Exploiting Non-local Features for Spoken Language Understanding

Minwoo Jeong and **Gary Geunbae Lee**
Department of Computer Science & Engineering
Pohang University of Science and Technology,
San 31 Hyoja-dong, Nam-gu
Pohang 790-784, Korea
{stardust,gblee}@postech.ac.kr

Abstract

In this paper, we exploit non-local features as an estimate of long-distance dependencies to improve performance on the statistical spoken language understanding (SLU) problem. The statistical natural language parsers trained on text perform unreliably to encode non-local information on spoken language. An alternative method we propose is to use trigger pairs that are automatically extracted by a feature induction algorithm. We describe a light version of the inducer in which a simple modification is efficient and successful. We evaluate our method on an SLU task and show an error reduction of up to 27% over the base local model.

1 Introduction

For most sequential labeling problems in natural language processing (NLP), a decision is made based on local information. However, processing that relies on the Markovian assumption cannot represent higher-order dependencies. This long-distance dependency problem has been considered at length in computational linguistics. It is the key limitation in bettering sequential models in various natural language tasks. Thus, we need new methods to import *non-local* information into sequential models.

There are two types of method for using non-local information. One is to add edges to structure to allow higher-order dependencies and another is to add features (or observable variables) to encode the non-locality. An additional consistent edge of a linear-chain conditional random field (CRF) explicitly models the dependencies between distant occurrences of similar words (Sutton and McCallum, 2004; Finkel et al., 2005). However, this approach requires additional time complexity in inference/learning time and it is only suitable for representing constraints by enforcing label consistency. We wish to identify ambiguous labels with more general dependency without additional time cost in inference/learning time.

Another approach to modeling non-locality is to use observational features which can capture non-local information. Traditionally, many systems prefer to use a syntactic parser. In a language understanding task, the head word dependencies or parse tree path are successfully applied to learn and predict semantic roles, especially those with ambiguous labels (Gildea and Jurafsky, 2002). Although the power of syntactic structure is impressive, using the parser-based feature fails to encode correct global information because of the low accuracy of a modern parser. Furthermore the inaccurate result of parsing is more serious in a spoken language understanding (SLU) task. In contrast to written language, spoken language loses much information including grammar, structure or morphology and contains some errors in automatically recognized speech.

To solve the above problems, we present one method to exploit non-local information – the trigger feature. In this paper, we incorporate trigger pairs into a sequential model, a linear-chain CRF. Then we describe an efficient algorithm to extract the trigger feature from the training data itself. The framework for inducing trigger features is based on the Kullback-Leibler divergence criterion which measures the improvement of log-likelihood on the current parameters by adding a new feature (Pietra et al., 1997). To reduce the cost of feature selection, we suggest a modified version of an inducing algorithm which is quite efficient. We evaluate our method on an SLU task, and demonstrate the improvements on both transcripts and recognition outputs. On a real-world problem, our modified version of a feature selection algorithm is very efficient for both performance and time complexity.

2 Spoken Language Understanding as a Sequential Labeling Problem

2.1 Spoken Language Understanding

The goal of SLU is to extract semantic meanings from recognized utterances and to fill the correct values into a semantic frame structure. A semantic frame (or template) is a well-formed and machine readable structure of extracted information consisting of slot/value pairs. An example of such a reference frame is as follows.

```
<s> i wanna go from denver to new york on
november eighteenth </s>
FROMLOC.CITY_NAME = denver
TOLOC.CITY_NAME = new york
MONTH_NAME = november
DAY_NUMBER = eighteenth
```

This example from air travel data (CU-Communicator corpus) was automatically generated by a Phoenix parser and manually corrected (Pellom et al., 2000; He and Young, 2005). In this example, the slot labels are two-level hierarchical; such as FROMLOC.CITY_NAME. This hierarchy differentiates the semantic frame extraction problem from the named entity recognition (NER) problem.

