Natural Language Understanding by Combining Statistical Methods and Extended Context-Free Grammars

Stefan Schwärzler*, Joachim Schenk*, Frank Wallhoff, and Günther Ruske

Institute for Human-Machine Communication
Technische Universität München
80290 Munich, Germany
{sts,joa,waf,rus}@mmk.ei.tum.de

Abstract. This paper introduces an novel framework for speech understanding using extended context-free grammars (ECFGs) by combining statistical methods and rule based knowledge. By only using 1st level labels a considerable lower expense of annotation effort can be achieved. In this paper we derive hierarchical non-deterministic automata from the ECFGs, which are transformed into transition networks (TNs) representing all kinds of labels. A sequence of recognized words is hierarchically decoded by using a Viterbi algorithm. In experiments the difference between a hand-labeled tree bank annotation and our approach is evaluated. The conducted experiments show the superiority of our proposed framework. Comparing to a hand-labeled baseline system ($\widehat{=}100\%$) we achieve 95,4 % acceptance rate for complete sentences and 97.8 % for words. This induces an accuray rate of 95.1 % and error rate of 4.9 %, respectively F1-measure 95.6 % in a corpus of 1 300 sentences.

1 Introduction

In this paper, we address the problem of developing a simple, yet powerful speech understanding system based on manually derived domain-specific grammars. Contrary to existing grammar based speech understanding systems (e. g. Nuance Toolkit platform), not only grammar decisions are included, but information from grammar and word-label connections are combined as well and decoded by Viterbi algorithm.

A two pass approach is often adopted, in which a domain-specific language model is constructed and used for speech recognition in the first pass, and the understanding model obtained with various learning algorithms is applied in the second pass to "understand" the output from the speech recognizer. Another approach handles recognition and understanding at the same time [**?**,2]. For this purpose so-called "concepts" are defined, which represent a piece of information on the lexical as well as on the semantic level. In this way all the statistical methods from speech recognition can be utilized for all hierarchies. This concerns especially the stochastic modeling solutions offered by Hidden Markov Models. The hierarchies consist of transition networks (TNs) whose nodes either represent terminal symbols or refer to other TNs [3]. This approach uses

* Note: Both authors contributed equally to this paper.

G. Rigoll (Ed.): DAGM 2008, LNCS 5096, pp. 254–263, 2008.

statistical methods for semantic speech understanding requiring fully annotated tree banks. To this end a fully annotated tree bank in German language is available in the NaDia corpus [2], which is similar to the corpus used in the ATIS task [4]. In our experiments we use the NaDia corpus as baseline system.

In general, fully annotated tree banks are not available for most application domains. However, a flat structure of words on the sentences level can commonly be realized rather easily. The sentences of natural language understanding have to obey certain grammatical rules, e. g. time data rules. Therefore we use extended context free grammars (ECFGs), whose production rules on the right-hand side consist of regular expressions (REs). REs are compact notations to describe formal languages. In our approach, we describe the working-domain's grammar of our speech understanding system with appropriate REs. The basic principles of RE can be found in [5]. Context-free grammars (CFGs) consist of terminal and nonterminal symbols. Nonterminal symbols are used to describe production rules in CFG. For this reason we obtain a hierarchical structure of production rules, which constitute rule sets of semantic units (e. g. rules for time, origin, and destination). The idea is that we start with the start nonterminal symbol S and replace it with any sequence in the language described by its RE. This hierarchical coding determines the semantic meaning of the terminal symbol. Our aim is to analyse a given sentence and to decode the hierarchical structure of each terminal symbol. A similar approach has been presented in combination of CFG and N-gram modeling in Semantic Grammar Learning [6]. However, our system uses ECFGs, and is built on REs to describe the grammar rules. Thereby we utilize the entire sentences for semantic decoding by the grammar. The next section gives a brief overview of extended context-free grammars and the hierarchical utilization. Afterwards we introduce our TN for speech understanding. In section 4 we present our Viterbi based parsing methods. The settings and the goals of our experiments are defined in section 5. Our system is evaluated in section 6 on a hand-labeled tree bank. A comparison between hand-crafted annotation and automatic annotation is presented. Finally, conclusions and outlook are given in section 7.

