Hierarchical Language Models for One-Stage Speech Interpretation

Matthias Thomae, Tibor Fabian, Robert Lieb, Günther Ruske

Institute for Human-Machine Communication
Technische Universität München, Germany
eurospeech05@thomae-privat.de, {fab,lie,rus}@mmk.ei.tum.de

Abstract

This paper presents a robust semantic model for one-stage interpretation of natural speech. Our semantic analysis uses no explicit syntactic and morphologic knowledge, which seems sufficient for narrow application domains. In contrast to previous approaches, our semantic model is embedded in a uniform, hierarchical, stochastic modeling framework together with acoustic-phonetic and lexical knowledge, and semantic representations are computed directly from acoustic observations through a one-stage decoding process. The decoder produces a hierarchical (tree-) structure of words and semantic category symbols by use of the so-called hierarchical language model (HLM). We discuss generation of HLM by mixing rule-based and data-driven language model (LM) generation techniques, namely weighted regular expressions, n-grams and exact LM. Different HLM configurations with varying discounting techniques, n-gram orders and scaling factors are examined. Experiments were conducted with an airport information dialogue application. The evaluation results are based on HLM perplexity and our previously published semantic tree accuracy.

1. Introduction

In [1] we introduced a One-stage Decoder for Interpretation of Natural Speech (ODINS), which tightly integrates automatic speech recognition (ASR) and natural language understanding (NLU) techniques in a one-stage decoding process. Experiments confirmed that the one-stage approach can be favorable over typically used multi-stage methods by avoiding errors caused by decisions in early processing stages. In contrast to typical hybrid speech understanding approaches, ODINS operates on a single, uniform model for acoustic-phonetic, lexical and syntactic-semantic knowledge. Uniformity is appealing because it enables the integration of different types of knowledge models without having to modify the decoder, as long as the expressive power of the underlying framework is sufficient. Furthermore, decoder complexity is potentially lower, as only one algorithmic framework needs to be considered, instead of many. On the downside, a hybrid modeling approach may lend itself to more efficient decoding.

Our uniform modeling approach is realized as a stochastic context-free grammar representation through weighed transition network hierarchies (WTNH). WTNH consist of transition networks whose nodes either represent terminal symbols or refer to other transition networks. WTNH are similar to stochastic recursive transition networks, but they employ the Moore machine representation instead of the Mealy one. As another fundamental difference, WTNH are logically divided into hierarchy levels by defining groups of transition networks. These groups can then be assigned different attributes, such as structural constraints or search parameters. In order to still maintain flexibility, hierarchy levels can possess an arbitrary number of sub-levels, and hierarchy levels may be skipped (see [1] for a more detailed description). Uniform modeling approaches based on composition of finite-state transducers enable even tighter integration of ASR and NLU, and automatic optimization of the search space by automata minimization [2]. However, an explicit preservation of hierarchical modeling structure as in WTNH allows a better control over the decoding process, which is e.g. useful for uncovering temporal alignment as needed for confidence measure computation [3]. In the explicitly hierarchical model, automata minimization can at least be performed locally for each network of a WTNH. The upper part of a WTNH, representing syntactic-semantic knowledge, is denoted as hierarchical language model (HLM). HLM consist of a hierarchical combination of local language models (LLM) for symbol sequences, each represented by a weighted transition network. We use HLM as a semantic grammar which combines syntax and semantics in a single model, omitting explicit syntactic and morphologic knowledge. Such an approach is typically more robust than a deep semantic analysis, especially if NLU processing has to deal with unreliable textual input, as is the case when users should be able to talk in a natural way and in natural situations. On the downside, combining syntax and semantics renders the model highly domain-specific.

In Section 2, we discuss HLM generation using a mixture of rule-based and data-driven LM techniques. Through this we aim to reduce the effort for model generation and achieve good speech interpretation performance even if training data is sparse. The use of WTNH enables to make independent modeling decisions for each network of the hierarchy. The HLM presented in this work were built from three different types of LLM: Weighted regular expressions, n-grams and exact LM. In order to cope with sparse and unreliably distributed training data, we apply smoothing techniques to LLM. Moreover, we discuss transformations of the HLM likelihood distribution. In speech recognition, LM factor and word insertion penalty aid in establishing a balance between the likelihood distributions of acoustic model (AM) and LM. We show how similar parameters are effectively applied to HLM, and introduce additional parameters to adjust the within-HLM likelihood distribution.

