors, and the performance of a system trained on such pseudo-labels will be limited by the amount of label noise.

Another option to obtain phoneme-level annotation is the use of Grapheme-to-Phoneme (G2P) systems. G2Ps take as input words or sentences, and output the corresponding phonemes. Those phonemes can then be used as pseudo-labels to train an APR system, allowing to train on utterance-text pairs rather than manual phoneme annotations.

However, as they only use text, the G2Ps output phonemes that correspond to the accepted standard pronunciation of the language, with no guarantees that they correspond to the phonemes that were actually uttered. This makes the pseudo-labels a noisy source of supervision for APR systems. To mitigate this lack of pronunciation variation, G2Ps often propose multiple pronunciations per word, either by specifying the possible pronunciations for a word separately [6], or by designing a set of rewriting rules allowing to produce multiple pronunciations from a single entry [7]. Those multiple pronunciations per word increase the chances that one of them matches the pronunciation in the utterance paired with the text. However, we still do not know which proposed pronunciation is actually instantiated in the utterance. This limits the usefulness of having access to multiple such pronunciations for each word.

In this paper, we propose the use of a variant of the Connectionist Temporal Classification (CTC) [8] loss, named the Graph Temporal Classification (GTC) [9], to leverage the multiple pronunciations per word outputted by a G2P during the training of an APR system. Instead of relying on a single ground-truth label sequence, GTC allows any concatenation of the proposed pronunciations for each word to be treated as a valid target sequence. Similar techniques have been applied in the field of semi-supervised ASR, where the issue of multiple pseudo-label sequences per utterance is also present [9], [10], [13], [15], motivating us to apply similar techniques to APR.

Through experiments in English and Dutch, we show that the use of our proposed loss leads to an improvement in Phoneme Error Rate (PER) for both languages over a CTC baseline using only one pronunciation per word.

II. Related Work

To mitigate the scarcity of manually annotated data for automatic phoneme and phone recognition, several approaches have been proposed in the literature.

One line of work uses unsupervised or self-supervised learning to extract acoustic representations, which are then fine-tuned using limited labeled data. For instance, [18] fine-tuned a self-supervised wav2vec 2.0 model on TIMIT and achieved state-of-the-art phoneme recognition accuracy. Similarly, [19] trained a pretrained wav2vec 2.0 model for ASR in a fully unsupervised manner and, after a so-called self-training stage, achieved competitive performance in fully unsupervised phoneme recognition.

A second line of research leverages the output of Grapheme-to-Phoneme (G2P) systems to generate phonemic pseudo-

This research was supported by the Flemish Government under "Onderzoeksprogramma AI Vlaanderen", FWO-SBO grant S004923N and KU Leuven grant C24M/22025

labels that are then used to train phoneme or phone recognizers. Representative examples include [20] and [21], which pursue universal phone recognition by introducing differentiable phone-to-phoneme layers and training on phonemic transcriptions generated using Epitran [22]. These methods implicitly assume that the G2P system provides a single transcription per word, and that this transcription exactly matches the pronunciation in the corresponding utterance.

In practice, pronunciation dictionaries and G2P systems frequently provide multiple candidate pronunciations. For instance, CMUDict [6] lists several variants for many English words. The English G2Ps bundled with the Montreal Forced Aligner (MFA) likewise output multiple pronunciations for ambiguous words by default [5]. In Dutch, the Fonilex dictionary generates variants through a set of pronunciation rules [7]. Other studies have also focused on designing rules to generate pronunciation variants for English [23].

The present work is inspired by advances in semi-supervised automatic speech recognition (ASR). Semi-supervised ASR typically assumes one or more noisy word-level transcriptions per utterance, obtained through beam search or from independent seed models; analogously, we assume multiple noisy phonemic transcriptions per word. Several studies have adapted the Connectionist Temporal Classification (CTC) [8] objective to accommodate uncertainty: [10] allows for the insertion of additional tokens between words; [11] allows for the insertion of tokens at the start and end of an utterance; [13] introduces alternative arcs in the CTC graph at uncertain positions; [12] allows for substitutions and insertions; [14] allows for substitutions, insertions and deletions; [9] compiles multiple beam-search hypotheses into a confusion network for training with a modified version of CTC; and [15] jointly exploits pseudo-transcriptions from two models trained on different acoustic features through a modified CTC that accounts for two reference label sequences.