Regardless of the fact that there are some differences between SLU and NER, we can still apply well-known techniques used in NER to an SLU problem. Following (Ramshaw and Marcus, 1995), the slot labels are drawn from a set of classes constructed by extending each label by three additional symbols, Beginning/Inside/Outside (B/I/O). A two-level hierarchical slot can be considered as an integrated flattened slot. For example, FROMLOC.CITY_NAME and TOLOC.CITY_NAME are different on this slot definition scheme.

Now, we can formalize the SLU problem as a sequential labeling problem, $\mathbf{y}^* = \arg\max_{\mathbf{y}} P(\mathbf{y}|\mathbf{x})$. In this case, input word sequences $\mathbf{x}$ are not only lexical strings, but also multiple linguistic features. To extract semantic frames from utterance inputs, we use a linear-chain CRF model; a model that assigns a joint probability distribution over labels which is conditional on the input sequences, where the distribution respects the independent relations encoded in a graph (Lafferty et al., 2001).

A linear-chain CRF is defined as follows. Let $\mathcal{G}$ be an undirected model over sets of random variables $\mathbf{x}$ and $\mathbf{y}$. The graph $\mathcal{G}$ with parameters $\Lambda = \{\lambda, \ldots\}$ defines a conditional probability for a state (or label) sequence $\mathbf{y} = y_1, \ldots, y_T$, given an input $\mathbf{x} = x_1, \ldots, x_T$, to be

$$P_\Lambda(\mathbf{y}|\mathbf{x}) = \frac{1}{Z_\mathbf{x}} \exp\left(\sum_{t=1}^{T} \sum_k \lambda_k f_k(y_{t-1}, y_t, \mathbf{x}, t)\right)$$

where $Z_\mathbf{x}$ is the normalization factor that makes the probability of all state sequences sum to one. $f_k(y_{t-1}, y_t, \mathbf{x}, t)$ is an arbitrary linguistic feature function which is often binary-valued in NLP tasks. λ_k is a trained parameter associated with feature f_k. The feature functions can encode any aspect of a state transition, $y_{t-1} \rightarrow y_t$, and the observation (a set of observable features), $\mathbf{x}$, centered at the current time, t. Large positive values for λ_k indicate a preference for such an event, while large negative values make the event unlikely.

Parameter estimation of a linear-chain CRF is typically performed by conditional maximum log-likelihood. To avoid overfitting, the 2-norm regularization is applied to penalize on weight vector whose norm is too large. We used a limited memory version of the quasi-Newton method (L-BFGS) to optimize this objective function. The L-BFGS method converges super-linearly to the solution, so it can be an efficient optimization technique on large-scale NLP problems (Sha and Pereira, 2003).

A linear-chain CRF has been previously applied to obtain promising results in various natural language tasks, but the linear-chain structure is deficient in modeling long-distance dependencies because of its limited structure (n-th order Markov chains).

2.2 Long-distance Dependency in Spoken Language Understanding

In most sequential supervised learning problems including SLU, the feature function $f_k(y_{t-1}, y_t, \mathbf{x}_t, t)$ indicates only local information for practical reasons. With sufficient local context (e.g. a sliding window of width 5), inference and learning are both efficient.

However, if we only use local features, then we cannot model long-distance dependencies. Thus, we should incorporate non-local information into the model. For example, figure 1 shows the long-distance dependency problem in an SLU task. The same two word tokens "dec." should be classified differently, `DEPART.MONTH` and `RETURN.MONTH`. The dotted line boxes represent local information at the current decision point ("dec."), but they are exactly the same in two distinct examples. Moreover, the two states share the same previous sequence `(O, O, FROMLOC.CITY_NAME-B, O, TOLOC.CITY_NAME-B, O)`. If we cannot obtain higher-order dependencies such as "fly" and "return," then the linear-chain CRF cannot classify the correct labels between the two same tokens. To solve this problem, we propose an approach to exploit non-local information in the next section.

