EURASIP Journal on Advances
in Signal Processing

**RESEARCH** **Open Access**

# Free resources for forced phonetic alignment in Brazilian Portuguese based on Kaldi toolkit

Cassio Batista[*], Ana Larissa Dias and Nelson Neto

*Correspondence:
cassiotb@ufpa.br
Computer Science Graduate Program, FalaBrasil Group, Federal University of Pará, Rua Augusto Corrêa, 1, Belém 66075–110, Brazil

**Abstract**

Phonetic analysis of speech, in general, requires the alignment of audio samples to its phonetic transcription. This could be done manually for a couple of files, but as the corpus grows large, it becomes infeasibly time-consuming. This paper describes the evolution process toward creating free resources for phonetic alignment in Brazilian Portuguese (BP) using Kaldi, a toolkit that achieves state of the art for open-source speech recognition, within a toolkit we call UFPAlign. The contributions of this work are then twofold: developing resources to perform forced alignment in BP, including the release of scripts to train acoustic models via Kaldi, as well as the resources themselves under open licenses; and bringing forth a comparison to other two phonetic aligners that provide resources for BP, namely EasyAlign and Montreal Forced Aligner (MFA), the latter being also Kaldi-based. Evaluation took place in terms of phone boundary and intersection over union metrics over a dataset of 385 hand-aligned utterances, and results show that Kaldi-based aligners perform better overall, and that UFPAlign models are more accurate than MFA's. Furthermore, complex deep-learning-based approaches still do not improve performance compared to simpler models.

**Keywords:** Forced aligner, Phonetic alignment, Speech segmentation, Acoustic modeling, Kaldi, Brazilian Portuguese

## 1 Introduction

Forced phonetic alignment is the task of aligning a speech recording with its phonetic transcription, which is useful across a myriad of linguistic tasks such as prosody analysis. However, annotating phonetic boundaries of several hours of speech by hand are very time-consuming, even for experienced phoneticians. As several approaches have been applied to automate this process, some of them brought from the automatic speech recognition (ASR) domain, the combination of hidden Markov models (HMM) and Gaussian mixture models (GMM) has been for long the most widely explored for forced alignment.

Regardless of the technique adopted, phonetic alignment resources for Brazilian Portuguese (BP) are still scarce. With respect to ASR-based frameworks, our research found only three forced aligners that provide pre-trained models for BP: EasyAlign [1], Montreal Forced Aligner (MFA) [2] and UFPAlign [3, 4]. To the best of our knowledge, EasyAlign is the only HTK-based aligner that ships with a model for BP, but appears to be no

Springer Open

longer maintained; MFA is the only Kaldi-based one; and UFPAlign has been evolving through time to work with both HTK and Kaldi as back-end.

As a matter of fact, UFPAlign was initiated in [3], providing a package with grapheme-to-phoneme (G2P) converter, syllabification system and GMM-based acoustic models trained over the HTK toolkit [5]. As usual, tests comparing the automatic versus manual segmentations were performed. An extra comparison was made to EasyAlign [1], which to our knowledge was the only aligner that supported BP at that moment. It was observed that the tools achieved equivalent behaviors, considering two metrics: boundary-based and overlap rate.

Later on, following Kaldi's success as the de facto open-source toolkit for speech recognition [6] due to its efficient implementation of neural networks for training hybrid HMM-DNN acoustic models, UFPAlign was updated in [4] with respect to its HTK-based version, yielding better results with both monophone and triphone GMM-based models, as well as with a standard feed-forward, DNN-based model trained using `nnet2` recipes. Both HTK- and Kaldi-based versions of UFPAlign were then evaluated over a dataset containing 181 utterances spoken by a male speaker, whose phonemes were manually aligned by an expert phonetician.

Therefore, as `nnet2` recipes became outdated, this work builds upon [4] by updating training scripts to Kaldi's `nnet3` recipe, which contains the current state-of-the-art scripts for ASR. Up-to-date versions of the acoustic models, phonetic and syllabic dictionaries were released to the public under the MIT license on FalaBrasil's GitHub account,[1] as well as the scripts to generate them. Assuming Kaldi is pre-installed as a dependency, UFPAlign pipelines works fine under Linux environments via command line, but also provides a graphical interface as a plugin to Praat [7], a popular free software package for speech analysis in phonetics.

Additionally, some intra- and inter-evaluation procedures were performed, the former considering all acoustic models trained within the Kaldi's default GMM and DNN pipeline, the latter applying the HTK former version of UFPAlign [3], EasyAlign [1], and MFA [2] aligners over the same dataset for the sake of a fair comparison. The evaluation dataset was extended from 193 utterances spoken by a male individual to include 192 sentences spoken by a female speaker, i.e., 385 manually aligned audio files in total. The similarity measure is given in terms of the absolute difference between the forced alignments with respect to manual ones, which is called phonetic boundary [2]. A second metric, known as intersection over union (IoU), is widely used in image segmentation for object detection [8]. IoU computes the ratio between the overlap regions of both forced and manual alignments (intersection) and their respective areas combined (union).

In summary, the contributions of this work include:

- Release of monophone-, triphone-, and DNN-based (`nnet3`) acoustic models, which comprise a total of five pre-trained, Kaldi-compatible models included as part of UFPAlign. Scripts used to train such models are also available.

[1] https://github.com/falabrasil.

**Table 1** List of open-source tools that perform forced phonetic alignment

| Tools | Ref. | Based on | Algorithm | License | BP | Allow train | Praat's plugin | Active |
|---|---|---|---|---|---|---|---|---|
| Aeneas | [13] | – | DTW-TTS | AGPL | No | No | No | ✓ |
| DSAlign | [14] | DeepSpeech | DNN | MPL | No | No | No | ✓ |
| EasyAlign | [1] | HTK | GMM | GPL | ✓ | No | ✓ | No |
| FAVE-align | [15] | HTK | GMM | GPL | No | No | No | ✓ |
| Gentle | [16] | Kaldi | DNN (`nnet3`) | MIT | No | No | No | ✓ |
| kaldi-dnn-ali-gop | [17] | Kaldi | GMM, DNN (`nnet3`) | GPL | No | No | No | ✓ |
| LaBB-CAT | [18] | HTK | GMM | GPL | No | ✓ | No | ✓ |
| MAUS | [19] | HTK | GMM | – | No | No | no | ✓ |
| MFA | [2] | Kaldi | GMM, DNN (`nnet2`) | MIT | ✓ | ✓ | No | ✓ |
| P2FA | [20] | HTK | GMM | – | No | No | No | No |
| Prosodylab-Aligner | [21] | HTK | GMM | MIT | No | ✓ | No | ✓ |
| SailAlign | [22] | HTK | GMM | GPL | No | No | No | No |
| SPPAS | [23] | Julius | GMM | GPL | No | ✓ | No | ✓ |
| UFPAlign | [3, 4] | HTK, Kaldi | GMM, DNN (`nnet2`, `nnet3`) | MIT | ✓ | No | ✓ | ✓ |

Characteristics like the ASR framework the system is built upon, main algorithms used, license, and whether it supports Brazilian Portuguese (BP), training models over the same dataset to be aligned, and interfacing with Praat's GUI are also discriminated. The last column also indicates whether the aligner was found to be still actively maintained

- Generation of multi-tier TextGrid files for Praat, based on phonetic and syllabic dictionaries built over a list of words in BP collected from multiple sources and post-processed by GNU Aspell [9] spell checker in order to remove potential misspellings.
- Embedding of FalaBrasil's G2P [10, 11] and syllabification [12] software tools within UFPAlign to generate on-the-fly phonemes and syllables, respectively, for words that are eventually missing in the dictionaries.
- Comparison to the only two ASR-based phonetic aligners that exist for Brazilian Portuguese (to the best of our knowledge), regarding the phone boundary metric [2] over a dataset of 385 hand-aligned utterances.

The remainder of this article is structured as follows. Section 2 lists the related academic work in the field and public-available toolkits concerning forced phonetic aligners. Section 3 presents the acoustic model training pipeline, as well as the forced phonetic alignment procedure with Kaldi, and the audio corpora used for training and evaluation. Evaluation tests and results are reported and discussed on Sects. 4 and 5 , respectively. Finally, Sect. 6 presents the conclusion and plans for future work. Appendix 1 shows the detailed per-phone results achieved with respect to the IoU metric for all forced aligner systems evaluated.

## 2 Related work and toolkits

Several automatic phonetic alignment tools have been developed to relieve the phoneticians of the laborious task that is performing manual alignment on an increasing amount of speech data. Table 1 summarizes the main characteristics of some of the currently available open-source tools to perform forced alignment.

Contrary to most automatic phonetic alignments tools, Aeneas [13] is a non-ASR-based forced aligner. Instead, it uses an approach called Sakoe-Chiba band dynamic time warping (DTW) algorithm and text-to-speech (TTS) to compute the alignments [24]. Aeneas is a Python/C library and provides built-in, multi-platform command-line interface (CLI) tools. Currently, Aeneas claims to work on 38 languages.

On the other hand, a well-known ASR-based forced aligner is Prosodylab-Aligner [21], developed at McGill University, Canada. It offers a multi-platform Python interface that essentially automates the HTK workflow. It uses an English dictionary and a monophone-based acoustic model pre-trained over a North American English speech corpus, but it also allows the use of models for other languages with even the possibility of training this language-tailored acoustic model over the same dataset to be aligned. The resulting word and phone alignments are written to Praat's TextGrid file.

Munich Automatic Segmentation (MAUS) [19] is GMM-based forced alignment system developed at the University of Munich, Germany. Although the CLI provides full language support for German only under Linux systems, another 26 languages, including European Portuguese, are supported by a web-based interface [25]. MAUS is distributed under an all rights reserved license and requires HTK as a third-party dependency. The result is stored in a Praat TextGrid file. Therefore, MAUS uses a hybrid approach consisting of statistically weighted rules to predict possible pronunciation variants and an HTK-based search algorithm that uses a statistical classification of the signal to find the most likely segmentation and labeling.

U.S. University of Pennsylvania's Penn Phonetics Lab Forced Aligner (P2FA) [20] provides a Python-based interface on the top of HTK and uses the CMU Pronouncing Dictionary (CMUDict) [26] along with a GMM-based monophone acoustic model trained over the SCOTUS corpus, the U.S. Supreme Court recordings. Although it supports only English, a different version is available for Chinese. This toolkit used to be also available as a web interface, but only the Python command-line interface is now obtainable. As output, P2FA generates a TextGrid file.

SailAlign [22] is a toolkit that implements an adaptive and iterative speech recognition and text alignment approach to allow large-scale data to be processed. It uses triphone-based acoustic models trained with HTK on both Wall Street Journal (WSJ) and TIMIT corpora, hence English is the only language supported. SailAlign is available only as a CLI for Linux.

The Language, Brain and Behaviour Corpus Analysis Tool (LaBB-CAT) [18] is a browser-based linguistics research system developed at the University of Canterbury, New Zealand. LaBB-CAT was designed to index audio corpora, orthographic transcripts, and other time-aligned annotations for easy online access in a central database. Alternatively, it can be also downloaded as an offline standalone package. LaBB-CAT can perform forced alignment using HTK through a train-and-align approach to produce speaker-dependent monophone models [27].

SPPAS [23] is an automatic annotation and analyses tool developed at the *Laboratoire Parole et Langage*, France. It is based on Julius decoder [28], which means the models included in the toolkit were trained with HTK. SPPAS was developed to be as language-and-task-independent as possible, including models for 11 languages including Portuguese, although the Portuguese acoustic model was trained over adapted French and Spanish data. SPPAS also offers both GUI and CLI interfaces on multi-platform environment.

FAVE-align [15] is a CLI tool developed to align sociolinguistic interviews and thus has some advantages when dealing with spontaneous speech, such as allowing multiple speakers and being robust to background noise. It is built upon P2FA, therefore relying on both CMUDict and HTK. Acoustic models were trained on 8000 h of hand-aligned U.S. Supreme Court oral arguments, hence, English is the only language supported. The output is also Praat-compliant.

EasyAlign [1] is one of the forced aligners that supports Brazilian Portuguese, as well as Spanish, French and Taiwan Min. It was developed at the University of Geneva, Switzerland. Relying on HTK, EasyAlign is developed as a Praat's plugin for Windows, having therefore a lower level of difficulty when compared to other tools, since its features are directly accessible from the Praat's menu. Besides, less manual steps are required to generate a multi-level TextGrid output file.

DSAlign [14] is a forced aligner based on DeepSpeech [29], an open-source speech recognition system developed using end-to-end (E2E) deep learning on the top of Google's TensorFlow library. Internally, DSAlign uses a voice activity detector (VAD) to split the provided audio data into voice fragments. Then, the resulting fragments are transcribed into textual phrases via DeepSpeech and finally the actual text alignment is based on a recursive divide and conquer approach, the Smith-Waterman alignment algorithm [30]. However, apart from the fact that DeepSpeech outputs characters instead of phonemes due to its E2E fashion, DSAlign produces only word alignment-based VAD decision boundaries, which might include one or more words per segment in JSON format.

As for Kaldi-based forced aligners, Gentle [16] is available either as a GUI in a web browser, or as a Python library. Gentle is built on top of Kaldi's time-delay neural network (TDNN) models [31, 32], a type of HMM-DNN acoustic model, pre-trained on Fisher English corpus following the Kaldi's ASpIRE recipe. Currently, Gentle performs forced alignment only on English data and it does not appear to be an academic work since no publications have been found. Therefore, to the best of our knowledge, there is no work regarding Gentle's performance compared to the others currently available automatic alignment tools.

Forced-alignment and Goodness of Pronunciation tool (`kaldi-dnn-ali-gop`) [17] is also a Kaldi-based aligner, available as a toolkit to be included under a Kaldi installation. It supports both GMM and DNN acoustic modeling architectures, the latter being built upon Kaldi's `nnet3` setup for TDNNs. Acoustic models are based on Kaldi's LibriSpeech recipe using LibriSpeech dataset [33]. This aligner is released under GPL and supports only English.

