STRUCTURE LEARNING FOR NATURAL LANGUAGE PROCESSING

Yizhao Ni, Craig J Saunders, Sandor Szedmak and Mahesan Niranjan

ISIS Group, School of Electronics and Computer Science, University of Southampton
Southampton, SO17 1BJ, United Kingdom
yn05r@ecs.soton.ac.uk, craig.saunders@xrce.xerox.com,
{ss03v,mn}@ecs.soton.ac.uk

ABSTRACT

We applied a structure learning model, Max-Margin Structure (MMS), to natural language processing (NLP) tasks, where the aim is to capture the latent relationships within the output language domain. We formulate this model as an extension of multi–class Support Vector Machine (SVM) and present a perceptron–based learning approach to solve the problem. Experiments are carried out on two related NLP tasks: part–of–speech (POS) tagging and machine translation (MT), illustrating the effectiveness of the model.

1. INTRODUCTION

Numerous fields in *natural language processing* (NLP) employ different machine learning methods. In this paper, we aim at applying a structure learning model for two related NLP problems: *part–of–speech* (POS) *tagging* and *machine translation* (MT).

1.1. Part–of–speech (POS) tagging

POS tagging is the process of "translating" the words in a text into a particular part of speech. It can be viewed as a simplified form of machine translation, which translates the words one by one into POS tags without word reordering.

Among recent top performing methods for automatic assignment of POS tagging, *Hidden markov models* [1] and *Maximum entropy models* [2] are the most popular. In both methods, a tag t_i given a word f_i with its context feature function h is connected via a conditional probability $p(\mathrm{t}_i|h(f_i))$ while different parametric forms are applied to model this probability. The models are then "generating" the POS tag sequence for a given sentence by maximizing the sequence probability $\prod_{i=1}^{N} p(\mathrm{t}_i|h(f_i))$. Alternatively, one can view POS tagging as a multi–class classification problem, which predicts a word f_i's tag label t_i according to a learnt function $\mathbf{w}$: $\mathrm{t}_i = \arg\max_{\hat{\mathrm{t}}} \mathbf{w}^T h(f_i, \hat{\mathrm{t}})$. In this way, general classification techniques such as SVM can be applied [3].

In POS tagging there are several problematic cases that come from confusing or ambiguous items in speech. For example, the word "good" in the phrase "good strategy" is an adjective (JJ) while in "the common good" it is a noun (NN). Current methods such as [2] try to solve the problematic cases by exploring richer feature sets, such as grammatical features and lemmas. However, this feature extension is highly dependent on the prior knowledge about the language and is expensive to collect in advance.

In contrast to the above, we aim at improving the performance by exploiting only a limited set of features, where we apply a word disambiguation technique for the problematic cases and use a *max–margin structure* (MMS) learning model for POS tagging. The framework is similar to structure SVM [3] that allows more flexible margins between classes, which however, has a lower computational complexity to enable its application to a large learning problem. In this paper we treat POS tagging as an MT task with a specific target language made of POS tags. The rest of this paper is organised as follows. We first briefly state the general framework for machine translation. Section 2 presents the MMS learning model and algorithms. Then in section 3, we describe the procedure for feature extraction and the model training. Section 4 evaluates the performance of the MMS model on POS tagging and MT tasks. Finally we draw conclusions and mention the future work in Section 5.

1.2. Machine translation

Phrase-based statistical machine translation (SMT) is a task where each source sentence f is segmented into a sequence of I phrases $\bar{\mathbf{f}}^I$ and translated into a target sequence $\bar{\mathbf{e}}^I$, often by means of a stochastic process that maximizes the posterior probability $\bar{\mathbf{e}}^I = \arg\max_{\hat{\mathbf{e}}^I \in E} \left\{ P(\hat{\mathbf{e}}^I|\bar{\mathbf{f}}^I) \right\}$. Usually the posterior probability $P(\hat{\mathbf{e}}^I|\bar{\mathbf{f}}^I)$ is modeled with a log–linear maximum entropy framework [4] which makes it easier to integrate additional models

$$P(\hat{\mathbf{e}}^I|\bar{\mathbf{f}}^I) = \frac{\exp\left(\sum_m \lambda_m h_m(\hat{\mathbf{e}}^I, \bar{\mathbf{f}}^I)\right)}{\sum_{I', \hat{\mathbf{e}}^{I'}} \exp\left(\sum_m \lambda_m h_m(\hat{\mathbf{e}}^{I'}, \bar{\mathbf{f}}^I)\right)}$$

