WORD-SEMANTIC LATTICES FOR SPOKEN LANGUAGE UNDERSTANDING

Jan Švec, Luboš Šmídl, Tomáš Valenta, Adam Chýlek, Pavel Ircing

NTIS - New Technologies for Information Society, Faculty of Applied Sciences
University of West Bohemia, Pilsen, Czech Republic

[honzas,smidl,valentat,chylek,ircing]@ntis.zcu.cz

ABSTRACT

The paper presents a method for converting word-based automatic speech recognition (ASR) lattices into word-semantic (W-SE) lattices that contain original words together with a partial semantic information – so-called semantic entities. Semantic entity detection algorithm generates semantic entities based on the expert-defined knowledge. The generated W-SE lattices have smaller vocabulary and consequently reduce the sparsity of the training data. The format of the W-SE lattices also naturally preserves the inherent uncertainty of the ASR output that can be exploited in subsequent dialog modules. The presented technique employs the framework of weighted finite state transducers which allows for efficient optimization of word-semantic lattices. We have evaluated the method in two different spoken language understanding tasks and obtained more than 10% reduction of concept error rate in comparison with using 1-best word hypothesis in both of those tasks.

Index Terms— Spoken language understanding, dialog systems, word-semantic lattices

1. INTRODUCTION

The spoken language understanding (SLU) module is a crucial part of a spoken dialog system. The state-of-the-art dialog managers currently use statistical methods to maintain the belief state [1, 2] which represents the probability distribution over all possible dialog states. Such statistical methods can deal with the uncertainty of an ASR and an SLU and effectively model the decision policy to solve possible ambiguities in the belief state.

The ASR output is usually represented as a word lattice (acyclic weighted finite state acceptor [3]). This paper presents a method for preprocessing such word lattices before processing them in the SLU module. The preprocessing algorithm identifies the semantic entities in the input word lattice and replaces them with the corresponding identifiers. Such preprocessing allows for better generalization in the SLU module: for example the preprocessing could identify that the input training utterance contains a *date*, a *time* and a *person's name*. Then the particular lexical realization of those entities is irrelevant for the statistical-based SLU and the SLU model can better generalize in the case when the new (unseen) input contains the same entities but expressed with different words.

The key idea of this paper is the integration of the semantic entity detection algorithm [4] into a statistical SLU model. This allows us to incorporate the expert knowledge into the statistically trained model. The preprocessing of SLU training data with some algorithm which reduces the effect of the data sparsity could be found in most SLU models [5, 6, 7, 8]. In the STC model (Mairesse et. al. [6]) the input words matching the domain database values are replaced with corresponding category labels. The LUNA framework [5] uses local regular grammars to identify local concepts in the input word lattice. The other methods use finite state transducers (FST) to identify local semantic concepts in input word lattices [7, 8]. By using FST the model must deal with ambiguities in the parsed output. For example the utterance "at ten past three" could be decoded as two valid times "at ten" or "at ten past three" if the part "past three" is optional in the local grammar. The ambiguity could be solved by a greedy approach [6] or by using statistical methods, e.g. HMM tagger [5].

Another issue is the ability to parse the uncertain ASR output. The uncertainty is usually expressed in the form of word lattices or word confusion networks. It has been shown that the uncertain ASR output improves the SLU performance [9, 10, 7]. While the word confusion networks are simpler for further processing and achieve lower oracle word error rates, they model the posterior probabilities of longer word sequences imprecisely [4]. The general structure of word lattices leads to higher computational demands during processing [9].

Semantic entity detection (SED) algorithm generates semantic entities based on the expert-defined knowledge to model the domain semantic entities (SEs) of common types (e.g. times, dates, names) [4]. The ambiguities in the SED process are solved using the integer linear programming (ILP) and the result is used to convert the word lattices into a generalized word-semantic (W-SE) lattices. Such W-SE lattices are then used in the statistical SLU model. The use of ILP allows us to define more strict rules for ambiguous or overlapping matches of local concepts than the simple finite state transduction. The method effectively combines the expert knowledge with the power of statistical learning. We use the theory of rational kernel functions [11, 12] to directly classify the W-SE lattices in the SLU model. This model is represented by a hierarchical discriminative model (HDM) [13] which is an extension of the Semantic Tuple Classifiers (STC) model [6].

