MISSISSIPPI STATE UNIVERSITY

RECOGNIZING TELEPHONE NUMBERS USING THE ISIP DECODER

Program #7

EE 8993: Speech Processing

Audrey Le

Professor: Dr. Joseph Picone

August 1, 1998

1. PROBLEM DESCRIPTION

In this project, we are to build a system that recognize spoken telephone number using the ISIP decoder. The system must be able to handle 4, 7, and 10 digit strings. For the acoustic model, we will use the ISIP context-dependent phone models currently packaged as part of the ISIP decoder demo. For the language model, we will build our own. We will use our own voice and home telephone number as data to demonstrate our system.

2. PROCEDURES

For the first step, the data were collected. Three digit strings of length four, seven, and ten were recorded using the audio equipment available in the lab. The digit strings were recorded at a sample rate of 8000Hz. They were encoded in one channel linear format. The three digit strings are given in the table below.

Data	Sample Rate (Hz)	Encoding	No. of Channels	Duration (Sec)
9030	8000	linear	1	2.00
3389030	8000	linear	1	3.25
6013389030	8000	linear	1	5.00

Table 1: Data used for demo of the system.

The data were converted from raw format to NIST audio format. The reason we changed the format is that our feature extraction software program requires the input be in either TIMIT, NIST, or ISO format. Since we have a utility to convert raw format to NIST format, we chose NIST format.

After conversion from raw to NIST format, the data were fed in the extraction program to get the feature vectors. The feature program, developed by Philip Loizou at University of Arkansas, can produce different types of feature vectors such as MFCC and LPC. Details of the feature extraction program can be found in [1]. For our decoder, we need a 39 dimension feature vector. The feature vector is made up of 12 MFCC, 1 energy coefficient, 12 delta MFCC, 12 delta-delta MFCC, 1 log speech energy, and 1 delta energy for a total of 39 coefficients.

Since our decoder takes in ASCII input, the features produced by the extraction program had to be converted from binary to ASCII format using another utility from the feature extraction package. The ASCII output from the extraction program also needed to be filtered before feeding it to the decoder.

Upon viewing the features produced by the extraction program, we noticed some “NaN”s and “Inf”s occurring toward the end of the file. Using Xwaves to view the waveform, we found trailing zeros at the end of the file. The data then were edited using w_edit to chop off the zeros at the end of the file. The trailing zeros caused the feature extraction program to produce “NaN” and “Inf” values. Apparently the feature extraction program did not take preventive measure to flag this situation.
In addition to the feature file, we need a acoustic model, a language model, and a lexicon as input

to the decoder. For the acoustic model, we used the generic ISIP context-dependent phone models given as part of the decoder demo package. For the language model, we used a bigram language model with each bigram having equal probability since for telephone number each digit is random. For the lexicon, we restricted our vocabulary to include only 10 digit. In addition, zero is referred as “zero” not “oh”.

Having the necessary files, we ran the decoder for the three digit strings. The results are reported in the next section.

3. RESULTS

The table below shows the output of the decoder for a given audio input. The performance of the decoder on the string level is very poor. However, the decoder performs a bit better on the word level. It probably performs even better on the phone level. However, the error rate for the phone level was not calculated since the phone error rate is not a good measure of the performance of the system. The argument is that even some phones in a word were recognized correctly, the word is still a different word. We are more interested in the overall understanding achieved from the recognizer’s output rather than the details of the output.

Audio Input	Decoder Output	String Error Rate (%)	Word Error Rate (%)
9030	50302	100	40
3389030	3325230	100	43
6013389030	601333250202	100	42

Table 2: Performance of the decoder on our input.

The main reason for poor performance is that our language model is a simple one. A better and more comprehensive language model will likely to improve performance. In addition, the speaker has a heavy accent, a combination of Vietnamese and Southern American accent. The latter accent is known in speech recognition’s literature to give problems.

4. CONCLUSIONS

We have presented here the results of our decoder experiments. From this experiment we learned the various parts of the speech recognition system. We learned what each part is composed of and how to obtain it. We also familiarized ourselves with the various audio recording equipments.

5. REFERENCES

[1] P. Loizou, “Feature Extraction Programs for Speech Recognition,” User Manual, Univ. of Arkansas, 1997.

Figure 1: Distribution of Word N-grams for the Top 1000 Bins.

Figure 2: OOV rate as a function of N most frequent words.