This work is also related to earlier research on ASR systems that leveraged multiple possible pronunciations during training. For example, the SPHINX system [16] attempted to incorporate multiple pronunciations in parallel during training using the Baum-Welch algorithm, but observed mixed results. Similarly, [17] incorporated multiple pronunciations during Lattice-Free Maximum Mutual Information (LF-MMI) training of a Hidden Markov Model-Deep Neural Network (HMM-DNN) system by embedding multiple pronunciations into the Finite State Transducer that encodes word pronunciations. However, unlike our work, their evaluation focused solely on ASR performance, and no experiments were conducted to assess the specific benefit of modeling multiple pronunciations for phoneme recognition.

III. Loss Definition

We assume that we have an input sequence of feature vectors X of length T processed by a neural network into an output sequence of posterior probabilities $Y = (\boldsymbol{y}^1, \ldots, \boldsymbol{y}^{T'})$ (where T might be different from T' due to subsampling) over the phoneme vocabulary as well as the special blank character ε. We can obtain the probability estimated by a model with parameters θ that a sequence of characters π of length T', containing either phoneme characters or blanks, is a valid alignment with the utterance X by computing:

$$P(\pi \mid X, \theta) = \prod_{t=1}^{T'} P(\pi_t \mid \pi_{(t-1):}, X, \theta) \tag{1}$$

$$\approx \prod_{t=1}^{T'} P(\pi_t \mid X, \theta) = \prod_{t=1}^{T'} \boldsymbol{y}^t_{\pi_t} \tag{2}$$

where π_t represents the t-th character of π, and $\boldsymbol{y}^t_{\pi_t}$ is the posterior probability assigned to character π_t by the model for the frame $\boldsymbol{y}^t$. The independence of π_t from the previous characters of the sequence $\pi_{(t-1):}$ assumed by the model is called the conditional independence assumption.

We also define a many-to-one reduction function $\beta(\pi)$ which takes a sequence of characters (including blanks), and collapses repeating consecutive characters into one, and then deletes the blank characters.

A. Connectionist Temporal Classification

The standard CTC loss between the sequence of posterior probabilities and a ground truth sequence $\hat{\boldsymbol{y}}$ is defined as:

$$\mathcal{L}_{CTC} = -\log P(\hat{\boldsymbol{y}} \mid X, \theta) = -\log \sum_{\pi \in \beta^{-1}(\hat{\boldsymbol{y}})} P(\pi \mid X, \theta) \tag{3}$$

where $\beta^{-1}(\hat{\boldsymbol{y}})$ represents the set of all sequences π that are such that $\beta(\pi) = \hat{\boldsymbol{y}}$. The computation of this sum is intractable if done naively, but can be performed in polynomial time using the forward-backward algorithm [9].

B. Graph Temporal Classification

We follow the definition of Graph Temporal Classification (GTC) of [9]. GTC is an extension of CTC, where there is not a unique ground truth sequence, but a set of acceptable ground truth sequences encoded in a graph $\mathcal{G}$. We define $\mathcal{G}$ as an arbitrary directed acyclic graph associated with a set of specified start and end nodes, and where each node has a label associated corresponding to one of the phonemes of the vocabulary. Any path within $\mathcal{G}$ which starts at one of the start nodes and ends at one of the end nodes specifies an acceptable ground truth sequence. We write $\boldsymbol{y} \in \mathcal{G}$ when the label sequence $\boldsymbol{y}$ is one of the label sequences encoded in $\mathcal{G}$. We then define the GTC loss as:

$$\mathcal{L}_{GTC} = -\log P(\mathcal{G} \mid X, \theta) = -\log \sum_{\pi \in \beta^{-1}(\mathcal{G})} P(\pi \mid X, \theta) \tag{4}$$

where we define $\beta^{-1}(\mathcal{G})$ as the set of sequences π such that $\beta(\pi) = \hat{\boldsymbol{y}}_k$, and $\hat{\boldsymbol{y}}_k \in \mathcal{G}$. As for standard CTC, the computation of the sum can be done in polynomial time with the forward-backward algorithm.

IV. Loss Implementation

Following prior work in semi-supervised ASR [13], we implement our version of GTC using Weighted Finite State Acceptors (WFSA) [24] in the k2 framework [25]. This implementation offers design flexibility and enables efficient training and inference on GPUs. The framework includes built-in support for forward-backward computations on arbitrary WFSAs.

A. Standard CTC Implementation

Standard CTC loss can be implemented by the intersection of two WFSAs, named the emission and label WFSAs [10].

