# Discriminative Spoken Language Understanding Using Statistical Machine Translation Alignment Models

Mohammad Aliannejadi[1(✉)], Shahram Khadivi[1], Saeed Shiry Ghidary[1], and Mohammad Hadi Bokaei[2]

[1] Department of Computer Engineering and Information Technology, Amirkabir University of Technology, Tehran, Iran
{m.aliannejadi,khadivi,shiry}@aut.ac.ir
[2] Department of Computer Engineering, Sharif University of Technology, Tehran, Iran
bokaei@ce.sharif.edu

**Abstract.** In this paper, we study the discriminative modeling of Spoken Language Understanding (SLU) using Conditional Random Fields (CRF) and Statistical Machine Translation (SMT) alignment models. Previous discriminative approaches to SLU have been dependent on *n-gram* features. Other previous works have used SMT alignment models to predict the output labels. We have used SMT alignment models to align the abstract labels and trained CRF to predict the labels. We show that the *state transition* features improve the performance. Furthermore, we have compared the proposed method with two baseline approaches; Hidden Vector States (HVS) and baseline-CRF. The results show that for the F-measure the proposed method outperforms HVS by 1.74 % and baseline-CRF by 1.7 % on ATIS corpus.

**Keywords:** Spoken language understanding · Statistical machine translation · Conditional random fields · Hidden vector state · Discriminative modeling · Sequential labeling · Natural language processing

## 1 Introduction

Spoken Language Understanding (SLU) is the problem of extracting the intention and aim of the user's utterance. More specifically, a SLU system tries to find a mapping from user's utterance in natural language, which follows no rules or restrictions, to the limited set of concepts that is structured and meaningful for the computer. The only restriction for the user's input is the domain of the utterance, i.e. a SLU system trained in Air Travel domain, can be used only in this domain.

Below is an example of Air Travel Information System (ATIS) [1] corpus. The input utterance is:

I want to return to Dallas on Thursday

and its corresponding output is:


A. Movaghar et al. (Eds.): AISP 2013, CCIS 427, pp. 194–202, 2014.
DOI: 10.1007/978-3-319-10849-0_20

```
    GOAL : RETURN
TOLOC.CITY = Dallas
RETURN.DATE = Thursday .
```

The output indicates that the goal of the utterance is to get the list of *RETURN* flights. The arrival city of the flights is "Dallas" (which is labeled as *TOLOC.CITY*) and the desired date of the flights is "Thursday" (which is labeled as *RETURN.DATE*). A simple algorithm is able to generate a SQL command from the system's output. Therefore the list of return flights to Dallas on Thursday will be available for the user automatically.

The first statistical method for SLU was based on Hidden Markov Model (HMM) using a finite state semantic tagger which was a part of AT&T's CHRONUS system [2]. In [2] semantic representation was flat-concept but later Hidden Vector State (HVS) extended the representation to a hierarchical structure and modeled the problem using a Push-down automaton [3].

The problem can been converted to a sequential labeling problem. Discriminative methods deal directly with the aligned data (fully annotated) using sequential classification and Conditional Random Fields (CRF) [4,5]. Discriminative sequence labeling was a success because of its power in utilizing problem specific features, comparing to HMM. Linear-chain CRF combines the advantage of discriminative modeling and sequence modeling [6].

In [7–9] the problem of SLU was dealt by SMT approaches. S.D. Pietra et al. In [7] have used fertility models to predict labels in ATIS corpus and R. Macherey et al. In [8] and later in [9] have applied numerous alignment models for this task where the source language is the user's utterance and the target langauge is a formal language of the labels. All these works have used SMT alignment models to *predict* the labels. These models are not able to capture many dependnecies and difficulties of this problem, thus they will not be appropriate for predicting the labels. On the other hand, these models are able to *align* the training data to their abstract (unaligned) labels at high accuracy.

