Automatic Speech Recognition using HTK

William L. and Andrea T.

Data Mining and Machine Learning

# Recognition Result with Static MFCC Features

Executing the Perl script that trains and tests the given data gives us a result that is outputted into the *result* directory under the name *recognitionFinalResult*.*res*. The recognition results of the clean tests using a system of only clean training data with static features are shown in Fig. 1.

Graphical user interface, text, application, email

Description automatically generated

**Fig. 1.** Recognition result without extra features using clean data. No changes in -p flag value.

From Fig. 1, the results on sentence (SENT) and word (WORD) level are very different. These values indicate how well the entire words and sentences are recognised, so it can be deduced that this system is good at recognizing words, but bad at recognizing sentences, at a low accuracy of 63.54%. Additionally, at the word-level result, there is an imbalance between the terms (deletion) and (insertion). That can be changed by modifying *-p* flag value when executing the *HVite* command.

After testing, it was found out that a *–p* value of -15 gave the most balance between *D* and *I* as seen in Fig. 2. It was also discovered that by increasing the -*p* flag, *I* increase, and further increasing -*p* would also increase the difference between *D* and *I.* As expected, decreasing -*p* increases *D* and decreases *I*, allowing us to find a value where *D* and *I* are identical. The balance of *D* and *I* has also increased the correctness at the sentence level and the accuracy at the word level.

Graphical user interface, text, application, email

Description automatically generated

**Fig. 2**. Recognition result without extra features using clean data. -p = -15.

# Delta and Delta-Delta Features

Similarly, we can include the and features into the recognition system using the same script with some modifications. Values in the Perl script containing MFCC\_E are changed into MFCC\_E\_D\_A. and features have a total of 39 dimensions, three times more than the initially 13 dimensions of the MFCC\_Efeature. The last modification in the Perl Script is what prototype to use for the HMM training. The prototype file is located inside the *lib* directory, and the modification to the prototype looks like $Proto = “$REC\_DIR/lib/proto\_s1d39\_st8m1\_LabDMML\_MFCC\_E\_D\_A”;, where $REC\_DIR is the directory of the project.

Other than Perl scripts, some config files inside the config directory will also need modifications, mainly the files: *config\_HCopy\_MFCC\_E*, *config\_test\_MFCC\_E*, and *config\_train\_MFCC\_E*. Values for TARGETKIND in all three files needs to be modified to MFCC\_E\_D\_A and in the *config\_HCopy\_MFCC\_E* file, set the DELTAWINDOW = 3 and ACCWINDOW = 2. After all the modifications in the Perl script and config files, as well as trying different -*p* values, the Perl script was executed using clean training and test data. As seen in Fig. 3, the result is outputted again into *recognitionFinalResult*.*res*, unless the output file name was changed in the Perl script.

Text

Description automatically generated with low confidence

**Fig. 3.** Recognition result including and features using clean data. -p = -50.

Fig. 3 has shown a significant increase in correctness and accuracy at both levels. The and values are balanced, showing maximum accuracy for the recognition system when testing clean data. Correctness at the sentence level has jumped up by around 20%, and accuracy at the word level is at 96.88% with only 21 missed words out of 865, while using and features.

With and features, the recognition system was able to recognize most of the testing data, given that it now has three times the features than before. The increase in accuracy was expected because the and features approximate the first and second derivatives of the signal. By deriving the signal, we get an approximation of the signal energy based of off its previous energy. If a signal energy was high previously and is now low, we would have a negative gradient, and a similar process happens to get a positive gradient. Compared to the system without and features, static systems would not be able to recognize if an energy signal came from a high or low energy, while this new system can. With this, the system now has more input data and parameters that it could learn from, which may or may not lead to an increase in accuracy. But in this case, the accuracy did increase and that means that a pattern was found in the approximation of the signals that the system could learn and recognize.