2 Extended Context-Free Grammars (ECFG)

CFGs are powerful enough to describe the syntax of most programming languages. On the other hand they are simple enough to allow construction of efficient parsing algorithms. In spoken language understanding, an extended CFG is used by the Phoenix parser, which allows to skip unknown words and performs partial parsing [7].

An ECFG G is specified by a tuple of the form (N, Σ, P, S), where N is a nonterminal alphabet, Σ is a terminal alphabet, P is a set of production schemes of the form $A \rightarrow L_A$, such that A is a nonterminal and L_A is a regular language over the alphabet $\Sigma \cup N$, and S is the sentence symbol, which is a nonterminal [8].

Nonterminal symbols are used to describe further production rules in ECFG. The production rules consist of a "Kleene closure" expressed by $*$, $+$, and $?$ operators in regular expression syntax. Using this construction we obtain a hierarchical structure of production rules in ECFG which constitute rule sets of semantic units (e. g. rules for time, origin, and destination), see Fig. 1. Each terminal symbol is generated by a nonterminal symbol, which is part of the hierarchical structure.

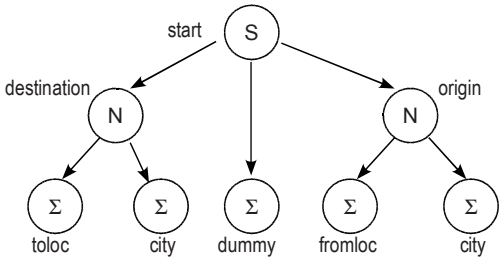

Fig. 1. Schematic tree consisting of terminal leaf nodes and nonterminal symbols

Hence, for every terminal symbol a hierarchical coding is provided by the sequence of nonterminal symbols leading from the root to the terminal symbol. This hierarchical coding determines the semantic meaning of the terminal symbol. Our aim is to analyse a given sentence and to decode the hierarchical structure of each terminal symbol.

To that end we formulate a TN that describes the transitions between each two succeeding terminal symbols in time. The TN cannot be derived directly from the ECFGs. Therefore we translate the production rules in ECFGs into non-deterministic automata (NDA), whose equivalence has been shown in [9].

3 Transition Network (TN)

3.1 Non-Deterministic Automata (NDA)

In this section we show how a TN can be derived from ECFGs using NDA. We may assume that the language L_A (see Sec. 2) is represented by a nondeterministic finite state automaton $M_A = (Q_A, \Sigma \cup N, \delta_A, s_A, F_A)$, where Q_A is a finite set of states, δ_A is the transition relation, s_A is the start state and F_A is a set of final states.

For example, consider sentences that contain any number of DUMMY followed by either ORIG (origin) or DEST (destination), and are ended by any number of DUMMY. The label DUMMY denotes a not-meaningful entity. The rather simple grammar describing these sentences formulated as a set of productions P in ECFG is shown in Eq. 1, where a bold-font denotes nondetermistic states, the remaining states are deterministic.

$$\begin{aligned} \mathbf{S} &= \text{DUMMY}^*.(\mathbf{ORIG} \mid \mathbf{DEST}).\text{DUMMY}^* \\ \mathbf{ORIG} &= \text{FROM.DUMMY}^*.\text{CITY} \\ \mathbf{DEST} &= \text{TO.DUMMY}^*.\text{CITY} \end{aligned} \tag{1}$$

The equivalent NDA is shown in Fig. 2. As seen in this figure, the NDA consists of states. A transition from one state to another is performed by emitting a terminal symbol. For NDA, arbitrary emissions can lead from a certain state to a certain other state. As explained above we build a structure of NDA with ECFGs. Each grammar rule is represented by one sub-NDA. ϵ-transitions denote transitions between states without emitting a terminal symbol and are used to connect sub-NDA.