3 Incorporating Non-local Information

3.1 Using Trigger Features

To exploit non-local information to sequential labeling for a statistical SLU, we can use two approaches; a syntactic parser-based and a data-driven approach. Traditionally, information extraction and language understanding fields have usually used a syntactic parser to encode global information (e.g. parse tree path, governing category, or head word) over a local model. In a semantic role labeling task, the syntax and semantics are correlated with each other (Gildea and Jurafsky, 2002), that is, the global structure of the sentence is useful for identifying ambiguous semantic roles. However the problem is the poor accuracy of the syntactic parser with this type of feature. In addition, recognized utterances are erroneous and the spoken language has no capital letters, no additional symbols, and sometimes no grammar, so it is difficult to use a parser in an SLU problem.

Another solution is a data-driven method, which uses statistics to find features that are approximately modeling long-distance dependencies. The simplest way is to use identical words in history or lexical co-occurrence, but we wish to use a more general tool; triggering. The trigger word pairs are introduced by (Rosenfeld, 1994). A trigger pair is the basic element for extracting information from the long-distance document history. In language modeling, n-gram based on the Markovian assumption cannot represent higher-order dependencies, but it can automatically extract trigger word pairs from data. The pair ($A \rightarrow B$) means that word A and B are significantly correlated, that is, when A occurs in the document, it triggers B, causing its probability estimate to change.

To select reasonable pairs from arbitrary word pairs, (Rosenfeld, 1994) used averaged mutual information (MI). In this scheme, the MI score of one pair is $MI(A;B) =$

$$P(A,B)\log\frac{P(B|A)}{P(B)} + P(A,\bar{B})\log\frac{P(\bar{B}|A)}{P(\bar{B})} + P(\bar{A},B)\log\frac{P(B|\bar{A})}{P(\bar{B})} + P(\bar{A},\bar{B})\log\frac{P(\bar{B}|\bar{A})}{P(\bar{B})}.$$

Using the MI criterion, we can select correlated word pairs. For example, the trigger pair (dec.→return) was extracted with score 0.001179 in the training data[1]. This trigger word pair can represent long-distance dependency and provide a cue to identify ambiguous classes. The MI approach, however, considers only lexical collocation without reference labels $\mathbf{y}$, and MI based selection tends to excessively select the irrelevant triggers. Recall that our goal is to find the significantly correlated trigger pairs which improve the model. Therefore, we use a more appropriate selection method for sequential supervised learning.

3.2 Selecting Trigger Feature

We present another approach to extract relevant triggers and exploit them in a linear-chain CRF. Our approach is based on an automatic feature induction algorithm, which is a novel method to select a feature in an exponential model (Pietra et al., 1997; McCallum, 2003). We follow McCallum's work which is an efficient method to induce features in a linear-chain CRF model. Following the framework of feature inducing, we start the algorithm with an empty set, and iteratively increase the bundle of features including local features and trigger features. Our basic assumption, however, is that the local information should be included because the local features are the basis of the decision to identify the classes, and they reduce the

[1]In our experiment, the pair (dec.→fly) cannot be selected because this MI score is too low. However, the trigger pair is a binary type feature, so the pair (dec.→return) is enough to classify the two cases in the previous example.

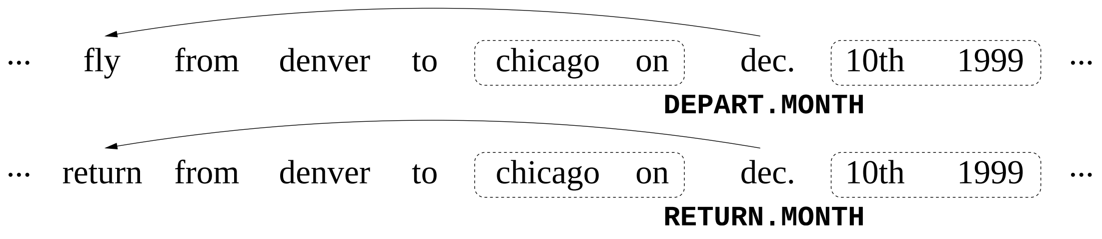
··· fly from denver to chicago on dec. 10th 1999 ···
DEPART.MONTH
··· return from denver to chicago on dec. 10th 1999 ···
RETURN.MONTH

Figure 1: An example of a long-distance dependency problem in spoken language understanding. In this case, a word token "dec." with local feature set (dotted line box) is ambiguous for determining the correct label (DEPART.MONTH or RETURN.MONTH).

mismatch between training and testing tasks. Furthermore, this assumption leads us to faster training in the inducing procedure because we can only consider additional trigger features.