The emission WFSA encodes the probability distributions over the vocabulary for each frame emitted by the models. It accepts any sequence of characters π and assigns a weight based on the log probabilities produced by the neural network for each frame.

The label WFSA encodes all character sequences π that collapse to the ground-truth label sequence under the collapse function $\beta(\pi)$. This WFSA has its weights assigned to 0, as its only role is to specify the set of acceptable sequences.

The intersection of the two yields a WFSA which only accepts the character sequences collapsing to the ground truth label sequence, and for each of those sequences outputs the log probability of that sequence. The log-sum-exp of the weights of all acceptable character sequences can then be computed using the forward-backward algorithm.

B. GTC Implementation

Our implementation of GTC consists of modifying the label WFSA of the standard CTC implementation, such that it does not only accept one ground truth sequence, but the set of all ground truth sequences encoded in the graph $\mathcal{G}$. To build this graph label WFSA, we use the procedure described in Algorithm 1. We assume that for each instance, we have a sequence of words $W = (w_1, \ldots, w_N)$, where each word w_i has an associated set of m_i possible pronunciations $\mathcal{S}_{w_i} = \{p_1, \ldots, p_{m_i}\}$. For every pronunciation of every word, we build a standard CTC label WFSA as done in other works [10], [13], then put the label WFSA of all the pronunciations of a given word in parallel. We then serialize the parallelized WFSAs of every word into the final label WFSA. The resulting WFSA will accept any sequence of phonemes that are the concatenation of any of the pronunciations of the words given by the G2P.

An example of the construction of such a label WFSA is given in Figure 1, for a sequence of two words, one with pronunciations "ab" and "ac" and the other with possible pronunciations "de" and "fe". For clarity, we did not include the arcs that allow skipping blanks when two consecutive characters are not the same.

Algorithm 1 Construction of the label graph WFSA

```
Label_WFSA ← ∅
for each word w_i from i = 1 to N do
    Word_WFSA ← ∅
    for each pronunciation p_j ∈ S_i do
        Pron ← Build_CTC_Graph(p_j)
        Word_WFSA ← Parallel(Word_WFSA, Pron)
    end for
    Label_WFSA ← Serial(Label_WFSA, Word_WFSA)
end for
return Label_WFSA
```

V. Experimental setup

A. Corpora

We train and evaluate models using the GTC loss for phoneme recognition in United States English and Belgian Dutch.

1) English Data: For US English, we use the Common Voice dataset (version 21.0) [26], where we selected all the utterances whose "accent" attribute contained "united states", which left us with a training set of 451 hours and 5,485 speaker ids, and a validation set of 1.8 hours and 420 speakers ids. For testing, we use the test part of the TIMIT dataset [27] which consists of 1.4 hours of utterances from 168 speakers that have been phonemically annotated. To compute the oracle LER (see Section VI-A), we use the train part of TIMIT, which consists of 3.9 hours from 462 speakers. As we do not use the TIMIT training set to train our models, we do not need to worry about data contamination. Therefore, we retain the SA sentences in the train and test sets to ensure more statistically significant results.

2) Dutch Data: For Belgian Dutch, we use the Belgian Dutch part of the Corpus Gesproken Nederlands (CGN) [28]. Around 10% of CGN has been phonemically annotated. We randomly select 1.3 hours from 32 speakers of the phonemically annotated data for validation, and 5 hours and 102 speakers for testing. For the training set, we take the remaining utterances of CGN that do not share speakers with the validation and test sets, which consists of 188 hours and 1,166 speakers.

B. G2P Resource

1) English: To generate multiple phonemic transcriptions per word, we use the G2P model developed by [29], trained on the CMUDict pronunciation dictionary [6]. By default, this model uses a threshold-based decoding strategy, where it dynamically prunes hypotheses and outputs all those whose scores are within a certain threshold of the best-scoring hypothesis. The model can also be configured to produce a fixed number of pronunciations, but doing so is computationally too expensive to apply to the entire vocabulary. On the other hand, the threshold-based approach can sometimes yield too few alternative pronunciations for words that occur frequently in the datasets.