One of the most recent automatic phonetic alignment tools is the Montreal Forced Aligner (MFA) [2]. MFA is a 29-language multilingual update to the English-only Prosodylab-Aligner [21] and maintains its key functionality of training on new data, as well as incorporating improved architecture (triphone GMMs and speaker adaptation), which also offers the possibility of using DNN-based acoustic models based on `nnet2` recipes.[2] BP support from MFA relies on a model trained over a 22-h corpus from GlobalPhone dataset [34].

[2] It seems MFA has dropped DNN support in version 2.0 pre-releases.

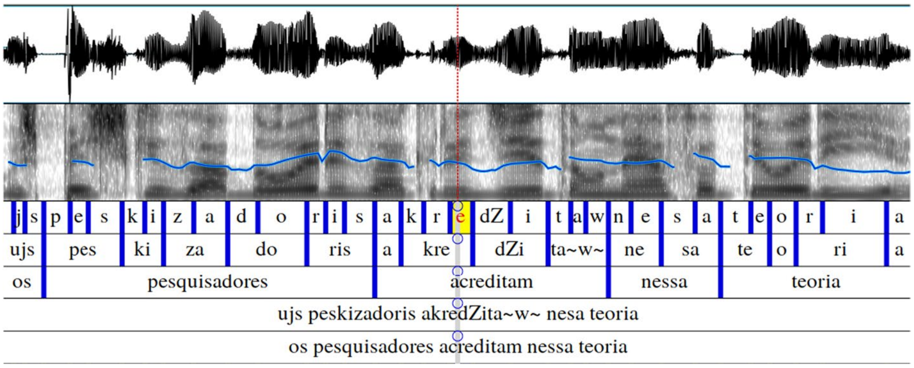

**Fig. 1** Visualization of a TextGrid file via Praat's editor, including five tiers: phonemes, syllables, words, and phonetic and orthographic transcription, respectively. FalaBrasil's G2P [10, 11] and syllabification [12] library provide phonetic and syllabic tokens, respectively, from text. Praat [7] is responsible for plotting the audio's waveform in both time and frequency domains, while Kaldi [6] provides the temporal marks for the dark-blue vertical bars that keep tokens apart from each other

Likewise, UFPAlign has been developed exclusively for BP by the FalaBrasil Research Group at Federal University of Pará (UFPA), Brazil. It is available as a Praat's plugin, but also works via CLI under Linux environments. UFPAlign consists of a set of tools, such as a grapheme-to-phone (G2P) converter, syllabification system and acoustic models that automatically produce segmentations audios in Brazilian Portuguese, initially via HTK toolkit, and more recently via Kaldi scripts running on the back-end [3, 4].

Undoubtedly, there are currently several open-source toolkits to perform automatic phonetic alignment. Thus, it is up to the user to choose which forced alignment tool is more appropriate for their goal and necessity according to the offered features, such as supported language, algorithm, interface, license, and so on.

Nevertheless, despite the diversity of available tools and resources for speech recognition as acoustic models, public resources and tools are still scarce for less representative languages, such as Brazilian Portuguese. Among the summarized automatic phonetic aligners, only three of them support BP.

Therefore, this work's main motivation is twofold: (1) build upon UFPAlign to mitigate that gap for BP by providing MIT-licensed monophone-, triphone- and DNN-based acoustic models trained with the latest recipes of Kaldi over a corpora of approximately 171 h of speech data, as well as phonetic and syllabic dictionaries constructed from a list of 200,000 words in Brazilian Portuguese using FalaBrasil's G2P and syllabification tools [10–12]; and (2) provide more consistent tests over hand-annotated time alignments from a dataset containing 193 and 192 utterances from a male and a female speaker, respectively, against the current only two ASR-based tools that also work for BP: EasyAlign and MFA.

## 3 Methodology

This section details the forced phonetic alignment process within UFPAlign, which is similar to a traditional decoding stage in speech recognition where one needs an acoustic model and a phonetic dictionary (or lexicon) to decide among senones, except the language model is not necessary in such case. UFPAlign works via command line on Linux, but also as a plugin for Praat, a popular speech-related software which is then used as graphical interface to display a visual representation of an audio with its respective time alignments over ortographic, phonetic and syllabic tokens, as shown in Fig. 1.

For that, UFPAlign uses Kaldi as the ASR back-end to automatically compute time stamps based on the knowledge of a previously trained acoustic model (also generated by Kaldi), and FalaBrasil's grapheme-to-phoneme (G2P) and syllabification tools to provide phonemes and syllables from regular words (also known as graphemes), given that users themselves provide such transcriptions as input alongside with the corresponding audio file. The output is stored in a TextGrid file—a well-known file format for Praat users.

### 3.1 UFPAlign tools: Kaldi, grapheme-to-phoneme and syllabification

Kaldi [6] is an open-source toolkit developed to support speech recognition researchers. Based on finite-state transducers (FST) built upon the OpenFst library [35], the toolkit provides standardized scripts written in Bash (called "recipes"), which wrap C++ executables to build all sort of input-speech-related tasks. Kaldi relies on hidden Markov models (HMM) to model the speech's sequential characteristics in a dual-fashion architecture for training acoustic models: HMMs combined either with Gaussian mixture models (GMM) [36], or with deep neural networks (DNN) [37]. While GMMs are used to model HMM output probability densities from scratch, the DNN training actually uses the GMM model to produce high-level alignments as reference for the final acoustic model [38].

The DNN training framework is provided by Kaldi in three distinct setups[3]: `nnet1` [39], `nnet2` [40, 41] and `nnet3`. Among the setups, there are some differences regarding the training, such as nonlinearity types, learning rate schedules, network topology, input features and so on. However, unlike `nnet1` and `nnet2`, `nnet3` offers an easier access to use and configure more specialized kinds of networks other than simple feed-forward ones, including long short-term memory (LSTM) [42] and time-delay neural networks (TDNN) [31, 32], for example.

As Kaldi requires a phonetic dictionary or lexicon to serve as the target being modeled by HMMs, this work uses a G2P converter provided by the FalaBrasil Group as an open-source library written in Java [10, 11]. This tool relies on a stress determination system that is based on a set of rules that do not focus in any particular BP dialect and provide only one pronunciation by word, which means it deals only with single words and does not implement co-articulation analysis between words (i.e., cross-word events are not considered). The phonetic alphabet is composed by 38 phonemes plus a silence phone, inspired by the Speech Assessment Methods Phonetic Alphabet (SAMPA) [43], a system of phonetic notation.

The syllabification tool, on the other hand, is not a requirement when training acoustic models for ASR, but rather just a feature of UFPAlign for composing another tier in the TextGrid output file. It is also provided by the FalaBrasil Group within the same library

[3] http://www.kaldi-asr.org/doc/dnn.html.

**Table 2** Speech corpora used to train acoustic models

| Dataset | Refs. | Hours | Words | Speakers |
|---|---|---|---|---|
| LapsStory | [11] | 5 h:18 m | 8257 | 5 |
| LapsBenchmark | [11] | 0 h:54 m | 2731 | 35 |
| Constitution | [44] | 8 h:58 m | 5330 | 1 |
| Consumer protection code | [44] | 1 h:25 m | 2003 | 1 |
| Spoltech LDC | [45] | 4 h:19 m | 1145 | 475 |
| West point LDC | [46] | 5 h:22 m | 484 | 70 |
| CETUC | [47] | 144 h:39 m | 3528 | 101 |
| Total | | 170 h:51 m | 14,518 | 687 |

as the G2P, the algorithm is also rule-based and do not focus on any particular Brazilian dialect either [12].

### 3.2 Training speech corpora and lexicon

The FalaBrasil speech corpora[4] consists of seven datasets in BP , as summarized in Table 2. The datasets contain audio files in an uncompressed, linear, signed PCM (namely, WAVE) format and are sampled at 16 kHz with 16 bits per sample.

A language model (LM), despite not being used during phonetic alignment, is necessary for training the acoustic model. The LM used here was built in [11] using SRILM [48] toolkit over ~1.5 million sentences from the CETENFolha dataset [49].

Finally, the phonetic dictionary was created via FalaBrasil G2P tool [10, 11] based on a list of words collected from multiple sources on the Internet, including a word list from University of Minho's Projecto Natura [50], LibreOffice's VERO dictionary [51], NILC's CETENFolha dataset [49], FrequencyWords repository based on subtitles from OpenSubtitles [52, 53], and the transcription of FalaBrasil's audio corpora described in Table 2. GNU Aspell [9] is responsible for checking out the spelling and consequently filtering the huge number of words collected, resulting in approximately 200,000 words in the final list.

### 3.3 Acoustic models

The deep-learning-based training approach in Kaldi actually uses the GMM training as a pre-processing stage. Figure 2 shows the pipeline to train a DNN acoustic model based on GMM triphones using Kaldi. For this work, AMs were trained by adapting the recipe for Mini-librispeech dataset [54], as opposed from our previous work in which scripts were originally based on recipes for Wall Street Journal (WSJ) [55] and Resource Management (RM) [4] datasets. The difference between recipes relies mainly on the architecture of the neural network, as will be shown later.

In the front-end, the acoustic waveforms from the training corpus are windowed at every 25 ms with 10 ms of overlap, being encoded as a 39-dimension vector: 12 Mel frequency cepstral coefficients (MFCCs) [56] using C0 as the energy component, plus

[4] https://github.com/falabrasil/speech-datasets.

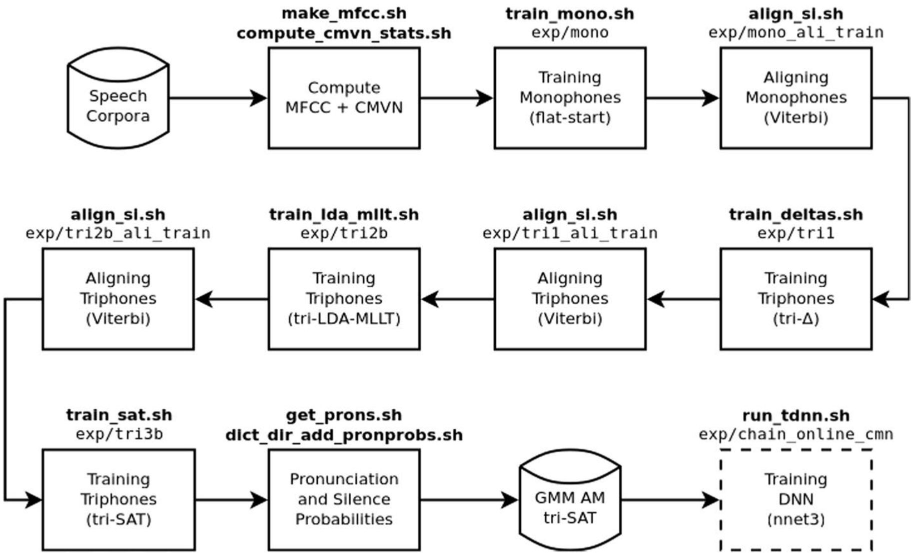

**Fig. 2** Stages for training a hybrid DNN-HMM acoustic model on the top of a triphone-based, speaker-adapted GMM one. Lexicon and LM are omitted, but are basic inputs for all training blocks as well as the MFCCs. Scripts are based on Kaldi's Mini-librispeech recipe, and the DNN in particular was trained following the `nnet3` setup. At the top of each block, there is a reference for a script (boldface) and a directory where resources created by that script are usually stored (normal text). More details on the DNN training (dashed block) are depicted in Fig. 3

13 delta ($\Delta$, first derivative) and 13 acceleration ($\Delta\Delta$, second derivative) coefficients are extracted from each window.

The flat-start approach models 39 phonemes (38 monophones plus one silence model) as context-independent HMMs, using the standard 3-state left-to-right HMM topology with self-loops. At the flat-start, a single Gaussian mixture models each individual HMM with the global mean and variance of the entire training data. Also, the transition matrices are initialized with equal probabilities.

Kaldi uses Viterbi training [57] to re-estimate the models at each training step. Likewise, in order to allow training algorithms to improve the model parameters, Viterbi alignment is applied after each training step. Subsequently, the context-dependent HMMs are trained for each triphone considering $\Delta$ and $\Delta\Delta$ coefficients. Each triphone is represented by a leaf on a decision tree. Eventually, leaves with similar phonetic characteristics are then tied/clustered together.

The next step is the linear discriminant analysis (LDA) combined with the maximum likelihood linear transform (MLLT) [58–60]. The LDA technique takes the feature vectors and splices them across several frames, building HMM states with a reduced feature space. Then, a unique transformation for each speaker is obtained by a diagonalizing MLLT transform. On top of LDA+MLLT features, a speaker normalization that uses feature-space maximum likelihood linear regression (fMLLR) as alignment algorithm is applied [61].

