AUTOMATION OF THE TRAINING PROCEDURES FOR NEURAL NETWORKS PERFORMING MULTI-LINGUAL GRAPHEME TO PHONEME CONVERSION

Horst-Udo Hain
SIEMENS AG, Corporate Technology
81730 Munich, Germany
horst-udo.hain@mchp.siemens.de

ABSTRACT

Any TTS system requires accurate grapheme to phoneme conversion routines due to the limited pronunciation dictionaries. Data driven approaches offer the possibility to train new languages without the knowledge of rules. A big problem is to prepare huge databases for the training. In this paper the automation of the data preparation and the pattern generation for the training of a neural network is addressed. This automation is language independent, and no native expert is required. The data preparation, the generation of the training patterns and the training itself are done completely automatically.

Keywords: multi-lingual TTS, neural networks, language independent

10.21437/Eurospeech.1999-463

1 INTRODUCTION

Data driven approaches for the grapheme to phoneme conversion are commonly used in multilingual TTS systems. They offer the possibility to handle languages without the help of native speakers or at a higher level without the knowledge about the language itself.

There are many data driven approaches used for the grapheme to phoneme conversion. In [1] the most likely phonetic transcription is found by a Bayes decision rule, in [2] the grapheme to phoneme alignment is done by a Viterby module and a tree structure is used to store the aligned grapheme phoneme pairs. Examples for neural networks are NETtalk [3] for American English, NETspeak [4] for British English and [5] for German.

All systems have one problem in common: the preparation of the data base with the grapheme to phoneme mapping. Either these mappings are done by hand or semiautomatic with manual corrections. This is time consuming and can lead to inconsistencies. On the other hand you need a human expert who knows the language well enough to make corrections.

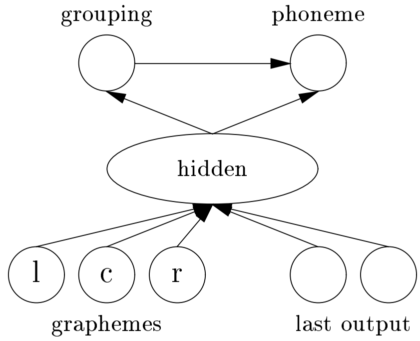

Figure 1: neural net for grapheme to phoneme conversion

This paper describes an approach that tries to create these mappings completely automatically. First the runtime functionality of our TTS system "Papageno" is introduced to show the demands on the data preparation. In the following sections the automation of the grapheme to phoneme mapping and the context width determination are explained. The last sections give a short summary and lists the work to be done in the future.

2 RUNTIME TRANSCRIPTION FUNCTIONALITY

The transcription in our TTS system is done in two steps. The first one converts the graphemes to the phonemes, and the second one inserts stress marks and syllable breaks in this phoneme string.

2.1 Grapheme to Phoneme

The input of this net consists of 9 nodes, while the output has two nodes. The input nodes are for seven graphemes (the center with three left and three right neighbours), the previous phoneme output of the net for this word and the previous grouping output of the net (see figure 1).

ISCA Archive
http://www.isca-speech.org/archive

6th European Conference on Speech Communication and Technology (EUROSPEECH'99)
Budapest, Hungary, September 5-9, 1999

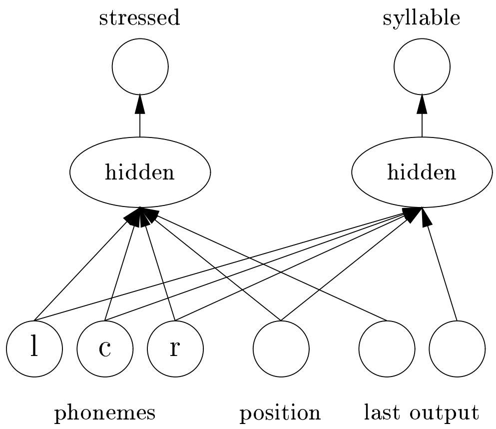

Figure 2: neural net for stress and syllable prediction

The grouping (or alignment) output is used to distinguish how many graphemes were used for the phoneme. For instance in the word *shall* (`/S@l/`) the graphemes *sh* were used to build the phoneme `[S]`, so the grouping is two. In this case the next grapheme to be handled is the *a*.

2.2 Word Stress and Syllable Breaks

This net uses as input a phoneme window of seven phonemes (the center with three neighbours on both sides), the position of the phoneme in the word and the previous output for stress and syllable. The output is whether the phoneme is stressed or not and if there is a syllable break after it or not (see figure 2).

