The training of Slovak speech recognition system based on Sphinx 4 for GSM networks

Juraj Vojtko 1, Juraj Kacur 1, Gregor Rozinaj 1

1 Department of Telecommunication, Faculty of Electrical Engineering and Information Technology, Slovak University of Technology, Ilkovicova 3, Bratislava 812 19, Slovakia
E-mail: kacur@ktl.elf.stuba.sk

Abstract – In the submitted paper we present the training process of HMM models that are designed to be used in ASR systems employed in GSM networks. First a brief overview regarding the current problems and applications of ASR systems is given, followed by the description of MOBILDAT-SK speech database and the SPHINX 4 and SphinxTrain capabilities. Then the process of HMM models training is presented utilizing the facility of the SphinxTrain system adjusted for the structure of MOBILDAT database and the Slovak language. The article is concluded by presenting the achieved results using the tools of the SHINX 4 by the means of 3 types of tests: application words, isolated digits, and looped digits. The WER for the looped digits and CD phoneme models is 1.8% which is roughly comparable to the performance of other systems.

Keywords – HMM, ASR, SPHINX 4, SphinxTrain, MOBILDAT-SK

1. INTRODUCTION

There is a great need to employ various kinds of dialog or information retrieval systems in the current telecommunication networks. These systems should be capable to operate on a real time bases, must be speaker independent and exhibit high accuracy. Practical systems must support dictionary size of at least several hundreds words. These requirements can be currently met by the statistical speech modeling using HMM models of tied context dependent phonemes with multiple Gaussian mixtures.

In order to secure both robust and accurate set of models huge speech databases must be assumed. There have been many ASR systems and databases collected and tested in the fixed line environment. However, as the latest communication technologies [1], [2] are becoming more ubiquities, so is the need for theses services to be available in GSM networks. It is known that the best results regarding the ASR accuracy can be achieved if the training and application environment fit each other even despite the existence of some advanced speech extraction methods like: MFCC, PLP, TIFFING, etc. Thus the described ASR and the results are related to the speech database gathered over the GSM networks.

Finally, as there is no analytical solution to the estimation of HMM models a great effort must be spent in the training phase. Thus a mature system must be used providing many tools that can be iterative and selectively applied and verified. There are more systems to do so but the most spread ones are HTK and SHINX [3]. In the following will focus our attention to the open source SPHINX system which opposes the more advanced HTK system.

2. THE MOBILDAT-SK DATABASE

As it was already mentioned a successful construct of the up to date ASR systems must take into account the matching between the training and employment environments. Thus we decided to use a newly built Slovak MOBILDAT database [4] which was recorded over GSM network and various types of cell phones. The concept of MOBILDAT database is based on the widely used structure of the SPEECHDAT [5] database, whose many versions were built for several languages using fix telephone lines. Also some reference recognition systems like REFREC 0.96 or MAPER [6] were designed to cooperate with them. Thus let us in the following to recap some basic facts about SPEECHDAT database and its differences to the MOBILDAT.

The Slovak SPEECHDAT database consists of 1000 speakers divided into the training set (800) and the testing set (200). Each speaker produces several recordings in a session with the duration between 4 to 8 minutes. There were 48 items spoken per a speaker for the SPEECHDAT-SK and 50 for the MOBILEDAT-SK. These items were categorized and marked into the following groups: isolated digit items (I), digit/number strings (B,C), natural numbers (N), money amounts (M), yes/no questions (Q), dates (D), times (T), application keywords (A), word spitting phrase (E), directory names (O), spellings (L), phonetically rich words (W), phonetically rich sentences (S, Z). Speech files were A-law coded into 8 bits with no header at 64kbit/s. Description files were provided for each utterance with the orthographical transcription but no time alignment was given. Beside the speech following non-speech events were labeled too: truncated recordings (~), mispronunciation (*), unintelligible speech (**), filed pauses (fil), speaker noise (spk),
stationary noise (sta) and intermitted noise (int). In the case of the MOBILDAT-SK the GSM specific distortion that may afflict the speech parts was marked too as (%). In the accompanying dictionary the phonetic transcription allowing multiple pronunciations was provided. Finally, there were 41739 useable speech recordings in the training portion, containing 51 Slovak phonemes and 10567 different context dependent phonemes.