Section 3 presents results of experiments within a speech dialogue scenario for an airport information application. Different experiments measure the influence of smoothing techniques, weighting parameters and n-gram orders on the performance of HLM and the whole speech interpretation system. HLM are evaluated by computing the test-set perplexity directly on the WTNH representation. The goodness of semantic representations decoded by ODINS from speech is measured with the tree

This work was funded partly by the German Research Council (DFG) project Ru 301/6-2.

node accuracy metric introduced in [4].

2. Hierarchical Language Models

A (flat) LM can be viewed as a likelihood distribution over symbol sequences. Given a sequence $\mathbf{S}$ of $|\mathbf{S}|$ symbols $s_1 \dots s_{|\mathbf{S}|}$ from an alphabet Σ, the likelihood $P(\mathbf{S})$ that this sequence occurs can be expressed as the product of the occurrence likelihood of the single symbols given their predecessors:

$$P(\mathbf{S}) \quad = \quad \prod_{i=1}^{|\mathbf{S}|} P(s_i | s_1 \dots s_{i-1}) \qquad (1)$$

Likewise, a HLM can be regarded as a likelihood distribution over ordered trees $\mathbf{T}$ of semantic symbols. For the moment, we assume that $\mathbf{T}$ is a constant-height tree, and that a horizontal line through $\mathbf{T}$ touches all tree nodes belonging to a hierarchy level l of $\mathbf{T}$, with $1 \leq l \leq L$. $\mathbf{T}^l$ denotes the sequence of $i = 1 \dots |\mathbf{T}^l|$ tree node symbols s_i^l contained in a hierarchy level l of $\mathbf{T}$. If we assume that the symbols on tree level l only depend on symbols of the next higher tree level $l + 1$, we can approximate $P(\mathbf{T})$ by:

$$P(\mathbf{T}) = P(\mathbf{T}^1, \dots, \mathbf{T}^L) \approx P(\mathbf{T}^L) \prod_{l=1}^{L-1} P(\mathbf{T}^l | \mathbf{T}^{l+1}) \qquad (2)$$

As there exists a sequential correspondence between adjacent hierarchy levels, the tree nodes of $\mathbf{T}^l$ can be segmented into $i = 1 \dots |\mathbf{T}^{l+1}|$ consecutive sub-sequences $\mathbf{S}_i^l$ so that each sub-sequence $\mathbf{S}_i^l$ directly corresponds to a tree node s_i^{l+1} on the next higher level. Hence, Equation 2 becomes:

$$P(\mathbf{T}) \quad \approx \quad P(\mathbf{S}_1^L) \prod_{l=1}^{L-1} \prod_{i=1}^{|\mathbf{T}^{l+1}|} P(\mathbf{S}_i^l | s_i^{l+1}) \qquad (3)$$

In a HLM, each LLM describes one of the terms of Equation 3, and is represented by a weighted transition network. The root language model represents the unconditional likelihood term $P(\mathbf{S}_1^L)$. In order to increase HLM flexibility, we allow LLM to refer to any hierarchy level below their own, not only to the direct subordinate. In this case, $\mathbf{T}$ is no longer a constant-height tree. Such level skipping is considered in Equation 3 by imagining a 'dummy' LLM with a single symbol for each skip transition, whose likelihood term is one and thus can be ignored.

For each of the likelihood terms of Equation 3, it can be decided independently which type of LM to use. More specifically, we can use a mixture of rule-based LM, whose structure and weighting is manually defined by human experts, and data-driven modeling, where structure and weights are derived automatically from annotated speech corpora. These two approaches can also be combined, e.g. by manually defining model structure but automatically deriving model weighting.

