A COMPARISON OF HTK, ISIP AND JULIUS IN SLOVENIAN
LARGE VOCABULARY CONTINUOUS SPEECH
RECOGNITION

Tomaž Rotovnik, Mirjam Sepesy Maučec, Bogomir Horvat, Zdravko Kačič

Faculty of Electrical Engineering and Computer Science, University of Maribor,
Smetanova 17, 2000 Maribor, Slovenia
tomaz.rotovnik@uni-mb.si, mirjam.sepesy@uni-mb.si

ABSTRACT
In this paper recognition results from different speech decoders are presented for Slovenian large vocabulary speech recognition task. For speech recognition two different types of lexica and language models were used. Word based models were used for baseline system and sub-word (stems and endings) based models for comparison. For all decoders a two-pass decoding strategy was used. With all three decoders better recognition results were achieved using stem-ending models (3% absolute on average). Experiments also showed slightly better recognition results with Julius decoder, as opposed to two other decoders, and improvement of real-time factor for 65%.

1. INTRODUCTION
Slovenian language is, like other Slavic languages, a highly inflectional language. Its rich morphology causes a big problem in Large Vocabulary Continuous Speech Recognition (LVCSR). The experiment on newswire corpora has shown that the Slovenian language requires a magnitude larger vocabulary than English in order to achieve the same text coverage. In Slovenian language many different word forms can be derived from the same root form.

At the moment the most advanced systems for speech recognition can handle from 20K to 60K of words; thus the only possible thing to do in case of inflectional language would be to restrict the vocabulary size. This would however result in a high Out Of Vocabulary (OOV) rate. Inflectional change to the word mostly affects word ending, whereas stem remains unchanged. Smaller lexical units (stems and endings) were used, to solve the problem with high OOV rate [1].

For speech recognition two different types of language models were used: word-based language model and model at a sub-word level (named morphological model). The results of experiments using three different decoders are reported. To compare decoders, a two-pass decoding strategy was used for all of them. All experiments were performed on the SNABI speech database [2].

2. ARCHITECTURE OF DECODERS
Three different decoders were used for recognition: HVite from HTK toolkit [3], trace projector from ISIP toolkit [4] and Julius decoder [5]. Some important characteristics and distinctions of the decoders will be presented. For the first recognition pass they all used a derivative of standard time-synchronous Viterbi beam search decoder with pre-compiled static recognition network [6]. Viterbi beam search belongs to a class of breadth-first dynamic programming techniques, where all hypotheses are parallelly pursued and gradually pruned as the correct hypothesis arises with its maximum score. Recognition system can be treated as a recursive transition network composed of the states of acoustical models, in which any state can lead to the acquisition of another state. Viterbi beam search is time-synchronous because all partial hypotheses, formed in the search, finish at the same time. ISIP and HTK decoders used word-graph algorithm at the second pass. Word graph is constructed as a result of keeping more then 1-best hypothesis at first-pass. First-pass defines possible word strings, which can be used as grammar constraints in the second pass decoding. Because the search space is strictly constrained at the word level by the word graph, second pass (word graph rescoring) is much more efficient than the Viterbi beam search decoding. Although the process of word-graph generation is very lengthy, using more complex acoustic and language models will in the long run give a better performance in accuracy.

HTK decoder uses token-passing implementation of the Viterbi algorithm [7]. First pass N-best recognition was successfully implemented with word-pair approximation [8]. Second pass uses word graph algorithm (HLRescore) with trigram or fourgram language models. To minimize parallel memory resources three different pruning techniques [9] were used. Conventional beam pruning is performed at the model level. Second pruning limits model instances and is called maximum model pruning. Third pruning technique reduces word-ends (word-end pruning) and limits the number of them in each frame. Decoding algorithm is written in ANSI C program language.

Decoder from ISIP is called trace projector. It includes similar algorithms and techniques to those, mentioned above [4]. To reduce search requirements a tree-structured representation of the lexicon is used. This limits the information of successive words until word-end point is reached. Language factoring technique was used to avoid this limitation [10]. At the first pass Viterbi beam searching algorithm with the tree lexicon and in the second pass word graph rescoring with trigram language models is used. The basic difference between this decoder and the HTK decoder is the use of lexical tree, which reduces the total number of active models at the beginning of recognition (faster...
recognition). Another distinction is in program language structure. ISIP toolkit is built with C++ program language which includes usage of classes (higher memory requirements). Trace projector also supports beam pruning on all three levels (state, model, word), which enables a more precise setting of pruning parameters.

