論文 / 著書情報
Article / Book Information

論題(和文)	
Title(English)	An Empirical Comparison of Sphinx and HTK models for Speech Recognition
著者(和文)	JOSEF R NOVAK, DIXON PAUL RICHARD, 古井 貞熙
Authors(English)	Josef R. Novak, Paul R. Dixon, Sadaoki Furui
出典(和文)	日本音響学会2010年春季講演論文集, , No. 2-6-9, pp. 73-74
Citation(English)	, , No. 2-6-9, pp. 73-74
発行日 / Pub. date	2010, 3

An Empirical Comparison of Sphinx and HTK models for Speech Recognition*

◎ Josef R. Novak Paul R. Dixon Sadaoki Furui
Tokyo Institute of Technology, Tokyo, Japan
{novakj,dixonp,furui}@furui.cs.titech.ac.jp

1 Introduction

It is often challenging to develop high quality baselines for ASR experiments. This can sometimes make it difficult to effectively evaluate the quality of new contributions, or the real gains that can be expected from new methods. Nevertheless it is clearly of interest to provide comparisons against a robust baseline, and it is also intriguing to compare our WFST-based decoder against the well-known HTK and Sphinx3 engines on an even footing. In this paper we investigate the relative performance of the T^3 decoder [5] on the November 1992 ARPA WSJ test set, and make an empirical comparison with HTK HDecode and Sphinx3. We compare the T^3 decoder against previously reported baselines, utilizing the same open source WSJ acoustic [2] and the standard WSJ languages models. Through the use of T^3 as a common decoding engine we are also evaluate both HTK and Sphinx acoustic models independent of their respective decoding engines.

The remainder of the paper focuses on the details of the experimental setup, the basic construction techniques employed to build the WFST cascades and the experimental results. The paper is structured as follows: Section 3 describes the HTK and Sphinx models, Section 4 describes the WFST cascade configuration, Section 5 outlines the experimental setup and reports the results and Section 6 provides additional analysis and concludes the paper.

2 Shared Knowledge Sources

In order to minimize the workload and help guarantee the repeatability of our experiments, we rely on the open-source, pre-tuned acoustic models described in [2], and the standard WSJ language models and pronunciation dictionaries. All of the HTK and Sphinx setups discussed below employed the WSJ 5K non-verbalized 5000 word closed vocabulary set in combination with the WSJ standard ARPA format 5K non-verbalized closed bigram language model.

In both case of Sphinx and HTK, the acoustic models were trained using the full set of WSJ0 and WSJ1 training data (211 hours). This included the long-term and journalist training data. In the case of T^3, in addition to the existing HTK conversion tool, a new tool was developed to convert arbitrary Sphinx format acoustic models into a format suitable for use with the T^3 decoder.

3 HTK and Sphinx

HTK and Sphinx are two of the more widely used open source speech recognition toolkits, and both have enjoyed considerable popularity over the years. Both of these toolkits contain tools for training acoustic models, and while these models are similar there are some differences that are worth mentioning. The most salient differences are that Sphinx trains positional triphones by default, and HTK trains back-off context-dependency phones. Furthermore Sphinx supports fairly fine-grained specification of the number of tied states or senones, whereas HTK does not. Finally, HTK supports training silence models with an independently variable number of Gaussians. As described in [2], the HTK and Sphinx models which were trained for this comparison were trained so as to produce comparable models, as far as possible given the above constraints and inherent differences in the respective training procedures. Both the HTK and Sphinx models employed roughly 8000 tied states, referred to as senones in the case of Sphinx. Both models used 32 Gaussian mixture models. This number of mixture models was chosen primarily because previous experiments showed that selecting a larger number provided little gain, while it did have a significant impact on the RTF for HDecode and Sphinx3.