The application of W-SE lattices in the SLU is described in the following sections. Sec. 2 introduces the ILP-based semantic entity detection, Sec. 3 describes the HDM model, Sec. 4 presents the method of generating the W-SE lattices. Sec. 5 shows the experimental results and Sec. 6 concludes the paper.

2. SEMANTIC ENTITY DETECTION

From the point of view of the SED algorithm the semantic entity (SE) is virtually a named entity with semantic interpretation assigned. Each SE consists of a type (e.g. time, date, names) and semantic tags used to describe the SE meaning. Semantic tags are also used to convert the lexical realisation of SE into a "computer readable" representation of the SE. In this approach each SE type is represented by separate CFGs. This assumption is not limiting, because the structure of many SE types is fixed and well-known.

Therefore the CFGs can be defined by an expert in a given domain or they can be generated automatically from the domain database.

In almost every modern speech recognizer the ASR lattices are a by-product of the Viterbi decoding. In this paper we use the lattice preprocessing method adapted from [14] to convert the raw ASR lattices into the form of a weighted finite state transducer (WFST). Alternative solution is to use a WFST-based decoder and generate the lattices directly [3].

Each type of semantic entity z has a corresponding CFG G_z. The grammars describing the semantic entities are non-recursive and therefore they can be converted into unweighted finite state transducers T_z without the need for an approximation. The grammars are designed in this way: the input side of the transducer T_z represents terminal symbols of the CFG G_z and the output side provides the interpretation of the corresponding path in T_z.[1]

For illustration, consider a sequence of words *ten past three*. Let's suppose that this sequence represents time information and we use the transducer T_{time} created from the corresponding grammar G_{time}. Thereafter the output semantic interpretation can be *time:10:p:3* and the corresponding machine representation *3:10 pm*. Naturally, there is usually more than one semantic entity type z in the real tasks. The complete expert knowledge about the task's semantic (the FST Z) is then represented by the union [16] of all individual T_zs, i.e. $Z = \bigoplus_z T_z$.

The process of generating the semantic interpretation is implemented by using a transducer composition algorithm [16, 17]. This approach has the advantage that it can be easily extended to the case where the input is the ASR lattice. There is also an optimized implementation of WFST composition [17]. At the same time it has some disadvantages: (1) we need to model all possible symbol sequences including meaningless words [7], (2) there could be ambiguities in the semantic entity assignment, (3) the given part of the utterance could have assigned multiple semantic representations, which is caused by the uncertain ASR hypotheses.

The solution of these problems was suggested in [4]. The issue (1) was solved by applying the approach of a factor automaton [18]. The factor automaton of a WFST represents a set of all paths and subpaths of the WFST. Therefore the subpaths containing meaningless words (i.e. words which are not among the terminal symbols of any G_z) are silently ignored during the composition of the factor automaton and the FST Z. Only the subpaths which bear the meaning according to the set of G_zs are processed. The problems (2) and (3) are solved by applying the heuristics of maximum unambiguous coverage [4], which can be converted to an optimization problem solvable using the integer linear programming (ILP). The solution of the ILP leads to a disambiguated set of semantic entities.

Now assume that U is a weighted finite state acceptor (WFSA) representing an ASR lattice and Z corresponds to compiled SE grammars in the form of an unweighted FST. The complete SED algorithm consists of the following steps: (1) convert WFSA U to WFST U_T by placing unique identifiers i to an input label of each transition in U, (2) create the factor automaton $F(U_T)$, (3) compose $F(U_T) \circ Z$ and (4) define and solve the ILP optimization to obtain the set $\mathcal{F}^*$ of unambiguous semantic entities.

Let $\mathcal{K}(U_T)$ and $\mathcal{W}$ denote the set of input and output labels of U_T, respectively. The assignment of unique identifiers can be expressed as a mapping $i = \mathrm{id}(w)$ where $i \in \mathcal{K}(U_T)$ is an unique identifier and $w \in \mathcal{W}$ is the output symbol (i.e. word) assigned to a transition with identifier i in U_T. Since $\mathrm{id}(\cdot)$ is a bijective mapping, the inverse mapping exists and will be denoted as $w = \mathrm{id}^{-1}(i)$.