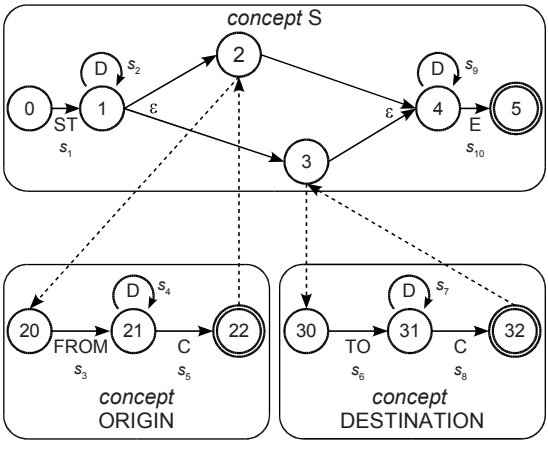

Fig. 2. Schematic diagram of a nondeterministic finite state automaton

The NDA are now transformed into the NT. To avoid confusion, we distinguish between NDA states, i. e. states that are introduced during the transformation from ECFG to the NDA, and rule-states $(s_1, \ldots s_N)$ which describe the TN. Each rule-state represents a terminal symbol, the transitions between the rule states describe the desired transitions between terminal symbols in time. Each emitted symbol of the NDA corresponds to a rule-state in the TN. Hence each rule-state emerges from and points to an NDA state.

The transition between the rule-states is also defined by the NDA-states: One rule-state (e. g. s_j) is followed by all rule-states emerging from the NDA-state where rule-state s_j is pointing to. This leads from the NDA depicted in Fig. 2 to the rule network described by matrix $\mathbf{A} = (a_{ij})$, $i, j = 1, \ldots, N$, defined by the ten rule-states $s_1, \ldots, s_{10}$:

$$\mathbf{A} = \begin{array}{c} \\ s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \\ s_7 \\ s_8 \\ s_9 \\ s_{10} \end{array} \begin{array}{c} \begin{array}{cccccccccc} s_1 & s_2 & s_3 & s_4 & s_5 & s_6 & s_7 & s_8 & s_9 & s_{10} \end{array} \\ \begin{pmatrix} 0 & 1/3 & 1/3 & 0 & 0 & 1/3 & 0 & 0 & 0 & 0 \\ 0 & 1/3 & 1/3 & 0 & 0 & 1/3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/2 & 1/2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/2 & 1/2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1/2 & 1/2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1/2 & 1/2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

The probabilities of each line i in $\mathbf{A}$ add up to $\sum_{j=1}^{N} a_{ij} = 1$, except for the last line. Every probability denotes a transition from s_i to s_j. In this work we assume each

transition from one state to another as equally probable as long as the transition is allowed by the rule-scheme. Each state s_i corresponds to a terminal symbol which is stored in the vector

$$\mathbf{b} = (\text{‘START’, ‘DUMMY’, ‘FROM’, ‘DUMMY’, ‘CITY’,} \\ \text{‘TO’, ‘DUMMY’, ‘CITY’, ‘DUMMY’, ‘END’}) \tag{2}$$

The vector $\mathbf{b}$ denotes the first level labels from an annotated corpus. By definition every sentence ends with the token “END”. Thus there are no transitions from the corresponding state, and the last line in Matrix $\mathbf{A}$ is always 1 in the last column.

In our case, each transition of one state to another is equally probable, as far as it is allowed by the TN.

3.2 Hierarchical Structure

As explained above, each state s_i of the transition matrix $\mathbf{A}$ represents a terminal symbol from the ECFG. To keep the hierarchical information of each terminal symbol, the list $\mathbf{H}$ is introduced, where

$$\mathbf{H} = (1 \rightarrow \text{S}; 2 \rightarrow \text{S}; 3 \rightarrow \text{S,ORIG}; 4 \rightarrow \text{S,ORIG}, \\ 5 \rightarrow \text{S,ORIG}; 6 \rightarrow \text{S,DEST}; 7 \rightarrow \text{S,DEST}; \\ 8 \rightarrow \text{S,DEST}; 9 \rightarrow \text{S}; 10 \rightarrow \text{S}) \tag{3}$$

contains the hierarchical information for each state. Matrix $\mathbf{A}$ can be illustrated in a transition-diagram with the possible transitions between two consecutive time instances t and $t + 1$, as shown in Fig. 3; additionally the hierarchical structure of each terminal symbol is shown in this figure. For example, the terminal symbol ‘DUMMY’ can occur both in concept ORIG and in concept DEST.