Now, we start the inducing process with local features rather than an empty set. After training the base model $\Lambda^{(0)}$, we should calculate the gains, which measure the effect of adding a trigger feature, based on the local model parameter $\Lambda^{(0)}$. The gain of the trigger feature is defined as the improvement in log-likelihood of the current model $\Lambda^{(i)}$ at the i-th iteration according to the following formula:

$$\begin{aligned} \hat{G}_{\Lambda^{(i)}}(g) &= \max_{\mu} G_{\Lambda^{(i)}}(g,\mu) \\ &= \max_{\mu} \left\{ L_{\Lambda^{(i)}+g,\mu} - L_{\Lambda^{(i)}} \right\} \end{aligned}$$

where μ is a parameter of a trigger feature to be found and g is a corresponding trigger feature function. The optimal value of μ can be calculated by Newton's method.

By adding a new candidate trigger, the equation of the linear-chain CRF model is changed to an additional feature model as $P_{\Lambda^{(i)}+g,\mu}(\mathbf{y}|\mathbf{x}) =$

$$\frac{P_{\Lambda^{(i)}}(\mathbf{y}|\mathbf{x}) \exp\left(\sum_{t=1}^{T} \mu g(y_{t-1}, y_t, \mathbf{x}, t)\right)}{Z_{\mathbf{x}}(\Lambda^{(i)}, g, \mu)}.$$

Note that $Z_{\mathbf{x}}(\Lambda^{(i)}, g, \mu)$ is the marginal sum over all states of $\mathbf{y}'$. Following (Pietra et al., 1997; McCallum, 2003), the mean field approximation and agglomerated features allows us to treat the above calculation as the independent inference problem rather than sequential inference. We can evaluate the probability of state $\mathbf{y}$ with an adding trigger pair given observation $\mathbf{x}$ separately as follows.

$$P_{\Lambda^{(i)}+g,\mu}(y|\mathbf{x},t) = \frac{P_{\Lambda^{(i)}}(y|\mathbf{x},t) \exp\left(\mu g(y_t, \mathbf{x}, t)\right)}{Z_{\mathbf{x}}(\Lambda^{(i)}, g, \mu)}$$

Here, we introduce a second approximation. We use the individual inference problem over the unstructured maximum entropy (ME) model whose state variable is independent from other states in history. The background of our approximation is that the state independent problem of CRF can be relaxed to ME inference problem without the state-structured model. In the result, we calculate the gain of candidate triggers, and select trigger features over a light ME model instead of a huge computational CRF model[2].

We can efficiently assess many candidate trigger features in parallel by assuming that the old features remain fixed while estimating the gain. The gain of trigger features can be calculated on the old model that is trained with the local and added trigger pairs in previous iterations. Rather than summing over all training instances, we only need to use the mislabeled N tokens by the current parameter $\Lambda^{(i)}$ (McCallum, 2003). From misclassified instances, we generate the candidates of trigger pairs, that is, all pairs of current words and others within the sentence. With the candidate feature set, the gain is

$$\begin{aligned} \hat{G}_{\Lambda^{(i)}}(g) &= N\hat{\mu}\tilde{E}[g] \\ &- \sum_{j=1}^{N} \log\left(E_{\Lambda^{(i)}}[\exp(\hat{\mu}g)|\mathbf{x}_j]\right) - \frac{\hat{\mu}^2}{2\sigma^2}. \end{aligned}$$