3 GRAPHEME TO PHONEME MAPPING

The generation of the training patterns for the stress and syllable prediction is quite easy because the SAMPA phoneme set allows to find the boundaries between phonemes and the syllable and stress marks are easy to be found. The problem for the generation of the grapheme to phoneme conversion is to find out which grapheme is mapped to which phoneme in the word. In a standard phonetic lexicon this boundaries are not marked.

This marks are usually inserted by hand, semiautomatic with a manually correction or rule based. Therefore an expert or at least a trained person is necessary. It takes a long time and can be inconsistent. The better way is to find a procedure which can do this automatically. Such a procedure will be described in the next sections.

An entry in a phonetic lexicon looks like this:

```
shall
/ S @ l /
```

What is needed is

```
sh a ll
/ S @ l /
```

There have to be marks between the graphemes to know which graphemes belong to which phonemes. This is necessary for the pattern generation for the training of a neural network.

4 AUTOMATION OF THE MAPPING

The mapping is performed completely automatically. There are only two sources required: the phonetic lexicon and a list of the phonemes where each phoneme has a flag set to distinguish whether it is a vowel or not.

Actually the mapping is done in the following steps:

1. Find the words that have the same number of graphemes and phonemes and map each other in this sequence.

2. If all but the last phoneme are mapped and there are graphemes left, then map all these graphemes to the last phoneme.

3. If an entry could not be mapped completely, then go to last mapping actually available and add to its longest matching grapheme string the next grapheme. Then try to map the whole word. If this it not successful, add the next grapheme, until there are as many graphemes left as phonemes are.

4. If an entry could not be mapped forward, then try to map it backward and look for the positions how far the mapping routines came. If there is only one phoneme left, then map all the graphemes actually not mapped to the neighbours to this phoneme.

5. Now try to connect phonemes. For instance in the word
   ```
   axes
   / @ k s i z /
   ```
   there are more phonemes than graphemes. To be able to map this word two phonemes have to be connected. The candidates to connect are established by the frequency of the possible mappings. In this case `k` and `s` are concatenated to `k+s`. The new phoneme string would be `/ @ k+s i z /`. Now the word can be completely mapped:
   ```
   a x e s
   / @ k+s i z /
   ```

This is the step where the attribute vowel/consonant is used. Only connections of two vowels or two consonants will be accepted.

6. The last step looks in the phoneme string for already connected phonemes. If such a phoneme string is found, the two phonemes are replaced by their concatenation. If the mapping is now successful, the concatenation will be accepted.

Every step is done for every entry before the next step is done. After each step the frequency of each mapping is checked. If this frequency is below a predefined global threshold the following strategy is applied: If the frequency of this mapping is much lower than the frequency of the mapping between this grapheme and the next phoneme or this grapheme and the previous phoneme, then this mapping is deleted.

All the steps are performed until no new mapping can be found. Then the mapping table is saved and the procedure is ready.

5 CONTEXT WIDTH FOR INPUT

During the training the neural network tries to learn rules (hidden in the weights and biases) how to convert the input patterns into the right output patterns. These rules should not be inconsistent with one another. The network has no chance to learn the "right" output if there are different outputs for the same input.

To ensure that no contradictions occur in the patterns, they have to be found during the pattern generation. If they occur, the context width is too small and has to be expanded. On the other hand, if the context is too large, it will become difficult to train the net. The generalization property significantly decreases as the amount of training material is limited while the degrees of freedom increase.

To find out which context would be an optimum, an approach was developed to compare the number of the patterns and the resulting contradictions for each context width. Also the resulting network structure has to be considered.

5.1 Pattern Generation

During the pattern generation for every grapheme to phoneme mapping the input/output pattern is created and added to an array. If the pattern already exists, a counter for this pattern is incremented. This counter is used to determine which pattern should be used if the output nodes are different for equal input nodes.

5.2 Approach

The parameter that sets the minimum for the right context is the maximal grouping that occurs in the mapped lexicon. If the grouping is four, then the net must see at least the grapheme in the center and the three right neighbours. Otherwise it does not see enough context to learn the decision for this grouping.

For the English lexicon the maximal grouping was four. That's why the procedure checks the results of all context combinations from one left and three right grapheme to six left and six right graphemes. The maximum of six neighbours was chosen to assure that the network structure becomes not too big.

It counts the following parameters:

- the created patterns:
  is the number of all patterns that have been added to the array.
- the total number of contradicting patterns:
  is the number of all contradictions in the array, regardless whether the compared patterns had the same input or not.
- the patterns that have to be deleted:
  is the number of patterns that have to be deleted because they had the same input as another pattern but different output and occurred less than the other pattern.