Given an arbitrary ordering, the i-th item of the set $\mathcal{F}^*$ is the triplet (y^i, u^i, P^i) where: y^i is a semantic entity, i.e. the SE type and SE interpretation tags, for example *time:10:p:3*, u^i is a sequence of unique transition identifiers from U_T; $u^i = (1, 5, 8)$ says that the SE y^i is assigned to transitions with identifiers 1, 5 and 8 in U_T. These transitions form a subpath of U_T, and P^i is a posterior probability of the semantic entity y^i.

The probabilities P^i are computed from the weights of U_T and the semantic entity detection does not modify these weights. The posterior probability of the semantic entity is equal to the posterior probability of the underlying lexical realization in the lattice U.

Given the set $\mathcal{F}^*$ we can construct the lattice of semantic entities as was described in [4]. Such lattice is useful in dialog management to distinguish between the following two cases (assume the lattice contains two semantic entities SE_1 and SE_2): (1) SE_1 and SE_2 are alternative hypotheses based on different ASR hypotheses or (2) the user uttered first SE_1 and then SE_2.

3. HIERARCHICAL DISCRIMINATIVE MODEL

This section describes the Hierarchical Discriminative Model (HDM) introduced in [13]. The description of HDM uses the terminology of feed-forward neural networks – the *input layer* computes lexical features, the *hidden layer* transforms these features into a new feature space. The *output layer* then predicts lexicalized probabilities. The probabilities are used to parameterize the generalized probabilistic context-free grammar (PCFG). The symbols used in this PCFG correspond to the set of domain-dependent *semantic concepts*. These concepts represent the atomic units of semantics important in the given task – for example TIME, STATION, ACCEPT, CREATE etc.

The features generated by the hidden layer correspond to the presence or absence of a given semantic concept or concepts in the predicted semantic tree. For example, one feature can represent the presence/absence of the pair ACCEPT-STATION, another feature *root*-TIME etc. The hidden layer is equal to the Semantic Tuple Classifiers (STC) model introduced in [6]. It uses support vector machines (SVMs) with kernel values computed in the input layer. The predictions of these classifiers are not directly used – the distance to the decision boundary is used as an input of the output layer. The output layer employs a set of multi-class SVMs and uses the feature vector computed in the hidden layer to predict expansion probabilities of the PCFG. The rules of the PCFG are inferred from training data.

The structure of HDM is fully discriminative – it models directly the posterior probability distribution. In experiments presented in [13], the HDM outperforms both the generative model (Hidden Vector State parser [19]) and the discriminative model (STC). Moreover the parameters of the input layer are encoded using the weighted finite state transducer (WFST). The computation of the SVM rational kernel function values uses WFST operations [11] over a set of training lattices. We developed a method based on the factor automaton [12] to speed-up the computation of the rational kernel function values. By using this method, it is possible to compute the vector of 20k kernel function values in times of the order of milliseconds. Computation of rational kernel functions is defined using the WFST operations such as composition or ϵ-removal. Then, the HDM is able to process many input structures such as one-best hypotheses, word lattices or phoneme lattices [20]. In this paper we will use this feature of HDM to train the SLU model on the W-SE lattices which

[1] The output labels correspond to CFG tags as specified in the W3C Speech Recognition Grammar Specification (SRGS) [15].

combine the lexical units (words) and the semantic units (semantic entities).

4. WORD-SEMANTIC LATTICES

This section presents an extension of semantic entity lattices (Sec. 2). The idea is to process the ASR lattice and replace the subpaths in the lattice corresponding to a semantic entity with the SE interpretation. All other transitions in the original ASR lattice are unchanged. The transitions in the word-semantic lattice are labelled with the mix of the original words from ASR lexicon and the symbols representing the SE interpretations (see Fig. 1.c).

To construct the word-semantic (W-SE) lattice from the lattice U, the set of all semantic entities $\mathcal{F}^*$ has to be computed. For the construction of the W-SE lattice we will reuse the intermediate result of the SED algorithm – the WFST U_T which is labelled with input symbols unique to each transition. Given the set $\mathcal{F}^*$ we can construct the *mapping transducer* M which transduces the sequence of such unique identifiers to some symbols x^i (see Sec. 4.1). The input alphabet of M is $\mathcal{K}(U_T)$ and the output alphabet $\mathcal{S}$. The next step is to compose the mapping transducer with the input lattice:

$$C = \text{invert}\,(M) \circ U_T \tag{1}$$

The operator invert represents a WFST inversion (swapping input and output labels on each transition). This is because the input alphabet of M is $\mathcal{K}(U_T)$ which must be matched with the input alphabet of U_T. After performing the composition, the input side of the transducer C is virtually the word-semantic lattice and the output side is the source word lattice. The way to obtain the word-semantic lattice *WSE* is to project C on the input side (Π_1). Then the standard set of WFST optimization algorithms is applied: ϵ-removal (rmeps), determinization (det), minimization (min) and weight-pushing with normalization of transition probabilities from the initial state to sum $\bar{1}$ (push) [16, 21]:

$$\mathit{WSE} = \text{push min det rmeps}\,\Pi_1(C) \tag{2}$$

The internal structure of *WSE* depends on the structure of the mapping transducer M and the original lattice U.

4.1. Mapping transducer M

The mapping transducer M is an unweighted transducer with an input alphabet $\mathcal{K}(U_T)$ and an output alphabet $\mathcal{S}$. The mapping transducer is constructed for each lattice U_T. First, the set $\mathcal{K}(U_T)$ is divided into to two disjoint sets $\mathcal{K}_E(U_T)$ and $\mathcal{K}_W(U_T)$ such that:

$$\mathcal{K}_E(U_T) = \bigcup_{u^i \in \mathcal{F}^*} u^i \tag{3}$$

$$\mathcal{K}_W(U_T) = \mathcal{K}(U_T) \setminus \mathcal{K}_E(U_T) \tag{4}$$

The set $\mathcal{K}_E(U_T)$ represents the unique identifiers of such transitions in U_T which have assigned an SE. The set $\mathcal{K}_W(U_T)$ contains unique identifiers of transitions which do not form any SE. Such transitions will be preserved in the W-SE lattice with the original lexical symbols (words).

The initial state s of the mapping transducer is also its final state. For each identifier $i \in \mathcal{K}_W(U_T)$ the transitions t is inserted into M: origin and destination state of t is s, input symbol of t is i and output symbol of t is a word given by $\text{id}^{-1}(i)$. This ensures that the identifiers of transitions in U_T not belonging to any SE are mapped back to the original words (for an example see Fig. 1.b).

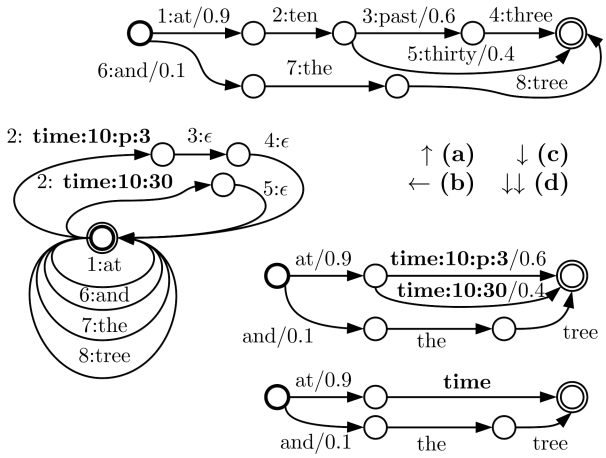

Fig. 1: **(a)** *The word lattice with unique identifiers U_T. The transition labels have the following structure. Assume G_{time} so that it assigns the following SEs to U_T:* time:10:p:3 *(transitions 2, 3, 4) and* time:10:30 *(2, 5). The SEs are typeset in boldface.* **(b)** *Corresponding mapping transducer M for the* full *derivation.* **(c)** *The resulting W-SE lattice,* full *derivation.* **(d)** *The resulting W-SE lattice,* type *derivation.*

For the i-th semantic entity from $\mathcal{F}^*$ the new path π^i in M is created. The sequence of input labels of π^i is equal to u^i. The sequence of output labels x^i of π^i is derived from y^i. The origin and destination state of π^i is again the state s. We evaluated the following four different derivations of x^i from y^i (in the following examples assume that $y^i =$ *time:10:p:3* and the corresponding word sequence is *ten past three*):

- *type* derivation – x^i is a single symbol corresponding to a SE type of y^i, e.g. $x^i = (time)$. This derivation is the most general and retains only the information that the W-SE lattice contains the specific SE types.
- *type*$_n$ derivation – x^i is a sequence of numbered SE types, the length of x^i is the same as the length of u^i, e.g. $x^i = (time_1, time_2, time_3)$. This derivation in addition to *type* preserves the number of tokens corresponding to a given SE in the W-SE lattice.
- *split* derivation – x^i is a sequence of tags from which the SE is composed, e.g. $x^i = (time, 10, p, 3)$. This derivation allows us to distinguish between different interpretations of the SEs.
- *full* derivation – x^i is a single symbol corresponding to whole SE y^i, e.g. $x^i =$ (*time:10:p:3*). This derivation uses the specific SE as the W-SE lattice transition label.