Using the estimated gains, we can select a small portion of all candidates, and retrain the model with selected features. We iteratively perform the selection algorithm with some stop conditions (excess of maximum iteration or no added feature up to the gain threshold). The outline of the induction

[2]The ME model cannot represent the sequential structure and the resulting model is different from CRF. Nevertheless, we empirically prove that the effect of additional trigger features on both ME and approximated CRF (without regarding edge-state) are similar (see the experiment section).

```
Algorithm InduceLearn(x,y)
$triggers \leftarrow \{\varepsilon\}$ and $i \leftarrow 0$
while $|pairs| > 0$ and $i < maxiter$ do
    $\Lambda^{(i)} \leftarrow \mathrm{TrainME}(\mathbf{x}, \mathbf{y})$
    $P(\mathbf{y}_e|\mathbf{x}_e) \leftarrow \mathrm{Evaluate}(\mathbf{x}, \mathbf{y}, \Lambda^{(i)})$
    $\mathbf{c} \leftarrow \mathrm{MakeCandidate}(\mathbf{x}_e)$
    $G_{\Lambda^{(i)}} \leftarrow \mathrm{EstimateGain}(\mathbf{c}, P(\mathbf{y}_e|\mathbf{x}_e))$
    $pairs \leftarrow \mathrm{SelectTrigger}(\mathbf{c}, G_{\Lambda^{(i)}})$
    $\mathbf{x} \leftarrow \mathrm{UpdateObs}(\mathbf{x}, pairs)$
    $triggers \leftarrow triggers \cup pairs$ and $i \leftarrow i + 1$
end while
$\Lambda^{(i+1)} \leftarrow \mathrm{TrainCRF}(\mathbf{x}, \mathbf{y})$
return $\Lambda^{(i+1)}$
```

Figure 2: Outline of trigger feature induction algorithm

algorithms is described in figure 2. In the next section, we empirically prove the effectiveness of our algorithm.

The trigger pairs introduced by (Rosenfeld, 1994) are just word pairs. Here, we can generalize the trigger pairs to any arbitrary pairs of features. For example, the feature pair (of→B-PP) is useful in deciding the correct answer `PERIOD_OF_DAY-I` in "in the middle of the day." Without constraints on generating the pairs (e.g. at most 3 distant tokens), the candidates can be arbitrary conjunctions of features[3]. Therefore we can explore any features including local conjunction or non-local singleton features in a uniform framework.

4 Experiments

4.1 Experimental Setup

We evaluate our method on the CU-Communicator corpus. It consists of 13,983 utterances. The semantic categories correspond to city names, time-related information, airlines and other miscellaneous entities. The semantic labels are automatically generated by a Phoenix parser and manually corrected. In the data set, the semantic category has a two-level hierarchy: 31 first level classes and 7 second level classes, for a total of 62 class combinations. The data set is 630k words with 29k entities. Roughly half of the entities are time-related information, a quarter of the entities are city names, a tenth are state and country names, and a fifth are airline and airport names. For the second level hierarchy, approximately three quarters of the entities are "`NONE`", a tenth are "`TOLOC`", a tenth are "`FROMLOC`", and the remaining are "`RETURN`", "`DEPERT`", "`ARRIVE`", and "`STOPLOC`."

For spoken inputs, we used the open source speech recognizer Sphinx2. We trained the recognizer with only the domain-specific speech corpus. The reported accuracy for Sphinx2 speech recognition is about 85%, but the accuracy of our speech recognizer is 76.27%; we used only a subset of the data without tuning and the sentences of this subset are longer and more complex than those of the removed ones, most of which are single-word responses.

All of our results have averaged over 5-fold cross validation with an 80/20 split of the data. As it is standard, we compute precision and recall, which are evaluated on a per-entity basis and combined into a micro-averaged F1 score (F1 = 2PR/(P+R)).