Due to the lack of suitable metrics, we select LLM types manually with the aid of informal decision criteria. Rule-based modeling is applied where the target language can be covered with an easily definable rule set, or the amount of available training data is too small for data-driven modeling. Moreover, if semantic objects required for an application are not seen during training, such as airline names or flight codes for our example domain, manual extension of LLM is useful. Otherwise, we estimate LM automatically from an annotated speech corpus. Currently, availability of full tree annotations is assumed. In [1] we briefly described a semi-automatic, iterative procedure developed for this purpose. Data-driven LM approaches can be categorized according to their generalization abilities, i.e. to assign non-zero probability to unseen events. Generalization is especially important at the surface level of HLM, in order to cover arbitrary-length utterances and variable ordering of semantic concepts.

A prominent representative for LM with generalization ability are n-grams, which limit the dependency of the current symbol s_i to the $n-1$ previous symbols, denoted as history h_i^n. Thus, the n-gram model likelihood $P_n(\mathbf{S})$ of a symbol sequence $\mathbf{S}$ is an approximation of Equation 1. In the basic case, the maximum-likelihood estimate of an n-gram is directly computed from the n-gram counts by normalization with the counts of all n-grams with the same history. The generalization ability of n-grams is often extended by combination with lower-order n-grams through interpolation or backoff [5]. In addition to generalization, discounting is a central issue in statistical language modeling. Discounting reduces the likelihood of the unreliable estimates from the observed counts and redistributes the freed probability mass. The joint application of discounting and generalization, in the sense that the probability mass freed by discounting is redistributed among unseen events created by generalization, is denoted as smoothing. A comparative study of different smoothing techniques for n-gram LM can be found in [5]. In our work, we use 'canonical' Katz backoff smoothing, and modified Kneser-Ney smoothing as proposed in [5]. All n-gram models were computed with the SRILM Toolkit [6].

In order to integrate n-gram LM into HLM, they need to be represented as weighted transition networks. For backoff n-gram LM efficient network representations are known (e.g. [7]). The backoff principle is implemented via failure transitions to null nodes, which realizes a context change to an $(n-1)$-gram if the n-gram does not exist. If the n-gram exists, its preference over the backoff path must be ensured. In the utilized SRILM Toolkit [6] this problem is solved by deleting the direct transition if it has a lower likelihood than the backoff path.

When considering the generalization abilities of the whole HLM, it has to be taken into account that the hierarchical structuring of semantic symbol sequences into equivalence classes itself has a generalizing effect. Therefore, generalization may not be desirable, especially for sub-surface LLM. In this case, we use so-called *exact LM*, which exactly cover the symbol sequences seen during training. Hence, the exact model likelihood $P_e(\mathbf{S})$ is an exact description of the likelihood $P(\mathbf{S})$ of Equation 1, and not an approximation like the n-gram likelihood. A transition network representation of an exact LM is created by representing the training symbol sequences for a LLM as a list of regular expressions. This is then compiled into a finite-state automaton and minimized by use of the Lextools and FSM Library toolkits [8, 9].

Again it must be assumed that the amount of training data is not sufficient to reflect the real distribution of events very well, so that discounting is also desirable for exact LM. We apply two different discounting techniques to exact LM, namely additive discounting and Good-Turing discounting (see [5]). The latter is also the basis for many smoothing techniques, such as Katz and Kneser-Ney smoothing. However, in contrast to n-gram LM, we directly discount on the network level, i.e. we use network transitions as the basic events. Through this, discounting techniques can be applied generally to any LM that has a network representation whose transitions can be marked with counts. This is of special interest for this work, because it enables data-driven weighting and smoothing of rule-based LM (see following section). In order to perform counting and smoothing of network transitions, the corpus statistics need to be transfered to the network. This is carried out by walking the paths through the network corresponding to the corpus' symbol sequences and counting how often each network transition is traversed. Note that the transition network needs to be deterministic in order to determine a unique path for each symbol sequence.

In addition to data-driven LM, HLM contain rule-based LM in the shape of manually defined weighted regular expressions. These are compiled into transition networks by use of the Lextools toolkit [8]. The weighting can either be defined manually, distributed uniformly, or derived from corpus data. The latter is performed as described in the previous section, by counting and discounting network transitions. This procedure effectively realizes a combined rule-based and data-driven LM, which can be useful if the model structure can easily be given manually, but the model weighting is not clear to the human expert.

