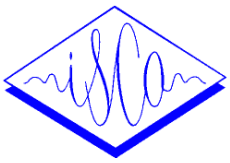

6th International Conference on Spoken Language Processing (ICSLP 2000)
Beijing, China
October 16-20, 2000

GRAPHEME BASED SPEECH RECOGNITION FOR LARGE VOCABULARIES*

Christoph Schillo, Gernot A. Fink, Franz Kummert
Bielefeld University, Faculty of Technology, 33594 Bielefeld, Germany
e-Mail: {schillo, gernot, franz}@techfak.uni-bielefeld.de

ABSTRACT

Common speech recognition systems use phonetically motivated subword units. To utilize words in these systems, one has to translate the available graphemic word representation into a phonetic one. To reduce this manual effort we propose to build grapheme based recognition systems. They can be used as speech interfaces for devices that can provide a graphemic representation of words like city names of navigation systems. Results of experiments on a 10,000 word lexicon of German cities are presented.

1. INTRODUCTION

Today most automatic speech recognition systems are limited to a fixed vocabulary of words. When utilizing these systems for new tasks usually out of vocabulary words will occur. The same problem can be observed during run time when introducing dynamically changing lexica. Because current recognition systems are working on subword units based on phonemes this requires a *phonetic* transcription of every newly encountered word. Therefore, additional effort arises which cannot completely be automated. However, for words in an augmented lexicon or for a new recognizer built from scratch usually the *graphemic* representations are already available. This leads to our experiments in building recognition systems based entirely on graphemes. Though the articulatory ambiguity of graphemes in German speech is not as high as in e.g. English, it still affects the performance [7]. Therefore, the accuracy of the graphemic recognizer will not be as good as a specialized phonetic one while the handling will be simplified. To use all advantages a combination of both systems is necessary: the phonetic recognizer handles standardized words and phrases and switches to the graphemic recognizer whenever a word from a large or dynamically specified list is likely to appear.

In this paper we present approaches to build grapheme based isolated word recognition systems for a 10,000 word lexicon, which is needed in e.g. navigation systems and directory assistance applications like proposed in [1] and [6]. But in contrast to them we focus on small scale computer platforms, i.e. DSP systems, a crucial requirement for an application that cannot be put on a desktop computer due to environmental restrictions.

*This work was partly founded by Böhme Datentechnik, Stuttgart, Germany

2. TASK

The chosen domain for our task is car based speech recognition. In this environment speech is an optimal interface for handling several devices without distracting the driver. Most important devices are the cellular phone, the car stereo, and the navigation systems. All three can provide names to control several features: the stereo can receive a list of currently available station names, the cellular phone contains a phonebook, and the navigation system includes cities and street names in it's maps. Although the control of the three devices is a similar task, the hardest by far is the navigation system with it's huge list of names.

When using the intended system, certain keyphrases are used to address the different devices and trigger the graphemic recognizer. For example to obtain a route the driver utters the phrase *'navigation: show me the way to ...'*. The next word in line is most likely a city or street name. So the graphemic recognizer receives the next sample for processing. Naturally, the graphemic recognition will have a less accurate performance, therefore, not only the best, but a list of n results will be presented, either via speech or the display of the navigation system. When the intended name is among the results the user can select it by saying it's number or by simply verbally interrupting the output when it is uttered. If all results are failures, the process can either be repeated or a spelling mode can be entered. Selection and rejection are handled by the more robust phonetic recognizer.

3. METHODS

We pursued two different approaches. The first one is a 'graphemic typewriter' which is supported by different kinds of language models (cf. figure 1 A), while the second one is a tree based recognition system with trigraphemic

subword units and access to the complete (graphemic) lexicon (cf. figure 1 B).