This work was supported by the European Commission under the IST Project SMART (FP6-033917)

where h_m represent symbol models with scaling factors λ_m. As the denominator only depends on the source phrase sequence $\bar{\mathbf{f}}^I$, it is usually discarded and the solution is also represented as $\bar{\mathbf{e}}^I = \arg\max_{\hat{\mathbf{e}}^I \in E} \left\{ \exp\left(\sum_m \lambda_m h_m(\hat{\mathbf{e}}^I, \bar{\mathbf{f}}^I) \right) \right\}$.

A combination of several symbol models, including a Phrase Translation Probability (PTP) model, a language model and a phrase reordering model [5], are commonly used and the decoder then searches a Viterbi–best string path according to the joint performance of these models.

In this paper, we focus on developing the Phrase Translation Probability (PTP) model, which has changed little over the past few years. Following the assumption that the target phrase translations are conditionally independent given their source phrases, the traditional SMT model assigns the phrase translation probability for a phrase pair $(\bar{f}_j, \bar{e}_i)$ using the Maximum Likelihood Estimation (MLE)

$$p(\bar{f}_j, \bar{e}_i) = \frac{count(\bar{f}_j, \bar{e}_i)}{\sum_{\hat{e}} count(\bar{f}_j, \hat{e})}$$

This model is commonly used in current SMT systems [6], although it has two major limitations: firstly, the phrase translation probability only depends on the frequency of the phrase pairs, the sentence context in which phrases occur is completely ignored. Secondly, for low–frequency phrase pairs, the covariance of the MLE would be considerably large, often making the prediction over–fit the training data.

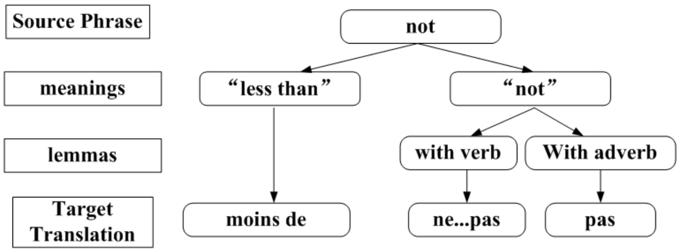
As machine learning techniques become more and more attractive in the NLP field, several discriminative methods have been applied for the PTP model. For example, a word sense disambiguation model is proposed in [7], that learns the translations of words basing on the word environment, syntactical information and lemma. By extending from words to phrases, [8] deals with the phrase translation as a classification problem and uses a set of SVMs to perform the classification. The SVM confidences are then transformed into the phrase translation probabilities. Although these approaches consider the sentence context, the potential connections between target translations are still ignored. Theoretically, the structure of the target translations can be learnt by structure learning techniques such as a structured SVM, while in practice the running time usually makes it infeasible to apply to a large data set. Our work inherits this idea of structured SVMs, but applies a perceptron-based algorithm in order to reduce the computation complexity. Our aim is to show that it is practical to apply this *max–margin structure* model to the MT field and produce reasonable results.

2. SYSTEM DESCRIPTION

Since the POS tagging is a simplified form of machine translation, in the system description our notation will follow the general MT notation as used in [6].

2.1. Phrase translation as structured prediction

Let us define the source phrase as $\bar{f}_{j_n}$ with $\bar{f}$ denoting the phrase label, j_n denoting the phrase position in the sentence and n denoting the n-th example. We use a similar notation $\bar{e}_{i_n}$ for target phrases. For each unique source phrase $\bar{f}$, we assign a cluster $\Omega_{\bar{f}}$ including all the target translations (candidates) and the number of candidates is denoted as $C_{\bar{f}}$. Figure 1 demonstrates an English–to–French translation example, in which $\bar{f} =$ "not", $\Omega_{\bar{f}} = \{$"moins de", "ne...pas", "pas"$\}$ and $n = 1, \ldots, 3$. Without loss of clarity, we also abbreviate the source and the target phrases as $\bar{f}_n$ and $\bar{e}_n$.



Example 1:
Source Sentence：Not five minutes ago.
Translation：Il y a moins de cinq minutes.
Example 2:
Source Sentence：I am not sure.
Translation：Je ne suis pas sûr.
Example 3:
Source Sentence：There is not even a pub.
Translation：Il n´y a même pas de bar.

