A Practical Analysis of a Forced Alignment Pipeline:
Solving OOV Errors and Comparing Acoustic Models

Atharva Naitam
B.Tech Computer Engineering
St. Vincent Pallotti College of Engineering & Technology

November 7, 2025

Abstract

This report details the implementation of a speech-to-text alignment pipeline using the Montreal Forced Aligner (MFA). The objective was to find precise time boundaries for words and phonemes in raw audio. The project documents the entire engineering process, from initial data validation to a critical failure and the implementation of a custom-trained solution.

The initial alignment attempt using the standard `english_us_arpa` model failed with a fatal `AssertionError`. This was traced to 5 Out-of-Vocabulary (OOV) words being assigned invalid 0-duration alignments. To solve this, a custom Grapheme-to-Phoneme (G2P) model was trained to generate valid pronunciations for these words, which successfully resolved the error.

Finally, this report presents a comparative analysis between the `english_us_arpa` model and the newer `english_mfa` model, concluding that the older `english_us_arpa` model and its G2P counterpart produced significantly more accurate alignments for this specific dataset. An automated batch script was also developed to make the successful 6-step pipeline reproducible.

1 Key Concepts in Speech Alignment

- **Phoneme:** The smallest unit of sound in a language that can distinguish one word from another. For example, the word 'cat' consists of three phonemes: `K`, `AE`, and `T`. The goal of this assignment is to find the exact start and end time of each phoneme in the audio.
- **Acoustic Model (AM):** The "engine" of the aligner. An acoustic model is a statistical model (in this case, a GMM-HMM) that has been trained on thousands of hours of speech. Its job is to calculate the probability that a small "frame" of audio (e.g., 10ms) corresponds to a specific phoneme.
- **Pronunciation Dictionary:** A "lookup table" that maps orthographic words (like 'hello') to their phoneme sequences (like `HH AH0 L OW2`). This dictionary is the "glue" that connects the text transcript to the acoustic model.
- **TextGrid:** The output file format, readable by the program Praat. It is a simple text file that contains a list of time intervals (start, end) and their corresponding labels (the word or phoneme).
- **MFCC (Mel Frequency Cepstral Coefficients):** Raw audio waveforms are too complex to be used directly. The audio is converted into **MFCCs**, which are a standard feature in speech recognition. This process extracts the key acoustic features relevant to human speech (like formants) while discarding noise and non-linguistic information.

- **CMVN (Cepstral Mean and Variance Normalization):** This is a data-cleaning step. The features of an audio file can be very different based on the microphone, the room, or the speaker's vocal tract. CMVN normalizes all the features to have a similar mean and variance, making the speaker's voice consistent and removing the "channel effects."

2 Methodology and Analysis

The alignment was an iterative process. This section details the steps of the MFA pipeline, the failure encountered, and the custom-engineering solution.

2.1 Phase 1: Pipeline Execution and Initial Failure

The alignment process begins with a series of data preparation steps, visible in the MFA log:

1. **'Normalizing text...'**: MFA first cleans the input transcripts. This involves converting all text to uppercase and stripping punctuation to ensure the words in the transcript can be found in the pronunciation dictionary.
2. **'Generating MFCCs...'**: The raw `.wav` files are processed. As defined by the Kaldi toolkit, the audio is segmented into 25ms frames, and their MFCC features are computed.
3. **'Calculating CMVN...'**: The extracted MFCCs are normalized on a per-speaker basis to create a consistent set of features for the aligner.
4. **'Compiling training graphs...'**: MFA builds a "search space" (a Finite State Transducer) that represents all possible pronunciation paths based on the dictionary.

2.1.1 Key Observation: OOV Words

The initial validation of the data against the `english_us_arpa` dictionary found 5 Out-of-Vocabulary (OOV) words:

- `dukakis`, `politicize`, `maffy`, `wbur`, `melnicove`

2.1.2 Key Observation: Fatal Error

When the main alignment was run (`mfa align . english_us_arpa...`), the process crashed. The OOV words were assigned 0-duration alignments, which caused the TextGrid exporter to fail. This was the primary "error or mismatch" observed.

```
...
File "D:\miniconda3\envs\mfa\Lib\site-packages\kalpy\gmm\data.py", line 65
assert begin < end
AssertionError:
```