3. SPHINXTRAIN AND SPHINX 4 TOOLS FOR THE ASR SYSTEMS

There are several systems that allow building and training HMM models which meet up to date accuracy, speed and size requirements. However the most famous are the HTK and SPHINX. There have been several versions of the SPHINX where the latest one exhibits some appealing features that can be further developed and used in modern technologies.

Sphinx4 is an open source speech recognition system created by Carnegie Mellon University, Sun Microsystems Laboratories, Mitsubishi Electric Research Labs and Hewlett Packard. As it is written in the JAVA programming language it makes it platform independent. Furthermore, Sphinx4 is not only a speech recognition tool, it is also an integrated platform for the future research in the speech domain. It is a very flexible and modular system capable of performing many recognition tasks. It supports both live and batch modes, and accepts both statistical language modeling (n-grams) and finite state grammars JSFG. The Sphinx4 architecture has been designed modular, so the system programmer can change each module to fit the particular demands. The main blocks of Sphinx4 are: Frontend, Decoder, and Knowledge base which contain lexicon, acoustic models (HMMs) and language model that create so called Linguist. Blocks like Frontend and Decoder are independently replaceable and all the necessary manipulations are carried out by setting the configuration file.

Frontend is the input block (performs the speech feature extraction) to Sphinx4 system and is made up of several communicating parts which are preemphasis, windowing, Fourier transform etc.

The primary role of the Decoder block is to use the features produced by the Frontend and in conjunction with the Linguist to generate the recognize hypotheses. The Decoder block further comprises a pluggable SearchManager block and other supporting components that simplify the decoding process for the use by external applications. The SearchManager is not restricted to any particular implementation and may utilize different algorithms like: Viterbi algorithm, A* search, etc. In our tests we used component named as SimpleBreadthFirstSearchManager, which implements a simple frame synchronous Viterbi algorithm.

The Linguist generates the SearchGraph that is used by the Decoder during the search process. A typical implementation of the Linguist includes Language model, topologic structure of Acoustic model (HMM) and the Dictionary that maps words from the language model to the strings of acoustic models. The Dictionary contains a pronunciation lexicon where multiple acoustic realizations are supported.

As the recognition is based on statistical speech modeling, each sound unit is modeled by a sequence of states that further model the probability distributions of the observed speech. The process of training is a very complex assignment which plays an important role. This task is performed by the different tool common to all Sphinx versions called the Sphinx trainer (version 3) [7]. It is an independent application written in C language, which computes acoustic models. This trainer is called SphinxTrain and actually it is not available as a final version of SphinxTrain, but only as a nightly build version.

This package contains source codes of SphinxTrain tools and control scripts written in perl which actually govern the running of the SphinxTrain during the training.

Generally the training process outlined by the SphinxTrain gradually undergoes these stages:

- Verification of the input data
- Training of context independent (CI) HMM models (no time alignment is supported by Sphinx trainer, only embedded training)
- Training of context dependent (CD) HMM phonemes – triphones
- Pruning of the decision trees and tying the similar states
- Training of the CD phonemes’ models with the reduced states number (tied triphones)
- Training of the tied triphones models with the gradually augmented number of Gaussians

In order that the training process would be successfully accomplished several input data must be carefully prepared ahead and presented to the training script. The currently supported speech features are MFCC and PLP parameters and their time derivatives. Except the training data (extracted speech) several accompanying files and lists must be generated:

- list of all phonemes
- dictionary (lexicon)
- filler dictionary contains non-speech elements
- orthographical transcription of all training records
• list of all filenames existing in the training set
• SphinxTrain configuration file

The output of the SphinxTrain consists of several files which all together represent acoustic models for CI phonemes with 1 Gaussian mixture, untied CD phonemes with 1 Gaussian mixture and finally tied CD phonemes with multiple Gaussian pdf per state. However, one of the main drawbacks of the Sphinx system is that during the training phase all models must exhibit the same structure regardless of whether they are speech units like phonemes or phrases or even non-speech events. Furthermore, they can only have either 3 or 5 states per model and must have strictly left-right transition matrices which makes it difficult to model short pauses or longer sequences of background noises. Each model has one terminating, non-emitting state to allow multiple models to be concatenated into more complex models of words or even sentences.