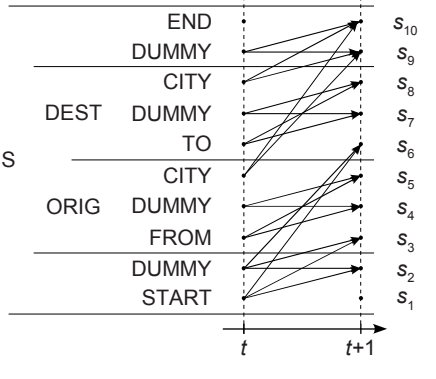

Fig. 3. Possible state transitions from time instance t to $t + 1$

4 Parsing

The unparsed sequence of first-level labels $\mathbf{b} = (b_1, \ldots, b_T)$ is hierarchically decoded by finding the best path $\hat{\mathbf{q}} = (\hat{q}_1, \ldots, \hat{q}_T)$ through a trellis – built by concatenating the transition diagrams – leading from the start symbol to the end symbol. However, in natural spoken sentences, a direct mapping between the words (terminals in the sentence) to the labels (terminals in the rule-network) is not available for each word. E. g. the word "to" has different meanings, depending on the context. First it may be an infinitive companion, second it can be used as a TOLOC (e. g. "to location") when combined with a city. In our approach all possible meanings of a word are pursued in the trellis leading to a variety of sentence hypotheses. To cope with different meanings of words, the word-label probability $p(b_i|w_t)$, describing the probability that a certain word w_t belongs to the given first level label b_i, is introduced. Applying Bayes' rule, one derives

$$p(b_i|w_t) = \frac{p(w_t|b_i) \cdot p(b_i)}{p(w_t)}. \tag{4}$$

The probabilities of the right hand side of Eq. 4 are estimated by using databases and are referred to as "world knowledge". Given the sentence "I want to go to Munich", hence

$$\mathbf{w} = (\text{'START'}, \text{'I'}, \text{'want'}, \text{'to'}, \text{'go'}, \text{'to'}, \text{'Munich'}, \text{'END'}),$$

the transition diagram shown in Fig. 3, and the trellis diagram displayed in Fig. 4 can be built, containing all parsing hypotheses.

The y-axis shows the terminal symbols $\mathbf{b}$ including the hierarchical structures, which were defined in $\mathbf{H}$. The x-axis represents time. All possible transitions can be displayed depending on their consecutive time instances (concatenation of Fig. 3). For the test example our system finds the optimal path through the trellis diagram (shown as bold arrows in Fig. 4). Light arrows represent possible alternative paths that have not been chosen due to the statistical constraints.

4.1 Viterbi Path

The best path through the trellis (the path with the highest probability) can be found by using the Viterbi algorithm. Introducing the quantity $\delta_t(i)$ describing the probability of a single path at time t that ends in state s_i, the Viterbi algorithm is formulated as displayed in the following:
Initialization:

$$\begin{matrix} \delta_1(i) = p(b_i|w_1) \\ \psi_1(i) = 0 \end{matrix}, 1 \leq i \leq N. \tag{5}$$

Recursion:

$$\begin{matrix} \delta_t(j) = \max\limits_{1 \leq i \leq N} [\delta_{t-1}(i) \cdot a_{ij}] \cdot p(b_j|w_t) \\ \psi_t(j) = \operatorname{argmax}\limits_{1 \leq i \leq N} [\delta_{t-1}(i) \cdot a_{ij}] \cdot p(b_j|w_t) \end{matrix}, \begin{matrix} 2 \leq t \leq T \\ 1 \leq j \leq N \end{matrix} \tag{6}$$