In this paper we automatically annotate the training data using SMT alignment models in order to have a better flat-concept version of the hierarchical semantic representation and trained CRF using the new annotated corpus. We show that this method outperforms other flat-concept fully annotation methods proposed before trained using CRF and also outperforms other methods dealing with abstract annotation directly.

The remainder of paper is organized as follows. Section 2 briefly describes CRF for sequential labeling. Section 3 briefly presents basic SMT alignment models. Section 4 presents the combination of SMT alignment for annotation and linear-chain CRF for sequential labeling task. The experimental evaluation using ATIS is presented in Sect. 5 and finally Sect. 6 concludes the paper.

## 2 Conditional Random Fields

Discriminative modeling of the problem leads to linear-chain CRFs for the task of mapping the input sequence ($\mathbf{x}$) to the sequence of labels ($\mathbf{y}$) [6]. $\mathbf{x}$ ranges over

all natural language sentences but $\mathbf{y}$ ranges over the finite label set $\mathcal{Y}$. In our problem the label set consists of labels such as `FLIGHT, TOLOC, FROMLOC, etc.` $\mathbf{x}$ and $\mathbf{y}$ are jointly distributed, but using CRF as a discriminative framework, the marginal $p(\mathbf{x})$ is not explicitly modeled, instead $\mathbf{x}$ and $\mathbf{y}$ are conditionally modeled as $p(\mathbf{y}|\mathbf{x})$ [6]. A linear-chain CRF is defined by a dependency graph $G$ and a set of features $f_k$ which the weights $\lambda_k$ are associated with them. The main difference of CRF as a dicriminative method to generative methods such as HMM is that CRF models the conditional probability $p(\mathbf{y}|\mathbf{x})$ directly utilizing problem-specific features, whereas HMM decomposes the probability function using Bayes theorem.

The conditional probability of a sequence of labels given a sequence of observations is given by:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(x)} \exp\Big(\Big\{\sum_{k=1}^{K} \lambda_k f_k(y_t, y_{t-1}, \mathbf{x}_t)\Big\} \tag{1}$$

with

$$Z(\mathbf{x}) = \sum_{y} \exp\Big\{\sum_{k=1}^{K} \lambda_k f_k(y_t, y_{t-1}, \mathbf{x}_t)\Big\}. \tag{2}$$

Feature functions are used to encode semantic features, concept relations, etc. in the model [4]. Mostly, these features are used in binary form, i.e. if the feature triggers, it returns 1 and otherwise 0. The parameters of the model are the weights of the features, $\lambda_k$, and the learning phase is the process of finding the optimum set of weights.

## 3 IBM Alignment Models

Given a source sentence $\mathbf{f} = f_1, \ldots, f_j, \ldots, f_J$, the best target sentence $\mathbf{e} = e_1, \ldots, e_i, \ldots, e_I$ is the one which maximizes $p(\mathbf{e}|\mathbf{f})$ [9–11]:

$$\begin{aligned} \hat{\mathbf{e}} &= \arg\max_{\mathbf{e}} \big\{p(\mathbf{e}|\mathbf{f})\big\} \\ &= \arg\max_{\mathbf{e}} \big\{p(\mathbf{f}|\mathbf{e}).p(\mathbf{e})\big\}. \end{aligned} \tag{3}$$

Introducing $\mathbf{a} = a_1 \ldots a_j \ldots a_J$ as the *hidden* alignment set, where each $a_j \in \{1, \ldots, I\}$, Eq. 3 will be updated as [10,12]:

$$\begin{aligned} p(\mathbf{f}|\mathbf{e}) &= \sum_{\mathbf{a}} p(\mathbf{f}, \mathbf{a}|\mathbf{e}) \\ &= p(J|\mathbf{e}) \; . \; \sum_{\mathbf{a}} \prod_{j=1}^{J} p(f_j, a_j | f_1, \ldots, f_{j-1}, a_1, \ldots, a_{j-1}, \mathbf{e}) \qquad (4) \\ &= p(J|\mathbf{e}) \; . \; \sum_{\mathbf{a}} \prod_{j=1}^{J} p(a_j | a_1 \ldots a_{j-1}, f_1, \ldots, f_{j-1}, \mathbf{e}) \\ &\quad . \; p(f_j | f_1 \ldots f_{j-1}, a_1, \ldots, a_{j-1}, \mathbf{e}). \qquad (5) \end{aligned}$$