Fig. 1. An English–to–French translation example. The latent connections between target translations are displayed in two levels: "meaning" and "lemma".

In our phrase translation system, we assign a separate model for each unique source phrase $\bar{f}$. Assume a set of training instances $\mathcal{S} = \{(\bar{f}_n, \bar{e}_n)\}_{n=1}^N$ with the same source phrase $\bar{f}$, each of which consists of a structured feature vector $\phi(\bar{f}_n, \bar{e}_n) \in \mathbb{R}^{d \cdot C_{\bar{f}}}$. Then the goal is to learn a linear evaluation function $F := \mathbf{w}^T \phi(\bar{f}_n, \bar{e}_n) \rightarrow \mathbb{R}$ that can "generate" an appropriate translation probability for $(\bar{f}_n, \bar{e}_n)$.

Instead of applying Maximum Likelihood Estimation, we adopt the max–margin formulation of [9] to find a linear operator $\mathbf{w}$ such that $\arg\max_{c \in \Omega_{\bar{f}}} \mathbf{w}^T \phi(\bar{f}_n, c) \approx \bar{e}_n, \forall n$. This is equivalent to minimizing a risk function $J(\mathbf{w})$, which corresponds to the sum of the classification errors associated with the translation candidates

$$J(\mathbf{w}) = \frac{1}{N} \sum_{n=1}^{N} \rho(\bar{f}_n, \bar{e}_n, \mathbf{w}) + \frac{\lambda}{2} \|\mathbf{w}\|^2 \qquad (1)$$

where ρ is a specific loss function and $\lambda \geq 0$ is a regularisation parameter.

As translations for the same source phrase tend to be interdependent, the phrase translation task is more than a multi–class classification problem. Consider Figure 1, if the phrase “not” in example 2 is translated into “pas” instead of “ne...pas”, intuitively the loss should be smaller than when it is translated into “moins de”. The output (target translation) domain has an inherent structure (e.g. “meaning” and “lemma”) and the loss function should respect the structure of the translation candidates. Hence, we model the phrase translation task as a structure learning problem and apply the soft–margin loss on the structured label domain

$$\rho(\bar{f}_n, \bar{e}_n, F) = \max\{0, \max_{c \neq \bar{e}_n}[\triangle(c, \bar{e}_n) + F(\bar{f}_n, c)] - F(\bar{f}_n, \bar{e}_n)\} \tag{2}$$

where the function $\triangle(c, \bar{e}_n)$ is applied to measure the “distance” between a pseudo candidate c and the correct translation $\bar{e}_n$. Theoretically, this loss requires that the pseudo candidates c which are “far away” from the true translation $\bar{e}_n$ must be classified with a large margin while nearby candidates are allowed to be classified with a smaller margin.

A variety of approaches have been suggested to evaluate the “distance” between strings, such as the “bag–of–words” method. Ideally, the measure function $\triangle(c, \bar{e}_n)$ should respect all aspects of influence on candidate connections, which is hard to achieve in practice however. To simplify the computation, we use a generation of the hamming distance – Levenshtein Distance, that measures the minimum number of modifications required to change one string into another. The algorithm can be found in [10] and then the distance function is computed as

$$\triangle(c, \bar{e}_n) = \frac{LevDist(c, \bar{e}_n)}{\max_{c' \in \Omega_{\bar{f}}} LevDist(c', \bar{e}_n)} \tag{3}$$

where the function $LevDist(c, \bar{e}_n)$ returns the levenshtein distance between c and $\bar{e}_n$. By definition, the distance value falls in $\triangle(c, \bar{e}_n) \in (0, 1]$ with $\triangle(c, \bar{e}_n) = 1$ indicating that c is the farthest from $\bar{e}_n$ among the translations.

2.2. Max-margin perceptron (MMP)

If we do not consider the regularisation term in (1) (i.e. $\lambda = 0$), we use a perceptron–based algorithm, named max-margin perceptron (MMP), to tune the parameters $\mathbf{w}$. The pseudo code of the algorithm is given in Table 1. Analogous to the standard Novikoff theorem, we provide an upper bound on the number of updates and a lower bound on the achievable margin for the MMP algorithm. Note that this algorithm is an extension of that provided by [11] where no distance between classes is considered (i.e. $\triangle(c, \bar{e}_n) = 1,\ \forall c$).

