Two-layer Semantic Entity Detection and Utterance Validation for Spoken Dialogue Systems

Adam Chýlek, Jan Švec, and Luboš Šmídl *

University of West Bohemia, Faculty of Applied Sciences
New Technologies for the Information Society
Univerzitní 22, 306 14 Plzeň, Czech Republic
{chylek,honzas,smidl}@kky.zcu.cz

Abstract. In this paper we present a novel method for semantic entity detection in a limited domain for spoken language understanding. The target domain of this method is a dialogue system for an interactive training of air traffic controllers (ATC). The method comprises of two layers of detection. First layer uses formerly proposed method for semantic entity detection to extract domain-dependent set of semantic entities. This semantic entities are modelled using context-free grammars. To detect mispronounced words or words which do not comply with the ATC radio-telephony rules we use the second layer of semantic entity detection. Together with that, we assign a semantic meaning to the utterance. We also discuss the possibility of using this approach for semantic-based correction of an utterance. The experiments were performed on transcribed data as well as on an output from speech recognizer.

Keywords: semantic entity detection, spoken language understanding, finite state machines

1 Introduction

The need for a novel approach for semantic entity detection and utterance validation comes from the development of a spoken dialogue system for interactive ATC training. Such training comprises of ATC-to-pilot communication lessons, where an ATC trainee controls virtual airspace in order to learn the rules and phrases that apply in air control environment. An ATC trainee communicates with several human pseudo-pilots that respond to commands or initiate the communication based on time plan created by an instructor. Pseudo-pilots process the commands into an input for virtual aircraft, which is visible on both pseudo-pilot's and trainee's radar screen.

The goal of this system is to provide a means of communication for ATC trainees mimicking real air traffic communication with pilots, without the actual need for human pseudo-pilots using dialogue system with automatic speech recognition (ASR), spoken language understanding (SLU) and text-to-speech (TTS) capability. At present, the dialogue system uses only semantic entity detection to highlight important semantic entities for human pseudo-pilot who then relays commands to several virtual aircraft

* This work was supported by the Technology Agency of the Czech Republic project No. TE01020197 and by an internal grant SGS-2013-032.

P. Sojka et al. (Eds.): TSD 2014, LNAI 8655, pp. 563–570, 2014.

under his control according to instructions from ATC. Such entities are *flight levels, communication frequencies, headings, clearances*, etc. We propose a method to not only extract those entities, but also assign another semantic meaning to the whole instruction, allowing us to manage virtual aircraft according to ATC's commands as a part of dialogue system.

Our method is also designed to validate user's utterance, because the domain of ATC radio communication is based on rules strictly defined in phraseology guides and as a part of their training, ATC trainees have to follow the rules without any deviation. The dialogue system can then use the information about utterance validity in order to give a feedback to its user on whether (and where) was the input incorrect. We also propose a method for utterance correction, giving the system an opportunity to not only validate against a set of rules, but also to offer the closest (in terms of Levenshtein distance) valid sequence of words and semantic entities the user might want to say instead.

Given that ATC communication is restricted by rules, we have chosen knowledge-based approach of spoken language understanding. The expert knowledge is represented as in many current commercial dialogue system by probabilistic context free grammars (PCFGs) which are used for both speech recognition and understanding. The developer of PCFG grammar does not need to be a dialogue system expert to write a grammar with good coverage of target semantic entities. On the other hand, the expansion probabilities are very hard to determine using knowledge only. Because of that, the probabilistic nature of PCFG is very rarely used and a large portion of recognition and understanding grammars are virtually deterministic context free grammars (CFG). The fact that CFG for a given semantic entity can be transferred and shared between many dialogue systems allows for rapid development of a SLU module for a new dialogue domain.