Julius decoder (C language implementation) uses a different approach than the other two decoders for two-pass recognition. At the first pass it uses word-trellis index method as described in [5]. This means that in every time-frame all remaining hypotheses within the beam are kept. So, context dependency can be handled on the later pass. During the compilation of the recogniser’s source code the 1-best approximation is chosen instead of the more time expensive word-pair approximation. 1-best approximation involves acoustic and language score errors, which are minimized in the second run. The second pass performs a best-first stack decoding search [11] in backward direction, using the word trellis index as both heuristics and word prediction. Decoder also enables the use of Phone-Tied Models (PTM) [12]. With these models recognition is faster but a little less accurate.

3. SPEECH AND TEXT CORPORA
The SNABI speech database [2] includes speech of 52 speakers where each person read in average more than 200 sentences and 21 speakers read also the text passage of 91 sentences. The complete database consists of approx. 14 hours of speech. For training language models [1], corpus of 60M words was used. It was obtained from the archives of the Slovenian newspaper VEČER, spanning from the period 1998 through 2000.

4. ACOUSTIC MODELS
Acoustic models were based on Hidden Markov models (HMM) with three emitting states and a left-right topology [3]. An acoustic optimiser based on expectation-maximisation (EM) with Baum-Welch algorithm was used for rough parameter estimation. Basic procedure for building context dependent models is shown in Fig. 1. 10% of all pronunciations were randomly extracted from the training set in order to build basic context-independent models. These models were iteratively trained from one mixture component to 32 mixture components, and then used to generate phone-level alignments. After aligning, a complete training set was used to build better context-independent models. Context-dependent phone models were seeded with better single mixture mono-phones and then state-tied to cluster those states and models, which were statistically similar. State-tying led to better parameter estimates, as the entire model clusters were seen in the training set for a sufficient number of times. It also allowed generating models for unseen contexts.

5. EXPERIMENTS AND RESULTS
5.1. Data preparation
Training data included 14786 pronunciations (sentences and isolated words). Test set consisted of 779 pronunciations spoken by 7 speakers, which presented 7% of all utterances. For the evaluation we used 12 FFT-derived cepstral coefficients and log-energy. Utterances were first pre-emphasized with FIR filter to compensate attenuation caused by the radiation from the lips. FFT features were computed using a 10 ms analysis frame and a 25 ms Hamming window. The first and the second derivative coefficients of the cepstral coefficients were appended to produce a thirty-nine-dimensional feature vector.

5.2. Language models and lexica
For recognition we used bigram, trigram, reversed trigram and fourgram backoff language models. Trigram, reversed trigram and fourgram versions were used only for rescoring, whereas bigram language model was used for generating word graphs and word trellis index. Words in pronunciation dictionary were evaluated from 20000 most common words in text corpora Večer. Word transcriptions were made automatically under basic grammatical principles using 30 phones. The number of phones was smaller than usual, because we did not differ between long and short vowels and also excluded some rare phones. Words were decomposed into smaller units based on stems and endings for the second set of experiment. 8497 different basic units were obtained. Out-of-vocabulary rate was decreased as can be seen in Table 1.

<table>
<thead>
<tr>
<th>Vocabulary</th>
<th>Number of units</th>
<th>OOV [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Word based</td>
<td>20000</td>
<td>22.26</td>
</tr>
<tr>
<td>Stem based</td>
<td>8662</td>
<td>6.5</td>
</tr>
</tbody>
</table>

Table 1: Vocabulary size and out-of-vocabulary rate

Table 2: Stems and endings in 20K-words vocabulary

Word-internal triphone models with 16 Gaussians mixtures for each state and each model were built for all recognition decoders. Modified Perl script described in [13] was used for training. Basic models had to be expanded with all unseen biphones for the stem-ending based models.
When decomposing words into stems and endings and using word internal triphones (biphones and monophones included) from word-based vocabulary, it is possible to acquire new biphones, as shown in Table 3. This way all missing biphones were created and added to the basic triphone models. The total number of models was 5983. The number of states was reduced from 10K to 3K after performing tree-based state tying. 4090 tied-state triphone models were obtained. These acoustical models were used for experiments with the HTK and the ISIP decoder. They were used for both vocabulary structures. The Julius decoder has different handling for short-pause model; it does not support short-pause (sp) model with transition from start to end node, but it supports transparent word handling for fillers, which are included in the language models. This was due to the characteristics of Japanese language. Because of the fact that in Slovenian language there is no large corpus that would include fillers, short pauses, notational symbols, a different approach was used. The extended model with minimum one transition for sp model was applied. Sp model was added as the last phone part of words in vocabulary. This can cause possible new errors: between two successive words in search space, at least one transition across sp model must be performed. This error was not tracked, as seen from experiments. Phone-Tied Models (PTM) were also generated. First monophone models with 64-mixture Gaussians were evaluated. Information about tied states from triphone models was then used to create PTM models [11].