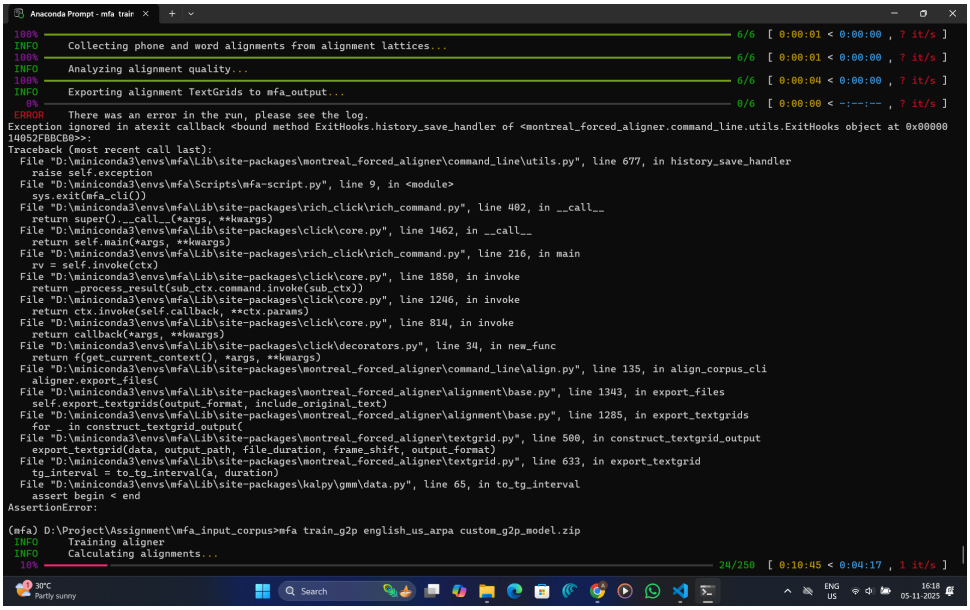

Figure 1: The initial alignment failed with an `AssertionError`, caused by a 0-duration interval for an OOV word.

2.2 Phase 2: Solution via Custom G2P Models

To solve the `AssertionError`, a valid pronunciation was required for each OOV word. This was achieved by training a custom **Grapheme-to-Phoneme (G2P) model**, which is a machine learning model that learns the relationship between a word's spelling (graphemes) and its sounds (phonemes).

2.2.1 G2P Model Architecture and Training

The G2P models in MFA are not neural networks, but **Weighted Finite-State Transducers (WFSTs)**. This architecture acts as a "probabilistic flowchart" that finds the lowest-cost (highest-probability) path from a sequence of letters to a sequence of sounds.

The training process (`mfa train_g2p`) uses the base dictionary (e.g., `english_us_arpa.dict`) as its "textbook."

1. **Alignment:** The model first aligns the graphemes and phonemes for all words in the dictionary (e.g., 'b-o-x' → 'B-AA-K-S').

2. **Training:** It then calculates the probabilities for all these letter-to-sound "rules" (e.g., what is the probability that 'c' is `K` vs. `S`?). These rules are "baked" into the final WFST model.

2.2.2 G2P Inference

Once trained, the G2P model ('custom_g2p_model.zip') was used to **predict** pronunciations for the 5 OOV words. The 'mfa g2p' command fed the spelling (e.g., 'dukakis') into the WFST, which returned the lowest-cost path: `D AH0 K AA1 K IH0 S`. These new pronunciations were used to create a `final_custom_dictionary.txt`, which successfully solved the alignment crash.

2.3 Phase 3: Comparative Analysis of Acoustic Models

A comparative analysis was performed to test the older `english_us_arpa` model against the newer `english_mfa` model.

My hypothesis was that the "newer" model might produce better results. To ensure a *fair test*, I could not use the same custom dictionary for both. A full, separate G2P pipeline was run for *both* models, using their respective base dictionaries as the training data. This process revealed that the two packages are not interchangeable, and the `english_us_arpa` model is conclusively superior for this dataset.

2.3.1 Key Finding 1: Phone Set and G2P Model Quality

The most significant finding was the vast difference in the underlying dictionaries, which directly impacted the G2P model quality.

- `english_us_arpa`**:** This package uses the **ARPABET** phone set. This set is very rich, as it includes stress markers (e.g., `AH0` for no stress, `AA1` for primary stress). The G2P model trained on this "textbook" learned complex rules and produced highly plausible pronunciations for OOV words (e.g., 'melnicove' as `M EH1 L N IH0 K OW2 V`).
- `english_mfa`**:** This package uses a different, simpler phone set (e.g., `@`, `i:`). This set does not encode lexical stress. The G2P model trained on this simpler data produced nonsensical and unusable pronunciations (e.g., 'melnicove' as `m i o: v`).

This demonstrates that the quality and richness of the base dictionary (the "textbook") is the most critical factor in training a G2P model that can successfully handle OOV words.

2.3.2 Key Finding 2: Dictionary Coverage

A secondary finding from the validation step was that the dictionaries have different word coverage. The `english_mfa` dictionary was missing the word `judgeships`, which **was** present in the `english_us_arpa` dictionary. This provided further evidence that the 'arpa' package was a better fit for this corpus.