The HTK and Sphinx projects both provide several different options for decoding. HTK provides HVite, as well as the more recent HDecode. Sphinx provides Sphinx2, Sphinx3, PocketSphinx, and Sphinx4. For these experiments we chose to focus on HDecode and Sphinx3. HDecode provides significantly better performance than HVite in terms of RTF, while Sphinx3 provides the best trade-off between speed and flexibility, and is perhaps considered the flagship decoder in the project. PocketSphinx has been shown to be faster in certain scenarios, but this generally involves the use of special "semi-continuous" acoustic models which tend to reduce accuracy.

4 T^3 WFST Setup

The WFST cascades for the HTK and Sphinx acoustic models were both constructed using the same input knowledge sources as were used as input to HDecode and Sphinx3. Each of these knowledge sources was compiled into a WFST, where G is the language model, L is the lexicon, and C represents the context-dependency. A generic conversion tool which was written to convert arbitrary ARPA format language models to WFSTs was employed to convert the WSJ 5K non-verbalized closed bigram LM according to the strategy outlined in [6]. This tool utilizes regular epsilon transitions as opposed to failure transitions, in order to represent back-off arcs in the resulting network. The use of regular epsilon transitions has the potential to introduce an element of non-stochasticity into the cascade however, other work such as [6] as well as our own experience indicate that in practice this has a negligible impact on both word accuracy and RTF. The lexicon and context-dependency transducers were constructed using standard techniques, as described in [3]. Silence modeling was implemented by introducing an optional "SIL" or "sp" loop to the lexicon transducer. In the case of the HTK cascade this consisted in an auxiliary self loop located at the start state, whereas in the case of the Sphinx-based cascade this consisted

*音声認識のための Sphinx と HTK 音響モデルの実験的比較
ノバック・ジョセフ、ディクソン・ポール、古井貞熙

in an additional, non-epsilon closure arc. Based on previous results from [3], as well as our own recent experiments in [4] all compilation, composition and optimization operations were conducted in the log semiring. The complete, exact series of operations is described below,

$$\pi(det(C \circ det(L \circ G))) \ .$$

The composition operation is denoted by $\circ$, determinization is denoted by det, and π represents the auxiliary symbol replacement operation. As denoted above, no final minimization step was performed on the final network. Minimization of the final network can sometimes lead to a significant size reduction, however in practice we have found this to rarely be significant enough to justify the unpredictable runtime, and cost to the RTF which can occur when the weights are pushed in a suboptimal way. The final HTK based cascade contained approximately 690K states and 2.35M arcs, whereas the Sphinx cascade was slightly larger at 720K states and 3.25M arcs. The difference in the number of states and arcs between the two cascades can be explained primarily by differences in the acoustic models, where the Sphinx acoustic models possessed a larger number of physical triphones, and the slight differences in the silence modeling implementations.

5 Experiments

All of the LVCSR evaluations were run on an Intel Core 2 based machine running at 3GHz with 6MB of cache and 4GBs of main system memory. The platform was a 64bit Linux machine and the T^3 decoder [5] was compiled for 64bit using the `g++` compiler, in order to allow the GPU to be fully exploited. For each search network the beams were run at widths from 90 to 380 at intervals of 10, while the band, language model weight and insertion penalty were set to 10000, 15 and 20 respectively. In the case of HDecode and Sphinx3, these auxiliary parameters were implemented exactly as recommended in [2].

In addition to the varied beam settings, each of the WFST cascades was evaluated both with GPU acoustic likelihood support enabled and with all computations performed on the CPU using only a single thread. In total 6 different evaluations were performed, the results of which can be seen in Figure 1.

The accuracy of HTK and Sphinx3 decoders is in-line with the results reported in [2]. When using the Sphinx models with the T^3 decoder, a significant increase in accuracy was observed in comparison to the baseline, this indicates that further tuning of the Sphinx3 decoder parameters could increase the accuracy, nevertheless it is also likely that this would have an added negative impact on the RTF.

The highest accuracy system was the GPU accelerated T^3 decoder using the Sphinx models. When enabling the GPU, both Sphinx and HTK models exhibited very similar RTF characersics, and this trend was similarly reflected in the case of the CPU only configurations.