Let us define the margin for the learner as

$$\gamma(\mathbf{w}, \mathcal{S}, \phi) := \min_{(\bar{f}_n, \bar{e}_n) \in \mathcal{S}} \frac{\langle \mathbf{w}, \phi(\bar{f}_n, \bar{e}_n) - \phi(\bar{f}_n, c_n^*) \rangle}{\|\mathbf{w}\|}$$

with c_n^* to be the maximizer of the $\max_{c_n \neq \bar{e}_n}$ operation in equation (2). Then we have:

```
input: The samples $\{(\bar{f}_n, \bar{e}_n)\}_{n=1}^{N}$, step size $\eta$
initialization: $k = 0$; $\mathbf{w}_k = \mathbf{0}$;
repeat
    for $n = 1, 2, \ldots, N$ do
        for $c \neq \bar{e}_n$ get
            $V = \max_c \{\triangle(c, \bar{e}_n) + \mathbf{w}_k^T \phi(\bar{f}_n, c)\}$
            $c^* = \arg\max_c \{\triangle(c, \bar{e}_n) + \mathbf{w}_k^T \phi(\bar{f}_n, c)\}$
        if $\mathbf{w}_k^T \phi(\bar{f}_n, \bar{e}_n) < V$ then
            $\mathbf{w}_{k+1} = \mathbf{w}_k + \eta\big(\phi(\bar{f}_n, \bar{e}_n) - \phi(\bar{f}_n, c^*)\big)$, $k = k + 1$
until converge
output: $\mathbf{w}_k \in \mathbb{R}^{d \cdot C_{\bar{f}}}$
```

Table 1. Max–Margin Perceptron (MMP) algorithm.

Proposition 1. *Let $\mathcal{S} = \{(\bar{f}_n, \bar{e}_n)\}_{n=1}^{N}$ be a sample set independently and identically drawn from an unknown distribution and let $\phi(\bar{f}_n, \bar{e}_n)$ be a feature vector with $\|\phi(\bar{f}_n, \bar{e}_n)\| = 1$ for all n, and that the learning rate η is a fixed positive number in Table 1. Suppose there exists a mapping operator $\mathbf{w}^*$ such that $\|\mathbf{w}^*\| = R$ and $\gamma(\mathbf{w}^*, \mathcal{S}, \phi) \geq \Gamma$, and the algorithm stops when the functional margin V in Table 1 is achieved for every data point.*

1. *Then the number of updates made by the MMP algorithm is bounded by $t \leq \frac{2}{\Gamma^2}(1 + \frac{1}{\eta})$.*
2. *Then for the solution $\mathbf{w}_t$ of MMP algorithm we have $\gamma(\mathbf{w}_t, \mathcal{S}, \phi) \geq \frac{\Gamma\xi}{2(\eta+1)}$, with $\xi = \min_k \triangle(c_k^*, \bar{e}_k)$ indicating the minimal distance between a pseudo candidate and a correct translation across all examples.*

Table 1 indicates that the computation complexity of MMP is $O(NdC_{\bar{f}})$, while the complexity of SVM with one–verse–all strategy is somewhere between $O(N^2d + NC_{\bar{f}})$ and $O(N^2d + N^2C_{\bar{f}})$ [12]. Since in practice the number of classes $C_{\bar{f}}$ would be much smaller than the number of examples N, this makes MMP substantially faster than the multi–class SVM used in [8] and obviously the structured SVM proposed in [3]. This time efficiency is verified by the POS tagging experiment results shown in Table 5.

Notice that in MMP $\mathbf{w}_t$ is tested on the example $(\bar{f}_t, \bar{e}_t)$ which are not available for training $\mathbf{w}_t$, so if we can guarantee a low cumulative loss we are already guarding against overfitting. If one wished to add regularisation to the model to further guard against overfitting, one could apply methods such as ALMA [13] or NORMA [14]. However, the requirement of normalising $\mathbf{w}$ at each step makes the implementation intractable for a large learning problem. As an alternative, the risk function (1) can be reformulated as a min–max optimisation problem which can be solved by a benchmark-based extra–gradient algorithm. Under mild conditions, the algorithm is guaranteed to converge linearly to a solution of $\mathbf{w}^*$ [15].

