Speech-Text Comparison as Speech-Speech Comparison

Methodology:

The existing speech-text matching solutions directly use text and speech data for matching, but the text does not contain the specific details of pronunciation. Based on this, I adopted a solution that can use the pronunciation details of the text: using TTS to generate standard speech for the text, and then using a speech encoder for same-modality comparison.

For Localization, text/speech can be divided into blocks according to needs, TTS speech of the text blocks can be generated, then the model compares human speech and each TTS speech block to obtain the performance of human speech in each block.

Explainability and Insight:

What is right, what is wrong, and what is the standard? I used MeloTTS as the TTS model. MeloTTS uses CMUdict (CMU Pronouncing Dictionary) as the basis for g2p (grapheme-to-phoneme) to generate speech. I use this speech as the "standard". The closer it is to this "standard", the more correct the speech is.

This will lead to a new problem: How to avoid voice personality characteristics such as pitch leading to biased comparison results? In this regard, I think of three solutions:

1. Standardize human speech. Voice cloning technology can be used to convert human voice to make it sound like TTS voice. Current data anonymization is an option.
2. Convert TTS speech. This is reverse voice cloning, making TTS voice sound like human voice.
3. Add various data augmentations when training the model. Data augmentation of various voice personality characteristics such as pitch, speaking speed, and volume is applied to TTS speech and human speech during training. This is also the current method.

How does the model judge similarity? The most direct way is to concatenate two speeches into one model, and the model directly outputs the results (see Figure 1). This solution is more effective, but lacks explainability. In one of my solutions, I used a speech encoder + a pooling layer to calculate the feature vectors of TTS speech and human speech, and then calculated the cosine similarity of the two vectors to represent the similarity of the two speeches (see Figure 2).

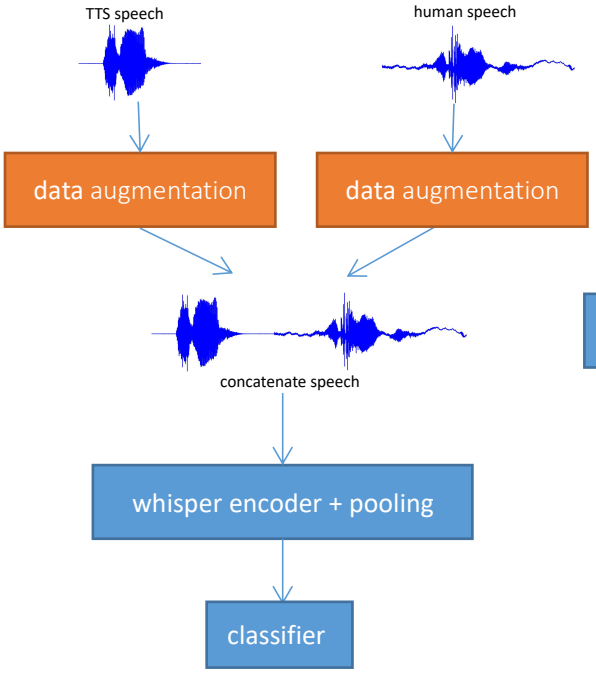

Figure 1

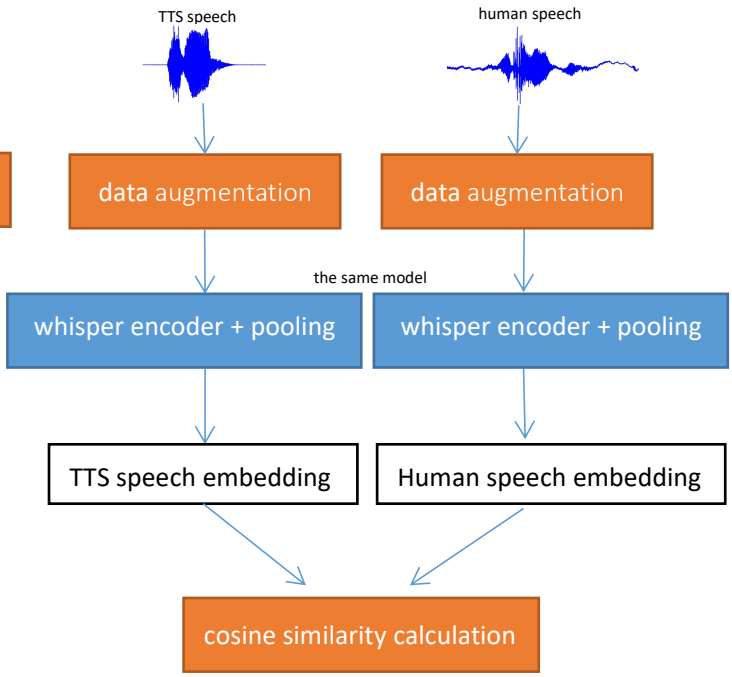

Figure 2

Combined with the above methods, what the model really outputs is **how similar the input speech is to the TTS standard speech based on CMUdict, excluding the influence of voice personality characteristics**.

Localization Ability:

Since human speech data does not contain location labels, it is relatively difficult to locate in this task. However, my solution can solve the problem of unlabeled localization.

For any text, we can divide it into blocks as needed, and then generate TTS speech blocks for each text block. The model can compare human speech with each TTS speech block to obtain the text error locations of human speech. Figure 3 and Figure 4 are example images of block matching.