The first term of Eq. 5 is the *length* model, the second is the *alignment/distortion* model and the last is the *lexicon/translation* model [9]. Different alignment models were proposed in [10,12], which are the base of the alignment phase of this work. The base of all these alignment models are the same as the HMM principle used in speech recognition [13] which we briefly introduce here.

### 3.1 HMM and IBM-1,2

Assuming a first-order dependence for $a_j$ in Eq. 5 and other suitable modeling assumptions [10] the HMM alignment model will be:

$$p(\mathbf{f}|\mathbf{e}) = p(J|I) \cdot \sum_{\mathbf{a}} \prod_{j=1}^{J} p(a_j|a_{j-1}, I) \cdot p(f_j|e_{a_j}). \tag{6}$$

IBM-1 and IBM-2 alignment models are obtained analogously by assuming zero-order dependencies.

### 3.2 IBM-3,4,5

In models of IBM-3 and 4, a new concept was introduced, *fertility*. Using the concept of *inverted alignment* to perform a mapping from target position $i$ to a set of source positions $j$, the fertility of a target word is the number of source words it can generate. Considering mappings $\mathcal{B}$ of the form:

$$\mathbf{b} : i \rightarrow b_i \subset \{1, \ldots, j, \ldots, J\},$$

with the constraint to cover each source position $j$ exactly once, Eq. 4 is rewritten using $\mathbf{a} = \mathbf{b}$:

$$p(\mathbf{f}, \mathbf{b}|\mathbf{e}) = p(J|I) \cdot \prod_{i=1}^{I} \left[ p(b_i|b_1, \ldots, b_{i-1}) \cdot \prod_{j \in b_i} p(f_i|e_i) \right].$$

IBM-3 and 4 has the problem which a source position could be chosen twice. Keeping track of the empty source positions IBM-5 overcomes this problem [10].

## 4 Training the Model Using IBM Alignments

In this section we show how to use IBM alignment results to train a linear-chain CRF for the sequence labeling task and describe the main advantage of it over previous works.

### 4.1 Alignment

The problem with discriminative models is that the data should be fully annotated, and because of that, in previous works some efforts were done to obtain

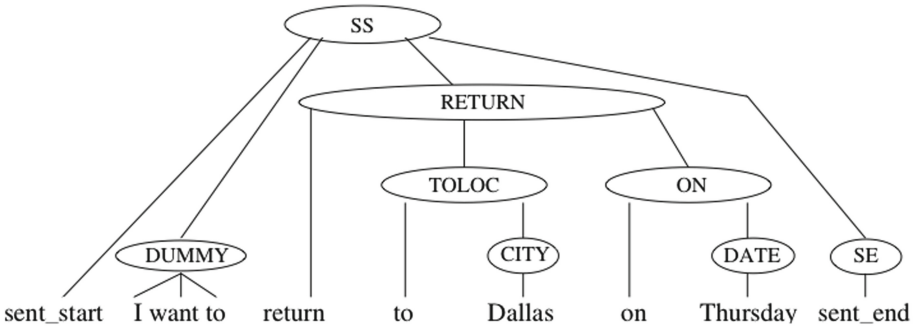


**Fig. 1.** The final parsed treed using HVS model [3]

fully annotated corpus from the unaligned annotation [4,5]. Figure 1 shows the annotation which was used in [3]. Note that in [3] the data is *unaligned*, so for the example shown in Fig. 1 the annotation for the sentence "sent_start I want to return to Dallas on Thursday sent_end" is:

```
SS(RETURN(TOLOC(CITY) ON(DATE)) SE)
```