The last step of the GMM training is the speaker adaptive training (SAT) [62, 63]. SAT is applied on top of the LDA+MLLT features performing adaptation and projecting training data into a speaker normalized space. This way, by becoming independent

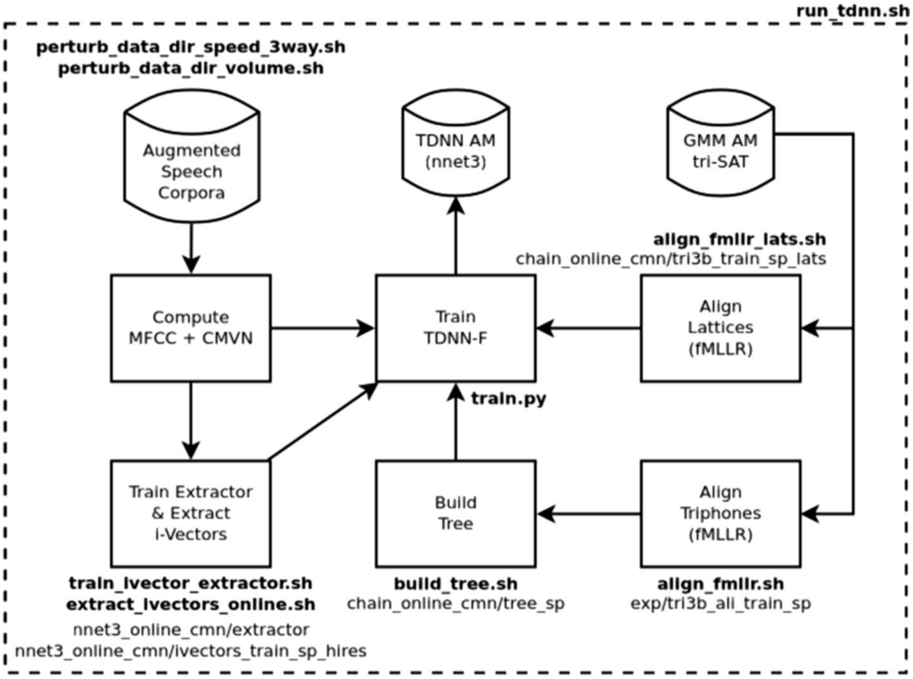

**Fig. 3** Stages for training the neural network (TDNN-F) following Kaldi's Mini-librispeech recipe. High-resolution, cepstral-normalized MFCCs (40 features instead of 13) are extracted from a speed/volume augmented corpora, as are the 100-dimensional i-vector features; to be used as input to the DNN. Training labels, on the other hand, are provided by a GMM tri-SAT acoustic model. At the top or bottom of each block, there is a reference for a script (boldface) and the directory where resources created by that script are usually stored (normal text). For details on the training pipeline of the tri-SAT and all previous GMM-based acoustic models, the reader is referred to Fig. 2

of specific training speakers, the acoustic model generalizes better to unseen testing speakers [64].

Figure 3 details how the DNN model is obtained as a final-stage AM by using the neural network to model the state likelihood distributions as well as to input those likelihoods into the decision tree leaf nodes [65]. In short terms, the network input (left side of the flowchart) are groups of feature vectors and the output (on the right side) is given by the aligned state of the SAT GMM system for the respective features of the input. The number of HMM states in the system also defines the DNN's output dimension [66].

The Mini-librispeech recipe also performs data augmentation on the original dataset through speed and volume perturbations, which increases the amount of data by five times [67]. Moreover, alongside normalized cepstral coefficients, the network is also fed i-vectors [68, 69], also extracted from the speech signal, as input features by default, which have proven to increase performance in speech recognition tasks by incorporating characteristics related to the speakers themselves.

Kaldi's `nnet3` scripts use factorized time-delay neural networks (TDNN-F) as default architecture [31], which are a type of feed-forward network that has a behavior similar to recurrent topologies like the long short-term neural network (LSTM) in the sense of capturing past and future temporal contexts with respect to the current speech frame to

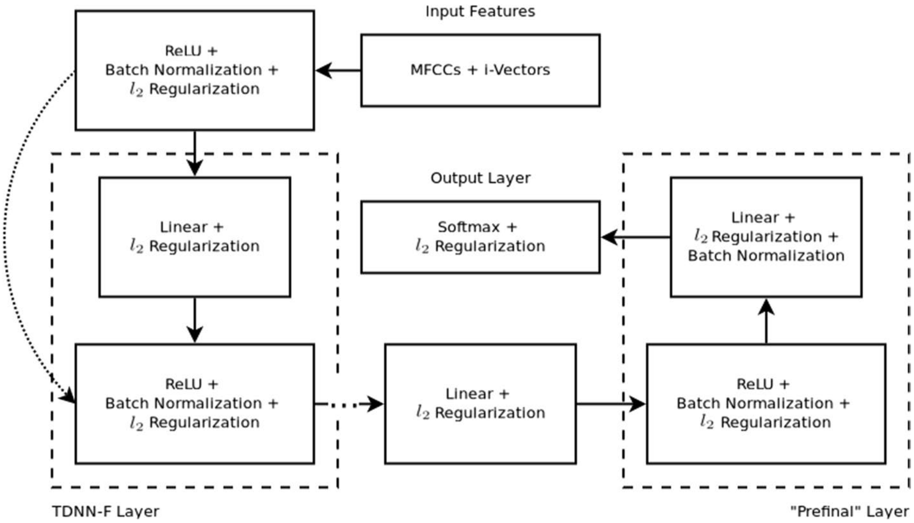

**Fig. 4** Default architecture of the TDNN-F from Kaldi's Mini-librispeech recipe. The setup is composed by multiple TDNN-F layers, although only one is being depicted. The dotted arrow represents a bypass connection similar to what happens in ResNet topologies. All layers but the output apply batch normalization, while $l_2$ regularization is applied without exception. Linear blocks are intentionally thinner by either being bottlenecks or just having a lower dimension with respect to other layers

be recognized, but with an easier procedure for parallelization. This opposes to previous `nnet2` recipes, for instance, which are pure vanilla networks.

The implementation in Kaldi uses a sub-sampling technique that avoids the whole computation of a feed-forward's hidden activations at all time steps and therefore allows a faster training of TDNNs. The "factorized" term distinguishes a TDNN-F from a traditional TDNN architecture by a singular value decomposition (SVD) that is applied at the hidden layer's weight matrices in order to reduce the number of model parameters without degrading performance [32].

Figure 4 illustrates the default architecture of the TDNN-F defined in Mini-librispeech recipe. Multiple instances of the so-called TDNN-F layers appear as a sequence of linear affine operations followed by a rectified linear unit (ReLU) activation function. Linear operations are here referred as the usual dot product affine function that multiplies the resulting coefficients of the immediate predecessor layer by the weight matrix [70], but without considering any bias vector in this case.

A bypass operation similar to what happens in residual networks (ResNet) also appears in between TDNN-F hidden layers. Batch normalization is applied after each ReLU activation, and after the last affine computation that precedes the output layer, while $l_2$ regularization (also known as $l_2$ norm or Euclidean norm) is applied after every single block. Finally, Euclidean norm is applied over the softmax output layer that models the probability distributions over senones. Table 3 summarizes some of the parameters used during training.

### 3.4 Kaldi forced phonetic alignment

UFPAlign uses Kaldi, a toolkit that is under active development and provides state-of-the-art algorithms for many speech-related tasks, including stable neural-network

**Table 3** Parameters used for TDNN-F training

| Parameter | Value |
|---|---|
| # TDNN-F layers | 12 |
| # Epochs | 10 |
| Time strides (TDNN-F layers) | {1, 1, 1, 0, 3, 3, 3, 3, 3, 3, 3, 3} |
| Dimension | 768 on both TDNN-F and prefinal layers |
| Bottleneck dimension | 96 on TDNN-F layers, 192 on prefinal layer |
| Bypass scale | 0.66 |
| Frame subsampling factor | 3 |
| Regularization parameter | 0.015 for output layer, 0.03 otherwise |
| Learning rate | 0.002 down to 0.0002 |

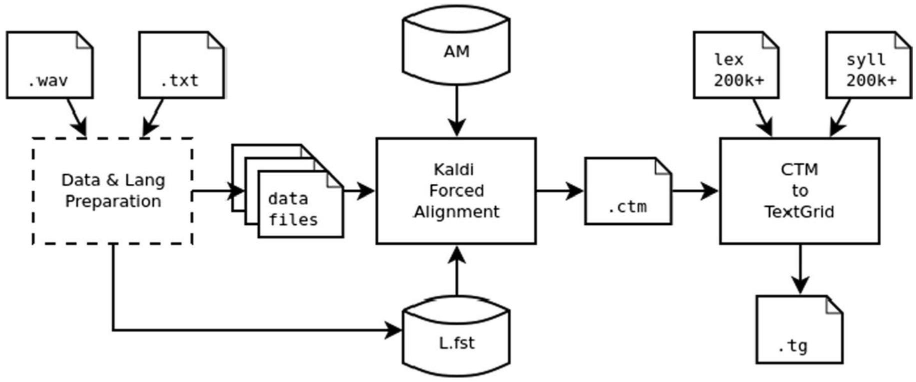

**Fig. 5** The phonetic alignment pipeline followed by UFPAlign. When a user feeds the system with an audio (`.wav`) and its respective transcription (`.txt`), they should expect a TextGrid file as output. Time marks are provided by Kaldi, which relies on the knowledge of the acoustic model (AM) and tokens of the lexicon (L.fst). Preparation of data and language files (dashed block) are especially detailed in Fig. 6

frameworks. Our aligner has also been developed as a plugin for Praat [7], a popular speech analysis software, which aims to ensure a user-friendly interface requiring only a few manual steps in the process. In fact, the plugin's interface was developed in Praat's programming language—Praat Scripting. Following a successful alignment, a multi-level annotation TextGrid (`.tg`) file can be loaded into Praat. Figure 5 shows the pipeline within UFPAlign to phonetically annotate speech samples. As usual, it requires an audio file (`.wav`) and its corresponding orthographic transcription (`.txt`) as input.

Kaldi forced alignment block itself performs several steps for obtaining the time-marked conversation (CTM) files, which contains a list of numerical indices corresponding to phonemes with both their start times and durations in seconds. After Kaldi scripts extract some features from time-domain audio data, the forced alignment step, that employs the aforementioned pre-trained acoustic models, is computed by Kaldi using Viterbi beam search algorithm [71]. Depending on the model, the input features could be simply normalized MFCCs for monophone and tri-Δ models, LDA for tri-LDA and tri-SAT models, or i-vectors for TDNN-F model.

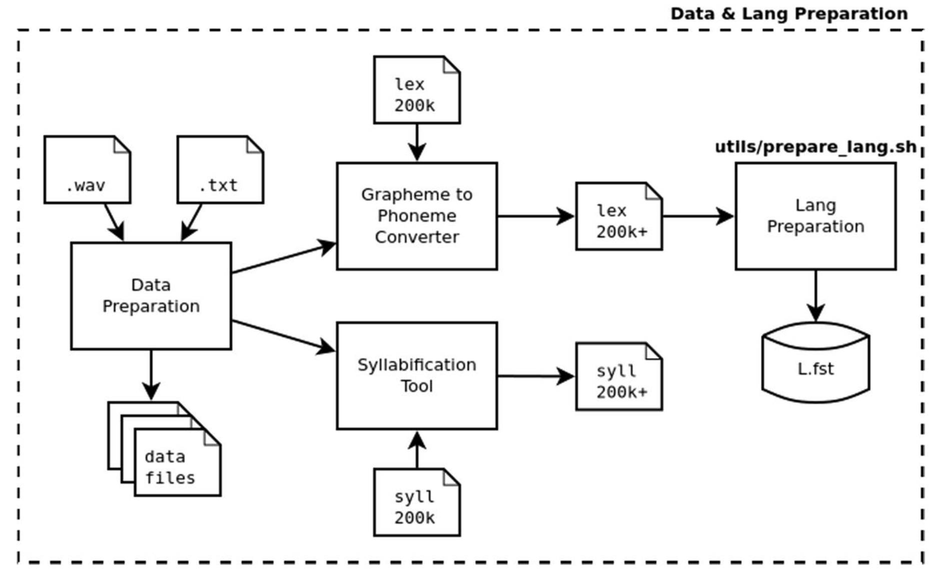

**Fig. 6** Data and language (lang) preparation as a pre-processing step for performing forced alignment with Kaldi. Eventual words that are missing in the vocabulary of the phonetic and syllabic dictionaries (`lex 200k` and `syll 200k`, respectively) are inserted during run time thanks to the embedding of FalaBrasil's G2P and syllabification tools within UFPAlign. Kaldi specific data and language files are also created

The data and language preparation stage in particular also creates some "data files" on the fly, which contain information regarding the specifics of the audio file and its transcription, namely `text`, `wav.scp`, `utt2spk`, and `spk2utt`. The language preparation stage, on the other hand, is given by a script provided by Kaldi to create another set of important files, the main one being the lexicon parsed into an FST format, called `L.fst`. The creation of Kaldi's data and language files is illustrated in Fig. 6.

For data preparation, the first step consists in checking whether there are any new words in the input data that were not seen during the acoustic model training. If any word in the transcriptions is not found in the pronunciation dictionary (lexicon), it calls the grapheme-phoneme conversion module (G2P) [10, 11] to extend the lexicon with each new word along with its respective phonemic pronunciation. For Praat's final visualization purposes, the word is also divided into syllables through the embedded syllabification tool [12]. Original phonetic and syllabic dictionaries originally contain approximately 200,000 entries and are represented as `lex 200k` and `syll 200k`, respectively, in Fig. 6. After missing words are appended, both become `lex 200k+` and `syll 200k+`.

The last block of the phonetic alignment process handles the conversion of both CTM files to a Praat's TextGrid (`.tg`), a text file containing the alignment information. Therefore, CTM files are read by a Python script that in the conversion process uses the `lex 200k+` and `syll 200k+` extended dictionaries to generate the output five-tier TextGrid that can be displayed by Praat's editor (c.f. Fig. 1).