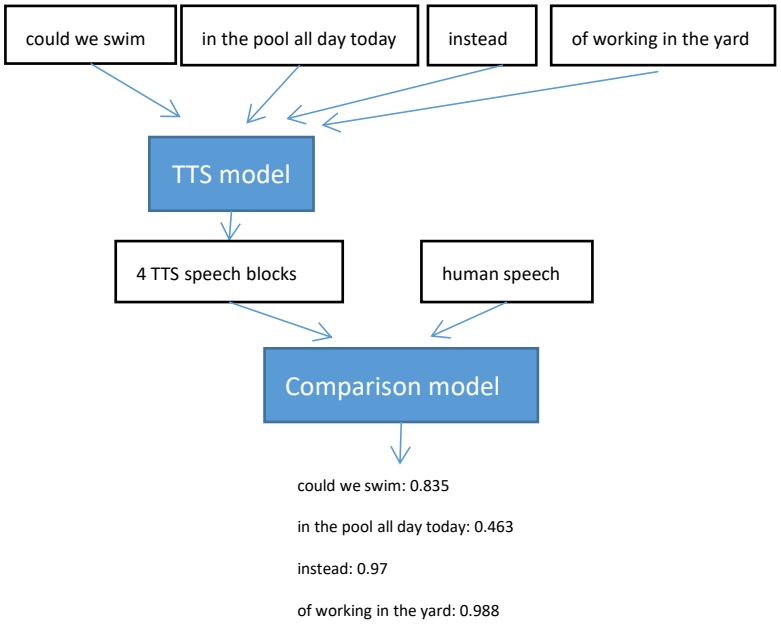

Figure 3 (sentence repetition task)

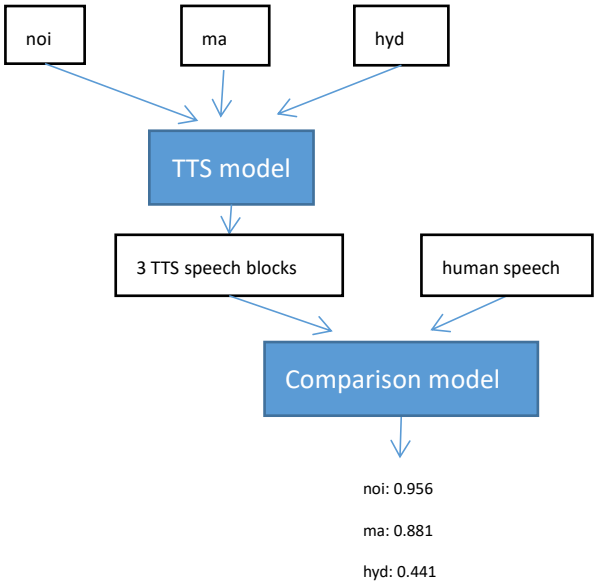

Figure 4 (nonword repetition task)

For some repetitive text, that is, the same text block is in different locations, or you need to know the error location of human speech, not just its text error location, you can use the sliding window method to segment human speech, and then compare each human speech window with the TTS speech blocks to obtain the scores of each window. As shown in Figure 5.

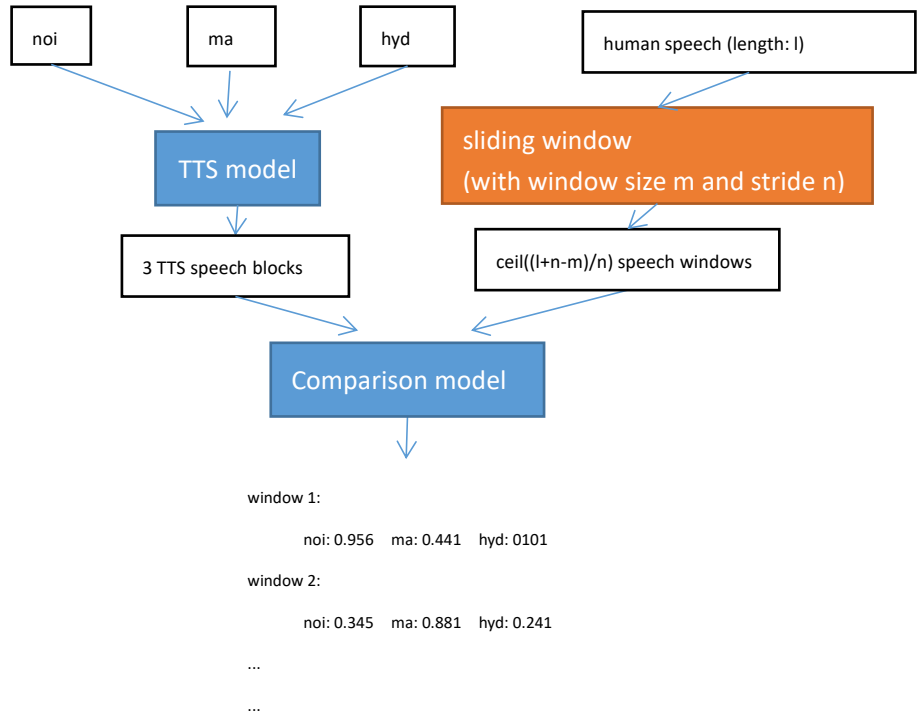

Figure 5

Before using the localization ability of the model, it is recommended to retrain or fine-tune the model, and add human speeches and TTS speech blocks to the dataset to obtain a new form of data before training.