Phonemes to Numbers

Professor Don Colton, BYUH

CS 441

1 Overview

Your task is to write a program that converts phonemes into numbers. The phonemes are presented as a time-aligned transcription where each phoneme has a starting time, an ending time, and a symbolic representation. The numbers are expressed in written characters as whole individual words.

2 Preparation

We will use as our standard the CSLU Numbers corpus. A sample of this corpus is available for free download from the CSLU Corpus website. At this writing, the homepage for CSLU corpora is:

```
http://www.cslu.ogi.edu/corpora/
corpCurrent.html
```

Find the Numbers Corpus, probably version 1.3.

Download the sample file, probably called `numbers_sample.zip` and store it somewhere so you can use it for the remainder of this project.

Open the file and review its contents. You should find a folder (directory) named "labels" and another folder named "trans". The labels directory contains the time-aligned phonetic labels in files with a .phn extension. The trans directory contains the orthographic (normal writing) transcriptions in files with a .txt extension.

All files are in plain ASCII text.

Display: Your instructor may ask you to display a matching pair of files, one .phn and one .txt, to show that you understand the file organization and to verify that you can work with the files.

3 Vocabulary

The full CSLU Numbers corpus consists of about 23,900 utterances, each with a phonetic transcription and an orthographic transcription. The sample corpus consists of 283 transcribed utterances, or about one percent of the full corpus. Within those sample utterances the following 46 words occur:

> and dash double eight eighteen eighth eighty eleven eleventh fifteen fifth fifty five forty four fourteen fourth hundred is it's nine nineteen ninety oh one right seven seventeen seventeenth seventh seventy six sixteen sixth sixty ten third thirteen thirteenth thirty thousand three twelve twenty two zero

These same 46 words occur in the full corpus, and there are probably additional words. Some words occur frequently. Others (like "right") may only occur once, and that by accident. You may find you get a better score if you ignore rare words.

4 Over Training

Your program is performing a recognition task. There are two ways to approach such a task. One is by strict memorization. You check to see which of the 283 cases you are viewing and you report the correct results for that case.

The other is by a judicious combination of memorization (basis cases) and rules (induction). Rather than memorizing the 283 sample results, you memorize the 46 words that occur in the sample results. And you create suitable rules to combine the memorized results to make additional, non-memorized results.

A brief example may help. When we calculate `4+3`, and we are very small, we may hold up four fingers on one hand and three on the other hand. Then we may count all the fingers to arrive at a total of seven. In this example, the meaning of `4` was memorized, the meaning of `3` was memorized, and the process of combining them was memorized. `4` and `3` are part of the basis. The process is part of the induction.

Later in life, we may actually memorize our "addition facts" or "addition tables" and come to know that `4+3=7` without ever counting them out. We can tackle a problem like `639+15` by breaking it up into smaller problems: `9+5=14`; write down the `4` and carry the `1`. `3+1+1` is `5`; write it down and carry zero. `6` comes down for a total of `654`. In this process, the single-digit additions were memorized and the rules for carry and column order provide the inductive step to we can add numbers of any size.

Because this same situation occurs in recognizing numbers, we must be careful to keep our basis set as small as possible, within reason. It is good to divide your corpus (the 283 utterances) into parts. One part can be used for development. You can look at those utterances in great detail and analyze them to understand your task. Another part can be used for testing. You use those utterances to see how well your program performs. The great risk is that you will train your program to recognize your 283 utterances very well, but you will fail to recognize the other 23,600 utterances. That is called overtraining. Use your resources carefully.

5 Evaluation

When you believe you have a reasonable performance rate for your program, you can request me to test it. I will test a maximum of five versions of your program. Your score for the assignment will be the score you earn on your fifth (or last) test.

When I test your program, I will report the number of utterances it correctly recognized, meaning that the entire utterance was exactly right in every way. I will provide you with my testing script, but not with the data files that I am using.

The input files (.phn) are given to you in their original form, redirected to you as STDIN standard input, with results captured from you as STDOUT standard out. Your file should present one word per line, interspersed at your discretion with comment lines that I will ignore. Comment lines are identified by a "`#`" as the first character of the line.

Each corpus transcription file (.txt) will be "normalized" by being stripped of all words that start with dot (such as `.pau`), and all characters that are enclosed in angle brackets (such as `<bn>`). Letters will be converted to lower case. Runs of whitespace will be converted to a single space. The 28 characters a-z, apostrophe, and space, will be kept. All other characters will be deleted.

It is impossible to get a perfect score because most but not all human transcriptions are correct. Also some transcriptions include word fragments. But try to get the best score you can.