2.1. Likelihood Balancing

The LM factor λ is an essential parameter in practical speech recognition, because it balances the likelihood values of AM and LM heuristically against each other, by a linear scaling of the LM likelihood distribution. The necessity for a likelihood transformation arises because HMM emissions are no 'real' probabilities, but values on a probability density function. In addition to adapting the ranges of the likelihood values, λ is simultaneously utilized to balance the relative influence of AM and LM on the decoding process. The higher the value of λ, the more the recognition results are dominated by the likelihood distribution of the LM. This means that in doubt a likely word transition with an unlikely acoustic match will be preferred over a less likely word transition with a more likely acoustic match. This property can be utilized to compensate differences in the qualities of AM and LM heuristically by giving more influence to the model of higher quality, thus avoiding errors caused by the lower-quality model.

As a side-effect of balancing the likelihood distributions, λ also affects the average length of a word. For relatively large values of λ, the LM likelihood values become comparatively small, so that a traversal from one word to another one is more 'costly' than staying within a word. Consequently, longer words are favored so that word deletion errors occur more likely than word insertions. Ideally, we would expect that the decoder achieves its maximum performance when it recognizes about as many words as were spoken. Hence, the number of deletions D should approximately equal the number of insertions I at the optimum value of λ. In order to describe this expectation numerically, we define the *insertion deletion ratio* IDR as the ratio between I and D in the optimum match between the reference transcriptions and recognizer hypotheses of a test set.

In order to illustrate the application of λ to our one-stage speech interpretation problem, we regard the maximum a-posteriori formulation of the search problem for the case that the HLM consists of $L = 4$ hierarchy levels, namely a word level $\mathbf{T}^1 = \mathbf{W}$, a word class level $\mathbf{T}^2 = \mathbf{K}$, a concept level $\mathbf{T}^4 = \mathbf{C}$ and a concept sub-level $\mathbf{T}^3 = \mathbf{C}'$. Denoting the acoustic observation sequence as $\mathbf{X}$, we get:

$$\underset{\mathbf{W},\mathbf{K},\mathbf{C}',\mathbf{C}}{\arg\max}\; P(\mathbf{X}|\mathbf{W})[P(\mathbf{W}|\mathbf{K})^{\lambda_K}[P(\mathbf{K}|\mathbf{C}')P(\mathbf{C}'|\mathbf{C})]^{\lambda_C}P(\mathbf{C})]^{\lambda} \tag{4}$$

Please ignore λ_K and λ_C for the moment. Hence, we apply λ in the WTNH by scaling the scores of all transitions within networks belonging to the HLM, i.e. on the word level and above. In order to quantify the likelihood balance between AM and HLM, we can use the evaluation method defined in [4], which performs a tree match between the annotated reference trees and the semantic hypothesis trees produced by ODINS. With the resulting statistics on matched tree nodes, we define a tree node IDR analogous to the word IDR above.

In order to balance the likelihood distribution *within* HLM, we introduce additional scaling factors for the main hierarchy levels, namely a word class factor λ_K and a concept factor λ_C for the example of Expression 4. Again, we can get a measure for the likelihood balance within HLM by looking at the numbers of inserted and deleted tree nodes on an evaluation set. In order to get separate IDR values for each hierarchy level, tree nodes belonging to different hierarchy levels are counted separately.

A more direct influence on the IDR is exercised by adding offsets to the likelihood transformation. In order to limit the number of parameters, we only examine one offset parameter in this work, namely the word insertion penalty p_W. We apply p_W to WTNH by adding it to the scores of all transition that lead into a word node.

3. Experimental Results

Experiments were conducted on a corpus of spontaneous speech utterances collected by simulating an airport information dialogue system through a wizard-of-oz setup. The corpus is an extended version of the one used in [1, 4], containing about 2700 utterances with 15000 words from 32 subjects in total. HLM were trained on a subset of 20 subjects, evaluation and cross-validation were performed on 6 subjects' utterances each. As in Equation 4, the semantic tree annotations yield 4 hierarchy levels. The word level contains about 580, the word class level 10 and the concept levels 40 unique symbols. As the speech corpus does not completely contain the word class contents relevant for the example application, the missing words (around 30) are added manually. HLM are generated using a mixture of data-driven and rule-based LM techniques, as described in Section 2. The root LM is a backoff n-gram LM, the other LLM are either exact LM or generated from regular expressions. Words within word classes are distributed uniformly. In all other LLM, weights are derived from corpus statistics and smoothed as described in Section 2. The unknown-word rate is 2.2% and 1.5% on evaluation and cross-validation set. The AM consists of the same speaker-independent tied intra-word triphone HMM with about 25k Gaussian mixture components as described in [1].