**Table 4** Speech corpus used to evaluate the automatic phonetic aligners

| Dataset | Duration | # Files | # Words | # Tokens |
|---|---|---|---|---|
| Male | 7 m:58 s (7 m:40 s) | 200 (193) | 1260 (665) | 5275 |
| Female | 7 m:34 s (7 m:18 s) | 199 (192) | 1258 (664) | 5262 |
| Total | 15 m:32 s (14 m:58 s) | 399 (385) | 2518 (686) | 10,537 |

Actual duration and number of files after discard are shown between parentheses, as well as the number of unique words

**Table 5** Cross-word mismatches between transcriptions manually aligned by a phonetician (top) versus generated by our G2P software (bottom)

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| (a) *"nada como um* **almoç o ao a r** *livre"* → *"nada como um* ***almoç oa r*** *livre"* | | | | | | | | | |
| a | w | m | o | s | O | ∅ | ∅ | a | h/ |
| a | w | m | o | s | u | a | w | a | X |
| (b) ***pair a u m ar*** *de arara rara no rio"* → *"**pair u m ar** de arara rara no rio"* | | | | | | | | | |
| p | a | j | 4 | ∅ | u ~ | m | a | h/ | |
| p | a | j | r | a | u ~ | ∅ | a | X | |
| (c) *"o baile inicia* ***às nov e e*** *meia"* → *"o baile inicia* ***às nov i*** *meia"* | | | | | | | | | |
| 6 | ∅ | Z | n | O | v | i | ∅ | | |
| a | j | s | n | O | v | i | i | | |

Word boundary losses, typically present in spoken language rather than in text, are represented by the empty set symbol (∅), as well as deletion or addition of phonetic tokens that can be later merged into one (/u~ m → /u~) or split into two or more (/6/ /Z/ → /a/ /j/ /s/), respectively.

## 4 Evaluation tests

The evaluation procedure takes place by comparing a bunch of TextGrid files: the hand-aligned reference and the ones automatically annotated by the forced aligners (i.e., by inference), as the phone boundary and IoU metrics consider the absolute difference between the ending time of both phoneme occurrences [2]. The calculation is performed for each acoustic model, and it takes place over all utterances from the evaluation dataset composed by one male and one female speaker.

It would be important to mention that only the phonetic information is considered during evaluation, i.e., the time stamps of the other tiers that compose the TextGrid file are used just as a product of the output of the aligner. In other words, time boundaries of syllables and words will not be part of the analysis. Furthermore, once again we remind the reader that syllabic tokens are not part of an ASR system, but rather just a feature of our forced aligner as a tool.

### 4.1 Evaluation speech corpus

In these experiments, the automatic alignment was estimated on the basis of the manual segmentation. The original dataset used for assessing the accuracy of the phonetic aligner is composed of 200 utterances spoken by a male speaker, and 199 utterances spoken by a female speaker, in a total of 15 min and 32 s of hand-aligned audio, as shown in Table 4. Praat's TextGrid files, whose phonetic time stamps were manually adjusted by a phonetician, are available alongside audio and text transcriptions.

Although we do acknowledge that this volume of test dataset is small, as is the number of different speakers in the corpus, we also emphasize as a disclaimer the difficulty

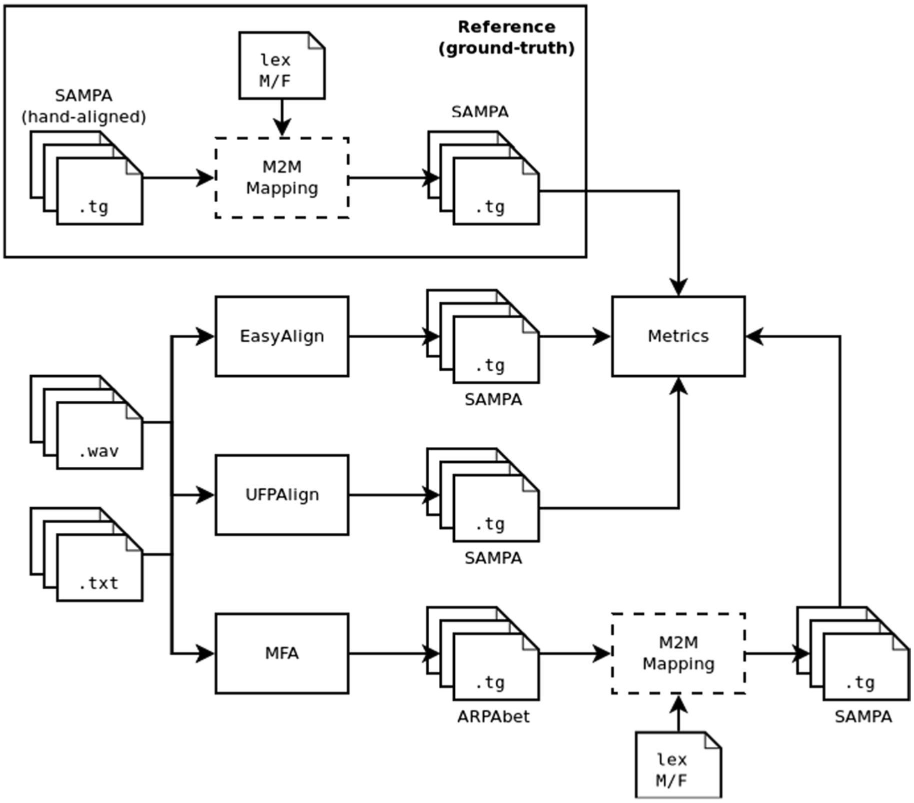

**Fig. 7** Evaluation procedure, in which a comparison on the output of all forced aligners to a hand-aligned ground-truth, is performed. However, metrics should be calculated considering a one-to-one mapping between inference and reference phonemes. This is where the M2M mapping is applied to make different phone sets inside TextGrid files match the SAMPA phonetic alphabet version used by FalaBrasil's G2P converter, which is provided by the lexicon generated over transcriptions from the evaluation speech corpora for both male and female speakers (`lex M/F`). That way, M2M mapping ensures each reference and inference utterance shares the same number of phones before undergoing evaluation

to have access to this kind of somewhat very specific labeled data. Moreover, the time it takes even for expert phoneticians to annotate each phoneme's time stamp by hand is insurmountably high [1].

This dataset was aligned with a set of phonemes inspired by the SAMPA alphabet, which in theory is the same set used by the FalaBrasil's G2P software that creates the lexicon during acoustic model training. Nevertheless, there are some problems of phonetic mismatches, and some cross-word phonemes between words, which makes the mapping between both phoneme sets challenging, given that FalaBrasil's G2P only handles internal-word conversion [10].

The example in Table 5 shows the phonetic transcription for three sentences given by the original dataset (top) and the acoustic model (bottom) which then suppress vowel sounds altogether due to cross-word rules (usually elision and apocope) when they occur at the end of the current word and at the beginning at the next. Such mismatches occur because the dataset was aligned by a phonetician considering acoustic information (i.e., listening) as the sentences are spoken in real life, which cannot be done by the G2P tool that creates the acoustic model's lexicon, since it is provided only with textual

**Table 6** Equivalence table between phonemes present in the hand-aligned dataset (original) and the ones yielded by FalaBrasil's G2P tool (FB)

| Orig. | FB | Orig. | FB | Orig. | FB | Orig. | FB | Orig. | FB | Orig. | FB |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 6 | a | e | e | j | j | n | n | h | R | v | v |
| a | a | e~ | e~ | j~ | j~ | o | o | s | s | w | w |
| 6~ | a~ | E | E | J | J | o~ | o~ | S | S | w~ | w~ |
| a~ | a~ | f | f | k | k | O | O | t | t | 4 | X |
| b | b | g | g | l | l | p | p | - | tS | z | z |
| d | d | i | i | L | L | h/ | r | u | u | Z | Z |
| - | dZ | i~ | i~ | m | m | h\ | R | u~ | u~ | | |

Most of the mappings are one-to-one, but some phones do not have a mapping, and phone swaps frequently occur (both cases are shaded in light gray)

information. Situations like these of phonetic information loss led to the removal of such audio files from the dataset before evaluation.

In the end, fourteen files were excluded from the dataset, so about 34 s of audio was discarded, and 193 and 192 utterances remained in the male and female datasets, respectively. The filtering also ignored intra- and inter-word pauses and silences, resulting in 2518 words (686 unique, since the utterances' transcriptions are identical for both speakers, i.e, they speak the very same sentences) and 10,537 phonetic segments (tokens) (c.f. Table 4).

### 4.2 Simulation overview

Figure 7 shows a diagram of the experiments where EasyAlign, UFPAlign and MFA forced aligners receive the same input of audio files (.wav) with their respective textual transcriptions (.txt). These are the files whose manual annotation is available. All three aligners output one TextGrid file (.tg) for each audio given as input, which then serve as the inference inputs to the phone boundary and IoU calculation. The reference ground-truth annotations, on the other hand, are provided by the 385 TextGrid files that contain the hand-aligned phonemes corresponding to the transcriptions in the evaluation dataset.

However, for computing the metrics, there must exist a one-to-one mapping between the reference and the inference phones, which was not possible at first due to the nature of the phonetic alphabets: UFPAlign and EasyAlign share the same SAMPA-inspired lexicon generated by FalaBrasil's G2P tool, while MFA is based on ARPAbet [72]. Furthermore, the hand-aligned utterances fall on a special case where the phonetic alphabet used (referred here as "original") is also SAMPA, but is not exactly the same as FalaBrasil's, as shown in Table 6.

Apart from the fact seen in Table 5 in which cross-word rules can insert or delete phones when considering word pairs rather than single words, some phonemes do not have an equivalent, such as /tS/ and /dZ/. Besides, there are also usual swaps between phonetically similar sounds: /h//, /h\/, /h/ and /4/, for instance, might be almost deliberately mapped to either /r/, /R/ or /X/. Obviously, the situation is worse for MFA where the set of phonemes is completely different.

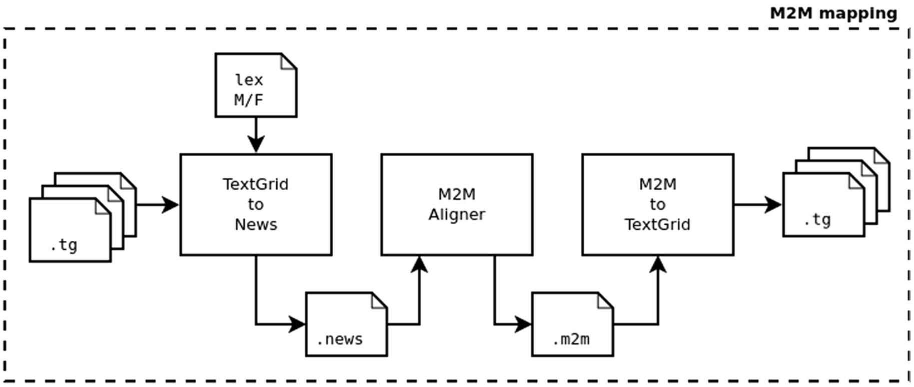

**Fig. 8** Pipeline to convert from many-to-many phonetic correspondence to one-to-one by the m2m-aligner software [73]. This is necessary given that the nature of the phone set used to train AMs is not the same used to manually align the evaluation dataset, although both are SAMPA-inspired; and that MFA uses a different set of phonemes based on ARPAbet

Thus, since the situation seemed to require a smarter approach than a simple one-to-one tabular, static mapping, it was necessary to employ a many-to-many (M2M) mapping procedure (c.f. dashed blocks on Fig. 7) based on statistical frequency of occurrence, e.g., how many times phones `/t/` and `/S/` from the original evaluation dataset were mapped to a single phone `/tS/` in the `lex M/F` file representing FalaBrasil's G2P SAMPA-inspired alphabet. This mapping also works when dealing with MFA's ARPAbet phonemes and will be further discussed in Sect. 4.3.

### 4.3 Many-to-many (M2M) phonetic mapping

By taking another look at Table 5, one might have also reasoned that the mapping between the two sets of phonemes is not always one-to-one. The usual situation is where a pair of phonemes from the dataset (original) is merged into a single one for the AM (FalaBrasil G2P), such as `/i~/` `/n/` → `/i~/` and `/t/ /S/` → `/tS/`. However, a single phoneme can also be less frequently split into two or more, such as `/u/ /S/` → `/u/ /j/ /s/`.

To deal with these irregularities, we used the many-to-many alignment model (m2m-aligner) software [73] in the core of a pipeline that converts the original TextGrid from the evaluation dataset to a TextGrid that is compatible with the FalaBrasil's phonetic dictionary (or lexicon) used to train the acoustic models, as shown in Fig. 8. We took advantage of the same pipeline to convert MFA's ARPAbet-based phonemes to SAMPA as well.

The m2m-aligner works in an unsupervised fashion, using an edit-distance-based algorithm to align two different (unaligned) strings from a file in the `news` format, in order for them to share the same length [73]. As this algorithm works based on frequency counts (e.g., how many times phonemes `/d/` and `/Z/` are merged to `/dZ/`), all 385 TextGrid files from our evaluation dataset, represented as short `.tg`, are used to compose a single `news` file, whose format is exemplified in Table 7. Notice the file is composed by the phonemes of the whole sentence rather than by isolated words, in order to mitigate the effects of the cross-word boundaries. The

**Table 7** Example of a single `news` file with phonemes from three out of 385 TextGrid files for sentences *"é bom pousar"*, *"os lindos jardins"* and *"vergonha do país"*

(a) Original dataset phone set (original SAMPA) vs. FalaBrasil's (SAMPA)

| Dataset phonemes (SAMPA, original) | Acoustic model phonemes (SAMPA, FB) |
|---|---|
| `E b o~ n p o w z a h` | `E b o~ p o w z a X` |
| `u S l i~ n d u S Z a h\ d Z i~ n S` | `u j s l i~ d u s Z a R dZ i~ s` |
| `v e h/ g o~ J 6 d u p a i S` | `v e R g o~ J a d u p a i j s` |

(b) MFA phone set (ARPAbet) vs. FalaBrasil's (SAMPA)

| MFA phonemes (ARPAbet) | Acoustic model phonemes (SAMPA, FB) |
|---|---|
| `E+ B O~+ W~ P O Z A+ RR` | `E b o~ p o w z a X` |
| `UX S L I~+ D UX S Z A RR DJ I~ S` | `u j s l i~ d u s Z a R dZ i~ s` |
| `V E RR G O+ NJ AX D O+ P A I+ S` | `v e R g o~ J a d u p a i j s` |

Each line contains a whole phonetic sentence to be converted, and different phone sets are separated into two distinct columns divided by a tabular `'\t'` character, so every other phonetic token is separated by a single space. Groups of phonemes which are supposed to be later merged by m2m-aligner in the `m2m` file are shaded in gray

**Table 8** Conversion of time stamps for the sentence *"onde existem"*

| **494** | **533**$^{*}$ | **558** | **565**$^{*}$ | **583** | **682** | **748** | **854** | **929** | **979**$^{\ddagger}$ | **1042**$^{\ddagger}$ | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| o~ | n | d | Z | i | i | z | i | S | t | e~ | |
| o~ | | dZ | | i | e | z | i | s | t | e~ | j~ |
| 533* | | 565* | | 583 | 682 | 748 | 854 | 929 | 979$^{\ddagger}$ | 1010 | 1042$^{\dagger}$ |

Markers depict either merging (*) or splitting (‡) of phones

string mapping is finished after a certain number of iterations when the m2m-aligner provides a one-to-one mapping in a file we called `m2m` (c.f. Fig. 8) that joins some phonemes together, as shown by shades of gray in Table 7.