3. TRAINING PROCEDURE

In this section, we describe two key steps for the experiments: feature extraction and model training.

3.1. Feature extraction

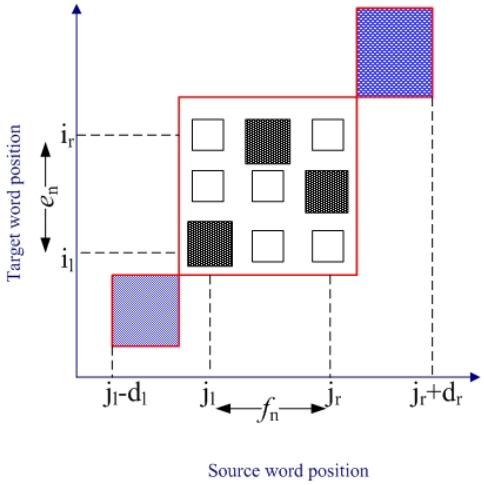


Fig. 2. Illustration of the phrase pair $(\bar{f}_n, \bar{e}_n)$ (word alignments are in black rectangle) and the phrase environment (shadow blue) used around the source phrase.

Following the Word Sense Disambiguation method [7], we consider different kinds of information extracted from the phrase environment (see Figure 2). The types of features we used are depicted in Table 2.

Types	Feature Extraction
Context	Source word n–grams within a window (length d) around the phrase edge $[j_l]$ and $[j_r]$
Syntactic	Source part of speech tag n-grams within a window (length d) around the phrase edge $[j_l]$ and $[j_r]$

Table 2. Features extracted from the phrase environment. n-gram indicates a word sequence of length n.

To specify the difference with respect to each source environment position d_z, we express the features as

$$\phi_u(s_p^{|u|}) := \delta(s_p^{|u|}, u) \qquad p = \{j_l - d_l, j_r + d_r\}$$

where $\delta(\cdot, \cdot)$ denotes the indicator function and string $s_p^{|u|} = [\bar{f}_p, \ldots, \bar{f}_{p+|u|}]$ with $|u|$ denoting the length of u. In this way, the phrase features are distinguished by both the content u and the start position p. For example, in Figure 1 the word "not" in example 2 has the following context features $\{\delta(s_0^1, \text{“I”}), \delta(s_1^1, \text{“am”}), \delta(s_3^1, \text{“sure”}), \delta(s_0^2, \text{“I am”})\}$. As required by the MMP algorithm, we then *normalise* the feature vector $\bar{\phi}_t = \frac{\phi_t}{\|\phi\|}$.

3.2. Model training

To form the training sample pool, all consistent phrase pairs $\{(\bar{f}_n, \bar{e}_n)\}_{n=1}^N$ with the corresponding features are derived from the training sentences using a phrase pair extraction procedure described in [6][1]. Then the instances having the same source phrase $\bar{f}$ are considered to be from the same cluster (see Figure 1 for example) and a mapping operator $\mathbf{w}_{\bar{f}}$ is tuned by the cluster samples only. When decoding, given a source phrase $\bar{f}_j$, we find the corresponding cluster model and predict the confidence–rated possibility for each candidate translation. For MT experiments, the confidence–rated values are then transformed to probabilities using the softmax function [12].

4. EXPERIMENTS

4.1. Part–of–speech (POS) tagging

In this paper, we view the POS tagging experiment as a subproblem of machine translation. The motivation of this experiment is to introduce the MMS model for capturing the relationships between target candidates, and we expect it to improve the performance of problematic cases described in [16]. In contrast to MT, the "distance" between POS tags for the same word are not clear and can not be measure by the Levenshtein Distance. Hence, the distance matrix $\triangle(c, \mathrm{t}_i)$ used is predefined heuristically, according to the problematic cases described in [16]. In general, the harder the problematic case is, the larger the distance will be.

The MMS model was then trained and tested on the POS tagged Wall Street Journal section of the Penn Treebank[2], in which sections 15–18 were used for training and section 20 as a test set. The data set sizes are shown in Table 3. To compare the performance, results derived from two other systems are also displayed. One is the Stanford Log–linear POS Tagger [2] that utilizing a maximum entropy based model; the other is a multi–class SVM model which is trained by SVM–Multiclass [17]. The performance is measured by the overall accuracy as well as several class–specific F1 scores for the most problematic cases.