Our primary evaluation metric for the complete speech interpretation process is the tree node accuracy Acc_n from [4], which we compute from the counts of correct C_n, substituted S_n, inserted I_n and deleted tree nodes D_n after matching reference and hypothesis trees by:

$$Acc_n = \frac{C_n - I_n}{C_n + S_n + D_n} \tag{5}$$

Similar to IDR, accuracies can also be computed for specific hierarchy levels, by considering only the tree nodes from that level. In order to evaluate HLM alone, we compute test-set perplexities *ppl* of HLM by determining the best path through the transition network representation, corresponding to the tree annotations of the test-set.

Figure 1 depicts the total tree node accuracy curves for different smoothing scheme combinations on the evaluation set. The bigram root LM are subjected to Katz (*katz*) or modified

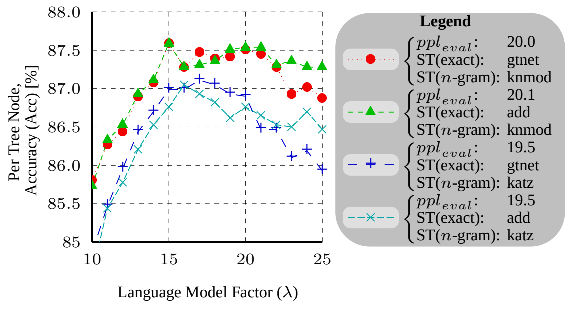

Figure 1: *Tree node accuracy for different smoothing schemes.*

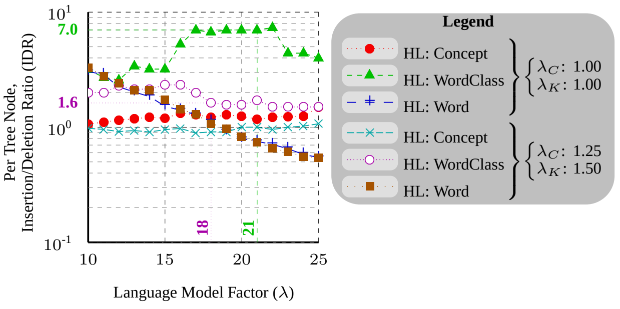

Figure 2: *IDR before and after likelihood balancing.*

Kneser-Ney (*knmod*) smoothing, exact networks are adjusted by additive (*add*) or Good-Turing (*gtnet*) discounting. As the plot shows, *knmod* consistently outperforms *katz* by about 0.5% absolute Acc_n over the whole tested range of λ, although the HLM perplexities displayed in the legend indicate the contrary. The superiority of *knmod* is in accordance with the findings of [5]. Acc_n and *ppl* of the two exact network discounting methods don't differ much, whereby *gtnet* seems marginally better. As *gtnet* is theoretically more promising than *add* and achieved better results in other studies, we select it for further experiments. In the second experiment, the n-gram order of the root LM was varied between 1 and 3. While the trigram root LM yields a significantly better HLM perplexity (18.5) than the bigram (20.0), the total tree node accuracy only improves slightly from 87.5% to 87.7% on the evaluation set. As expected, omitting the n-gram context by setting $n = 1$ yields a substantial loss in both perplexity (37.6) and accuracy (82.7%). For other experiments we used $n = 2$.