Finally, as the m2m-aligner provides the mapping for phonemes, another script provides the time stamps calculations prior to creating the converted TextGrid file. Table 8 illustrates how the phonetic time stamps, in milliseconds, are mapped accordingly. Basically if two or more phonemes are mapped into a single one (merging), as in /o~/ /n/ → /o~/ or /d/ /Z/ → /dZ/ (marked with an $*$), the time stamp of the last phoneme is considered. However, if one phoneme is mapped to two or more (splitting) as in /e~/ → /e~/ /j~/, then linearly spaced time stamps are generated in between the phone to be split (†) and its immediate predecessor (‡).

We acknowledge that, after splitting a single phoneme into two or more, attributing equal durations to new phonemes does not reflect the physical events of speech, as it is known that vowels have longer durations than consonants and semivowels. However, at first, we kept this model for the sake of simplicity. Moreover, as splitting occur more or less at the same proportion across the output of all forced aligners we tested, we believe this does not influence the accuracy of such.

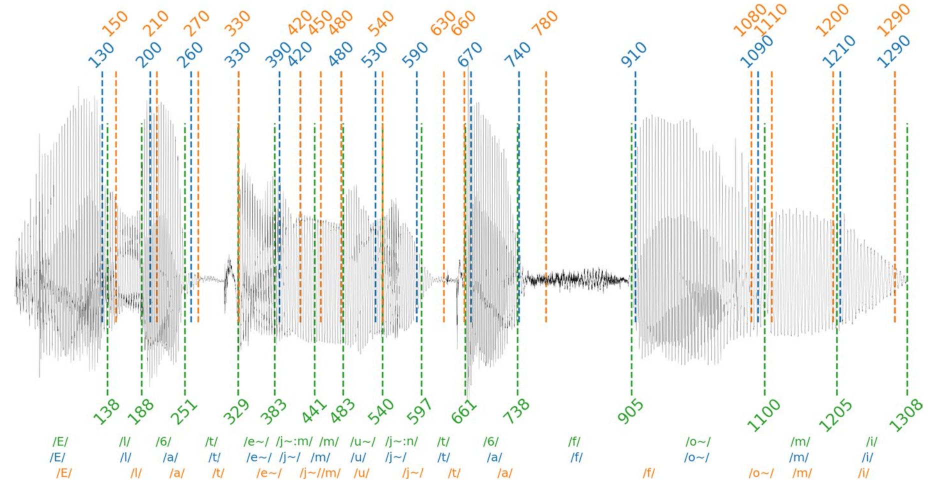

**Fig. 9** Example of alignments for the utterance "*ela tem muita fome*." Ground-truth boundaries are shown in green, while automatic ones produced by Kaldi's monophone- and TDNN-F-based models are shown in blue and orange, respectively. Tokens (phonemes) are shown at the bottom at their respective in-between-boundaries middle position. Notice that phonemes `/j~:m/` and `/j~:n/` have been created in the ground-truth through a merge of the individual phones `/j~`, `/m/` and `/n/`, based on M2M mappings

**Table 9** Original (orig) and FalaBrasil (FB) SAMPA-based phonetic transcriptions for the phrase "*ela tem muita fome*"

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Orig. | `E` | `l` | `6` | `t` | `e~` | `j~:m` | `m` | `u~` | `j~:n` | `t` | `6` | `f` | `o~` | `m` | `i` |
| FB | `E` | `l` | `a` | `t` | `e~` | `j~` | `m` | `u` | `j~` | `t` | `a` | `f` | `o~` | `m` | `i` |
| mono | 8 | 12 | 9 | 1 | 7 | 21 | 3 | 10 | 7 | 9 | 2 | 5 | 10 | 5 | 18 |
| TDNN-F | 12 | 22 | 19 | 1 | 37 | 9 | 3 | 0 | 33 | 1 | 42 | 175 | 10 | 5 | 18 |
| mono | .94 | .71 | .71 | .87 | .87 | .52 | .62 | .78 | .75 | .80 | .86 | .96 | .92 | .88 | .78 |
| TDNN-F | .92 | .53 | .50 | .75 | .58 | .31 | .71 | .95 | .63 | .47 | .64 | .37 | .10 | .86 | .79 |

Respective phone boundary (integer) and intersection over union (floating point) values achieved by the monophone and TDNN-F models for the example in Fig. 9 are also shown

### 4.4 Example of phone boundary and intersection over union

Here, we depict a practical example of the phone boundary and IoU calculation. This is meant to introduce the reader to Sect. 5, in which the results are presented. Figure 9 shows an example of manual and forced alignments for the utterance "*ela tem muita fome*." The time stamps are given in milliseconds. Table 9 shows both the phone sets, the phone boundary, and the IoU values for each phoneme.

To calculate the phone boundary, one would need only to subtract one of the values at the top (either blue or orange) from their respective vertical pair at the bottom (the reference value in green) and then ignore negative signals by considering the absolute value. In a perfect segmentation, all values would be zero, which corresponds to computing the metric on the reference annotations against itself.

The intersection over union, on the other hand, considers the intersection area between the start and end boundaries of each phoneme. For unidimensional signals such as speech, the area is simply the difference (subtraction) between the ending and the starting times. This is then simply divided by the area of the union between the reference and automatic aligned phonemes' time stamps.

Taking the phoneme `/f/` as example, one can see that the TDNN-F boundaries are really off, while the monophone model could almost perfectly align the phoneme (c.f. Fig. 9.) Therefore, the IoU "score" for the monophone would be closer to one (0.96, i.e., better), as their intersection (numerator) is large, which consequently reduces the area of their union (denominator); while for the TDNN-F model, the score would be closer to zero (0.37, i.e., worse), as the area of their union is greater than their intersection (c.f. Table 9, rows 5–6, 13th column.) Analogously, the boundary for phoneme `/f/` should have ended at 905 ms (green), but ended at 910 ms (blue, better) and 1080 ms (orange, worse) when aligned with monophone and TDNN-F models, yielding therefore a phone boundary value of 5 and 175, respectively (c.f. Table 9, rows 3–4, 13th column.)

Finally, taking the results from the monophone-based model as target example, one can see that from the total of 15 phonemes, ten are less than 10ms, and the remaining five are less than 25ms off the manual alignment references, which means 66.7% and 33.3% of the tokens (i.e., all of them), *for this single utterance*, were respectively aligned within these two pre-specified thresholds. With the TDNN-F model, the results were worse: it achieved 40% and 33.3%, but alone these values do not provide an early sum to 100% as they do with the monophone model. Furthermore, for the real evaluation of phone boundary, these percentages are calculated over the whole reference dataset, which means there will be one instance of Table 9 for each of the 385 utterances, and the percentage values are computed with regard to the overall number of tokens (e.g., $\sim$ 5200 for each speaker, as shown in Table 4.) IoU scores are grouped in a phoneme-wise fashion per speaker, however.

A summary of the expected goals for each numeric analysis is as follows:

- The lower the phone boundary values, i.e., closer to zero, the better;
- The higher the IoU score, i.e., closer to one, the better;
- The higher the percentage of phonetic tokens aligned at a lower threshold value (in milliseconds), i.e., 100% below the 10ms threshold, the better.

## 5 Results and discussion

Results for the phone boundary metric will be reported in terms of a tolerance threshold that shows the how many phonetic tokens were more precisely aligned with respect to the manual alignments. Besides, in order to support the phone boundary evaluation, the intersection over union metric was also computed in forced alignments values against the reference ones, and results will be shown in a per-phoneme basis for both speakers from the evaluation dataset.

For IoU, however, only the most accurate results we have achieved will be discussed in detail for the sake of simplification, but as the values seem to follow a relatively consistent pattern across all systems, Appendix 1 shows the complete graphical results for all HTK-, MFA- and UFPAlign-based acoustic models.

### 5.1 Phone boundary

Numerical values, in milliseconds, are presented in Tables 10 and 11 for the female and male portions of the evaluation dataset, respectively. The best ones are highlighted in bold.

**Table 10** Results for the female dataset regarding the cumulative percentage below a tolerance threshold, in milliseconds, of the differences between forced aligned audio and ground-truth phonemes, also known as phone boundary

| **Toolkit** | **Cumulative tolerance** | | | |
|---|---|---|---|---|
| | **< 10 ms** | **< 25 ms** | **< 50 ms** | **< 100 ms** |
| UFPAlign (HTK) | 31.40% | 63.94% | 88.19% | 97.08% |
| EasyAlign | 36.59% | 78.12% | 94.06% | 98.91% |
| MFA ( A ) | 39.34% | 75.99% | 87.77% | 95.65% |
| MFA (T&A) | 37.65% | 78.69% | 95.16% | 99.08% |
| UFPAlign (mono) | 47.47% | 87.70% | 97.55% | 99.57% |
| UFPAlign (tri-Δ) | **50.44%** | **89.88%** | **98.34%** | 99.62% |
| UFPAlign (tri-LDA) | 47.48% | 89.22% | 98.27% | 99.76% |
| UFPAlign (tri-SAT) | 45.69% | 88.20% | 98.15% | 99.77% |
| UFPAlign (TDNN-F) | 34.41% | 75.94% | 97.61% | **99.87%** |

Notations on MFA stand for align-only (A) and train-and-align (T&A) procedures, while on UFPAlign, they denote either the nature of the toolkit or the acoustic model

**Table 11** Results for the male dataset regarding the cumulative percentage below a tolerance threshold, in milliseconds, of the differences between forced aligned audio and ground-truth phonemes, also known as phone boundary

| **Toolkit** | **Cumulative tolerance** | | | |
|---|---|---|---|---|
| | **< 10 ms** | **< 25 ms** | **< 50 ms** | **< 100 ms** |
| UFPAlign (HTK) | 30.73% | 62.45% | 86.55% | 96.42% |
| EasyAlign | 31.53% | 67.51% | 89.69% | 96.95% |
| MFA ( A ) | 32.81% | 64.85% | 78.49% | 90.61% |
| MFA (T&A) | 45.12% | 83.34% | **97.23%** | 99.66% |
| UFPAlign (mono) | 43.51% | 83.42% | 96.29% | 99.42% |
| UFPAlign (tri-Δ) | **46.28%** | **85.55%** | 97.13% | 99.74% |
| UFPAlign (tri-LDA) | 43.49% | 84.50% | 97.19% | 99.74% |
| UFPAlign (tri-SAT) | 42.14% | 83.51% | 97.19% | 99.78% |
| UFPAlign (TDNN-F) | 32.02% | 70.62% | 96.65% | **99.94%** |

Notations on MFA stand for align-only (A) and train-and-align (T&A) procedures, while on UFPAlign they denote either the nature of the toolkit or the acoustic model

As far as MFA train-and-align (T&A) feature is concerned, roughly only 1% of phoneme tokens aligned by Kaldi-based aligners are off the 100 ms tolerance, against 3% of tokens aligned by HTK-based tools. In fact, approximately 96–97% of phonemes were under the 50 ms tolerance when aligned by acoustic models trained with MFA and UFPAlign, considering an average of all models. Unfortunately, this is not true for MFA's pre-trained model for Brazilian Portuguese (in align-only mode), which on the other hand, for larger tolerance threshold values, performed a little worse than HTK.

Among HTK-based aligners, EasyAlign performed best considering all statistics and tolerance thresholds for both male and female speakers. However, as already pointed out in [3], the same ground-truth dataset used for evaluation in this work was also used to train the BP acoustic model shipped with EasyAlign, so this might have had some bias when comparing it to UFPAlign. Overall, UFPAlign (HTK) achieved very similar values across metrics for both speakers of the dataset, while EasyAlign's behavior shows a greater accuracy on the female voice. Nevertheless,

**Table 12** Mean and median of IoU scores over all phonemes for both speakers

| Toolkit | Female | | Male | |
|---|---|---|---|---|
| | Mean | Median | Mean | Median |
| UFPAlign (HTK) | 0.562 | 0.600 | 0.558 | 0.597 |
| EasyAlign | 0.634 | 0.686 | 0.578 | 0.625 |
| MFA ( A ) | 0.650 | 0.723 | 0.567 | 0.674 |
| MFA (T&A) | 0.678 | 0.722 | 0.691 | 0.736 |
| UFPAlign (mono) | 0.711 | 0.755 | 0.686 | 0.730 |
| UFPAlign (tri-Δ) | **0.729** | **0.769** | **0.704** | **0.743** |
| UFPAlign (tri-LDA) | 0.717 | 0.751 | 0.694 | 0.726 |
| UFPAlign (tri-SAT) | 0.710 | 0.743 | 0.690 | 0.721 |
| UFPAlign (TDNN-F) | 0.600 | 0.645 | 0.563 | 0.624 |

Best results are highlighted in bold

the parcel of phonetic tokens whose difference to the manual segmentation was less than 10 ms stayed below the 40% even for EasyAlign.