The derivations above are sorted from the most general derivation (*type*) to the most specific derivation (*full*). By using different derivations, the SLU model is able to select "the level of details" retained in the W-SE lattices. More details preserved in the W-SE lattice mean more parameters and the need for more examples in order to train the SLU.

5. EXPERIMENTAL RESULTS

We used two semantically annotated corpora collected for SLU in a spontaneous dialog system. The first one was the Human-Human Train-Timetable (HHTT) corpus [22]. The corpus contains inquiries

Table 1: *Corpora characteristics.*

	HHTT	TIA
# different concepts	28	20
# train sentences	5240	6425
# devel. sentences	570	519
# test sentences	1439	1256
ASR Vocabulary size	13886	42615
ASR Acc (Oracle Acc)	75.0% (84.6%)	77.9% (87.0%)

and answers about train connections. The second one was a Czech Intelligent Telephone Assistant (TIA) corpus containing utterances about meeting planning, corporate resources sharing and conference call management. These corpora contain unaligned semantic trees together with word-level transcriptions. We have split the corpora into train, development and test data sets (approximately 72:8:20) at the dialog level, so that the speakers do not overlap.

To evaluate the SLU performance we used the *concept accuracy* measure [13] defined as

$$cAcc = \frac{N - S - D - I}{N} = \frac{H - I}{N} \quad (5)$$

where H is the number of correctly recognized concepts, N is the number of concepts in reference and S, D, I are the numbers of substituted, deleted and inserted concepts. We used our in-house LVCSR decoder to obtain the word lattices [23]. The recognition accuracy and other characteristics of both corpora are summarized in Tab. 1. The CFG grammars G_z for the HHTT task were collected during the development of the knowledge-based spoken dialog system [24]. In this task the following SE types were used: *station*, *time*, *date*, *train_type*. The grammars for the second task were partially reused from the HHTT task (*date* and *time*) and two additional SE types were introduced: *name* (first and last names in arbitrary order and optional titles before and after the name) and *resname* (shared company resources such as *laptop* or *meeting room*).

First the W-SE lattices for a specific type of derivation (*type*, *type*$_n$, *split* and *full*) were generated using the task grammars G_z. Then, these lattices were used in the same way as ASR lattices during the HDM model training and prediction. The metaparameters of HDM were optimized on the development set. The results (concept accuracy $cAcc$) on the test data are reported in Tab. 2.

This table shows that the use of W-SE lattices improves the concept accuracy. The row *W-SE 1-best* shows the results for the case where the best word hypothesis is converted to word-semantic hypothesis using *type*$_n$ derivation for HHTT and *split* derivation for TIA. These results are better than the results of HDM trained from the best word hypothesis (without SEs), but worse than the results for full W-SE lattices. The best results were achieved for the *type*$_n$ derivation in the HHTT task and for *split* derivation in the TIA task. The table shows that the achieved concept accuracy varies with the level of details of the W-SE lattice. The low number of details and also the high number of details lead to lower parsing performance.

The table also shows the results for two additional cases: (1) SE completely removed from the W-SE lattices (row *SE removed*), i.e. W-SE lattice contains just the transitions which do not belong to any SE, and (2) the W-SE lattice consists only from SEs (row *SE only*). The second case is equal to semantic entity lattices generated by the SED algorithm. These two rows show that the word and SEs are complementary – it is not possible to reach the performance of the baseline with the use of just SE lattices or word lattices with SE removed. Just the fusion of these two data sources leads to an improvement in the concept accuracy of the HDM.