In another experiment, we investigated the effects of the likelihood transformation parameters described in Section 2.1, starting with baseline settings of $\lambda_K = 1$, $\lambda_C = 1$ and $p_W = 0$. At first, we carried out a joint optimization of only λ, λ_K and λ_C with regards to Acc_n on the cross-validation set, yielding $\lambda = 18$, $\lambda_K = 1.5$ and $\lambda_C = 1.25$. In a second experiment, we also added p_W to the joint optimization, which yielded $\lambda = 18$, $\lambda_K = 1.5$, $\lambda_C = 1.25$ and $p_W = -10$. Table 1 summarizes the resulting total and per-level tree node accuracies of the two experiments on the evaluation set. Error rate reductions compared to the baseline are also given. In the baseline experiment, only λ was optimized, yielding $\lambda = 21$. Although the within-HLM likelihood balancing yields no gain in total accuracy and even a small loss on the word level, the word class errors are reduced significantly by 13.5% relative. The use of p_W yields further improvements on the word and concept levels. In order to illustrate the effects of within-HLM likelihood balancing, the per-level *IDR* curves of baseline and first optimization are plotted in Figure 2. Most noticeable is the improvement on the word class level. Where 7 times more word class insertions than deletions occur in the baseline setting, this is reduced to 1.6 for the optimized setting. The word and concept *IDR* are also brought closer to the expected optimum value of 1.

	baseline	λ_K, λ_C opt.		$\lambda_K, \lambda_C, p_W$ opt.	
	Acc_n	Acc_n	Err_{rel}	Acc_n	Err_{rel}
Word	85.1%	84.9%	$+1.3\%$	85.3%	-1.3%
WordClass	94.8%	95.5%	-13.5%	95.5%	-13.5%
Concept	89.0%	89.3%	-2.7%	89.6%	-5.5%
Total	87.5%	87.5%	-0.0%	87.8%	-2.4%

Table 1: *Results of optimized likelihood balancing.*

4. Conclusion

We discussed HLM for robust semantic modeling in our one-stage speech interpretation framework. As previously shown, a comparable two-stage system only performs as accurate as our one-stage system if its speech recognition stage propagates a large number of alternative hypotheses to the second stage. We showed for an airport information test system, that the use of advanced smoothing methods for HLM such as modified Kneser-Ney smoothing significantly improves the accuracy of the whole system. We also applied LM factor and word penalty known from speech recognition to our system, and showed that an extension of the transformation with hierarchy level dependent scaling factors improves the likelihood balance within the HLM, yielding a significant reduction of the word class errors. We also gave evidence that the ratio of insertions and deletions aids as an indicator for the likelihood balance of HLM.

5. References

[1] M. Thomae, T. Fabian, R. Lieb, and G. Ruske, "A One-Stage Decoder for Interpretation of Natural Speech," in *Proc. NLP-KE'03*, Beijing, China, October 2003. [Online]. Available: http://www.thomae-privat.de/publications/nlpke2003.pdf

[2] M. Mohri, "Finite-State Transducers in Language and Speech Processing," *Computational Linguistics*, vol. 23, no. 2, 1997.

[3] R. Lieb, T. Fabian, G. Ruske, and M. Thomae, "Estimation of Semantic Confidences on Lattice Hierarchies," in *Proc. ICSLP*, Jeju Island, Korea, October 2004.

[4] M. Thomae, T. Fabian, R. Lieb, and G. Ruske, "Tree Matching for Evaluation of Speech Interpretation Systems," in *Proc. ASRU*, St. Thomas, USVI, Nov. 2003.

[5] S. Chen and J. Goodman, "An Empirical Study of Smooting Techniques for Language Modeling," CRTC, Harvard Univ., Cambridge, MA, Tech. Rep. TR-10-98, Aug. 1998.

[6] A. Stolcke, "SRILM - An Extensible Language Modeling Toolkit," in *Proc. ICSLP*, Denver, Colorado, USA, September 2002.

[7] C. Allauzen, M. Mohri, and B. Roark, "Generalized Algorithms for Constructing Statistical Language Models," in *41st ACL Meeting*, Sapporo, Japan, Juli 2003, pp. 40–47.

[8] R. Sproat, "Lextools: a toolkit for finite-state linguistic analysis." [Online]. Available: http://www.research.att.com/sw/tools/lextools/synth.pdf

[9] M. Mohri, F. C. N. Pereira, and M. Riley, "A Rational Design for a Weighted Finite-State Transducer Library," in *WS on Implementing Automata*, 1997, pp. 144–158.