5.3. Results

Experiments were divided into two parts. In the first part word-based models were used. Results are shown in Table 4. Low recognition accuracy was achieved. OOV words produced a major source of recognition errors. The second source of errors were different word forms derived from the same lemma, which are phonetically very similar.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>HTK 1</td>
<td>41.62</td>
<td>70</td>
<td>9.66</td>
<td>42.05</td>
<td>45</td>
<td>6.9</td>
</tr>
<tr>
<td>HTK 2 – 3</td>
<td>44.16</td>
<td>200</td>
<td>15.18</td>
<td>42.24</td>
<td>155</td>
<td>11.0</td>
</tr>
<tr>
<td>HTK 2 – 4</td>
<td></td>
<td></td>
<td></td>
<td>45.03</td>
<td>530</td>
<td>11.1</td>
</tr>
<tr>
<td>ISIP 1</td>
<td>43.7</td>
<td>122</td>
<td>9.55</td>
<td>44.0</td>
<td>122</td>
<td>7.9</td>
</tr>
<tr>
<td>ISIP 2 – 3</td>
<td>44.6</td>
<td>424</td>
<td>13.41</td>
<td>48.0</td>
<td>380</td>
<td>11.9</td>
</tr>
<tr>
<td>Julius 2 – 3</td>
<td>44.7</td>
<td>247</td>
<td>6.3</td>
<td>48.6</td>
<td>178</td>
<td>3.6</td>
</tr>
<tr>
<td>Julius PTM 2 – 3</td>
<td>41.03</td>
<td>243</td>
<td>3.2</td>
<td>43.79</td>
<td>169</td>
<td>2.1</td>
</tr>
</tbody>
</table>

Table 4: Results of continuous speech recognition experiments for three different decoders

HTK decoder with one-pass strategy achieved 41.6% accuracy. Two-pass strategy improved accuracy by 2.5% absolute against memory consumption and real-time performance (increased by 57%). ISIP achieved better results in both strategies with slightly better real-time performance; however memory requirements were larger (Fig. 2) by a factor of 2. The most probable reason was the use of C++ classes to implement memory handling. Julius decoder uses two-pass procedure as default. The best recognition results and also very fast recognition (6.3 RT) was achieved with expanded acoustical models.

![Figure 2: Memory consumption](image)

When compared to the slowest recognition time (HTK 2-3), Julius was 140% faster (Fig. 3). The fastest recognition time was achieved with PTM models (3.2 RT); however recognition accuracy was not as good (only 41%). Second part included stem-ending based models. We predicted that if vocabulary was smaller then search space would also be smaller. This means smaller memory requirements and faster recognition times.
Also better accuracy was expected because of a smaller OOV rate. But the result was a bit different than expected. HTK decoder delivered worse recognition results with trigram stem-ending models than with word based models. Results were better when four gram stem-ending language model was used.

The reason is connected to HLRescore, which performs only language rescoring, while ISIP and Julius also consider acoustical information in second pass. The difference can be seen in Table 5. The first row presents lattice error rate after the first pass. In next rows recognition results using cross combination of ISIP and HTK decoders are displayed. The best recognition result (49.5%) was achieved using HTK decoder in the first pass and ISIP decoder in the second.

<table>
<thead>
<tr>
<th>First pass</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Lattice error [%]</td>
<td>ISIP</td>
<td>HTK</td>
</tr>
<tr>
<td></td>
<td>37,1</td>
<td>32,6</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Second pass</th>
<th>ACC(%)</th>
<th>ACC(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISIP - 3</td>
<td>48,0</td>
<td>49,5</td>
</tr>
<tr>
<td>HTK - 3</td>
<td>44,34</td>
<td>42,24</td>
</tr>
<tr>
<td>HTK - 4</td>
<td>46,04</td>
<td>45,03</td>
</tr>
</tbody>
</table>

Table 5: Results with cross combination of ISIP and HTK decoder

The best real time performance was achieved by Julius decoder (3.6RT). Julius, using PTM models with stem-ending models, gave the best real-time factor in all test experiments (2.1RT), while recognition accuracy was somewhere in the middle according to other results in the second part of experiments. Final recognition results are presented in Fig. 4.

6. CONCLUSION

This paper presented results of LVCSR for Slovenian language using three different decoders with two different types of language models. As seen by the results, Julius has met most of the requirements, important for large vocabulary continuous speech recognition: accuracy, memory consumption and real-time performance. Recognition performed at the sub-word level in Slovenian language showed many advantages over word-level recognition. As shown in the experiments with sub-word level models, it established faster recognition time, less memory consumption and higher recognition accuracy, while the OOV rate significantly improved.

7. REFERENCES