Table 2: *Results of HDM trained on two different tasks. The results should be compared with the baseline trained on the best hypothesis (words 1-best) and on the word lattice (words).*

Structure of HDM input	HHTT $cAcc$	TIA $cAcc$
words 1-best	73.96	81.96
words	74.90	83.55
W-SE 1-best	75.18	82.56
W-SE type	75.88	83.90
W-SE type$_n$	**76.67**	83.87
W-SE split	75.77	**84.40**
W-SE full	69.36	82.24
SE removed	57.97	74.59
SE only	59.54	51.32

Table 3: *Detailed results for specific semantic concepts. The following measures for each semantic concept are shown: F-measure (F), Precision (P) and Recall (R).*

	word lattice			W-SE lattice		
HHTT concept	F	P	R	F	P	R
TIME	**93.0**	93.0	93.0	**93.5**	93.1	94.0
TRAINTYPE	**88.6**	95.1	83.0	**87.3**	92.1	83.0
STATION	**79.7**	96.1	68.1	**84.4**	96.4	75.1

	word lattice			W-SE lattice		
TIA concept	F	P	R	F	P	R
TIME	**93.9**	94.7	93.0	**94.0**	94.6	93.4
NAME	**92.3**	97.0	88.1	**93.7**	97.1	90.5
RES	**80.8**	87.1	75.3	**86.8**	93.0	81.5

The Tab. 3 shows the detailed results for semantic concepts which relate to some SE. This table shows that the additional expert-defined knowledge encoded in SE grammars improves mainly the recall of these semantic concepts. By improving the prediction performance for these concepts the overall concept accuracy is also improved. This is caused by the use of semantic trees – the presence of a parent concept influences the set of its child concepts and vice versa.

6. CONCLUSION

The presented method generates combined word-semantic lattices. The W-SE lattices are useful in the SLU task due to reduction of data sparsity and uncertainty of the original ASR lattices. The presented method was evaluated in the SLU framework. This framework employs the SED algorithm and the HDM model. The overall reduction in the error rate is more than 10% for both of the tasks (best word hypothesis vs. W-SE lattice) with the p-value of two-tailed t-test less than 0.01. In comparison with the word lattices, the W-SE lattices reduce 7.1% of errors for the HHTT task and 5.2% of errors for the TIA task. This is mainly due to the improved concept recall caused by the additional expert knowledge.

7. ACKNOWLEDGMENTS

This research was supported by the Grant Agency of the Czech Republic, project No. GAČR GBP103/12/G084.

8. REFERENCES

[1] Filip Jurčíček, Blaise Thomson, Simon Keizer, François Mairesse, Milica Gašić, Kai Yu, and Steve Young, "Natural Belief-Critic: a reinforcement algorithm for parameter estimation in statistical spoken dialogue systems," in *Proceedings of Interspeech*, Chiba, 2010, number 1.

[2] Milica Gašić, C. Breslin, M. Henderson, D. Kim, M. Szummer, Blaise Thomson, Pirros Tsiakoulis, and Steve Young, "On-line policy optimisation of bayesian spoken dialogue systems via human interaction," in *Proceedings of IEEE International Conference on Acoustics Speech and Signal Processing ICASSP*, Vancouver, Canada, 2013, pp. 8367–8371, IEEE.

[3] Daniel Povey, Mirko Hannemann, Gilles Boulianne, Lukáš Burget, Arnab Ghoshal, Miloš Janda, Martin Karafiát, Stefan Kombrink, Petr Motlíček, Yanmin Qian, Korbinian Riedhammer, Karel Veselý, and Ngoc Thang Vu, "Generating Exact Lattices in the WFST Framework," in *IEEE International Conference on Acoustics Speech and Signal Processing*, Kyoto, Japan, 2012, vol. 213850, pp. 4213–4216, IEEE.

[4] Jan Švec, Pavel Ircing, and Luboš Šmídl, "Semantic entity detection from multiple ASR hypotheses within the WFST framework," in *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, Olomouc, Czech Republic, Dec. 2013, pp. 84–89, IEEE.

[5] Géraldine Damnati, Frédéric Béchet, and Renato De Mori, "First implementation of the LUNA Spoken Language Understanding strategy on a telephone service application," *Intelligent Information Systems 2008*, pp. 499–505, 2008.

[6] François Mairesse, Milica Gašić, Filip Jurčíček, Simon Keizer, Blaise Thomson, Kai Yu, and Steve Young, "Spoken language understanding from unaligned data using discriminative classification models," in *IEEE International Conference on Acoustics, Speech and Signal Processing, 2009. ICASSP 2009.*, Taipei, 2009, pp. 4749–4752, IEEE.