2 Semantic Entity Detection

A widely used approach in SLU systems is preprocessing of input sentences with respect to lexical classes. In the Semantic Tuple Classifier model [4], prior to training and decoding, lexical classes are replaced with class labels. The description of lexical classes consist of a list of lexical units pertaining to a given class. Having more complicated structures such as aircraft callsigns or time entities it is usual to replace only parts of the semantic entity with the corresponding lexical class. This paper uses an approach which allows to detect semantic entities described using a grammar (or finite state transducer) in a generic weighted finite state transducer (WFST). Our in-house ASR decoder [8] is able to compute directly the transition probabilities in such WFST. In other words, the lattices are normalized, the probabilities of all hypotheses sums to one. For manipulation with WFST we are using OpenFST framework [1].

We have considered using word confusion networks (WCN) for optimised representation of ASR output, appearing for example in [2]. Their results promise the same detection performance as with word lattices, but allow higher processing speeds. We have encountered problems with this approach. Using a simple lattice F (Fig. 1) we've created WCN F' (Fig. 2) using an algorithm from [2], where the best path is chosen as the pivot path. Consider a sequence of words ab. We can clearly see that in the original lattice the probability of this sequence was 0.5, whereas the probability in WCN was

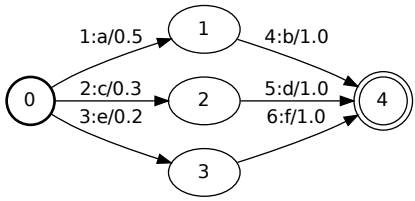

Fig. 1. Lattice example

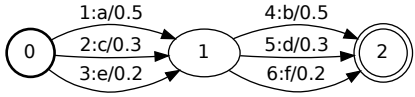

Fig. 2. Word confusion network example

reduced to 0.25. Experiments in [10] pointed out that the dominance of WCN stands only for single-word entities. Because our entities consist mostly of multiple words, we have chosen to use word lattices instead.

2.1 First Layer

Each type of semantic entity z has a corresponding CFG G_z. Generally, the CFG is parsed using the pushdown automaton with unlimited stack depth. In the case where CFG is not recursive, the stack depth is limited and such an automaton can be converted into a FST where the input symbols correspond to CFG terminal symbols and output symbols are the so called tags assigned by the CFG. If the CFG is recursive, it can be approximated by FST using for example algorithm presented in [7].

In this work, we use the standardized W3C speech recognition grammar specification (SRGS) [3] notation which allows us to use tags inside the rules definitions (Fig. 3). The use of a knowledge-based CFGs does not imply that the method is not probabilistic – it allows to assign posterior probabilities to every semantic entity detected by CFGs.

```
$number = ($d | ten{10} | twenty{20}[$d]);
$d = (one {1} | two {2} | three {3});
$runway = runway {RWY} $number;
```

Fig. 3. Example of grammar with tags for semantic entity detection

Considering our ASR hypotheses are represented as a WFST $\tilde{u}$, we first compile the knowledge base expressed as set of CFGs G_z into a transducer which accepts a string of symbols representing exactly one semantic entity and transduces this string onto a sequence of semantic tags e_i. The output symbols of T_z directly form the entity type and interpretation and allow the construction of "machine readable" objects (database

entries, time objects) in a dialogue manager. It is supposed that the first symbol in e_i indicates the type of semantic entity and the remaining symbols are its interpretation.

We represent all the transducers T_z created from grammars G_z as a single transducer by making an union $Z = \bigoplus_z T_z$. Because such transducer does not model all the words an input lattice can contain, it is infeasible to generate desired output by simple composition $\tilde{u} \circ Z$. As an alternative to creation of filler model which would match any of the words unseen in grammars, we create a factor automaton $F(\tilde{u})$ from the input lattice $\tilde{u}$. The factor automaton accepts all subpaths of the original lattice $\tilde{u}$ [6].

We can then generate transducer which encodes a mapping from a lattice subpath to a semantic entity by a composition $R = F(\tilde{u}) \circ Z$. Such composition also contains paths representing only partial matches of G_z. We then use heuristics of maximum unambiguous coverage to obtain subset F^* containing only unambiguously assigned entities from the set of all ambiguous semantic entities F. From the set F we use the subset F^* where each transition in the lattice has assigned at most one semantic entity and the number of transitions with assigned semantic entities is maximal.

