Can a syntactic language model be incorporated into an HMM-based ASR framework?

Johan Hall

January 17, 2005

1 Introduction

Automatic speech recognition (ASR) is the process which a computer interprets human speech into some kind of meaningful representation. Usually, the task for the automatic speech recognition is to identify the correct word-sequence. Speech recognizer is used in different applications for example simple command-control programs, transcription and speech understanding.

A good automatic speech recognizer should be able to recognize spontaneous continuous speech and should not require the speaker to break up their speech into discrete words. The task of recognizing speech from newspapers or news broadcast with a state of the art speech recognition systems can be done with accuracy higher than 90 percent. The accuracy drops significant when using spontaneous speech, due to the fact that the acoustic and language models usually have been built using written language or speech from written language [9].

This term paper is a literature review in the area of automatic speech recognizer and we will investigate different techniques to incorporate syntactic information into the speech recognizer. How can a speech recognizer benefit from syntactic information? Is it possible to integrate a parser?

The paper is organized as follows. We begin with a brief overview of automatic speech recognition. Section 3 investigates different methods of integrating language models. Finally, the paper ends with conclusions.

2 Automatic speech recognition

Speech recognition is a complicated task and have to deal with many dimensions of difficulties. The first step is to extract a number of features from the acoustic signal. The feature extraction has to be robust to acoustic variation but sensitive to linguistic content. Hence, the recognizer has to deal with various kinds of acoustic environments (surrounding noise and quality

of the microphone) and different speaker (genders, ages and health conditions). The speech signal is cut up in frames, a time slice around 10 ms. For each frame a feature vector is extracted with appropriated parameters [10].

The feature vectors are handed over to the classification module, these features are used to differentiate among the phonemes that are spoken for each word. Furthermore, the phonemes allow the recognizer to identify words.

The state of the art speech recognition systems of today use Hidden Markov Models (HMMs) to model this classification module. HMM is a statistical model used in many applications in Language technology. Markov models are state-space models that can be used to model a sequence of random variables that are not necessarily independent. The probability of each state is only dependent on immediately preceding state. In an HMM, we do not know the state sequence that the model passes through, but only the output sequence which is a probabilistic function of it. The generation of a random sequence of states $Q = \{q_1, q_2...q_n\}$ is then the result of a random walk in the chain and of a draw at each visit of a state [10, 14].

In speech recognition the statistical model consists of an acoustic model and a language model.This term paper will concentrate on the language model and how syntactic information can help the speech recognizer, see next chapter. First a brief overview of the acoustic model is done.

2.1 The Acoustic model

The feature analysis of the speech signal continuously generates feature vectors, which are fed into a recognition process of matching with reference patterns. Usually the acoustic models are HMMs trained on sub-word units such as phonemes (about 44 phonemes are used to represent all English words). A sequence of phonemes builds up a word in a pronunciation dictionary. It can contain information about different pronunciation variants of the same word [10].

The realization of one and the same phonemes depends on its neighboring phones. Therefore the context needs to be incorporated into models. Triphone models are often used, which consider the preceding and succeeding phone.

An automatic speech recognition system (with or without a language model) operates in two phases. It must first train the models, during this phase the system learns the reference patterns representing the different speech sounds and store the model in an appropriate representation. Usually probability for each distinct example is stored. Second phase, the recognition phase, during which an unknown input pattern is identified by using the stored models.