[7] Christian Raymond, Frédéric Béchet, Renato De Mori, and Géraldine Damnati, "On the use of finite state transducers for semantic interpretation," *Speech Communication*, vol. 48, no. 3-4, pp. 288–304, Mar. 2006.

[8] Christophe Servan, Christian Raymond, Frédéric Béchet, and Pascal Nocéra, "Conceptual decoding from word lattices: application to the spoken dialogue corpus media," in *Proceedings of International Conference on Spoken Language Processing*, Pittsburgh, 2006, pp. 1–4, ISCA.

[9] Dilek Hakkani-Tür, Frédéric Béchet, Giuseppe Riccardi, and Gokhan Tur, "Beyond ASR 1-best: Using word confusion networks in spoken language understanding," *Computer Speech & Language*, vol. 20, no. 4, pp. 495–514, Oct. 2006.

[10] Natthew Henderson, Milica Gašić, Blaise Thomson, Pirros Tsiakoulis, Kai Yu, and Steve Young, "Discriminative Spoken Language Understanding Using Word Confusion Networks," in *Spoken Language Technology Workshop (SLT), 2012 IEEE*, 2012, pp. 176–181.

[11] Corinna Cortes and Patrick Haffner, "Rational kernels: Theory and algorithms," *The Journal of Machine Learning*, vol. 5, pp. 1035–1062, 2004.

[12] Jan Švec and Pavel Ircing, "Efficient Algorithm for Rational Kernel Evaluation in Large Lattice Sets," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, 2013*, Vancouver, Canada, 2013, pp. 3133–3137, IEEE.

[13] Jan Švec, Luboš Šmídl, and Pavel Ircing, "Hierarchical Discriminative Model for Spoken Language Understanding," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, 2013*, Vancouver, Canada, 2013, pp. 8322–8326, IEEE.

[14] Dogan Can and Murat Saraclar, "Lattice Indexing for Spoken Term Detection," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 19, no. 8, pp. 2338–2347, 2011.

[15] A. Hunt and S. McGlashan, "Speech recognition grammar specification version 1.0," W3C Recommendation, Mar. 2004.

[16] Mehryar Mohri, "Weighted automata algorithms," *Handbook of weighted automata*, 2009.

[17] Cyril Allauzen, Michael Riley, and Johan Schalkwyk, "OpenFst: A general and efficient weighted finite-state transducer library," *Implementation and Application of Automata*, vol. 4783, pp. 11–23, 2007.

[18] Mehryar Mohri, Pedro Moreno, and Eugene Weinstein, "Factor automata of automata and applications," *Implementation and Application of Automata*, vol. 4783, pp. 168–179, 2007.

[19] Filip Jurčíček, Jan Švec, and Luděk Müller, "Extension of HVS semantic parser by allowing left-right branching," in *Proceedings of IEEE International Conference on Acoustics Speech and Signal Processing 2008*. 2008, number 1, pp. 4993–4996, IEEE.

[20] Jan Švec and Luboš Šmídl, "On the Use of Phoneme Lattices in Spoken Language Understanding," *Text, Speech and Dialogue*, vol. 8082, pp. 369–377, 2013.

[21] Mehryar Mohri, Fernando C. N. Pereira, and Michael Riley, "Weighted automata in text and speech processing," in *Proceedings of the 12th biennial European Conference on Artificial Intelligence*, Budapest, 1996, pp. 46–50, John Wiley and Sons.

[22] Filip Jurčíček, Jií Zahradil, and Libor Jelínek, "A human-human train timetable dialogue corpus," *Proceedings of EUROSPEECH, Lisboa*, pp. 1525–1528, 2005.

[23] Aleš Pražák, Josef V. Psutka, Jan Hoidekr, Jakub Kanis, Luděk Müller, and Josef Psutka, "Automatic online subtitling of the Czech parliament meetings," *Text, Speech and Dialogue*, vol. 4188, pp. 501–508, 2006.

[24] Tomáš Valenta, Jan Švec, and Luboš Šmídl, "Spoken Dialogue System Design in 3 Weeks," *Text, Speech and Dialogue*, vol. 7499, no. IV, pp. 624–631, 2012.