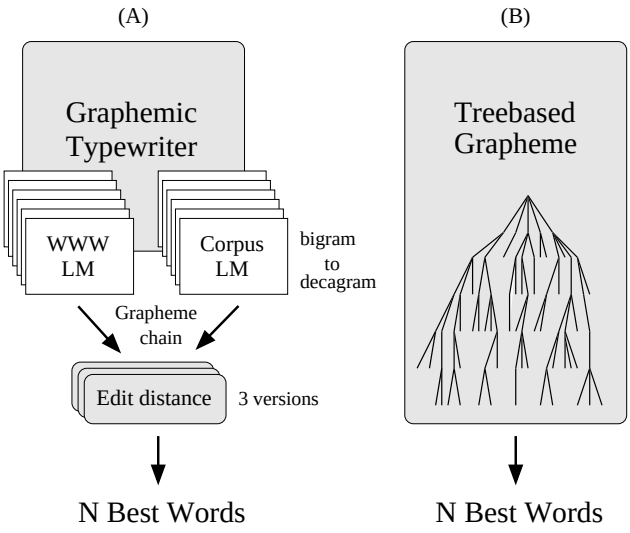

Figure 1: Two basic systems are compared: (A) a graphemic typewriter with language models of different origin and depth. (B) A lexicon tree based graphemic recognizer.

3.1. Graphemic Typewriter

Our first approach is a recognizer for context-independent graphemic units – a so called graphemic typewriter – with different statistical language models. The resulting grapheme chain is compared with the whole lexicon using a special distance measure. The best results are presented as the recognizer's output.

3.1.1 Language models

Two types of language models are considered. The first one is based on the recognizers' training material, the second one on the 10,000 city name lexicon with word frequencies equal to the importance of the city. To measure this importance an automatic World Wide Web search was used and the number of hits for each name on pages of German domains were counted. The higher the count is, the more important is the city. Afterwards this score table was purged to cope with names that are identical to common words or persons, like 'Eiche' (German for Oak) or 'Spielberg' (which was mistaken with the american director) and therefore produced too many hits. Furthermore, the counts were limited to the maximum count of 'Berlin', as the capital is supposed to be the most important city. Both language models ranged from bigram to decagram (cf. table 1 for results).

3.1.2 Lexicon mapping

Because the result of this recognition system is only a graphemic chain whereas the desired result is a word from the given lexicon of 10,000 cities, an additional processing step is necessary. The difference of the chain to all words is measured and the final output contains the best n results.

The distance measure for comparing the recognizer's word chain is based on the edit distance (or Levenshtein distance [4]), a dynamic programming approach in which both words are mapped, minimizing the overall cost function consisting of the number of substitutions, insertions and deletions. The standard method can only produce discrete values, which yields too many equally rated names from the lexicon. To obtain a more accurate and more continuous measure, a weight for every possible grapheme substitution is introduced. An additional empty grapheme ϵ makes it possible to express insertions as a substitution of ϵ by a grapheme and deletions as a substitution of a grapheme by ϵ. We used 33 graphemes and the ϵ symbol so 34 graphemes can possibly be substituted by 34 graphemes on every position of the word resulting in a 34 by 34 substitution weight matrix. The weight matrix was initialised using the actual output of the graphemic typewriter and the reference text. Every match of the two words with the same rating as the best match was considered, accumulating the counts for every substitution along the single mappings. These accumulated values were normalized and subtracted from 1 to get a proper weight. After iterating these calculations several times the weight matrix did not change anymore.

Another approach was tested, in which the diagonal of the weight matrix was fixed to a negative value after each iteration, giving the equality of graphemes in both words a competitive edge.

The advantage of the graphemic typewriter and an additional lexicon mapping is the strict separation of the speech recognition process and the lexicon search.

3.2. Tree-based recognizer

Our second approach is a tree-based recognizer. The whole lexicon is presented as a trigraphemic prefix tree and the result is an n-best list of actual names. No additional module like the sophisticated distance measurement has to be used. The tree is generated automatically based on the lexicon, but has to be represented in the system's main memory, therefore enlarging the overall memory needed. Nevertheless, the intended system does not need to recognize continuous words so no tree copies must be managed.

4. RESULTS

The framework of all experiments was the ESMERALDA development environment for statistical pattern recognition

systems [2]. We used 39 dimensional feature vectors consisting of the first 12 mel frequency cepstral coefficients (MFCC), the signal energy and the first and second time derivatives of these values within every 10 msec frame of the speech signal. The hidden markov models share 1009 gaussian distributions. As graphemes all 26 characters from the alphabet, the three German umlauts (*ä, ö, ü*), the sz ligature (ß) and the three simple translations of *sch, ch,* and *ck* to a single character were used. All graphemes were initialized by their most likely phonetic representation prior to the training.