3 The Language model

For small vocabularies, the acoustic models can be enough to recognize the speech with sufficient accuracy. However, for large vocabularies the system needs to take linguistic knowledge into account. The purpose of a language model is to allow all possible word sequences, but penalize the word sequence which is incorrect or impossible in the language. A common way to modelling various natural language phenomena is to estimate the distribution among them, to capture regularities in a Statistical Language Modelling (SLM). Let a sequence of K words be denoted by $W = w_1, w_2, ..., w_K$ and the sequence probability $P(W)$.

$$P(W) = \{P(w_1, w_2, ..., w_K) = \prod_{k=1}^{K} P(w_k|w_1, ..., w_{k-1})\}$$

This probabilistic language model will generate a huge number of probabilities and the training data available isn't enough to build adequate model. Instead more simplified n-gram model is usually implemented, where n is a small number and $n-1$ denotes how many previous words should be included in the model.

$$P(W) = \prod_{k=1}^{K} P(w_k|w_{k-N+1}, ..., w_{k-1})\}$$

Usually bigram ($n = 2$) models or trigram ($n = 3$) models are used [10].

In a typical application, the purpose of an n-gram language model may be to constrain the acoustic analysis, guide the search through various (partial) text hypotheses, and/or contribute to the determination of the final transcription [1, 2]. The SLM techniques using n-grams are the most popular choice. These approaches do not take advantage of the fact that what is being modelled is language, it may as well be a sequence of arbitrary symbols [15]. Furthermore, n-gram language models are fast and robust and can be seen as weighted finite state automata. However, n-gram modelling have problems to capture even relatively local dependencies that exist beyond scope of model [16].

The research community have proposed many solutions to exploit linguistic knowledge for enhancing or replacing the n-gram language models.

To compare different language models the researchers often use the term perplexity, which is a measurement of quality of a given language modelling technique. Perplexity can be interpreted as the average branching factor of the language according to the model, a value between 1 and infinitely large. The better the model, the lower the perplexity [15].

3.1 Integration into the recognizer

There are different ways to integrate the language models with speech recognition systems. Harper suggest a classification of three categories [7, 8]:

- Tightly-coupled: the language and acoustic models are highly integrated and not separable, which makes it difficult to evaluate the models independently. It is hard to scale up the the system and it tends to be hard to understand. However, a tightly-coupled design can directly reduce the search space of the acoustic model.
- Loosely-coupled: the language and acoustic models are developed in isolated modules, which makes it easier to train and evaluate them independently. A difficulty of this design is that it is hard to determine how the models should interact with it each other and the language model cannot directly reduce the search space of the acoustic model.
- Semi-coupled or moderately integrated, falls in between the previous two and the language models can be used to guide search in the acoustic models. Problem with this design is that it may require communication in both direction and can be difficult to engineer.

Wachsmuth report another categorization of different couplings or integration of parsing [17]:

- Two-stage: a view of the system as a loosely-coupled two-stage process. First a classical HMM-based recognizer generates n-best hypotheses. In the second phase the parser rescores the results.
- Compiled: constraints imposed by a grammar is complied into the recognizer, usually finite state machines. This approach is tightly-coupled and has the same properties.
- Word-prediction, make use of a parsing algorithms. The parser can be used to predict allowed successor words for a given word sequence.
- Word-verification: the recognizer is synchronized on word-level with the parser. Every extension of a hypothesis chain is scored by the parsing module.

3.2 Parsing

To develop a speech recognition system which makes use of syntactic information produced by a parser in a tightly-coupled or semi-coupled fashion, it has to meet few requirements [17]:

- an efficient interaction between the parser and recognizer has to be developed.
- the parser must be able to process the input incrementally, which means that it can only generate results depending on information that has been generated so far.
- to combine statistical and declarative constraints, a scoring mechanism has to be developed for the parser which has a good match with the statistical recognizer.

Kita and Ward have tried semi-coupled approach to incorporate LR parsing into the SPHINX speech recognition system in the early nineties. LR parsers read their input from left to right and generate a rightmost derivation and perform two kinds of actions: shifting a symbol onto the stack and reduction of the stack without reading any symbol. The parser is guided by a parse table. SPHINX uses HMMs of phonemes and words (built up by concatenating phoneme models). The SPHINX-LR system incorporates a stack to keep track of constituents built up during the parse. A stack is associated with each path and the path is abandoned if the system fails to update the stack, which means that no rule has been found in the parse table. The SPHINX-LR system improved the sentence accuracy from 69.3% to 78.7% compared with the original SPHINX system, but the word accuracy droped from 94.1% to 85.5% [13].