In align-only (A) mode, MFA models performed slightly better than EasyAlign's until 10 ms, but increasingly worse for larger values of tolerance for both male and female speakers. These poor results may be due to the nature of the dataset used to generate MFA's pre-trained acoustic models (GlobalPhone [34]), which contains only 22 h of transcribed audio. In contrast, training and aligning (T&A) on the same evaluation dataset with MFA proved better than HTK for the male speaker, and the results are similar for the female speaker.

The monophone- and triphone-based GMM models we trained with Kaldi for UFPAlign achieved the best performance with respect to phone boundary when compared to both MFA and HTK-based aligners. On average, approximately, 45% of tokens were accurately aligned within the 10 ms margin for all GMM models. Mean and median values are the lowest (except for tri-SAT on the male dataset, which was greater than MFA's T&A) and at most ~4 ms distant from each other. With respect to the speakers' gender, UFPAlign (Kaldi) performed approximately 4% better for the woman's voice until the 50 ms of tolerance, and about 2 ms more accurate according to the average mean.

Finally, TDNN-F simulation was definitely disappointing. We expected that results from a `nnet3` DNN-based setup would be at least similar to GMM-based ones, as it was in [4] with `nnet2`, but cumulative tolerance values were instead just slightly better than EasyAlign. The discussion Section sheds some light on possible reasons, as well as further evidence. Therefore, even though one can say that the best result was achieved by tri-delta (Δ) models on both male and female datasets, since it holds the rows with most boldface values in Tables 10 and 11 (except MFA was better off after 50 ms on the man's voice, but the values compared to UFPAlign's tri-Δ model are fairly and virtually the same), holding the greatest percentage of tokens more accurately aligned under 10 ms, we would rather prefer to state that all GMM-based AMs in UFPAlign achieved similar results.

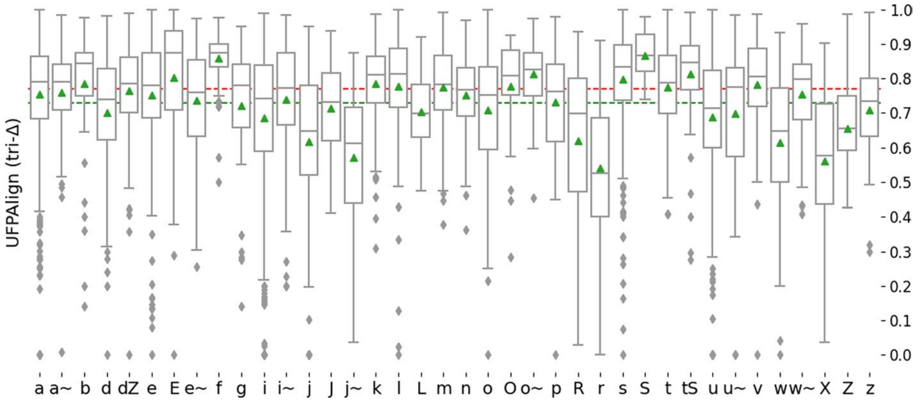

**Fig. 10** IoU values per phoneme on the utterances of the female speaker regarding the tri-Δ acoustic model trained with Kaldi

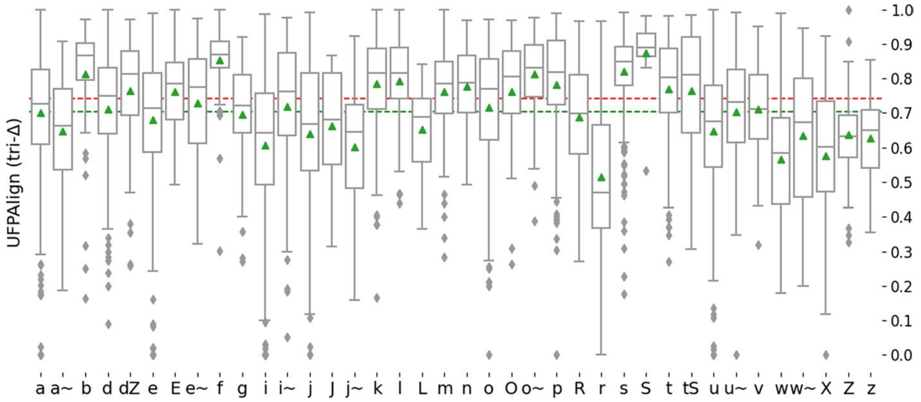

**Fig. 11** IoU values per phoneme on the utterances of the male speaker regarding the tri-Δ acoustic model trained with Kaldi

### 5.2 Intersection over union

Table 12 shows the average mean and median values for all acoustic models with respect to the IoU metric. As it can be seen, on average, the pattern already seen during phone boundary evaluation is maintained: Kaldi GMM-based models perform better overall, MFA train and align feature achieves close results, and finally, HTK-based aligners, MFA in align-only mode, and Kaldi's chain TDNN-F model are the worse ones.

Once again, the HMM-GMM tri-delta (Δ) model trained with Kaldi was the overall winner, even though all GMM-based models also achieved pretty similar results. Moreover, the median value is slightly higher (∼ +0.04), possibly due to a couple of outliers that may have dragged the average mean down.

Figures 10 and 11 show boxplots on the values achieved for the tri-Δ model on the female and male speakers, respectively, in a per-phoneme fashion. Horizontal, dashed lines represent the mean and median values, green triangles are the per-phoneme mean, and gray diamonds depict the outliers.

Some curious patterns can be extracted, though. Others do make sense indeed. For instance, the IoU values for the fricatives /f/, /s/ and /S/, as well as for the open and nasal realizations of grapheme "o," /O/ and /o~/, respectively, and plosives in general

**Table 13** Phone boundary results for the female dataset

| TDNN-F model | Features | Cumulative tolerance | | | |
|---|---|---|---|---|---|
| | | < 10 ms | < 25 ms | < 50 ms | < 100 ms |
| Chain | MFCCs + i-Vectors | 33.09% | 70.49% | 92.37% | 99.08% |
| Chain | MFCCs | 32.24% | 68.76% | 91.84% | 99.02% |
| Chain-free | MFCCs + i-Vectors | 47.05% | 83.46% | 96.08% | 99.36% |
| Chain-free | MFCCs | **48.78%** | **85.30%** | **96.80%** | **99.64%** |

are very high on both speakers, even though they seem slightly higher for the male speaker. For phonemes /R/, /r/, and /X/, on the other hand, which all map to the same grapheme "r" in BP, the accuracy was very poor, especially for the female speaker.

Some low IoU values for phonemes /i/, /j/ and /j~/ also draw attention. The latter, a nasal semivowel for grapheme "i," appears lots of times in merged phones, which may indicate something to be looked upon with more care at the M2M procedure. Perhaps an unrelated event, /w/ and /w~/, both semivowels for grapheme "o," also got below-average scores that are easy to see.

For a visualization of boxplots for all AMs, the reader is referred to Appendix 1.

### 5.3 Discussion

A possible reason for such a difference between HTK- and Kaldi-based aligners might be that HTK uses Baum-Welch algorithm for training HMMs while Kaldi uses Viterbi training [57]. On the other hand, among Kaldi models, tri-Δ stands out as being virtually the best one. However, with just a ~1–3% difference in tolerance, and ~0.02 difference in IoU scores, we cannot tell whether it is significant enough to classify one model into being better than the others, as they appear pretty close at glance. The linear sequence of model training just does not result in lower errors in phonetic boundaries as it resulted in lower word error rates for speech recognition [55].

The poorest results were produced by the TDNN-F, which needs careful investigation. Data insufficiency could have been the issue in the first place, as ~171 h of training data are far from the ideal volume to train a neural network efficiently. Other reasons include the use of frame subsampling, since Fig. 9 proves that time alignments (in orange) are always a multiple of 3; and the modified topology of HMMs which the TDNN-F trains upon, also known as chain model [74], which is further discussed with preliminary results in Sect. 5.3.1.

Moreover, navigating through all the burden to train a DNN model with Kaldi (which requires at least one GPU card) may not be the more appropriate move if the final task's goal is to align phonemes rather than to recognize speech. As MFA seem to have dropped support to DNN models, and our previous results with a nnet2 neural network setup only took tolerance values so far as to match tri-Δ models [4]. Nevertheless, conjectures still need to be experimented to remove doubts and prove hypothesis empirically.

**Table 14** Phone boundary results for the male dataset

| TDNN-F model | Features | Cumulative tolerance | | | |
|---|---|---|---|---|---|
| | | < 10 ms | < 25 ms | < 50 ms | < 100 ms |
| Chain | MFCCs + i-Vectors | 34.14% | 69.51% | 91.42% | 99.23% |
| Chain | MFCCs | 33.67% | 68.84% | 91.38% | 99.25% |
| Chain-free | MFCCs + i-Vectors | 46.60% | 83.14% | 96.16% | 99.34% |
| Chain-free | MFCCs | **46.75%** | **83.59%** | **96.87%** | **99.68%** |

#### *5.3.1 Investigation on TDNN-F chain models*

To further investigate some of the hypotheses to why the neural network performed so poorly in comparison with GMM models, we trained another four TDNN-F-based models, but this time varying some of the input features as well as the topology of the HMMs the neural network trains upon. The insight for the latter comes from the experience of others on the goodness of pronunciation[5] task in Kaldi.

We refer back to Fig. 3, where there is a block called "build tree." This stage recreates HMMs for the tri-SAT model that contain a single-state instead of the traditional three-state, left-to-right topology that is used to train the GMM-based models [74]. The decision tree that models senones is therefore also modified.

Tables 13 and 14 show the results for the female and male speakers, respectively, from four additional models trained over the same tri-SAT GMM model, with (chain) and without (chain-free) the use of modified HMM topology. At the input of the network, we also tested the high-resolution MFCCs with and without the i-Vector features stacked.

As it can be seen, even though i-Vectors seem to help chain models (∼1–2%), removing them from the training stage in chain-free models actually does improve results, even if sometimes just marginally (∼0.1–2%.) Nevertheless, the difference is not significant as the comparison chain vs. chain-free: the absolute gains at the smallest thresholds are of ∼15% and ∼13% for the female and male speakers, respectively. Also, in spite of the clear improvement with respect to previous results, the values for phone boundary are still behind the GMM-based models.

## 6 Conclusion

This paper presented contributions for the problem of forced phonetic alignment in Brazilian Portuguese (BP). An update to UFPAlign [4] was offered by providing adapted Kaldi recipes for training acoustic models on BP datasets, as well as properly releasing all the acoustic models for free under an open-source license on the GitHub of the FalaBrasil Group.[6] UFPAlign works either via command line (Linux) or in a graphical interface as a plugin to Praat. Up-to-date phonetic and syllabic dictionaries created over a list of 200,000 words for BP are also provided, as well as standalone grapheme-to-phoneme and syllabification systems for handling out-of-vocabulary words.

For evaluation, a comparison among the Kaldi-based acoustic models trained with an updated version of the scripts from [4] was performed, as well as a comparison to an

[5] https://github.com/kaldi-asr/kaldi/tree/master/egs/gop_speechocean762.

[6] https://github.com/falabrasil.

outdated HTK-based version of UFPAlign from [3]. Results regarding the absolute difference between forced and manual aligned utterances (phone boundary metric) and the overlap rate (intersection over union, or IoU) showed that the HTK-based aligner performed worse when compared to any of the Kaldi-based models, and that our acoustic models we trained from scratch performed better than MFA's pre-trained models.

### 6.1 Future work

As future work, there are a couple of experiments to be investigated. The simplest one would be to train GMM-based tri-Δ, tri-LDA, tri-SAT and even monophone-based acoustic models with a higher number of Gaussian mixtures per senone. We are already training DNNs on the top of tri-Δ and other triphone-based models other than the default tri-SAT, since that was the one that yielded the most accurate results according to phone boundary, but with smaller datasets the results did not seem to improve. Besides, training a DNN on the top of context-independent monophones also does not seem to help.

We also plan on testing on a new dataset of hand-aligned utterances spoken by a single male speaker that we recently had access to. Unfortunately that only leaves us with a three-speaker test set in total, but at least the volume of data is much greater than it once was approximately one and a half hour of speech whose phonemes' times were annotated by a phonetician.

Aiming at creating a more trustworthy mapping between phone sets, there could be an estimation of the durations of phones from the evaluation dataset in order to avoid attributing linearly spaced time stamps after the splitting procedure during M2M mapping. This is probably more complex as coarticulation between phones always occur, and we are aware that the volume of hand-aligned annotation per speaker may note be enough to perform a biphone analysis, for example. However, we plan to compute one overall duration per phone considering an average of all occurrences of that single (mono) phone to see whether automatically inferred boundaries vary.

Regarding the DNN, some preliminary results already suggest that chain models [74] are not well suited for phonetic alignment, and that the input features do not affect phone boundary values by a large margin. Even so, splicing cepstral features with LDA would also be a valid test. In addition, the TDNN-F setup has not been altered from Mini-librispeech's default recipe, which means some parameters such as layer dimension, number of layers, context width, and the application of frame subsampling could still undergo tuning for different languages of different training dataset sizes. It seems natural that the research shall continue now on chain-free models. Finally, other architectures like LSTMs should have its use evaluated.

At last, although UFPAlign can be used as a plugin to Praat, we plan in the future to train models compatible with MFA or Gentle under the same licensing, as to avoid open-source competition. Unfortunately, such effort did not work by the time of this submission, but as both codebases are more well documented and well maintained, they may potentially cover a broader community. The provision of a train-and-align feature for UFPAlign is also an ongoing plan.