To obtain a sufficient number of alternative pronunciations for the frequent words with a tractable computational budget, we follow the procedure below. First, we force the G2P model to output 5 pronunciations for the 2000 and 1000 most frequent words of Common Voice and TIMIT train, respectively. For the remaining words of Common Voice and TIMIT train, we use

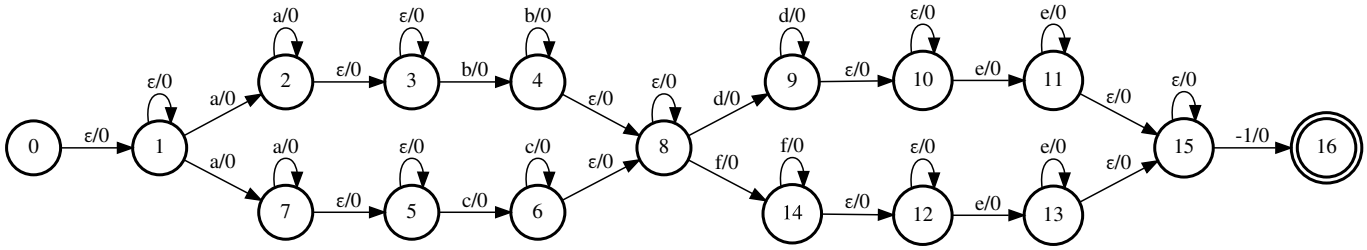


Fig. 1: Example of a label WFSA for a sequence of two words, one with possible pronunciations "ab" and "ac", and the other with possible pronunciations "de"and "fe". ϵ represents the special blank character. The arc with label -1 pointing to the final state of a design requirement for all k2 WFSAs. For ease of visualization, we omitted the arcs allowing to skip blanks when two successive characters are not the same.

the G2P in dynamic pruning mode, with a pruning threshold of 3.0. To select the top-n pronunciations, we rank the model's proposed outputs by their assigned scores.

2) Dutch: For Dutch, we use the rule-based pronunciation dictionary Fonilex [7]. Fonilex is a pronunciation dictionary containing a compact representation of Flemish pronunciations for 170,000 Dutch words, from each of which various pronunciation variants can be obtained by the application of pronunciation rewriting rules. The application of the pronunciation rules allows us to obtain high-spelling, normal-spelling, and low-spelling pronunciations for every word, which correspond to pronunciations used in careful, casual, and rapid informal speech, respectively. Words for which pronunciation rules could not encode all pronunciation variants have multiple entries in the database.

To select the top n pronunciations for each word in our experiments, we use the following heuristic, developed empirically. We begin by including the normal-spelling pronunciation provided by Fonilex. Next, we add the low-spelling and high-spelling variants. If a word has multiple entries in the Fonilex database, we select one of the entries at random (with the choice fixed across runs to ensure consistency). If more than three pronunciations are needed or some of the previously selected pronunciations are identical, we continue selecting from the remaining Fonilex entries in the same order: normal-spelling, then low-spelling, then high-spelling.

C. Automatic Phoneme Recognition System

For both languages, we train a conformer encoder [30] with embedding dimension 256, feedforward dimension 1,024, 4 attention heads, 8 layers and a dropout rate of 0.1. Before the encoder, we apply a convolution layer with a stride of 2 and a kernel size of 3. As input features to the model, we use Mel spectrogram and pitch features with standard ESPNet parameters.

We train our model on ESPNet [31], using dynamic batch sizing, with batches averaging 40 and 140 seconds of speech, and an epoch consisting of 11,000 and 16,000 steps for the Dutch and English model, respectively. We use the Adam optimizer [32] with a learning rate of 10^{-4} and weight decay 10^{-6}, and a learning rate schedule with a warmup of 10000 steps and power-law decay. We train until reaching the minimum loss value on the validation set, which typically requires 21 epochs.

VI. Results

A. Oracle Phoneme Label Error Rates and Upper Bounds on Performance

The possible pronunciations produced for each word by the G2Ps can be seen as pseudo labels. Similarly to what was done in [9], we can compute the phoneme Label Error Rate (LER) between our test set and the output of the G2P for each word, either by selecting one pronunciation or choosing the best one in an oracle fashion, i.e. computing the LER between the most appropriate pronunciation(s) in the dictionary and the ground truth, purposefully selecting those that minimize the error. Assuming that no system can achieve a better Phoneme Error Rate (PER) than the corresponding LER, the first quantity can be interpreted as an upper bound on the performance of our APR model if it was trained with standard CTC with only one of the G2P pronunciations per word. Similarly, the second quantity can be interpreted as an upper bound of the performance that we can expect to achieve with our APR system when training with GTC. That is, if the GTC loss enables the model to consistently select the most appropriate pronunciation, its performance would be bounded by the oracle LER. Additionally, the gap between the LER of a single pronunciation and the oracle LER is an indicator of the potential gains from using GTC to take advantage of multiple pronunciations.