4. TRAINING OF HMM MODELS ON THE MOBILDAT-SK USING SPHINXTRAIN

The outlined training procedure for the context dependent (CD) and context independent (CI) models in the SphinxTrain has to be slightly adjusted for the structure of the MOBILDAT database and the Slovak language. Thus in the following a brief description of the training process is provided together with some statistical information.

The training was performed on the Slovak MOBILDAT database to suite the requirements for the models to be successfully employed in GSM networks. The application of the MOBILDAT to the training procedure defined by SphinxTrain was not so straightforward as it is in the case of Masper or Refrec which were designed directly for its structure which is rather complex and distributed (5 CD). Thus several modifications had to be done prior to the training which can be summarized as follows: conversion of SAMPA symbols, generation of the list of all utterances, selection of the training portion, transcription information must have been gathered, transformed and saved in the format required by Sphinx tools, records with the unusable content had to be founded and removed, lexicon must have been transformed from that one provided by MOBILDAT which was designed as case sensitive, list of phonemes had to be created together with the list of fillers (non-speech events), and finally the configuration file had to be adopted to the structure of MOBILDAT.

During the process we have created CI models with 1, 2, 4, 8, 16, 32, 64, and 128 Gaussians and CD models with 1, 2, 4, 8, 16, 32, 64 Gaussians pdf, just to verify and compare their performance. There were 51 phonemes and 3 types of filler models determined to model non-speech acoustic artefacts. They are SIL for silence (including the silence at the beginning and the end of recordings), FIL for the filled pauses and SPK for the speaker’s noise. The unified HMM structure for all models used 3 regular states plus 1 non-emitting state for both CI and CD models. We set the number of tied states of triphones to 9000 in the configuration file. This number comes from another test in our department involving HTK system where this number is determined automatically in the state slitting process (also based on the decision trees). So, all CI models contain 216 states from which 54 are non-emitting ones. Then the number of CD triphones (word external context expansion was used) is 86209 with 557600 states, and after the tying process it ended up in 9000 unique states for triphones and 162 for phonemes. There were altogether 86209 logical triphones but only 18841 physical ones.

Non-speech filler INT (interrated noise) was ignored and mapped into SIL model just as well as the STA (stationary noise). Records containing incomplete (truncated) or mispronounced words were rejected from the training process, unlike the records contaminated by GSM noise, which were let included. Finally, the training set consists of 41739 records spoken by 880 persons. The dictionary contains the pronunciations of 14909 words, some of which has more pronunciations. However the SphinxTrain process can handle only one- the first, the remaining ones can be utilized only by the recognizer.

The initialization of the models was performed over all training recordings by the so called flat start initialization. Speech files were described by the MFCC parameters and their derivatives, which were divided into 3 streams for every processed block of the length of 20ms. Thus there were 13 cepstral coefficients, 13 delta and 13 delta-delta coefficients, computed form 16 bit linear PCM waveforms. During the training the Baum-Welch algorithm was used with the convergence ratio set to 0.04 and the maximum and minimum iteration numbers were 30 and 3 respectively.