The feature set for the Stanford system is described in [2] and a beam search decoder is applied to generate the predicted tag sequence. In our case, the MMS model and

[1] For POS tagging experiments, the word–to–tag samples with their features are derived directly from the training corpus.

[2] Data supplied by Conference on Natural Language Learning (CoNLL) 2004 shared task and can be downloaded at http://www.lsi.upc.edu/∼srlconll/soft.html.

Data Set	Tokens	Unknown Tokens
Training	$211,727$	0
Test	$47,377$	$3,092$ (6.5%)

Table 3. Data Sizes.

Feature	Description
Capital	the word contains capital character(s)
number	the word contains number(s)
hyphen	the word contains hyphen symbol
"-ed"	the word ends with "ed"
"-ing"	the word ends with "ing"
"-s"	the word ends with "s"

Table 4. Word specific features for POS tagging.

the multi–class SVM model used the context features in Table 2, as well as the word specific features demonstrated in Table 4. Observing that POS features might help the prediction, we also incorporated the syntactic features (see Table 2) by applying two–stage prediction. That is, first predicting the POS tags using the context and the specific features only, then predicting the POS tags again by incorporating the POS features predicted in the first stage. Since unknown words are unable to be assigned to certain word clusters, they are assigned to certain environment clusters instead. That is, for a sample with an unknown word f_j, it is assigned to a cluster with samples having the same environment POS tags $\{\mathrm{t}_{j-1}, *, \mathrm{t}_{j+1}\}$. In this way, we are able to predict some tags for $2,581$ out of $3,092$ unknown words. We will further investigate the other 511[3] cases and present a refined model in our future work.

The results are shown in Table 5. The accuracy figure for our model is the highest, although only slightly better than SVM predictions. We postulate that this is due to our rough definition of the distance matrix for taggers. Table 6 depicts the class–specific F1 scores for different POS taggers that have the most confusing cases. In many cases, our MMS model performed better than the multi–class SVM. As an informal comparison, to reach the same error tolerance the training time for MMS is (coded in Python) much better than SVM (coded in C++), implying that it is more applicable to larger learning problems (e.g. MT problems).

Model	Known Accuracy	Unknown Accuracy	Training Time
Stanford POS tagger	93.93%	65.26%	3.7 hours
SVM + context	94.02%	72.14%	1.3 hours
SVM + context + POS	94.27%	70.94%	1.7 hours
MMS + context	94.08%	**72.35**%	0.6 hour
MMS + context + POS	**94.30**%	70.52%	1.0 hour

Table 5. Test accuracy for known words (Known accuracy) and unknown words (Unknown accuracy) of the three systems. "context" denotes using the context and the word specific features; and "POS" denotes using the predicted POS features. The bold number indicates the best result.

Tag	F1 score	Tag	F1 score
IN	**98.2**% / 98.1%	JJ	**92.4**% / 92.3%
NN	93.7% / 93.7%	NNP	96.9% / 96.9%
NNPS	**52.4**% / 51.2%	RB	90.9% / 90.9%
RP	76.7% / **77.0**%	VB	75.4% / **75.6**%
VBD	**80.6**% / 80.5%	VBN	**69.9**% / 69.2%
VBP	77.0% / **77.2**%	VBZ	**86.5**% / 86.4%

Table 6. F1 scores for the most confusing POS classes, using "MMP + context + POS" (left) and "SVM + context + POS" (right). Bold numbers indicate better results.

4.2. Machine translation experiment

In this experiment, our goal is to verify the effect of the MMS model for two complex MT tasks: French-to-English and English-to-French translation using the NAACL2006 *EuroParl* corpus. Sentences of lengths between 1 and 100 words from the corpus were extracted where the ratio of source/target lengths was no more than $5 : 1$, and we used training and test sizes of $100,000$ and $1,000$ respectively. To compare the performance, a traditional Statistical Machine Translation system – Pharaoh [18], whose PTP model uses maximum likelihood estimation, is taken as the baseline system. To keep the comparison fair, our MT system just replaces Pharaoh's PTP model with our MMS prediction while sharing all other models (i.e. language model, phrase reordering model and decoder).

For parameter tuning, minimum-error-rating training is applied to Pharaoh while a simple grid search is applied to our system. Experiments are repeated five times to asses variance and the performance are evaluated by four MT measurements used in [6].