4.1. Corpora

The training material was taken from the German *Verbmobil* corpus [3]. It contains more than 300,000 words of spontaneous speech taken from human-human appointment scheduling dialogues. The material was spoken by a total of 654 speakers. 6,266 uniq words provide a great number of different grapheme combinations, forming a good base for our experiments.

As test material 978 city names (306 uniq) uttered by 10 speakers were extracted from another German corpus, *ERBA* , which deals with train scheduling queries [5]. The complete *ERBA* corpus contains read material from 85 speakers and consists of 99,688 words from a 942 word lexicon. Compared to *Verbmobil* this corpus does not offer as much graphemic variety but contains more city names, making it an ideal test set for our experiments. Nevertheless, to get another training set besides *Verbmobil* we repeated all experiments with additional *ERBA*-material. The new training set contained 95,560 words from the 75 speakers not included in the test set.

4.2. Graphemic typewriter

The first experiments were carried out on the graphemic typewriter with different language models (cf. table 1). As mentioned before, the output of the typewriter is a graphemic chain which has to be mapped to the lexicon. Therefore the results of the typewriter cannot be measured in word accuracy (WA) but grapheme accuracy (GA), the number of correctly recognized graphemes subtracted by the sum of deletions, insertions and substitutions, weighted by the total number of graphemes in this word.

The GA without any language model (LM) is just 24.24%, because several German orthographical peculiarities like matching a single vowel to a double vowel, inserting 'h' after vowels to prolong them etc. prevent a better result. The results can be improved by using grapheme based language models.

We tried to utilize a training corpus based LM first (table 1, Corpus LM and figure 2, lower graph), but found that the test set perplexity of the LM for the 33 graphemes was too high, 21.59 for the bi-gram and more than 47 for the n-grams with $n > 4$. The increasing perplexity for higher

	Corpus LM		WWW LM	
n-gram	GA	(PPX)	GA	(PPX)
2	38.77	(21.59)	43.93	(13.52)
3	39.28	(31.86)	52.07	(9.02)
4	40.73	(43.08)	60.02	(6.25)
5	41.40	(47.97)	66.96	(4.82)
6	41.13	(48.14)	70.90	(4.22)
7	41.32	(47.77)	72.07	(3.99)
8	41.25	(47.51)	**72.89**	(3.93)
9	41.24	(47.44)	72.55	(3.93)
10	41.24	(47.42)	72.67	(3.93)
none	24.24			

Table 1: Grapheme accuracies (GA) of the grapheme recognizer without language model and with language models from bi- to decagram of different origin and the corresponding perplexity (PPX) for the 10,000 cities set.

n clearly shows that the LM cannot model the grapheme sequences for the task properly. Nevertheless the few restrictions from the corpus based LM enhanced the GA up to 41%.

A more appropiate LM was calculated on our 10,000 city set, weighted by the importance of the cities (table 1, WWW LM and figure 2, higher graph). The perplexity of the n-grams is declining for higher n and reaches a minimum of 3.93 for n-grams with $n > 6$, leading to a GA of 72.89%.

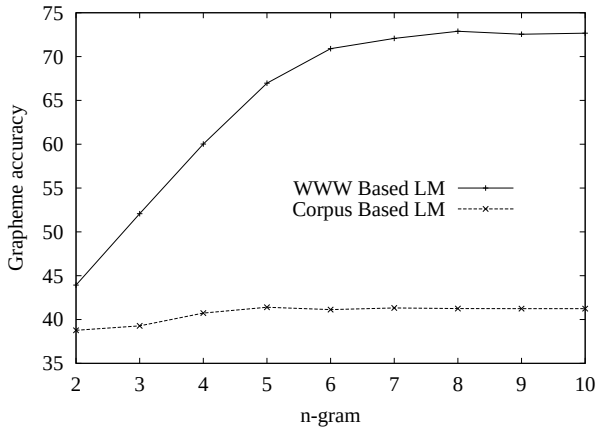

Figure 2: Grapheme accuracies of the graphemic typewriter with language models from bi- to decagram of different origin.