Termination:

$$\hat{q}_T = \operatorname*{argmax}_{1 \leq i \leq N} \delta_T(i) \tag{7}$$

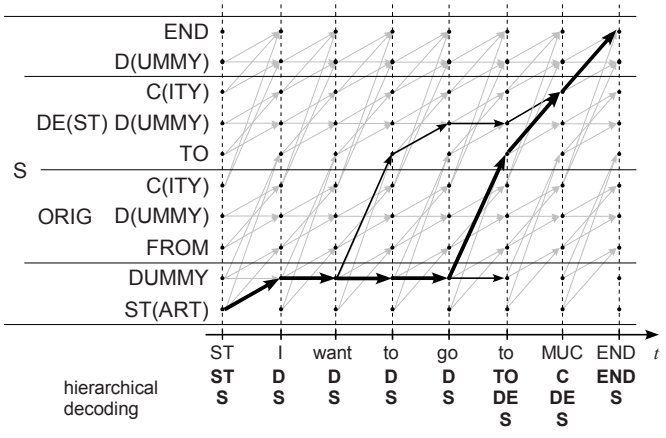

Fig. 4. Trellis diagram and best path for "I want to go to Munich"

The best path is then found by backtracking

$$\hat{q}_t = \psi_{t+1}(\hat{q}_{t+1}). \tag{8}$$

With the states revealed from the backtracking path the hierarchical structure of the sentence $\mathbf{w}$ is derived using the hierarchy list $\mathbf{H}$. Considering the above reviewed sentence, backtracking leads to the hierarchical decoding of the sentence shown in Fig. 4, as well as to the best path found by the Viterbi algorithm.

5 Experiments

We verified our approach on a database containing more than 1 300 sentences (1 000 for training, 300 for testing) of the air traveling information domain from the German NaDia corpus [2]. Each word in each sentence is manually hierarchically labeled (baseline system), according to its meaning. We used our system to hierarchically decode the sentences, given a certain ECFG. The crucial probabilities $p(w_t|b_i)$ for the decoding are trained using the NaDia corpus. However any other corpus in the same language could be used for training. The production rules of an ECFG describing the transition matrix $\mathbf{A}$ are defined to match the requirements of the corpus and suit the German grammar. The performance of the system depends on the production rules which are used in ECFG. On the one hand the rules must be precise enough to take care of the semantic content, on the other hand the rules have to capture all variants of natural language formulations. To show this, we created three different types of production rules in ECFG (in rising the number of states of the transition matrix), differing by their perplexity. In our experiments we compare the performance of the production rules in ECFG with the manually hierarchically labeled annotations (baseline system).

The goal is to identify the concepts TIME , ORIGIN, and DESTINATION independently of their order and combination, in more than 300 sentences of the air traveling

information domain. Each word in the sentences is labeled by one of 14 tags, like Dummy, AOrigin, ADestination, ATime, etc.

6 Results

The proposed types of ECFGs (see Sec. 5) are evaluated using the following measurement methods. A sentence is accepted, if the hierarchical structure automatically obtained by our system fully matches the baseline labeling. We define the sentence acceptance rate r_{sent} as

$$r_{\text{sent}} = \frac{\text{\# of accepted sentences}}{\text{total \# of sentences}}. \tag{9}$$

As explained above, a sentence is rejected as soon as the hierarchical decoding of one single word differs from the baseline annotation. To investigate our system's decoding abilities on a word level we use the word acceptance rate

$$r_{\text{word}} = \frac{\text{\# of accepted words}}{\text{total \# of words}}. \tag{10}$$

Tab. 1 shows further the word accuracy, the error rate and the well-known F1-measure results of three different types of ECFGs. In Tab. 1 it is assumed that the hand-labeled annotation works perfectly ($\widehat{=}100\%$). The results shown in Tab. 1 represent the automatic method for comparison. The best results were achieved in ECFG3 with a word acceptance rate of 97.8 % respectivley 95.6 % F1-measure and 95.4 % for complete sentences. The columns of Tab. 1 represent three different types of ECFGs, which differ in the number of terminal symbols. In ECFG1 a rough description of the concepts TIME, ORIGIN, and DESTINATION exists, whereas the perplexitiy is reduced in ECFG2 and ECFG3. The main difference consists in the description of TIME rules. As shown in Tab. 1 the sentence and word acceptance rate increases by the complexity of their grammar, which increases also to the number of states. The slight difference between the word and sentence acceptance rates is caused by multiple word errors in same sentences. In result most of our 1 300 sentences were accepted by the grammar. However, some sentences are refused as they do not follow the German grammar rules.