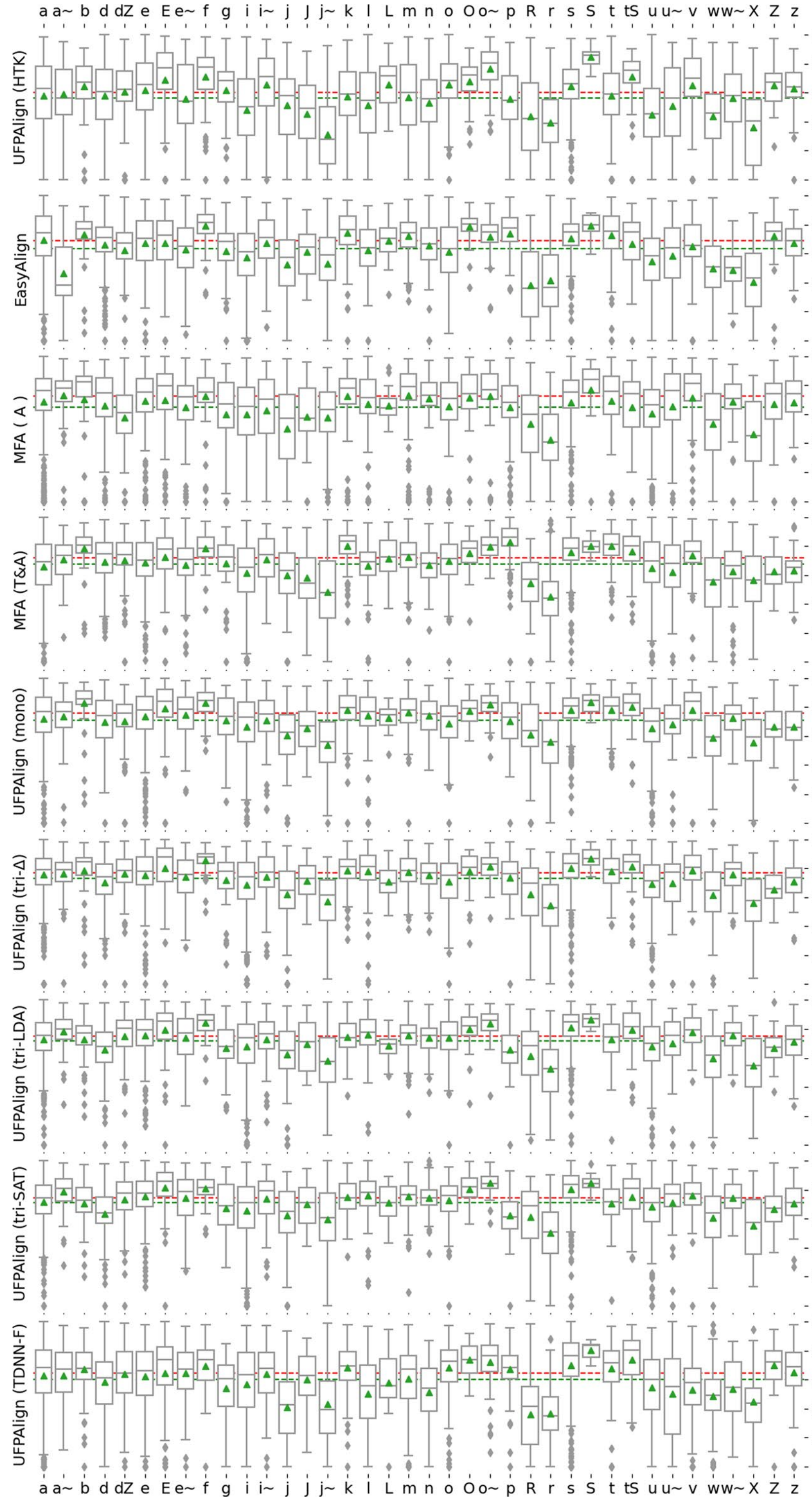

**Fig. 12** IoU values per phoneme on the utterances of the female speaker

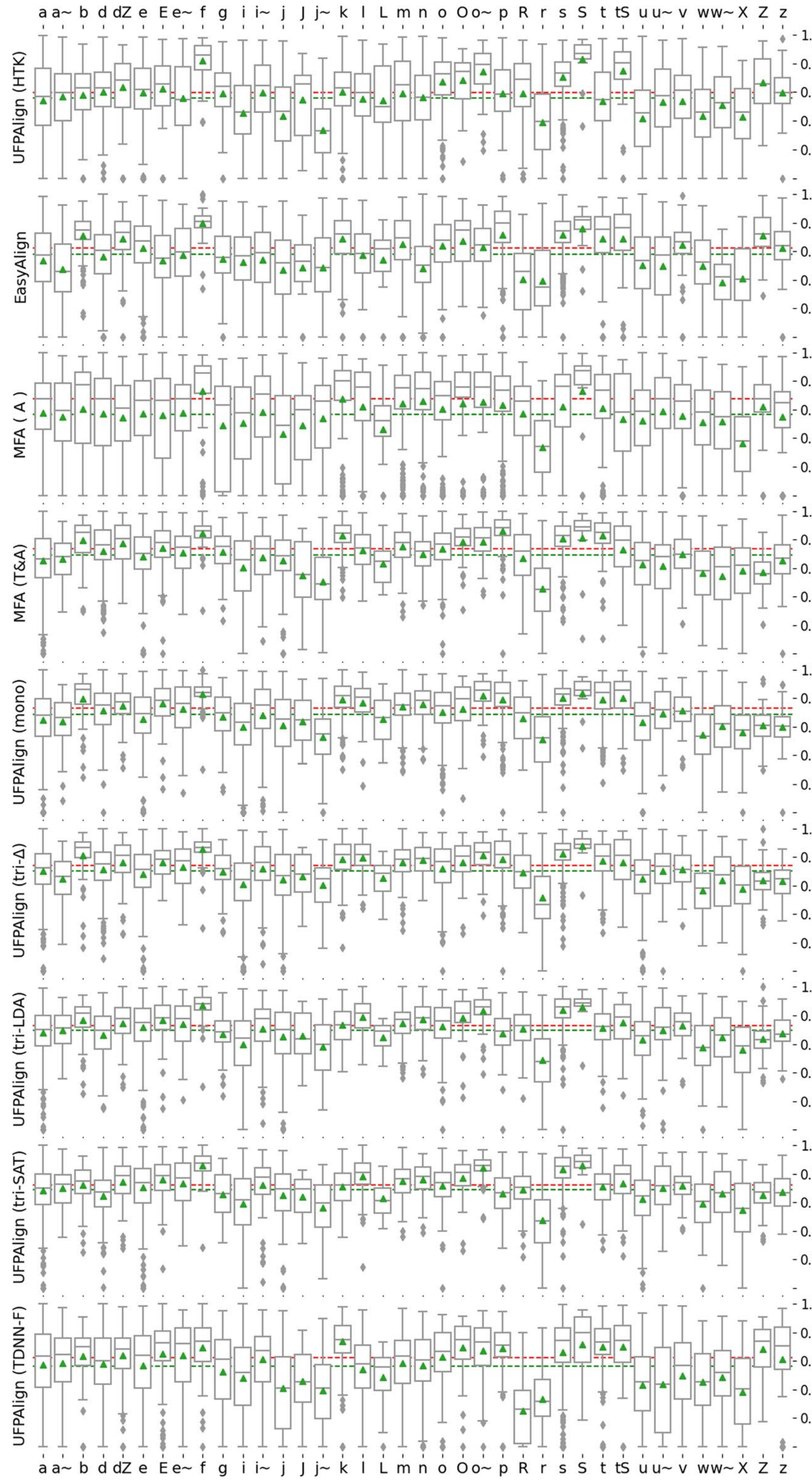

**Fig. 13** IoU values per phoneme on the utterances of the male speaker

## Appendix 1: Phoneme-wise analysis on intersection over union

Figures 12 and 13 show boxplots on the values achieved for the female and male speakers, respectively, in a per-phoneme fashion for all acoustic models evaluated. The horizontal, dashed lines on both plots are the average mean (green) and median (red) across all phonemes, which are previously summarized in Table 12. Furthermore, green triangles and gray diamonds represent the mean and the outliers for each phoneme.

The boxplots provide a great deal of information that can be overwhelming at glance. Still, they offer a great tool to analyze the behavior across models and phonemes in general. For a more in-depth discussion on the overall "best" performant system, the reader is referred to Sect. 5.2. We found that most of the patterns already discussed can also be extended and visualized on results for the remaining forced aligners.

### Abbreviations

AGPL: Affero GNU General Public License; AM: Acoustic model; ASR: Automatic speech recognition; BP: Brazilian Portuguese; CETUC: Centro de Estudos em Telecomunicações; CLI: Command line interface; CMU: Carnegie Mellon University; CTM: Time marked conversation; DTW: Dynamic time warping; E2E: End-to-end; FB: FalaBrasil; FST: Finite state transducer; G2P: Grapheme-to-phoneme; GMM: Gaussian mixture models; GPL: GNU General Public License; GPU: Graphic Processing Unit; GUI: Graphical user interface; HMM: Hidden Markov models; HTK: Hidden Markov model toolkit; IoU: Intersection over union; IPA: International phonetic alphabet; JSON: JavaScript object notation; LDA: Linear discriminant analysis; ASpIRE: Automatic SPeech recognition In Reverberant Environments; CAPES: Coordenação de Aperfeiçoamento de Pessoal de Nível Superior; CNPq: Conselho Nacional de Desenvolvimento Científico e Tecnológico; fMLLR: Feature space maximum likelihood linear regression; NILC: Núcleo Interinstitucional de Linguística Computacional; LDC: Linguistic data consortium; LM: Language model; LSTM: Long short-term memory; MAUS: Munich automatic segmentation system; MFA: Montreal forced aligner; MFCC: Mel frequency cepstral coefficients; MIT: Massachusetts Institute of Technology; MPL: Mozilla Public License; NLP: Natural language processing; P2FA: Penn phonetics lab forced aligner; PCM: Pulse code modulation; RM: Resource management; SAMPA: Speech assessment methods phonetic alphabet; SAT: Speaker adaptive training; SCOTUS: Supreme Court of the United States; TDNN-F: Factorized Time Delay Neural Network; TDNN: Time Delay Neural Network; TIMIT: Texas instruments and MIT; TTS: Text-to-speech; UFPA: Universidade Federal do Pará; VAD: Voice activity detection; VERO: VERificador Ortográfico; WSJ: Wall Street Journal.

### Acknowledgements

We gratefully acknowledge NVIDIA Corporation with the donation of the Titan Xp GPU used for this research. The authors would also like to thank CAPES research funding agency and, on behalf of all contributors of this project, PROPESP/UFPA and FAPESPA (under grant 001/2020, process 2019/583359) for the financial support.

### Authors' contributions

All authors contributed to this research, including the design of the simulations and analyses of the results. CB adapted Kaldi scripts to work with data in Brazilian Portuguese, prepared the evaluation dataset to fit a uniform pattern for comparison, generated results and wrote the first version of the manuscript. ALD worked in the Praat plugin and continually revised the manuscript, while NN contributed to substantial revisions of the text. All authors read and approved the final manuscript.

### Authors information

Cassio Batista received his B.S. degree in computer engineering from the Federal University of Pará (UFPA), Brazil, in 2016, and his M.S. degree in computer science in 2017, at the same institution. He is currently pursuing Ph.D. from the Computer Science Graduate Program at Federal University of Pará. His current research areas include speech and natural processing for Brazilian Portuguese.

Ana Larissa Dias received her B.S. degree in computer engineering from the Federal University of Pará (UFPA), Brazil, in 2018. Currently, she is a master's student in Computer Science at the same institution. Her research areas include speech recognition and processing for Brazilian Portuguese.

Nelson Neto received his B.S. degree in electrical engineering from the Federal University of Pará (UFPA), Brazil, in 2000, his MS degree in electrical engineering in 2006, and his Ph.D. in electrical engineering in 2011, at the same institution. He is currently a Professor in the Computer Science Graduate Program at UFPA. His research areas include speech recognition, speech synthesis and natural language processing for Brazilian Portuguese.

### Funding

This work is sponsored by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) and Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) through the provision of graduate scholarship fundings.

### Availability of data and materials

From the speech corpora used to train the acoustic models, CETUC, LapsBenchmark, Constitution and Consumer Protection Code datasets are freely available in https://github.com/falabrasil/speech-datasets. LapsStory is not publicly available for licensing issues, since it was extracted from private audio books. Spoltech and West Point can be purchased from Linguistic Data Consortium (LDC). As for the evaluation dataset of hand-aligned utterances, it was ceded by the group and cannot be released, but can be requested. Language model and lexicon files can be found in https://gitlab.com/fb-nlp under the MIT license.

## Declarations

### Competing interests

The authors declare that they have no competing interests.