Zue and Goddeau continue the work on using LR parsing and suggest a probabilistic LR parser. The recognizer keeps track of partial sentence hypotheses in a priority queue. At each step, the highest scoring hypothesis is dequeued and sent to the parser, which produces a list of next word candidates with associated word probabilities [19].

Harper have developed a speech recognition system which is divided into three loosely-coupled modules. The first module handles the acoustic and prosody processing, which selects word candidates and produces a word graph (directed acyclic graph representing the possible word paths through the utterance, are generated by postprocessing the word lattice). The prosodic module rules out word candidates with unlikely stress and duration patterns and annotates the word graph with information. The second module consists of a Constraint Dependency Grammar (CDG) parsing mechanism, which employs constraint propagation to prune word graphs. An Spoken Language Constraint Network (SLCN) represents all possible parses for the sentence hypotheses in a compact form, and is operated on by

constraints based upon syntactic, lexical, semantic, prosodic and pragmatic information. The final module merges the previous two modules to obtain the best sentence hypothesis, by annotating the word graph likelihood information [7, 11].

Jurafsky are using a probabilistic Earley parser and Stochastic Context-Free Grammar (SCFG) to produce word transition probabilities at each frame for a Viterbi decoder [12]. The Earley parsing algorithm has a cubic time complexity in worst-case scenario on the input length (in this case number of frames) and this parsing algorithm is too slow. To overcome this drawback the parser probabilities have to be locally approximated and this leads to a (suboptimal) approximation.

Chappelier report another way of parsing in a speech recognizer: lattice parsing. It can be used in both tightly and more loosely coupled systems [5]. In the tightly coupled, the parser takes the phoneme lattice (represents the output of the acoustic recognizer, it contains a complete record of all tokens which were not pruned during the recognition process) and a phoneme grammar as input. In a more loosely coupled system, it parses the word lattice with some phrase grammar. They have developed a parser [4] which is able to simultaneously deal with output produced by the acoustic modules and integrating syntactic models like Stochastic Context-Free Grammars (SCFG). On an average over all experiments conducted by Chappelier shows that the loosely coupled system with SCFG improves the results for 35% in the test case and for 67% of the cases it did at least as well as without SCFG.

3.3 Other structured language models

It has been proven hard to incorporate linguistic knowledge and integrate sophisticated methods like parsing [3]. N-gram language models remain the state of the art, and are used in almost every speech recognition system. It has been/is frustrating for research community of language models that n-gram models still is the best way to solve the problem. Around year 2000 seems the effort of creating language models more been focused to capture nonlocal dependencies without using parsing algorithms, instead of combining nonlocal syntactic information and n-gram models.

Wu and Khudanpur have tried to create a new language model which incorporates local n-gram dependencies with long-range dependencies: the syntactic structure and the topic of a sentence [18]. They use maximum entropy methods to integrate these dependencies and report substantial improvements over a trigram model in both perplexity and speech recognition accuracy. Furthermore, Chelba and Jelinek investigate a two-pass speech recognizer, which enables the use of extended distance dependencies [6].

4 Conclusion

After reading several articles about automatic speech recognition and language modelling, I am more confused if whether or not a syntactic language models improves the speech recognition. It seems that syntactic information captured by a parser can improve the accuracy, but the improvement is not as good as the researcher want it to be. The effort to integrate more advanced language models are too high and overall efficiency of the recognizer drops. The simple n-gram language models seems to be the best way to capture the local dependencies in the language and to enhance the models using a probabilistic model to capture the long-range dependencies.

References

[1] L. R. Bahl, F. Jelinek, and R. L. Mercer. A maximum likelihood approach to continuous speech recognition. *IEEE Trans. Pattern Anal. Machine Intell.*, PAMI-5:179–190, 1983.