In this example, the hierarchical semantic relationships [3] are presented in an abstract manner. In [4] a simple algorithm is proposed. Using this simple algorithm, a fully annotated data can be obtained in the following form:

$$\underbrace{\text{I}}_{null}\ \underbrace{\text{want}}_{null}\ \underbrace{\text{to}}_{null}\ \underbrace{\text{return}}_{null}\ \underbrace{\text{to}}_{null}\ \underbrace{\text{Dallas}}_{toloc.city}\ \underbrace{\text{on}}_{null}\ \underbrace{\text{Thursday}}_{return.date}$$

The only words that are tagged non-*null* are *Dallas* and *Thursday* and their tags are in two parts, the first part is the parent node in Fig. 1 and the second is the child. The main disadvantage of this annotation is that many words are labeled as *null*. In Fig. 1 it is obvious that many words such as "return" could be labeled to non-*null* (*RETURN*) labels. We will call this alignment method *Raymond's alignment*[1] in the remainder of the paper.

Another full annotation which was proposed in [5] is:

$$\underbrace{\text{I}}_{SR.ACity.pre}\ \underbrace{\text{want}}_{SR.ACity.pre}\ \underbrace{\text{to}}_{SR.ACity.pre}\ \underbrace{\text{return}}_{SR.ACity.pre}$$

$$\underbrace{\text{to}}_{SR.ACity.pre}\ \underbrace{\text{Dallas}}_{SR.ACity.start}\ \underbrace{\text{on}}_{SR.RDate.pre}\ \underbrace{\text{Thursday}}_{SR.RDate.start}$$

This annotation presents the labels in three parts. The first part indicates the goal of the utterance. In this example the goal of the utterance is "ShowReturn", therefore the first part of all the labels is *SR*. The second part indicates the slot and the third part indicates the state of the current word regarding to its slot. "Dallas" is the ArrivalCity (which is labeled *toloc.city* in Raymond's alignment),

[1] The first author of [4] is Christian Raymond, so we name this alignment in such way.

therefore the second part of its label is *ACity*. It is the first word of the *ACity* slot, so its status is *start*[2] This alignment labels "Dallas" and "Thursday" exaclty like Raymond's alignment, but main difference is that instead of labeling other words as *null*, it labels them regarding to the nearest labeled word, e.g. the nearest labeled word to "return" is "Dallas" - which is labeled as *SR.ACity.start* - so "return" is labeled as *SR.ACity.pre* which means the next labeled word has *ST.ACity.start* labels [5].

In this work, we propose another full annotation method, using the abstract annotation of [3] to train linear-chain CRF. This annotation is:

$$\underbrace{\text{I}}_{DUMMY}\ \underbrace{\text{want}}_{DUMMY}\ \underbrace{\text{to}}_{DUMMY}\ \underbrace{\text{return}}_{RETURN}\ \underbrace{\text{to}}_{TOLOC}\ \underbrace{\text{Dallas}}_{CITY}\ \underbrace{\text{on}}_{ON}\ \underbrace{\text{Thursday}}_{DATE}$$

Many useful information is saved by using this annotation, e.g. the word "return" is labeled as *RETURN* instead of *null* and this label is more informative. We have trained IBM alignment models on the parallel corpus taking the natural sentences as the source language and the labels as the target language.

### 4.2 Sequential Labeling

In order to bring the advantages of discriminative modeling and sequence modeling together, the problem should be modeled by linear-chain CRF [6].

Using this annotation, the probabilistic model of labels $p(\mathbf{y})$ which is modeled in linear-chain CRF plays its real role. $p(\mathbf{y})$ is modeled by the feature function $f(x_t, y_t, y_{t-1})$ i.e. the label transition from $y_{t-1}$ to $y_t$ [6]. In [4] label transitions are going from *null* state to other states, or going to *null* state from other states and or staying at *null* state. In [5] some words which could have better labels, such as *return* have labels depending on the nearest slot in the future or past.