2.3.3 Final Alignment Analysis

The final alignments in Praat (seen in Figures 2 and 3) are a direct consequence of the G2P quality. The `english_us_arpa` model's alignments are logical and accurate. The `english_mfa` model's alignments are unusable because the acoustic model was given a "bad map" (the incorrect G2P pronunciations) and failed to find the sounds in the audio.

3 Results and Key Observations

The visual analysis in Praat (Figures 2 and 3) confirms the findings.

3.1 Alignment Visualization

The figures below show a side-by-side comparison of the final alignments from both models.

- **Left Panels:** 'english_mfa' (Newer model, poor results)
- **Right Panels:** 'english_us_arpa' (Older model, good results)

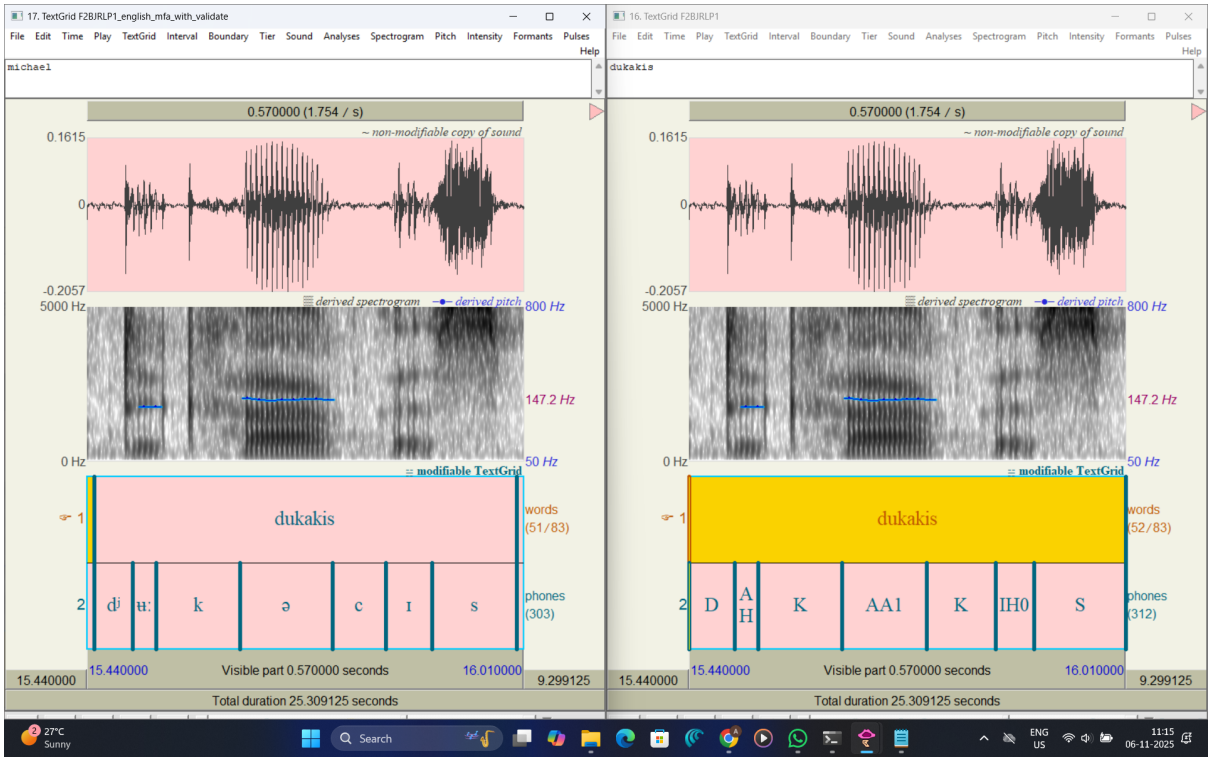

Figure 2: Alignment for **'dukakis'**. The `arpa` model (right) produces a clean, logical alignment. The `mfa` model (left) produces a different, less precise phoneme sequence (`d h: k @ c i s`) and a word boundary that appears slightly offset from the audio.