A final model (a first-order linear chain CRF) is trained for 100 iterations with a Gaussian prior variance of 20, and 200 or fewer trigger features (down to a gain threshold of 1.0) for each round of inducing iteration (100 iterations of L-BFGS for the ME inducer and 10∼20 iterations of L-BFGS for the CRF inducer). All experiments are implemented in C++ and executed on Linux with XEON 2.8 GHz dual processors and 2.0 Gbyte of main memory.

4.2 Empirical Results

We list the feature templates used by our experiment in figure 3. For local features, we use the indicators for specific words at location i, or locations within five words of i ($-2, -1, 0, +1, +2$ words on current position i). We also use the part-of-speech (POS) tags and phrase labels with partial parsing. Like words, the two basic linguistic features are located within five tokens. For comparison, we exploit the two groups of non-local syntax parser-based features; we use Collins parser and extract this type of features from the parse trees. The first consists of the head word and POS-tag of the head word. The second group includes governing category and parse tree paths introduced by semantic role labeling (Gildea and Jurafsky, 2002). Following the previous studies

[3]In our experiment, we do not consider the local conjunctions because we wish to capture the effect of long-distance entities.

Local feature templates
-lexical words -part-of-speech (POS) tags -phrase chunk labels
Grammar-based feature templates
-head word / POS-tag -parse tree path and governing category
Trigger feature templates
-word pairs ($w_i \rightarrow w_j$), $\|i - j\| > 2$ -feature pairs between words, POS-tags, and chunk labels ($f_i \rightarrow f_j$), $\|i - j\| > 2$ -null pairs ($\varepsilon \rightarrow w_j$)

Figure 3: Feature templates

of semantic role labeling, the parse tree path improves the classification performance of semantic role labeling. Finally, we use the trigger pairs that are automatically extracted from the training data. Avoiding the overlap of local features, we add the constraint $|i - j| > 2$ for the target word w_j. Note that null pairs are equivalent to long-distance singleton word features w_j.

To compute feature performance, we begin with word features and iteratively add them one-by-one so that we achieve the best performance. Table 1 shows the empirical results of local features, syntactic parser-based features, and trigger features respectively. The two F1 scores for text transcripts (Text) and outputs recognized by an automatic speech recognizer (ASR) are listed. We achieved F1 scores of 94.79 and 71.79 for Text and ASR inputs using only word features. The performance is decreased by adding the additional local features (POS-tags and chunk labels) because the pre-processor brings more errors to the system for spoken dialog.

The parser-based and trigger features are added to two baselines: word only and all local features. The result shows that the trigger feature is more robust to an SLU task than the features generated from the syntactic parser. The parse tree path and governing category show a small improvement of performance over local features, but it is rather insignificant (word vs. word+path, McNemar's test (Gillick and Cox, 1989); $p = 0.022$). In contrast, the trigger features significantly improve the performance of the system for both Text and ASR inputs. The differences between the trigger and the others are statistically significant (McNemar's test; $p < 0.001$ for both Text and ASR).

Table 1: The result of local features, parser-based features and trigger features

Feature set	F1 (Text)	F1 (ASR)
word (w)	94.79	71.79
w + POStag (p)	94.57	71.61
w + chunk (c)	94.70	71.64
local (w+p+c)	94.41	71.60
w + head (h)	94.55	71.76
w + path (t)	95.07	72.17
w + h + t	94.84	72.09
local + head (h)	94.17	71.39
local + path (t)	94.80	71.89
local + h + t	94.51	71.67
w + trigger	**96.18**	**72.95**
local + trigger	96.04	72.72

Next, we compared the two trigger selection methods; mutual information (MI) and feature induction (FI). Table 2 shows the experimental results of the comparison between MI and FI approaches (with the local feature set; w+p+c). For the MI-based approach, we should calculate an averaged MI for each word pair appearing in a sentence and cut the unreliable pairs (down to threshold of 0.0001) before training the model. In contrast, the FI-based approach selects reliable triggers which should improve the model in training time. Our method based on the feature induction algorithm outperforms simple MI-based methods. Fewer features are selected by FI, that is, our method prunes the event pairs which are highly correlated, but not relevant to models. The extended feature trigger ($f_i \rightarrow f_j$) and null triggers ($\varepsilon \rightarrow w_j$) improve the performance over word trigger pairs ($w_i \rightarrow w_j$), but they are not statistically significant (vs. ($f_i \rightarrow f_j$); $p = 0.749$, vs. ($\{\varepsilon, w_i\} \rightarrow w_j$); $p = 0.294$). Nevertheless, the null pairs are effective in reducing the size of trigger features.