Investigating the nature of SLU and the hierarchical semantic labels of [3], it is obvious that $p(\mathbf{y})$ is important, i.e. knowing that a word at time $t-1$ has been labeled as `TOLOC`, the word at time $t$ is highly probable to be labeled as `CITY` and so on. It is also possible to reduce the number of features because somehow we are keeping track of the history, so the number of words used as features in previous works [4,5] could be reduced to zero and the features are only the current word and future words. In other works such as [4] because many informative labels have been thrown out, actually existence of $p(\mathbf{y})$ will make no improvement and in order to overcome this information loss, they had to expand the features to see the words at even time $t-4$ but we will show that using this annotation $p(\mathbf{y})$ improves the model and the reduced number of features are enough for this task.

[2] City names may contain more than one words and so the other labels, in this case the other words' status will be *cont*, e.g. for the city name "Los Angeles", "Los" is labeled as *SR.ACity.start* and "Angeles" is labeled as *SR.ACity.cont*.

## 5 Experiments

In this section we evaluate our work on ATIS [1] which consists of 4978 training utterances selected from Class A in ATIS-2 and ATIS-3 corpora. The test set contains both ATIS-3 NOV93 and DEC94 test sets. In order to evaluate our work, results for HVS [3] and results from [4] are included.

### 5.1 Experimental Setup

We have used GIZA++ [14] for aligning the parallel corpus and used CRF++ [15] to train CRF. The resulting alignment is then corrected using the output alignment of [4]. This assures that no alignment errors will be injected to the CRF model. The features are indicators for lexical classes of words in a window around the decision state. We first trained the model using $[-4, 2]$ window using first order Markov chain dependency graph (linear-chain CRF) in order to have comparable results. We also trained the CRF using the zero order Markov chain to show the impact of the label model on the performance.

### 5.2 Results

We have post processed the sequence of labels to obtain the slots and their values. The slot-value pair is compared to the reference test set and the result is reported in precision (P) and recall (R) and F-measure (F) of slot classification.

Table 1 compares our method with the results of HVS baseline model and the baseline-CRF. The feature window for both CRF models is $[-4, 2]$. Our method outperforms both models. It shows the impact of our alignment method on the performance of CRF by utilizing state transitions or in other words $p(\mathbf{y})$. It shows that a more precise and informative labeling method could improve the performance. It's because an output decision at time $t$ is not only dependent on the lexical features but also on the previous decisions, i.e. if the model decides `FLIGHT` at time $t-1$, it will be very probable for `TOLOC` to be chosen at time $t$.

**Table 1.** Results of HVS, CRF and CRF + SMT on ATIS corpus

| Model | Recall | Precision | F-measure |
|---|---|---|---|
| HVS | 89.82 % | 88.75 % | 89.28 % |
| CRF | 89.25 % | 89.41 % | 89.32 % |
| CRF + SMT | 91.21 % | 90.83 % | 91.02 % |

In order to show the impact of $p(\mathbf{y})$ on the performance, we have trained zero order CRF using Raymond's alignment and our method. The results can be found in Table 2. Both CRF models use same feature window $[-4, 2]$. It is obvious when we used one order Markov chain in the CRF model, the results on

**Table 2.** Results of using first-order and zero-order dependecy in CRF and CRF + SMT on ATIS corpus

| Model | Recall | | Precision | | F-measure | |
|---|---|---|---|---|---|---|
| | 0 order | 1 order | 0 order | 1 order | 0 order | 1 order |
| CRF | 90.00 % | 89.25 % | 89.84 % | 89.41 % | 89.92 % | 89.32 % |
| CRF + SMT | 89.46 % | 91.21 % | 88.67 % | 90.83 % | 89.07 % | 91.02 % |

our annotation have significant improvement. On the other hand, surprisingly the performance of CRF which is trained using Raymond's alignment is degraded when involving state transitions. That's because it has thrown many informative data away, therefore the decisions depend only on the lexical features.