Table I shows the LER on the TIMIT train and CGN-phoneme dev datasets for the English and Dutch G2Ps, respectively. The *1-best* column refers to selecting the most likely pronunciation based on a context heuristic, i.e. only by looking at the available pronunciations. The *2-best* and *3-best* column respectively refer to selecting the 2 and 3 most likely pronunciations (if there are any), with the LER then computed in an oracle fashion (we select the pronunciations that lead to the lowest LER). The *all* column refers to the LER when all the possible pronunciations outputted by the G2P are considered to compute the oracle LER with the data sets. We can observe for both languages that the LER decreases as we include more pronunciations, demonstrating that the

most appropriate pronunciation is often not the first one, and that taking into account the other proposed pronunciations is beneficial for the error rate.

TABLE I: Oracle Label Error Rates (LER) [%] for English and Dutch G2P outputs

language	data set	1-best	2-best	3-best	all
English	TIMIT train	11.8	9.3	8.5	8.0
Dutch	CGN-phoneme dev	15.6	12.5	11.9	11.6

B. Phoneme Error Rate of the Trained Models

Table II shows the performance of the models trained either with CTC using a single pronunciation proposed by the G2P per word (*1-best*), or with GTC on at most 2 (*2-best*), 3 (*3-best*), or all (*all*) of the pronunciations proposed by the G2P, for both languages.

For English, the best model is the one trained on a maximum of 2 pronunciations per word. However, the differences between the *2-best*, *3-best*, and *all* systems are not statistically significant based on a Wilcoxon signed-rank test of the null hypothesis ($p = 0.04$ between *2-best* and *3-best*, and $p = 0.11$ between *2-best* and *all*).

For Dutch, the best model is the one trained on a maximum of 3 pronunciations per word. The difference between the performance of the *1-best* (CTC) and *2-best* systems are significant when doing a Wilcoxon signed-rank test ($p = 10^{-5}$), but the difference between the performance of the *3-best* and *all* systems is not significant ($p = 0.21$).

TABLE II: Phoneme Error Rates (PER) [%] on English (TIMIT test) and Dutch (CGN-phoneme test) sets

numb. pronunciations	TIMIT test	CGN-phoneme test
1-best (CTC)	32.9	23.9
2-best	28.8	23.5
3-best	29.1	23.0
all	29.0	23.1

VII. Conclusion

In this work, we proposed a modified version of the Connectionist Temporal Classification (CTC) loss that leverages multiple pronunciations per word, as generated by grapheme-to-phoneme (G2P) models, for Automatic Phoneme Recognition. The proposed approach was evaluated on datasets in both English and Dutch. In both languages, our experiments show that incorporating multiple G2P-derived pronunciations improves performance over a baseline trained with standard CTC on only a single pronunciation per word.

Since our method relies on word-level outputs from G2P models, it does not capture coarticulation effects across word boundaries, which presents a promising direction for future research. Furthermore, the G2P models used do not generate exhaustive pronunciation variants, as this would lead to an unmanageable number of alternatives. One possible solution is to incorporate pronunciation variability directly into the decoding graph via insertion, substitution, and deletion arcs. Finally, applying our method to non-Germanic languages remains an open question and merits further investigation.

References

[1] C. Cucchiarini and H. Strik, "The nature of phonetic transcription," in *Manual of Clinical Phonetics*, M. J. Ball, Ed. London: Routledge, 2021, pp. 141–149. [Online]. Available: https://www.taylorfrancis.com/chapters/edit/10.4324/9780429320903-11/nature-phonetic-transcription-catia-cucchiarini-helmer-strik

[2] A. Neri, C. Cucchiarini, and H. Strik, "ASR-based corrective feedback on pronunciation: does it really work?," in *Proc. Interspeech 2006*, Pittsburgh, PA, USA, 2006, paper 1372-Wed3A3O.2. doi: 10.21437/Interspeech.2006-543.