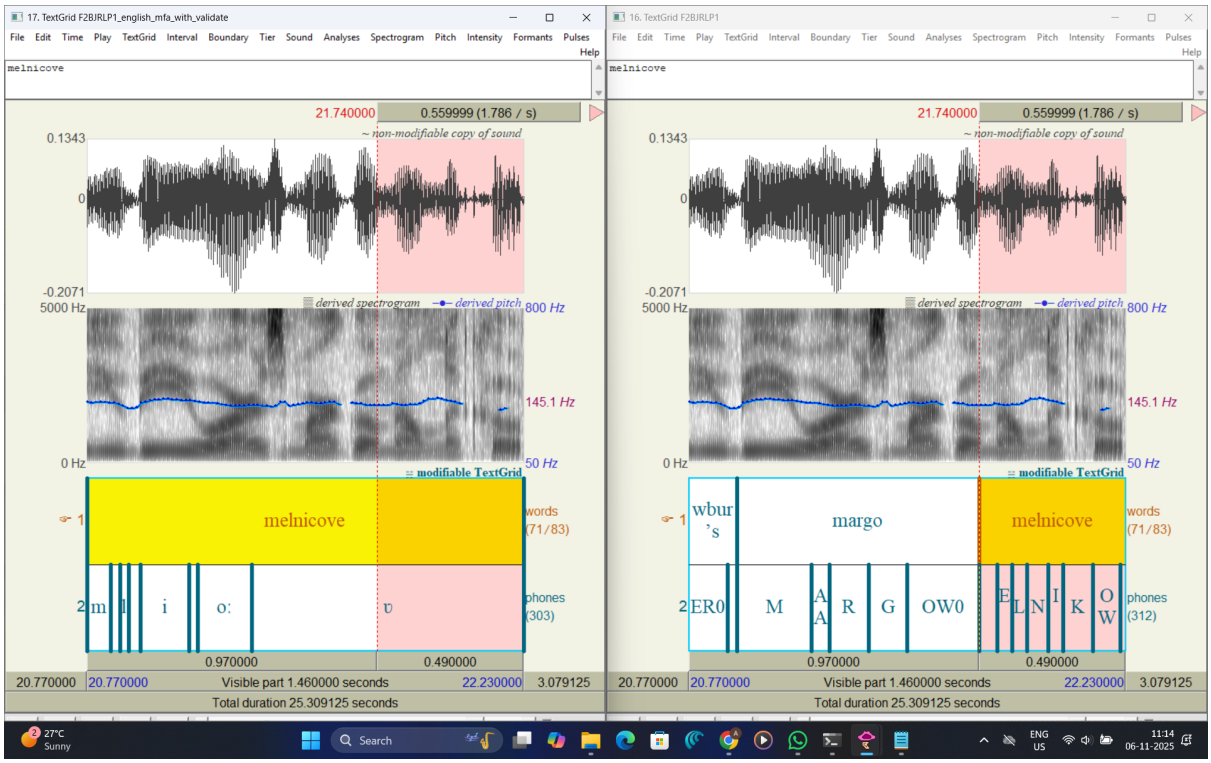

Figure 3: Alignment for **'melnicove'**. The `arpa` model (right) provides a complex, plausible alignment. The `mfa` model (left) provides a completely incorrect G2P guess (`m i o: v`) that fails to align with the audio, leaving large unaligned gaps.

3.2 Key Observations

1. **Errors and Mismatches:** The primary error was the `AssertionError` crash, caused by OOV words. This was solved by the custom dictionary. The secondary mismatches are seen in Figure 3, where the `english_mfa` model's alignment is completely wrong, assigning phonemes like `i` and `o:` to audio segments that do not match.

2. **Phoneme/Word Boundaries:** In the successful 'arpa' alignments (right panels), the word boundaries on Tier 1 correctly segment the audio. The phoneme boundaries on Tier 2 then sub-divide that word block into its G2P-generated sounds.

3. **Model Comparison:** The 'english_us_arpa' model was conclusively superior for this dataset. Its dictionary had better coverage, and its G2P model produced far more accurate guesses, leading to usable and correct alignments.

4 Conclusion

This project successfully demonstrated an end-to-end forced alignment pipeline. The primary finding was that default models are not guaranteed to work on real-world data and that OOV words are a critical failure point. By training a custom G2P model, this failure was solved.

Furthermore, a comparative analysis showed that the "newer" `english_mfa` model was inferior to the `english_us_arpa` model for this task. Finally, the entire successful workflow was automated into a `pipeline.bat` script, making the complex, multi-step process reproducible with a single command.

5 References

- **Kaldi ASR Toolkit** (2025). *Feature extraction documentation*. https://kaldi-asr.org/doc/feat.html#feat_intro

- **Montreal Forced Aligner** (2025). *Official Documentation (v3.3.8)*. https://montreal-forced-aligner.readthedocs.io/en/latest/

- **Boersma, P., & Weenink, D.** (2025). *Praat: doing Phonetics by Computer (v6.x)*. http://www.fon.hum.uva.nl/praat/

- **Gorman, K. (2016).** *Pynini: A Python library for weighted finite-state grammar compilation.* In Proc. ACL Workshop on Statistical NLP and Weighted Automata.

- **Novak, J., et al. (2012).** *WFST-Based Grapheme-to-Phoneme Conversion: Open Source tools for Alignment, Model-Building and Decoding.* In Proc. ACL.

- **McAuliffe, M., et al. (2017).** *Montreal Forced Aligner: Trainable Text-Speech Alignment Using Kaldi.* In Proc. Interspeech.