## 6 Conclusion

In this paper we proposed a new method to convert the abstract annotation automatically and without any human effort to full annotation, avoiding the loss of informative labels, reshaping the annotation from the hierarchical to a flat-concept annotation. We showed that using statistical machine translation alignments, improve the model's performance comparing to similar works. The results show that using this annotation improves the F-measure by 1.74 % comparing to HVS and using the same features comparing to CRF improves it by 1.7 %.

## References

1. Dahl, D.A., Bates, M., Brown, M., Fisher, W., Hunicke-Smith, K., Pallett, D., Pao, C., Rudnicky, A., Shriberg, E.: Expanding the scope of the atis task: the atis-3 corpus. In: Proceedings of the workshop on Human Language Technology, Association for Computational Linguistics, pp. 43–48 (1994)
2. Pieraccini, R., Tzoukermann, E., Gorelov, Z., Gauvain, J.L., Levin, E., Lee, C.H., Wilpon, J.G.: A speech understanding system based on statistical representation of semantics. In: IEEE International Conference on Acoustics, Speech, and Signal Processing 1992, ICASSP-92, vol. 1, pp. 193–196. IEEE (1992)
3. He, Y., Young, S.: Semantic processing using the hidden vector state model. Comput. Speech Lang. **19**(1), 85–106 (2005)
4. Raymond, C., Riccardi, G.: Generative and discriminative algorithms for spoken language understanding. In: International Conference on Speech Communication and Technologies, Antwerp, Belgium, August 2007, pp. 1605–1608 (2007)
5. Wang, Y.Y., Acero, A.: Discriminative models for spoken language understanding. In: International Conference on Speech Communication and Technologies, Citeseer (2006)
6. Lafferty, J., McCallum, A., Pereira, F.C.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proceedings of 18th International Conference on Machine Learning, Morgan Kaufmann, pp. 282–289 (2001)

7. Pietra, S.D., Epstein, M., Roukos, S., Ward, T.: Fertility models for statistical natural language understanding. In: Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics, Association for Computational Linguistics, pp. 168–173 (1997)
8. Macherey, K., Och, F.J., Ney, H.: Natural language understanding using statistical machine translation. In: INTERSPEECH, Citeseer, pp. 2205–2208 (2001)
9. Macherey, K., Bender, O., Ney, H.: Applications of statistical machine translation approaches to spoken language understanding. IEEE Trans. Audio Speech Lang. Process. **17**(4), 803–818 (2009)
10. Brown, P.F., Pietra, V.J.D., Pietra, S.A.D., Mercer, R.L.: The mathematics of statistical machine translation: parameter estimation. Comput. Linguist. **19**(2), 263–311 (1993)
11. Khadivi, S., Ney, H.: Automatic filtering of bilingual corpora for statistical machine translation. In: Montoyo, A., Muńoz, R., Métais, E. (eds.) NLDB 2005. LNCS, vol. 3513, pp. 263–274. Springer, Heidelberg (2005)
12. Vogel, S., Ney, H., Tillmann, C.: Hmm-based word alignment in statistical translation. In: Proceedings of the 16th conference on Computational Linguistics- Volume 2, Association for Computational Linguistics, pp. 836–841 (1996)
13. Khadivi, S., Zolnay, A., Ney, H.: Automatic text dictation in computer-assisted translation. In: International Conference on Speech Communication and Technologies, pp. 2265–2268 (2005)
14. Och, F.J., Ney, H.: Giza++: Training of statistical translation models (2000)
15. Kudo, T.: Crf++: Yet another crf toolkit. Software available at http://crfpp.sourceforge.net (2005)