[3] M. Lehr, K. Gorman, and I. Shafran, "Discriminative pronunciation modeling for dialectal speech recognition," in *Proc. Interspeech*, Singapore, 2014, pp. 1458–1462. doi: 10.21437/Interspeech.2014-356. [Online]. Available: https://www.isca-archive.org/interspeech_2014/lehr14_interspeech.html

[4] S. White, A. Hurren, S. James, and R.-A. Knight, "I think that's what I heard? I'm not sure: Speech and language therapists' views of, and practices in, phonetic transcription," *Int. J. Lang. Commun. Disord.*, vol. 57, no. 5, pp. 1071–1084, 2022. [Online]. Available: https://doi.org/10.1111/1460-6984.12740

[5] M. McAuliffe, M. Socolof, S. Mihuc, M. Wagner, and M. Sonderegger, "Montreal Forced Aligner: Trainable Text-Speech Alignment Using Kaldi," in *Proc. Interspeech*, Stockholm, Sweden, Aug. 2017, pp. 498–502. [Online]. Available: https://montrealcorpustools.github.io/Montreal-Forced-Aligner/images/MFA_paper_Interspeech2017.pdf

[6] R. L. Weide, "The CMU Pronouncing Dictionary," Carnegie Mellon University, 1993. [Online]. Available: http://www.speech.cs.cmu.edu/cgi-bin/cmudict

[7] P. Mertens and F. Vercammen, *FONILEX Manual*, version 1.0b, March 1998. [Online]. Available: https://lirias.kuleuven.be/bitstream/handle/123456789/389231/Mertens_1997_Fonilex_manual.pdf?sequence=1

[8] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proc. 23rd Int. Conf. Mach. Learn. (ICML)*, Pittsburgh, PA, USA, 2006, pp. 369–376. [Online]. Available: https://www.cs.toronto.edu/~graves/icml_2006.pdf

[9] N. Moritz, T. Hori, and J. Le Roux, "Semi-supervised speech recognition via graph-based temporal classification," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, Toronto, ON, Canada, June 2021, pp. 6548–6552.

[10] V. Pratap, A. Hannun, G. Synnaeve, and R. Collobert, "Star Temporal Classification: Sequence Modeling with Partially Labeled Data," in *Advances in Neural Information Processing Systems 35 (NeurIPS 2022)*, 2022, pp. 13392–13403. [Online]. Available: https://papers.neurips.cc/paper_files/paper/2022/file/57587d8d6a7ede0e5302fc22d0878c53-Paper-Conference.pdf

[11] X. Cai, J. Yuan, Y. Bian, G. Xun, J. Huang, and K. Church, "W-CTC: a Connectionist Temporal Classification Loss with Wild Cards," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2022. [Online]. Available: https://openreview.net/forum?id=0RqDp8FCW5Z

[12] D. Gao, M. Wiesner, H. Xu, L. P. Garcia, D. Povey, and S. Khudanpur, "Bypass Temporal Classification: Weakly Supervised Automatic Speech Recognition with Imperfect Transcripts," in *Proc. INTERSPEECH 2023*, Incheon, South Korea, Aug. 2023, pp. 924–928. doi: 10.21437/Interspeech.2023-2258.

[13] H. Zhu, D. Gao, G. Cheng, D. Povey, P. Zhang, and Y. Yan, "Alternative pseudo-labeling for semi-supervised automatic speech recognition," *IEEE/ACM Trans. Audio Speech Lang. Process.*, vol. 31, pp. 3320–3330, 2023. [Online]. Available: https://doi.org/10.1109/TASLP.2023.3306709

[14] D. Gao, H. Xu, D. Raj, L. P. G. Perera, D. Povey, and S. Khudanpur, "Learning from flawed data: weakly supervised automatic speech recognition," in *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, Taipei, Taiwan, Dec. 2023. doi: 10.1109/ASRU57964.2023.10389684.

[15] C.-T. Do, R. Doddipatla, and T. Hain, "Multiple-hypothesis CTC-based semi-supervised adaptation of end-to-end speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, Toronto, ON, Canada, June 2021, pp. 6978–6982. [Online]. Available: https://ieeexplore.ieee.org/document/9414414

[16] K.-F. Lee, H.-W. Hon, and R. Reddy, "An overview of the SPHINX speech recognition system," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 38, no. 1, pp. 35–45, Jan. 1990.

[17] D. Povey, V. Peddinti, D. Galvez, P. Ghahremani, V. Manohar, X. Na, Y. Wang, and S. Khudanpur, "Purely sequence-trained neural networks for ASR based on lattice-free MMI," in *Proc. Interspeech*, San Francisco, CA, USA, Sep. 2016, pp. 2751–2755. doi: 10.21437/Interspeech.2016-595.