An algorithm which takes F^* and time alignment of semantic entities to produce a lexical-semantic lattice F_e where each path encodes one sequence e consisting of semantic entities and words that did not match any grammar G_z is described in [10]. The weights of F_e correspond to posterior probability distribution $P(E = e|W = \tilde{u})$.

2.2 Second Layer for Validation

After creating a lexical-semantic lattice F_e in the first layer, we can now use a set of rules G_v to validate that the sequence of words and semantic entities correspond to phraseology rules. Because most of the phraseology rules have also a semantic information associated with it (for example *change of flight level, report of heading*, we chose to design this validating layer to not only emit information about validity, but also assign semantic meaning to the utterance.

Because the input for the second layer is a lexical-semantic lattice, we are using an lexical-semantic context free grammar in SRGS format made by expert using phraseology guide as the source of rules. These rules have also assigned tags to them to allow the dialogue manager to decide on further actions. An example of such rules are shown in Fig. 4, where the first rule states that path "climbing to _FL_ _CS_" in lexical-semantic lattice means the system should react to a "change of flight level confirmed from pilot.". We use "_" at the beginning and end of the names of semantic entities in order to distinguish them from words.

```
{air_response} climbing to _FL_ _CS_ {level_change} |
{air_response} maintaining _FL_ _CS_ {level_report}
```

Fig. 4. Example of grammar for utterance validation and further semantic annotation with semantic entities surrounded by "_"

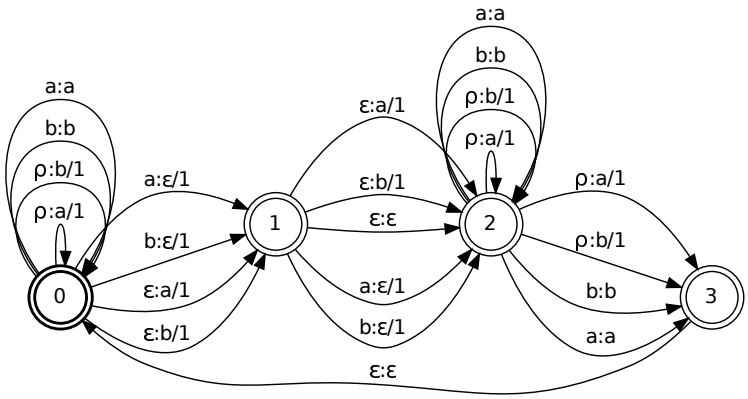

Fig. 5. Finite state transducer E for evaluation of edit distance between transducer F_1, F_2 using composition $F_1 \circ E \circ F_2$ with symbol table containing symbols *a* and *b* with only 2 consecutive deletions or insertions allowed.

The set of rules G_v is then converted into FST F_v. Because of the way the grammar was created, there is guaranteed to be no recursion. That allows us to convert to FST without any need of approximation as in Sec. 2.1. F_v is created in such way that semantic entities from first layer are considered only on the level of entity types, independent of their interpretation (i.e. entity `_FL_:2:3:0` with type `_FL_` and interpretation `230` matches any `_FL_` in rules G_v).

We are able to validate user's utterance by composing $R_v = F_e \circ F_v$ with lexical-semantic lattice F_e from first layer. If the resulting transducer R_v is empty, then we consider the input invalid, because we didn't find any rules the utterance would match. Otherwise we consider the input valid and we can find the semantic information assigned to it by traversing transducer R_v and noting the output symbols on its arcs.

As this layer does not use any probabilistic approach (rules in G_v do not have any probability assigned to them), the cost of each path is determined only from the score given by previous layer. In case of multiple hypotheses is the best hypotheses based solely on those scores.

2.3 Second Layer for Utterance Validation and Correction

To further enhance our utterance-validating layer, we extend the final step of composing $F_e \circ F_v$ by using intermediary FST that allows us to correct user's input. The algorithm produces lattice where can all paths be considered as valid (based on the same grammar created by an expert as in Sec. 2.2), with cost of the paths being a measure of how many correction had to be made to make them valid.