Table 1. Results for different ECFGs with decreasing perplexity

	ECFG1	ECFG2	ECFG3
$r_{\text{sent}}[\%]$	92.8	94.7	95.4
$r_{\text{word}}[\%]$	95.5	96.9	97.8
$\#_{\text{states}}$	89	98	113
Accuracy [%]	93.1	94.0	95.1
Error [%]	6.9	6.0	4.9
F1-Measure [%]	93.3	95.0	95.6

7 Conclusions and Outlook

In this work a novel approach for speech understanding has been presented. We showed how ECFGs can be used instead of tree bank annotations, e. g. understanding time data, etc. Our approach combines semantic grammar rules with the advantage of statistical methods. The system provides all possible rules by working in parallel running from the first frame till the last frame of the sentence. Thus a set of alternative trails on the trellis diagram results. The best path is found by using the Viterbi algorithm. We showed how to build a transition matrix $\mathbf{A}$ from ECFGs. The probability at postion (ij) in this matrix, denoting a transition from s_i to s_j (see Sec. 3.1), is set equal for each possible transition allowed by the rule-scheme up to now. In future work, we will be able to weight the transitions according to a training set. The results already (see Sec. 6) showed that our approach is comparable to a full hand-labeled annotated tree bank (baseline system). Our best verification reaches an acceptance rate of 95.4 % for complete sentences and 97.8 % for words. Furthermore in contrast to hand-labeled tree bank annotations our approach only needs a non-hierarchical annotation from a corpus, but up to now just suits parts of the German grammar. In future investigations we therefore plan to learn general parts of grammar (ECFGs) from a training set.

At this stage we presented only the principles of our approach. For further results we will apply our approach on the ATIS-Tasks. Additionally the perplexity of our grammar will be increased by automatic grammar reduction.

In this work our approach used just the decoder's best word hypotheses (having maximum probability). However our approach offers the possibility to evaluate word-confidences utilizing the ECFGs. In future work, we aim to introduce alternative word hypotheses and confidence measures from the one-stage decoder [**?**].

Acknowledgements

This work was funded partly by the German Research Council (DFG).

References

1. Thomae, M., Fabian, T., Lieb, R., Ruske, G.: A One-Stage Decoder for Interpretation of Natural Speech. In: Proc. NLP-KE 2003, Beijing, China (2003)
2. Lieb, R., Ruske, G.: Natural Dialogue Behaviour Using Complex Information Services in Cars. Institute for Human Machine Communication, Techn. University, Munich, Germany, Tech. Rep (2003)
3. Thomae, M., Fabian, T., Lieb, R., Ruske, G.: Hierarchical Language Models for One-Stage Speech Interpretation. In: Proc. Eurospeech, Lisbon, Portugal (2005)
4. Ward, W.: The CMU Air Travel Information Service: Understanding Spontaneous Speech. In: Proc. Workshop on Speech and Natural Language. Hidden Valley, Pennsylvania, pp. 127–129 (1990)
5. Blackburn, P., Bos, J.: Representation and Inference for Natural Language. Leland Stanford Junior University: CLSI Publications (2005)
6. Wang, Y., Acero, A.: Combination of CFG and N-Gram Modeling in Semantic Grammar Learning. In: Proc. Interspeech (2003)

7. Ward, W., Issar, S.: Recent Improvements in the CMU Spoken Language Understanding System. In: Proc. of ARPA Human Language Technology Workshop, pp. 213–216 (1994)
8. Brüggemann-Klein, A., Wood, D.: The Parsing of Extended Context-Free Grammars. In: HKUST Theoretical Computer Science Center Research Report, no. 08 (2002)
9. Hopcroft, J.E., Motwani, R., Ullman, J.D.: Introduction to automata theory, languages and computation. Addison-Wesley, Reading (2006)