[18] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, 2020, pp. 12449–12460.

[19] A. Baevski, W.-N. Hsu, A. Conneau, and M. Auli, "Unsupervised speech recognition," in *Advances in Neural Information Processing Systems*, vol. 34, 2021, pp. 27826–27839. [Online]. Available: https://openreview.net/forum?id=QmxFsofRvW9

[20] X. Li, S. Dalmia, J. Li, M. Lee, P. Littell, J. Yao, A. Anastasopoulos, D. R. Mortensen, G. Neubig, A. W. Black, and F. Metze, "Universal phone recognition with a multilingual allophone system," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, May 2020, pp. 8249–8253. [Online]. Available: https://par.nsf.gov/servlets/purl/10175774

[21] B. Yan, S. Dalmia, D. R. Mortensen, F. Metze, and S. Watanabe, "Differentiable allophone graphs for language-universal speech recognition," in *Proc. Interspeech*, Brno, Czech Republic, Aug. 2021, pp. 2471–2475. [Online]. Available: https://www.isca-speech.org/archive/interspeech_2021/yan21_interspeech.html

[22] D. R. Mortensen, S. Dalmia, and P. Littell, "Epitran: Precision G2P for Many Languages," in *Proc. Eleventh Int. Conf. Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan, May 2018. [Online]. Available: https://aclanthology.org/L18-1429/

[23] N. Cremelie and J.-P. Martens, "In search of better pronunciation models for speech recognition," *Speech Communication*, vol. 29, no. 2–4, pp. 115–136, Nov. 1999.

[24] M. Mohri, F. Pereira, and M. Riley, "Weighted finite-state transducers in speech recognition," *Computer Speech & Language*, vol. 16, no. 1, pp. 69–88, Jan. 2002. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0885230801901846

[25] W. Kang, L. Guo, F. Kuang, L. Lin, M. Luo, Z. Yao, X. Yang, P. Żelasko, and D. Povey, "Fast and parallel decoding for transducer," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, Rhodes Island, Greece, Jun. 2023, pp. 1–5, doi: 10.1109/ICASSP49357.2023.10094567.

[26] R. Ardila, M. Branson, K. Davis, M. Henretty, M. Kohler, J. Meyer, R. Morais, L. Saunders, F. M. Tyers, and G. Weber, "Common Voice: A Massively-Multilingual Speech Corpus," in *Proc. 12th Language Resources and Evaluation Conf. (LREC)*, Marseille, France, May 2020, pp. 4218–4222. [Online]. Available: https://aclanthology.org/2020.lrec-1.520/

[27] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, D. S. Pallett, and N. L. Dahlgren, "DARPA TIMIT Acoustic-Phonetic Continuous Speech Corpus CD-ROM," NIST Speech Disc 1-1.1, NIST Interagency or Internal Report (NISTIR) 4930, February 1993.

[28] I. Schuurman, M. Schouppe, H. Hoekstra, and T. van der Wouden, "CGN, an annotated corpus of spoken Dutch," in *Proceedings of the 4th International Workshop on Linguistically Interpreted Corpora (LINC-03) at EACL 2003*, Budapest, 2003. [Online]. Available: https://aclanthology.org/W03-2414/

[29] M. Bisani and H. Ney, "Joint-sequence models for grapheme-to-phoneme conversion," *Speech Communication*, vol. 50, no. 5, pp. 434–451, May 2008. [Online]. Available: https://doi.org/10.1016/j.specom.2008.01.002

[30] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu, and R. Pang, "Conformer: Convolution-augmented Transformer for Speech Recognition," in *Proc. Interspeech*, Shanghai, China, Oct. 2020, pp. 5036–5040. [Online]. Available: https://www.isca-archive.org/interspeech_2020/gulati20_interspeech.pdf

[31] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N. E. Yalta Soplin, J. Heymann, M. Wiesner, N. Chen, A. Renduchintala, and T. Ochiai, "ESPnet: End-to-End Speech Processing Toolkit," in *Proc. Interspeech*, Hyderabad, India, Sep. 2018, pp. 2207–2211. [Online]. Available: https://doi.org/10.21437/Interspeech.2018-1456

[32] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *Proc. 3rd Int. Conf. Learn. Representations (ICLR)*, San Diego, CA, USA, May 2015. [Online]. Available: https://arxiv.org/abs/1412.6980