[2] Jerome R. Bellegarda. A multi-span language modeling framework for large vocabulary speech recognition. *IEEE Transactions on Speech and Audio Processing*, 6(5):456–467, 1998.

[3] E. Brill, R. Florian, C. Henderson, and L. Mangu. Beyond N-Grams: Can Linguistic Sophisticantion Improve Language Modeling. In *Proceedings of the 36th Annual Meeting of the ACL*, 1998.

[4] J.-C. Chappelier and M. Rajman. A generalized cyk algorithm for parsing stochastic cfg. In *Procedings of TAPD'98 Workshop*, pages 133–137, 1998.

[5] J.-C. Chappelier, M. Rajman, R. Aragues, and A. Rozenknop. Lattice parsing for speech recognition. In *6eme conference sur le Traitement Automatique du Langage Naturel (TALN99), Cagese, France*, 1999.

[6] C. Chelba and F. Jelinek. Recognition performance of a structured language model. In *Proceedings of Eurospeech'99, Budapest, Hungary*, volume 4, pages 1567–1570, September 1999.

[7] M. Harper et al. Integrating Language Models with Speech Recognition. In *Proceedings of the AAAI-94 Workshop on Integration of Natural Language and Speech Processing*, 1994.

[8] M. P. Harper et al. Interfacing a cdg parser with an hmm word recognizer using word graphs. In *Proceedings of International Conference on Acoustics, Speech, and Signal Processing (ICASSP-99)*, 1999.

[9] Sadaoki Furui. Recent advances in spontaneous speech recognition and understanding. In *Proceedings of the ISCA-IEEE Workshop on Spontaneous Speech Processing and Recognition*, 2003.

[10] John Holmes and Wendy Holmes. *Speech Synthesis and Recognition.* Taylor and Francis, 2001.

[11] Michael T. Johnson, Mary P. Harper, and Leah H. Jamieson. Interfacing Acoustic Models with Natural Language Processing Systems. In *Proceedings of the International Conference on Spoken Language Recognition, Sydney, Australia*, volume 6, pages 2419–2422, 1998.

[12] D. Jurafsky, C.Wooters, J. Segal, A. Stolcke, E. Foster, G. Tajchman, and N. Morgan. Using a stochastic context-free grammar as a language model for speech recognition. In *Proceedings of the IEEE ConferenceonAcoustics, Speech, and Signal Processing, Detroit*, pages 189–192, May 1995.

[13] K. Kita and W. Ward. Incorporating lr-parsing into sphinx. In *Proceedings of International Conference on Acoustics, Speech, and Signal Processing (ICASSP-91)*, volume 1, pages 269–272, 1991.

[14] Lawrence R. Rabiner. A Tutorial on Hidden Markov Models and Selected Application in Speech Recognition. In *Proceedings of the IEEE*, volume 77, pages 257–286, 1989.

[15] R. Rosenfeld. Two decades of statistical language modeling: Where do we go from here? In *Proceedings of the IEEE*, pages 1270–1278, 2000.

[16] B. Srinivas. "Almost Parsing" Technique for Language Modeling. In *Proceedings of the ICSLP'96*, volume 2, pages 1173–1176, 1996.

[17] Sven Wachsmuth, Gernot A. Fink, and Gerhard Sagerer. Integrating of Parsing and Incremental Speech Recognition. In *Proceedings of the European Signal Processing Conference, Rhodes*, volume 1, pages 371–375, 1998.

[18] J. Wu and S. Khudanpur. Combining nonlocal, syntactic and n-gram dependencies in language modeling. In *Proceedings of Eurospeech'99, Budapest, Hungary*, volume 5, pages 2179–2182, September 1999.

[19] V. Zue and D. Goddeau. Integrating probabilistic lr parsing into speech understanding systems. In *Proceedings of International Conference on Acoustics, Speech and Signal Processing (ICASSP-92)*, pages 181–184, 1992.