The task featured clean speech and relatively complex acoustic models in comparison to the bigram language model and small vocabulary size, and this helps to explain the substantial speed gains which can be seen in the results for the GPU accelerated configurations. The GPU accelerated systems also exhibited increased accuracy due to the full logsum being performed in computing the mixture scores. In contrast, in the case of the CPU only configurations the less computationally intensive, yet somewhat less accurate logmax operation was employed.

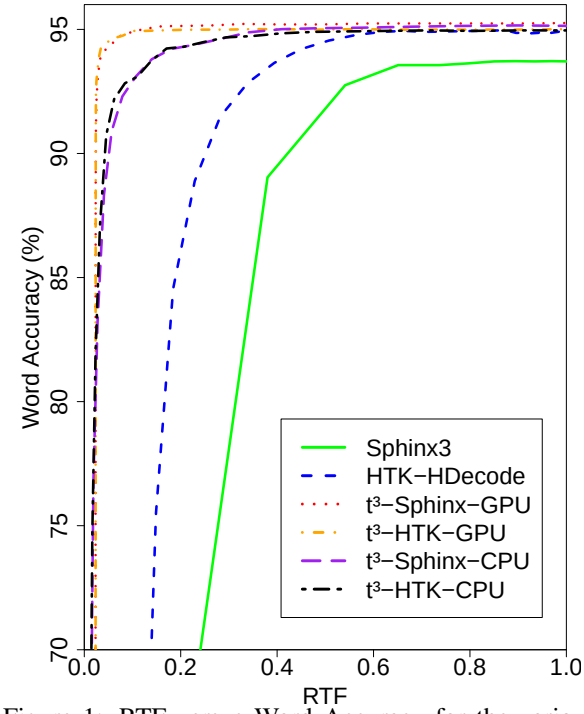

Figure 1: RTF versus Word Accuracy for the various decoders and acoustic models.

6 Conclusion and Future Work

In this paper we have presented an empirical comparison of the T^3 decoder alongside HDecode and Sphinx3 decoders. In all of our evaluations the T^3 decoder performed best in terms of both accuracy and speed. We have shown that an important feature of the T^3 decoder is the ability to operate on HTK or Sphinx models with comparable performance, and we have further confirmed the supplementary gains that may be achieved through use of the GPU.

In future work we plan to conduct further exhaustive comparisons by comparing even more decoders across a larger set of tasks. In particular it will be advantageous to look larger vocabulary tasks in order to better determine the reliability of the gains observed in these experiments. If the Sphinx and HTK models generate complementary errors we plan to investigate a tightly coupled combination T^3 system that can operate on both sets of models.

References

[1] S. Young, G. Evermann, D. Kershaw, G. Moore, J. Odell, D. Ollason, V. Valtchev, and P. Woodland. *The HTK Book (for HTK Version 3.2)*. Cambridge University Engineering Department, 2006.

[2] K. Vertanen. *Baseline WSJ Acoustic Models for HTK and Sphinx: Training Recipes and Recognition Experiments*. Cavendish Laboratory, University of Cambridge, 2006.

[3] C. Allauzen, M. Mohri, M. Riley and B. Roark *A generalized construction of integrated speech recognition transducers*. Proc ICASSP, pp. 761–764, 2004.

[4] P. R. Dixon, J. Novak, T. Onishi and S. Furui *Recent Evaluations of a WFST-Based Speech Recognition Decoder*. IEICE Tech. Report, SP2009-78, 2009.

[5] P. R. Dixon, D. A. Caseiro, T. Oonishi and S. Furui *The Titech Large Vocabulary WFST Speech Recognition System*. Proc. ASRU, pp. 1301–1304, 2007.

[6] C. Allauzen, M. Mohri, and B. Roark *Generalized Algorithms for Constructing Language Models*. Proc. ACL, pp. 40–47, 2003.