Received: 16 April 2021 Accepted: 3 February 2022
Published online: 19 February 2022

### References

1. J.-P. Goldman, Easyalign: an automatic phonetic alignment tool under praat, in *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, pp. 3233–3236 (2011)
2. M. McAuliffe, M. Socolof, S. Mihuc, M. Wagner, M. Sonderegger, Montreal forced aligner: trainable text-speech alignment using kaldi, in *Proceedings of Interspeech*, pp. 498–502 (2017). https://doi.org/10.21437/Interspeech.2017-1386
3. G. Souza, N. Neto, An automatic phonetic aligner for Brazilian Portuguese with a Praat interface, in *Computational Processing of the Portuguese Language*. ed. by J. Silva, R. Ribeiro, P. Quaresma, A. Adami, A. Branco (Springer, Cham, 2016), pp. 374–384
4. A.L. Dias, C. Batista, D. Santana, N. Neto, Towards a free, forced phonetic aligner for Brazilian Portuguese using Kaldi tools, in *Intelligent Systems*. ed. by R. Cerri, R.C. Prati (Springer, Cham, 2020), pp. 621–635
5. S. Young, D. Ollason, V. Valtchev, P. Woodland, *The HTK Book*. Cambridge University Engineering Department, version 3.4 (Cambridge, 2006)
6. D. Povey, A. Ghoshal, G. Boulianne, N. Goel, M. Hannemann, Y. Qian, P. Schwarz, G. Stemmer, The Kaldi speech recognition toolkit, in *IEEE 2011 Workshop* (2011)
7. P. Boersma, D. Weenink, Praat: Doing Phonetics by Computer (Version 6.1.15) [computer Program]. https://www.fon.hum.uva.nl/praat/
8. H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, S. Savarese, Generalized intersection over union: a metric and a loss for bounding box regression, in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 658–666 (2019). https://doi.org/10.1109/CVPR.2019.00075
9. K. Atkinson, GNU Aspell. https://aspell.net
10. A. Siravenha, N. Neto, V. Macedo, A. Klautau, Uso de regras fonológicas com determinação de vogal tônica para conversão grafema-fone em português brasileiro (2008). https://gitlab.com/fb-nlp/nlp-generator
11. N. Neto, C. Patrick, A. Klautau, I. Trancoso, Free tools and resources for Brazilian Portuguese speech recognition. J. Braz. Comput. Soc. **17**(1), 53–68 (2011). https://doi.org/10.1007/s13173-010-0023-1
12. N. Neto, W. Rocha, G. Sousa, An open-source rule-based syllabification tool for Brazilian Portuguese. J. Braz. Comput. Soc. (2015). https://doi.org/10.1186/s13173-014-0021-9
13. Pettarin, A.: Aeneas. https://github.com/readbeyond/aeneas
14. Mozilla: DSAlign: DeepSpeech Based Forced Alignment Tool. https://github.com/mozilla/DSAlign
15. Rosenfelder, I., Fruehwald, J., Evanini, K., Seyfarth, S., Gorman, K., Prichard, H., Yuan, J.: *FAVE: Forced Alignment and Vowel Extraction*. https://github.com/JoFrhwld/FAVE/
16. Ochshorn, R.M., Hawkins, M.: *Gentle Forced Aligner*. https://github.com/lowerquality/gentle
17. M. Tu, A. Grabek, J. Liss, V. Berisha, Investigating the role of l1 in automatic pronunciation evaluation of l2 speech, in *Proceedings of Interspeech 2018*, pp. 1636–1640 (2018). https://doi.org/10.21437/Interspeech.2018-1350
18. R. Fromont, J. Hay, LaBB-CAT: an annotation store, in *Proceedings of the Australasian Language Technology Association Workshop* (Dunedin, 2012), pp. 113–117. https://www.aclweb.org/anthology/U12-1015
19. F. Schiel, Automatic phonetic transcription of non-prompted speech, in *Proceedings of the ICPhS* (San Francisco, 1999), pp. 607–610
20. J. Yuan, M. Liberman, Speaker identification on the *Scotus corpus*. J. Acoust. Soc. Am. **123**(5), 3878–3881 (2008). https://doi.org/10.1121/1.2935783
21. K. Gorman, J. Howell, M. Wagner, Prosodylab-aligner: a tool for forced alignment of laboratory speech. Can. Acoust. **39**(3), 192–193 (2011)
22. A. Katsamanis, M.P. Black, P. Georgiou, L. Goldstein, S. Narayanan, SailAlign: robust long speech-text alignment, in *Proceedings of Workshop on New Tools and Methods for Very-Large Scale Phonetics Research* (2011)
23. B. Bigi, SPPAS: multi-lingual approaches to the automatic annotation of speech. J. Int. Soc. Phonetic Sci. **111–112**, 54–69 (2015)
24. F. Malfrère, T. Dutoit, *High-Quality Speech Synthesis for Phonetic Speech Segmentation*, vol. 3329 (1997)
25. F. Schiel, The Munich Automatic Segmentation System (MAUS). https://www.bas.uni-muenchen.de/Bas/BasMAUS.html
26. R. Weide, The CMU Pronouncing Dictionary (version 0.7b). http://www.speech.cs.cmu.edu/cgi-bin/cmudict

27. R. Fromont, Forced alignment of different language varieties using LaBB-CAT, in *Proceedings of the 19th International Congress of Phonetic Sciences (ICPhS)* (Melbourne, 2019), pp. 1327–1331. https://www.aclweb.org/anthology/U12-1015
28. A. Lee, T. Kawahara, K. Shikano, Julius-an open source real-time large vocabulary recognition engine **3**, 1691–1694 (2001)
29. A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates, A.Y. Ng, *Deep Speech: Scaling Up End-to-End Speech Recognition* (2014). arXiv:1412.5567
30. T.F. Smith, M.S. Waterman, Identification of common molecular subsequences. J. Mol. Biol. **147**(1), 195–197 (1981). https://doi.org/10.1016/0022-2836(81)90087-5
31. V. Peddinti, D. Povey, S. Khudanpur: a time delay neural network architecture for efficient modeling of long temporal contexts, in *Proceedings of Interspeech*, pp. 3214–3218 (2015)
32. D. Povey, G. Cheng, Y. Wang, K. Li, H. Xu, M. Yarmohammadi, S. Khudanpur, Semi-orthogonal low-rank matrix factorization for deep neural networks, in *Proceedings of Interspeech 2018*, pp. 3743–3747 (2018). https://doi.org/10.21437/Interspeech.2018-1417. http://dx.doi.org/10.21437/Interspeech.2018-1417
33. V. Panayotov, G. Chen, D. Povey, S. Khudanpur, Librispeech: an ASR corpus based on public domain audio books, in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5206–5210 (2015). https://doi.org/10.1109/ICASSP.2015.7178964
34. T. Schultz, N.T. Vu, T. Schlippe, Globalphone: a multilingual text speech database in 20 languages, in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 8126–8130 (2013). https://doi.org/10.1109/ICASSP.2013.6639248
35. C. Allauzen, M. Riley, J. Schalkwyk, W. Skut, M. Mohri, *Openfst: A General and Efficient Weighted Finite-state Transducer Library*, vol. 4783, pp. 11–23 (2007). https://doi.org/10.1007/978-3-540-76336-9_3
36. L.R. Rabiner, A tutorial on hidden Markov models and selected applications in speech recognition. Proc. IEEE **77**(2), 257–286 (1989). https://doi.org/10.1109/5.18626
37. G. Hinton, L. Deng, D. Yu, G.E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T.N. Sainath, B. Kingsbury, Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. IEEE Signal Process. Mag. **29**(6), 82–97 (2012). https://doi.org/10.1109/MSP.2012.2205597
38. A. Georgescu, H. Cucu, C. Burileanu, Kaldi-based DNN architectures for speech recognition in romanian, in *2019 International Conference on Speech Technology and Human–Computer Dialogue (SpeD)*, pp. 1–6 (2019). https://doi.org/10.1109/SPED.2019.8906555
39. Vesely, K., et al., Sequence-discriminative training of deep neural networks, in *INTERSPEECH 2013*, pp. 2345–2349 (2013)
40. X. Zhang, J. Trmal, D. Povey, S. Khudanpur, Improving deep neural network acoustic models using generalized maxout networks, in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 215–219 (2014). https://doi.org/10.1109/ICASSP.2014.6853589
41. D. Povey, X. Zhang, S. Khudanpur, *Parallel training of DNNs with Natural Gradient and Parameter Averaging* (2015). arXiv:1410.7455
42. V. Peddinti, Y. Wang, D. Povey, S. Khudanpur, Low latency acoustic modeling using temporal convolution and LSTMs. IEEE Signal Process. Lett. **25**(3), 373–377 (2018). https://doi.org/10.1109/LSP.2017.2723507
43. D. Gibbon, R. Moore, R. Winski, *SAMPA Computer Readable Phonetic Alphabet*. https://www.phon.ucl.ac.uk/home/sampa/
44. PCD Legal: PCD Legal: Acessível Para Todos. http://www.pcdlegal.com.br/
45. LDC: CSLU: Spoltech Brazilian Portuguese Version 1.0. https://catalog.ldc.upenn.edu/LDC2006S16
46. LDC: West Point Brazilian Portuguese Speech. https://catalog.ldc.upenn.edu/LDC2008S04
47. PUC-Rio: Centro de Estudos em Telecomunicações (CETUC). http://www.cetuc.puc-rio.br/
48. A. Stolcke, Srilm—an extensivle language modeling toolkit, in *Proceedings of the 7th International Conference on Spoken Language Processing (ICSLP)*, vol. 2, pp. 901–904 (2002)
49. Interinstitutional Center for Computational Linguistics: CETENFolha Dataset. https://www.linguateca.pt/cetenfolha/index_info.html
50. J.J. Almeida, A. Simões, *Projecto Natura*. https://natura.di.uminho.pt/wiki/doku.php
51. R. Moura, LibreOffice's VERO Dictionary. https://github.com/LibreOffice/dictionaries/tree/master/pt_BR
52. GitHub: FrequencyWords. https://github.com/hermitdave/FrequencyWords
53. Opensubtitles.org: OpenSubtitles. https://www.opensubtitles.org/
54. D. Povey, *OpenSLR: Open Speech and Language Resources*. https://openslr.org/index.html
55. C. Batista, A.L. Dias, N. Sampaio Neto, Baseline acoustic models for brazilian portuguese using Kaldi tools, in *Proceedings of IberSPEECH*, pp. 77–81 (2018). https://doi.org/10.21437/IberSPEECH.2018-17
56. S. Davis, P. Mermelstein, Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. IEEE Trans. Acoust. Speech Signal Process. **28**(4), 357–366 (1980). https://doi.org/10.1109/TASSP.1980.1163420
57. S. Buthpitiya, I. Lane, J. Chong, A parallel implementation of viterbi training for acoustic models using graphics processing units, in *2012 Innovative Parallel Computing (InPar)*, pp. 1–10 (2012). https://doi.org/10.1109/InPar.2012.6339590
58. R.O. Duda, P.E. Hart, D.G. Stork, *Pattern Classification*, 2nd edn. (Wiley-Interscience, New York, 2000)
59. R.A. Gopinath, Maximum likelihood modeling with Gaussian distributions for classification, in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP*, vol. 2, pp. 661–6642 (1998). https://doi.org/10.1109/ICASSP.1998.675351
60. S.P. Rath, D. Povey, K. Veselý, J. Černocký, Improved feature processing for deep neural networks, in *Proceedings of Interspeech*, pp. 109–113 (2013). https://www.isca-speech.org/archive/interspeech_2013/i13_0109.html
61. M.J.F. Gales, Maximum likelihood linear transformations for hmm-based speech recognition. Comput. Speech Lang. **12**(2), 75–98 (1998). https://doi.org/10.1006/csla.1998.0043

62. T. Anastasakos, J. Mcdonough, R. Schwartz, J. Makhoul, A compact model for speaker-adaptive training, in *Proceedings of ICSLP*, pp. 1137–1140 (1996)
63. T. Anastasakos, J. McDonough, J. Makhoul, Speaker adaptive training: a maximum likelihood approach to speaker normalization, in *1997 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, pp. 1043–10462 (1997)
64. Y. Miao, H. Zhang, F. Metze, Speaker adaptive training of deep neural network acoustic models using i-vectors. IEEE/ACM Trans. Audio Speech Lang. Process. **23**(11), 1938–1949 (2015)
65. S. Guiroy, R. de Cordoba, A. Villegas, Application of the Kaldi toolkit for continuous speech recognition using Hidden–Markov models and deep neural networks, in *IberSPEECH'2016 On-line Proceedings, IberSPEECH 2016* (Lisboa, Portugal, 2016), pp. 187–196. https://iberspeech2016.inesc-id.pt/wp-content/uploads/2017/01/OnlineProceedings_IberSPEECH2016.pdf
66. I. Kipyatkova, A. Karpov, Dnn-based acoustic modeling for Russian speech recognition using kaldi, in *Speech and Computer* (Springer, Cham, 2016), pp. 246–253
67. T. Ko, V. Peddinti, D. Povey, S. Khudanpur, Audio augmentation for speech recognition, in *Proceedings of Interspeech* (2015)
68. N. Dehak, P.J. Kenny, R. Dehak, P. Dumouchel, P. Ouellet, Front-end factor analysis for speaker verification. IEEE Trans. Audio Speech Lang. Process. **19**(4), 788–798 (2011). https://doi.org/10.1109/TASL.2010.2064307
69. D. Snyder, D. Garcia-Romero, D. Povey, S. Khudanpur, Deep neural network embeddings for text-independent speaker verification, in *Proceedings of Interspeech 2017*, pp. 999–1003 (2017). https://doi.org/10.21437/Interspeech.2017-620. http://dx.doi.org/10.21437/Interspeech.2017-620
70. G. Strang, *Introduction to Linear Algebra*, 5th edn. (Wellesley-Cambridge Press, Wellesley, 2016)
71. X. Huang, A. Acero, H.-W. Hon, *Spoken Language Processing: A Guide to Theory, Algorithm, and System Development*, 1st edn. (Prentice Hall PTR, Upper Saddle River, 2001)
72. J.E. Shoup, Phonological aspects of speech recognition, in *Trends in Speech Recognition*, pp. 125–138 (1980)
73. S. Jiampojamarn, G. Kondrak, T. Sherif, Applying many-to-many alignments and hidden Markov models to letter-to-phoneme conversion, in *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, Association for Computational Linguistics (Rochester, New York, 2007), pp. 372–379. http://www.aclweb.org/anthology/N/N07/N07-1047
74. D. Povey, V. Peddinti, D. Galvez, P. Ghahremani, V. Manohar, X. Na, Y. Wang, S. Khudanpur, Purely sequence-trained neural networks for ASR based on lattice-free MMI, in *Proceedings of Interspeech 2016*, pp. 2751–2755 (2016). https://doi.org/10.21437/Interspeech.2016-595