Table 1. Recognition results for digits in the loop

<table>
<thead>
<tr>
<th>Gaussians No.</th>
<th>WER CI</th>
<th>WER CD</th>
<th>SER CI</th>
<th>SER CD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>13.98</td>
<td>4.62</td>
<td>70.33</td>
<td>44.01</td>
</tr>
<tr>
<td>2</td>
<td>7.71</td>
<td>2.64</td>
<td>49.76</td>
<td>31.71</td>
</tr>
<tr>
<td>4</td>
<td>5.51</td>
<td>2.74</td>
<td>38.76</td>
<td>30.35</td>
</tr>
<tr>
<td>8</td>
<td>4.81</td>
<td>2.46</td>
<td>37.26</td>
<td>28.5</td>
</tr>
<tr>
<td>16</td>
<td>4.04</td>
<td>2.01</td>
<td>35.68</td>
<td>26.79</td>
</tr>
<tr>
<td>32</td>
<td>3.76</td>
<td>1.8</td>
<td>34.27</td>
<td>26.54</td>
</tr>
<tr>
<td>64</td>
<td>3.15</td>
<td>1.56</td>
<td>30.52</td>
<td>26.42</td>
</tr>
</tbody>
</table>

4. RESULTS

To be able to provide results in a form comparable to the other systems we have executed
following tests on the test section of the MOBILDAT database: application words, isolated digits, and looped digits. Actual recognition was done by Sphinx4 as it supports finite state grammar. In the recognition process similar restrictions were used as in training phase – recordings containing incomplete or mispronounced words were disqualified, while the records with GSM noise were accepted.

Because of an easy comparison the accuracy evaluation was actually done by the HTK tool HResults which computes WER and SER values in the standard way. The achieved results for CI and CD models with several Gaussians mixtures are shown in table 1 for the case of digits uttered in the loop while in table 2 there are isolated digits and application words.

Table 2. Recognition results for isolate digits and application words

<table>
<thead>
<tr>
<th>Gaussians No.</th>
<th>Isolated digits</th>
<th>Application words</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>WER CI</td>
<td>WER CD</td>
</tr>
<tr>
<td>1</td>
<td>58.06</td>
<td>26.27</td>
</tr>
<tr>
<td>2</td>
<td>46.33</td>
<td>26.15</td>
</tr>
<tr>
<td>4</td>
<td>40.73</td>
<td>17.43</td>
</tr>
<tr>
<td>8</td>
<td>33.33</td>
<td>18.72</td>
</tr>
<tr>
<td>16</td>
<td>27.85</td>
<td>17.35</td>
</tr>
<tr>
<td>32</td>
<td>32.42</td>
<td>14.16</td>
</tr>
<tr>
<td>64</td>
<td>29.22</td>
<td>10.05</td>
</tr>
</tbody>
</table>

5 CONCLUSIONS

We have trained both CI and CD HMM models on the MOBILDAT-SK database using the SphinxTrain procedure. MOBILDAT-SK was specially designed to reflect specific features of GSM networks and thus serves as a good base to train robust HMM models for any kind of ASR system to be employed in the GSM environment.

As it can be seen from table 1, the achieved results in the terms of WER for the digits in the loop falls in to the usually reported ranges even for fixed line environments, however the sentence errors are higher than normally should be. Bigger discrepancies are observed in the cases of isolated digits and application words. These are too high to be considered as error free. However, closer investigation gives us some explanations and reminds us of the implementation problems related to a particular task. All the tests were performed using the same settings of configuration parameters, however different grammars were used. As in the JSFG the non-speech events are not explicitly mentioned different techniques should be used. To control the insertion of these models one can use the following settings: wordInsertionProbability, languageWeight, silenceInsertionProbability, skip, and probably others. Despite that a priory probability of any filler can not be directly set. As these settings were the same as for the looped words, where they exhibited good performance, it turned out they were inappropriate for different items like isolated words. These 2 tests differ in the terms of speech presence which is much higher in the case of looped digits. In the case of isolated digits and application words longer parts of silences or non speech events were observed. As these segments are not properly modeled their deficiency is much more prevalent.

All these findings signify the great need for the careful setting of the configuration parameters that do differ for particular tasks. Unfortunately there are many parameters to be set which are mutually and non-linearly related so this task is bit tricky, but necessary to accomplish as we have just documented.

ACKNOWLEDGEMENT

The authors hereby declare that the article was founded by the grant: VEGA 1/3110/06.

REFERENCES