Figure 4 shows a sample of triggers selected by MI and FI approaches. For example, the trigger "morning $\rightarrow$ return" is ranked in first of FI but 66th of MI. Moreover, the top 5 pairs of MI are not meaningful, that is, MI selects many functional word pairs. The MI approach considers only lexical collocation without reference labels, so the FI method is more appropriate to sequential supervised learning.

Finally, we wish to justify that our modified

Table 2: Result of the trigger selection methods

Method	Avg. # triggers	F1 (Text)	F1 (ASR)	McNemar's test (vs. MI)
MI ($w_i \rightarrow w_j$)	1,713	95.20	72.12	-
FI ($w_i \rightarrow w_j$)	702	96.04	72.72	$p < 0.001$
FI ($f_i \rightarrow f_j$)	805	96.04	72.76	$p < 0.001$
FI ($\{\varepsilon, w_i\} \rightarrow w_j$)	**545**	**96.14**	**72.80**	$p < 0.001$

Mutual Information	Feature Induction
[1] from→like	[1] morning→return
[2] on→to	[2] morning→on
[3] to→i	[3] morning→to
[4] on→from	[4] afternoon→on
[5] from→i	[5] afternoon→return
[41] afternoon→return	[6] afternoon→to
[66] morning→return	[15] morning→leaving
[89] morning→leaving	[349] december→return
[1738] london→fly	[608] illinois→airport

Figure 4: A sample of triggers extracted by two methods

version of an inducing algorithm is efficient and maintains performance without any drawbacks. We proposed two approximations: starting with local features (Approx. 1) and using an unstructured model on the selection stage (Approx. 2), Table 3 shows the results of variant versions of the algorithm. Surprisingly, the selection criterion based on ME (the unstructured model) is better than CRF (the structured model) not only for time cost but also for the performance on our experiment[4]. This result shows that local information provides the fundamental decision clues. Our modification of the algorithm to induce features for CRF is sufficiently fast for practical usage.

5 Related Work and Discussion

The most relevant previous work is (He and Young, 2005) who describes an generative approach – hidden vector state (HVS) model. They used 1,178 test utterances with 18 classes for 1st level label, and published the resulting F1 score of 88.07. Using the same test data and classes, we achieved the 92.77 F1-performance, as well as 39% of error reduction compared to the previous result. Our system uses a discriminative approach, which directly models the conditional distribution, and it is sufficient for classification task. To capture long-distance dependency, HVS uses a context-free model, which increases the complexity of models. In contrast, we use non-local trigger features, which are relatively easy to use without having additional complexity of models.

Trigger word pairs are introduced and successfully applied in a language modeling task. (Rosenfeld, 1994) demonstrated that the trigger word pairs improve the perplexity in ME-based language models. Our method extends this idea to sequential supervised learning problems. Our trigger selection criterion is based on the automatic feature inducing algorithm, and it allows us to generalize the arbitrary pairs of features.

Our method is based on two works of feature induction on an exponential model, (Pietra et al., 1997) and (McCallum, 2003). Our induction algorithm builds on McCallum's method which presents an efficient procedure to induce features on CRF. (McCallum, 2003) suggested using only the mislabeled events rather than the whole training events. This intuitional suggestion has offered us fast training. We added two additional approximations to reduce the time cost; 1) an inducing procedure over a conditional non-structured inference problem rather than an approximated sequential inference problem, and 2) training with a local feature set, which is the basic information to identify the labels.