We are using Levenshtein distance to find the closest valid input interpretation. Concretely, we are using a finite state transducer E to compute edit distance between two transducers with restrictions on number of consequent deletions or insertions c_{id} based on [5].

The transducer E is created over a tropical semiring with $c_{id} + 2$ states, with state s_0 as a start state and $s_{c_{id}+1}$ as final state. For each symbol w from a set of symbols $W = W_e \cup W_v$, where W_e is a set of symbols from the first layer's output F_e and W_v is a set of symbols from F_v, we create several arcs in E. Considering s_0 as a starting state, we create arc from s_0 to s_0 transcribing input w to ρ with a weight of 1 (we will denote this arc as $[s_0, s_0, w, \rho, 1]$). The symbol ρ stands for "any symbol not seen on any of the arcs starting from the same state". This arc represents the operation of substitution. For substitution we also add arcs $[s_{c_{id}}, s_{c_{id}}, w, \rho, 1]$ and $[s_{c_{id}}, s_{c_{id}+1}, w, \rho, 1]$. We then add arcs $[s_i, s_{i+1}, w, \epsilon, 1]$ representing deletion and $[s_i, s_{i+1}, \epsilon, w, 1]$ representing insertion for $i \in \{0, \ldots, c_{id}\}$. To allow for unchanged words we add arcs $[s_0, s_0, w, w, 0]$,$[s_{c_{id}}, s_{c_{id}}, w, w, 0]$ and $[s_{c_{id}}, s_{c_{id}+1}, w, w, 0]$. Finally, we add arcs $[s_j, s_{j+1}, \epsilon, \epsilon, 0]$ for $j \in \{1, \ldots, c_{id}\}$ and $[s_{c_{id}+1}, s_0, \epsilon, \epsilon, 1]$, so that we can repeat whole pattern or use less than two insertions/deletions.

An example of such transducer allowing for computation of edit distance over a small set of symbols $W = \{a, b\}$ is shown in Fig. 5. Restriction on two deletions or insertions in a row can be seen between states number 0 and 2.

Composing input transducer, edit distance transducer and grammar transducer results in FST $R = F_e \circ E \circ F_v$. An important step before the composition itself is unweighting of transducer F_e, because weights are being used to compute the edit distance. We are using tropical semiring in order for the resulting distance between strings α and β (with insertion, deletion and substitution being penalized by 1) to be obtained as a sum of costs of transitions transcribing string α into β.

It is possible that transducer R contains paths where an edit distance (represented as cost of the path) is too high to be of any use for utterance correction. We can prune the FST R with a reasonably low threshold r, leaving us with transducer R_r that contains only paths with cost lower than the cost of the shortest path increased by r and the dialogue system can then offer all the closest valid variations to the user. We can also choose to use only n-best hypotheses instead of pruning, leaving us with guaranteed number of suggestions to the user.

The algorithm marks the input utterance as valid if there is a path in R_r with cost $\bar{0}$ (i.e. with edit distance equal to 0). Finding this path is a task of finding best path in R_r and then summing up the weights.

Table 1. Percentage of utterances corresponding exactly to the phraseological rules (referred to as valid) in test set using two-layer validation with precision, recall and F1-measure against reference

	1-best	raw	reference
valid	25.17%	29.29%	23.22%
precision	0.76	0.72	–
recall	0.83	0.91	–
F1	0.79	0.80	–

3 Experiments

The experiments were performed on data [9] that were obtained from real communication between pilots and ATCs. The test set contains 1434 sentences. The ASR recognizer vocabulary contains 10.3k words with accuracy of ASR being 83.9%. We were using the following representation of ASR hypothesis: the best hypothesis (1-best) and the raw unoptimized lattice. We've also used reference transcriptions made by human annotators.

The grammars used in our experiments were defined by phraseology guide for semantic entities in first layer as well as for validity rules in second layer.