The results for an English lexicon with about 59000 entries are listed in table 1. The fifth column contains the number of weights and biases for a neural network with such a context.

It shows that even with a context of six left and six right neighbours there are contradicting patterns! The fifth column shows the disadvantage of a big context. The bigger the input is, the bigger gets the neural network. A huge amount of patterns and a big network structure slow down the training speed. It also implicates that it becomes difficult for the training algorithm to find a minimum in the error space. Another point is the calculation time in the runtime process.

It is quite hard to find an optimal context width that complies all demands. Therefore the aspired task should be used for this decision. It is necessary to set a priority either for quality or for speed.

6 FUTURE WORK

The less the grouping is, the less is the maximum context width. That is why one criterion for the

context left/right	patterns	total contr.	deleted patterns	weights and bias
1 / 3	81259	5374	4737	22556
1 / 4	117528	4184	3777	26606
1 / 5	147621	3720	3356	31253
1 / 6	171313	3513	3154	35642
2 / 3	116964	3599	3393	26606
2 / 4	155625	2633	2536	31253
2 / 5	187223	2178	2104	35642
2 / 6	211722	1939	1867	40628
3 / 3	155435	2670	2555	31253
3 / 4	195775	1769	1739	35642
3 / 5	228243	1295	1285	40628
3 / 6	253272	1054	1045	45563
4 / 3	190223	2200	2097	35642
4 / 4	231506	1317	1294	40628
4 / 5	264537	835	832	45563
4 / 6	289854	584	582	51147
5 / 3	215846	2006	1905	40628
5 / 4	257694	1119	1097	45563
5 / 5	291005	632	630	51147
5 / 6	316449	377	376	56628
6 / 3	231808	1930	1829	45563
6 / 4	273958	1041	1019	51147
6 / 5	307395	553	551	56628
6 / 6	332905	298	297	61928

Table 1: contradicting patterns for different context widths

automated grapheme to phoneme mapping is to use as small groupings as possible.

The maximal grouping for the English lexicon is currently four. This grouping occurs for example in the word

`th ough t`
`/ T c t /`

To decrease the grouping there are some possibilities. One would be to use `oug ht` or `ou gth` as the mapping. This does not solve the problem yet. It would increase the grapheme to phoneme combinations, which only confuses the net and increases the number of patterns.

A way out can be the usage of a "not spoken grapheme" as used in [5]. That would increase the number of phonemes by one, but it would decrease the possible combinations for the grapheme to phoneme mappings. In this example a "null-phoneme" could be added between the `[c]` and `[t]` and the not spoken grapheme "gh" can be mapped to it:

`th ou gh t`
`/ T c 0 t /`

7 CONCLUSION

In this paper a completely data driven approach for the generation of the training patterns for a neural network was described. The first step inserts tags in the grapheme strings of a usual phonetic lexicon to mark the assignments between the graphems and the phonemes. The second step counts the number of contradicting patterns for a given context to find out the optimal context width for the input of a neural network.

The grapheme to phoneme mapping shows different results for the two tested languages English and German. 99.2% of 305000 entries of the German CELEX could be completely mapped and 91.8% of 59000 entries of the English PRONLEX.

The determination of the optimal context width cannot be done completely automatically because there are too many contradicting demands. Therefore the required quality and the training or calculation speed have to be taken into consideration.

In the future work the mapping algorithm has to be improved. The less the maximal grouping is and the less the amount of grapheme to phoneme mappings, the less is the number of contradicting patterns which will lead to better results for the neural network.

REFERENCES

[1] Stefan Besling, Heuristical and Statistical Methods for Grapheme-to-Phoneme Conversion, *Proceedings KONVENS 94, Vienna*, pp. 23-31

[2] Andersen, Ove, and Dalsgaard, Paus, A Self-Learning Approach to Transcription of Danish Proper Names, *ICSLP 94*, pp. 1627-1630

[3] Terrence J. Sejnowski, and Charles R. Rosenberg, Parallel Networks that Learn to Pronounce English Text, *Complex Systems 1*, pp 145-168, 1987

[4] N. McCulloch, M. Bedworth, and J. Briddle, NETspeak - A Re-Implementation of NETtalk, *Computer, Speech and Language 2*, pp. 289-301, 1987

[5] Katrin Rosenke, Realisierung der linguistisch-phonetischen Transkription für die Sprachsynthese durch neuronale Netze mit Multilayer-Perceptron-Struktur, *Verlag Dr. Köster*, 1996