In this paper, our approach describes how to exploit non-local information to a SLU problem. The trigger features are more robust than grammar-based features, and are easily extracted from the data itself by using an efficient selection algorithm.

[4]In our analysis, 10$\sim$20 iterations for each round of inducing procedure are insufficient in optimizing the model in CRF (empty) inducer. Thus, the resulting parameters are under-fitted and selected features are infeasible. We need more iteration to fit the parameters, but they require too much learning time ($>$ 1 day).

Table 3: Comparison of variations in the induction algorithm (performed on one of the 5-fold validation sets); columns are induction and total training time (h:m:s), number of trigger and total features, and f-score on test data.

Inducer type	Approx.	Induction/total time	# triggers/features	F1 (Text)	F1 (ASR)
CRF (empty)	No approx.	3:55:01 / 5:27:13	682 / 2,693	90.23	67.60
CRF (local)	Approx. 1	1:25:28 / 2:56:49	750 / 5,241	94.87	71.65
ME (empty)	Approx. 2	20:57 / 1:54:22	618 / 2,080	94.85	71.46
ME (local)	Approx. 1+2	**6:30 / 1:36:14**	**608 / 5,099**	**95.17**	**71.81**

6 Conclusion

We have presented a method to exploit non-local information into a sequential supervised learning task. In a real-world problem such as statistical SLU, our model performs significantly better than the traditional models which are based on syntactic parser-based features. In comparing our selection criterion, we find that the mutual information tends to excessively select the triggers while our feature induction algorithm alleviates this issue. Furthermore, the modified version of the algorithm is practically fast enough to maintain its performance particularly when the local features are offered by the starting position of the algorithm.

In this paper, we have focused on a sequential model such as a linear-chain CRF. However, our method can also be naturally applied to arbitrary structured models, thus the first alternative is to combine our methods with a skip-chain CRF (Sutton and McCallum, 2004). Applying and extending our approach to other natural language tasks (which are difficult to apply a parser to) such as information extraction from e-mail data or biomedical named entity recognition is a topic of future work.

Acknowledgements

We thank three anonymous reviewers for helpful comments. This research was supported by the MIC (Ministry of Information and Communication), Korea, under the ITRC (Information Technology Research Center) support program supervised by the IITA (Institute of Information Technology Assessment). (IITA-2005-C1090-0501-0018)

References

J. R. Finkel, T. Grenager, and C. Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of ACL'05*, pages 363–370.

D. Gildea and D. Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.

L. Gillick and S. Cox. 1989. Some statistical issues in the comparison of speech recognition algorithms. In *Proceedings of ICASSP*, pages 532–535.

Y. He and S. Young. 2005. Semantic processing using the hidden vector state model. *Computer Speech & Language*, 19(1):85–106.

J. Lafferty, A. McCallum, and F. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML*, pages 282–289.

A. McCallum. 2003. Efficiently inducing features of conditional random fields. In *Proceedings of UAI*, page 403.

B. L. Pellom, W. Ward, and S. S. Pradhan. 2000. The cu communicator: An architecture for dialogue systems. In *Proceedings of ICSLP*.

S. Della Pietra, V. J. Della Pietra, and J. Lafferty. 1997. Inducing features of random fields. *IEEE Trans. Pattern Anal. Mach. Intell*, 19(4):380–393.

L. A. Ramshaw and M. P. Marcus. 1995. Text chunking using transformation-based learning. In *3rd Workshop on Very Large Corpora*, pages 82–94.

R. Rosenfeld. 1994. Adaptive statistical language modeling: A maximum entropy approach. Technical report, School of Computer Science Carnegie Mellon University.

F. Sha and F. Pereira. 2003. Shallow parsing with conditional random fields. In *Proceedings of HLT/NAACL'03*.

C. Sutton and A. McCallum. 2004. Collective segmentation and labeling of distant entities in information extraction. In *ICML Workshop on Statistical Relational Learning*.