In our experiments, we did not evaluate the number of semantic entities and validation rules being covered by expert-defined grammars. Based on fixed set of grammars, we evaluated the influence of different ASR hypotheses representations on the validation performance. As a reference for validation, we used results of semantic entity detection and validation on the manually annotated transcription. We consider utterance valid, if it corresponds exactly to the rules.

Results can be seen in Table 1. The number of valid utterances was increased by 14% by using raw lattices instead of 1-best hypothesis. Confusion matrices are in Table 2 for 1-best hypotheses and Table 3 for raw lattices.

We assume the difference between valid sentences in reference transcription and ASR output is caused by the fact that language model in ASR tries to create sequences based on n-grams seen in training data, whereas human annotators that do not know the common word order in a pilot-to-ATC communication (hence do not posses any training data) prefer to mark words only as unintelligible instead of trying to guess.

Table 2. Confusion matrix for validation results using 1-best ASR hypotheses

	valid in reference	invalid in reference
valid from 1-best	275	86
invalid from 1-best	58	1015

Table 3. Confusion matrix for validation results using raw lattice ASR hypotheses

	valid in reference	invalid in reference
valid from lattice	303	117
invalid from lattice	30	984

4 Conclusion and Future Work

We presented a novel algorithm for semantic entity detection with utterance validation and its extension to utterance correction. Using this algorithm we were able to determine the percentage of valid sentences in our data set and assign semantic information to user's input which will be useful in further dialogue system development.

The results showed that by using a raw lattice from an ASR we are not only able to extract a valid hypothesis, but also increase the percentage of valid utterances our system recognized in comparison to the ones retrieved from 1-best hypothesis.

Further work can be done in utterance correction. Our approach using unweighted input lattices causes a loss of information about hypothesis probability obtained from language and acoustic model. This can be improved by using other than tropical semirings, for example lexicographic semiring would allow us to store both the edit distance and the probability from an ASR encoded into weights of transitions.

References

1. Allauzen, C., Riley, M., Schalkwyk, J., Skut, W., Mohri, M.: OpenFst: A general and efficient weighted finite-state transducer library. In: Proceedings of the Ninth International Conference on Implementation and Application of Automata, (CIAA 2007). Lecture Notes in Computer Science, vol. 4783, pp. 11–23. Springer (2007), `http://www.openfst.org`
2. Hakkani-Tür, D., Béchet, F., Riccardi, G., Tur, G.: Beyond asr 1-best: Using word confusion networks in spoken language understanding. Computer Speech & Language 20(4), 495–514 (2006)
3. Hunt, A., McGlashan, S.: Speech recognition grammar specification version 1.0. World Wide Web Consortium, Recommendation REC-speech-grammar-20040316 (March 2004)
4. Mairesse, F., Gasic, M., Jurcicek, F., Keizer, S., Thomson, B., Yu, K., Young, S.: Spoken language understanding from unaligned data using discriminative classification models. In: Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on. pp. 4749–4752 (April 2009)
5. Mohri, M.: Edit-distance of weighted automata. In: Implementation and Application of Automata, pp. 1–23. Springer (2003)
6. Mohri, M., Moreno, P., Weinstein, E.: Factor automata of automata and applications. In: Implementation and Application of Automata, pp. 168–179. Springer (2007)
7. Mohri, M., Nederhof, M.J.: Regular approximation of context-free grammars through transformation. In: Robustness in language and speech technology, pp. 153–163. Springer (2001)
8. Pražák, A., Psutka, J.V., Hoidekr, J., Kanis, J., Müller, L., Psutka, J.: Automatic online subtitling of the Czech parliament meetings. In: Text, Speech and Dialogue. pp. 501–508. Springer (2006)
9. Šmídl, L.: Air traffic control communication corpus. Published in LINDAT/CLARING repository, available under CC BY-NC-ND 3.0 from `http://hdl.handle.net/11858/00-097C-0000-0001-CCA1-0` (2012)
10. Švec, J., Ircing, P., Šmídl, L.: Semantic entity detection from multiple ASR hypotheses within the WFST framework. In: Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on. pp. 84–89 (Dec 2013)