4.3. Full evaluation

For the second step of the recognition system, the lexicon mapping, we used the best result from the grapheme based

n best	WA Grapheme$*$ mapped	WA Grapheme$*$ mapped, negative diag	WA Tree *Verbmobil*
1	49.69	53.16	61.45
5	53.06	61.55	78.62
10	53.98	64.95	79.65

Table 2: Word accuracies (WA) on n-best recognition of the different graphemic systems, trained with *Verbmobil*

n best	WA Grapheme$*$ mapped	WA Grapheme$*$ mapped, negative diag	WA Tree *ERBA*
1	48.87	49.07	82.71
5	57.97	59.10	89.57
10	60.73	62.57	89.67

Table 3: Word accuracies (WA) on n-best recognition of the different graphemic systems, trained with part of *ERBA*

recognizer and LM combinations (WWW LM 8-gram with WA 72.89). As mentioned above the plain edit distant provided too many results, so only the two weighted versions were considered. The described simple weighted version reached a word accuracy of just 49.69% on the best result, though the grapheme accuracy was 72.89%. Unfortunately even with nearly three quarter of a name correctly recognized there are still some names from the lexicon that fit better than the intended one. Therefore, the best solution was not always among the best 5 or 10 hits, giving a WA of 53.06% and 54%, respectively.

The results could be improved by amplifying the grapheme equality at the expense of the substitutions by introducing a fixed, negative diagonal in the weight matrix. The WA for a single result is increased to 53.16, for the best 5 and 10 results to 61.55% and 64.95%.

When using a tree based recognizer trigraphemes are considered instead of monographemes, making the recognition more robust. Therefore, the recognition rate on the tree is 61% for the best result and nearly 80% for the 10 best results (cf. table 2).

All experiments were performed on the test corpus, too, that differed from our trainings corpus. When using the material from the test corpus, that contained speakers not in the test, better results can be achieved (cf. table 3). The *ERBA* based test set differs from the *Verbmobil* trainings set in recording condition, it uses a different vocabulary and read speech instead of spontaneous speech. It is obvious that the tree based recognizer of the *ERBA* trained system (WA up to 89.67%) is superior to the *Verbmobil* recognizer (WA 79.65) due to the similarity of training and test material. But when comparing the graphemic typewriters for both training sets, the systems show a nearly equal behavior, with a slight advantage for the *Verbmobil* material. By removing the higher level restrictions the greater number of training examples increase the WA.

5. CONCLUSION

We compared several approaches to build a grapheme based speech recognition system which is able to handle even large scale vocabularies without any transcription effort. Because of the articulatory ambiguity of graphemes in context these systems will not be able to reach the performance of a standard recognizer, though the accuracy loss can be compensated partly by expanding the output from one solution to an n-best list. The results show the feasibility of our approach, though a test on very large vocabulary has to be done.

6. REFERENCES

1. B. Buntschuh, C. Kamm, G. Di Fabbrizio, A. Abella, M. Mohri, S. Narayanan, I. Zeljkovic, R. D. Sharp, S. Marcus, J. Shaffer, R. Duncan, and J. Wilpon. VPQ: A spoken language interface to large scale directory information. In *International Conference on Spoken Language Processing*, pages 2863–2866, 1998.
2. G. A. Fink. Developing HMM-based recognizers with ESMERALDA. In V. Matoušek, P. Mautner, J. Ocelíková, and P. Sojka, editors, *Lecture Notes in Artificial Intelligence*, volume 1692, pages 229–234, Berlin Heidelberg, 1999. Springer.
3. K. Kohler, G. Lex, M. Pätzold, M. Scheffers, A. Simpson, and W. Thon. Handbuch zur Datenaufnahme und Transliteration in TP 14 von VERBMOBIL – 3.0. Technical Report 11, Institut für Phonetik und digitale Sprachverarbeitung, Universität Kiel, 1994.
4. V. I. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Cybernetics and Control Theory*, 10:707–710, 1966.
5. S. Rieck. *Parametrisierung und Klassifikation gesprochener Sprache*. PhD thesis, Lehrstuhl für Informatik 5 (Mustererkennung), Universität Erlangen-Nürnberg, 1994.
6. H. Schramm, B. Rueber, and A. Kellner. Strategies for name recognition in automatic directory assistance systems. *Speech Communication*, 31:329–338, 8 2000.
7. E. G. Schukat-Talamazzini, H. Niemann, W. Eckert, T. Kuhn, and S. Rieck. Automatic speech recognition without phonemes. In *Proc. European Conf. on Speech Communication and Technology*, pages 129–132, Berlin, 1993.