Table 7 depicts the translation results, where we observed consistent improvements in all evaluations. Especially, the word accuracy and the NIST score concern more about the contents (rare n-grams), an improvement in both indicates that the MMS model is better in picking up correct words and phrases, which implies the potential benefits of the structure disambiguation.

5. CONCLUSION

In this paper, we applied a Max-margin structure (MMS) learning model for two related NLP tasks: POS tagging and

[3] Currently they are assigned to a class indicated as un-predictable.

Tasks	Measure	Pharaoh	MMS
FR–EN	BLEU [%]	26.5 ± 0.4	$\mathbf{27.1} \pm 0.4$
	word accuracy	61.0 ± 0.3	$\mathbf{61.8} \pm 0.3$
	NIST	6.69 ± 0.04	$\mathbf{6.80} \pm 0.04$
	METEOR [%]	50.8 ± 0.7	$\mathbf{51.1} \pm 0.7$
EN–FR	BLEU [%]	25.1 ± 0.4	$\mathbf{26.0} \pm 0.5$
	word accuracy	57.4 ± 0.5	$\mathbf{58.6} \pm 0.5$
	NIST	6.49 ± 0.06	$\mathbf{6.65} \pm 0.06$
	METEOR [%]	47.9 ± 0.2	$\mathbf{48.6} \pm 0.3$

Table 7. Four evaluations for machine translation experiments. Bold numbers refer to the best results.

machine translation. We have shown that when using certain distance measures between output classes (e.g. tags or target translations), the MMS model showed improved performance for both tasks. Furthermore the MMP algorithm is faster than SVM without decreasing the performance in practice, making this model more applicable to the large scale learning problems (e.g. MT problems).

For future work, we will further develop our model for these NLP problems. For POS tagging, we will extend the solution for the unknown words and create a complete model. For machine translation, we will focus on the integration between the MMS model and other MT models (e.g. language model, phrase reordering model), as performance could be improved if the influence of these models are more effectively balanced in an end-to-end MT system.

6. REFERENCES

[1] T. Brants, “Tnt – a statistical part–of–speech tagger,” in *Proc. of the Sixth conf. of the ANLP*, Seattle, WA, 2000, pp. 224–231.

[2] K. Toutanova, D. Klein, C. Manning, and Y. Singer, “Feature–rich part–of–speech tagging with a cyclic dependency network,” in *Proc. HLT–NAACL*, 2003, pp. 252–259.

[3] I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun, “Support vector learning for interdependent and structured output spaces,” in *Proc. ICML*, 2004.

[4] A. Berger, S. Della Pietra, and V. Della Pietra, “A maximum entropy approach to natural language processing,” *Computational Linguistics*, vol. 22, no. 1, pp. 39–72, March 1996.

[5] Y. Ni, C. Saunders, S. Szedmak, and M. Niranjan, “Handling phrase reorderings for machine translation,” in *proc. of ACL-IJCNLP*, Singapore, 2009.

[6] P. Koehn, A. Axelrod, A. B. Mayne, C. Callison-Burch, M. Osborne, and D. Talbot, “Edinburgh system description for the 2005 iwslt speech translation evaluation,” in *Proc. of IWSLT*, Pittsburgh, PA, 2005.

[7] D. Vickrey, L. Biewald, M. Teyssier, and D. Koller, “Word–sense disambiguation for machine translation,” in *Proc. of EMNLP*, 2005, pp. 771–778.

[8] J. Giménez and L. Màrquez, “Context–aware discriminative phrase selection for statistical machine translation,” in *Proc. the Second Workshop on Statistical Machine Translation*, Prague, June 2007, pp. 159–166.

[9] B. Taskar, C. Guestrin, and D.Koller, “Part–of–speech tagging guidelines for the penn treebank project,” in *Proc. NIPS*, Vancouver, Canada, December 2003.

[10] Dan Gusfield, *Algorithms on strings, trees, and sequences: computer science and computational biology*, Cambridge University Press, New York, 1997.

[11] M. Collins, “Discriminative training methods for hidden markov models: Theory and experiments with percepton algorithms,” in *ICML*, 2002.

[12] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.

[13] C. Gentile, “A new approximate maximal margin classification algorithm,” *Journal of Machine Learning Research*, vol. 2, pp. 213–242, 2001.

[14] J. Kivinen, A. J. Smola, and R. C. Williamson., “Online learning with kernels,” *IEEE Transactions on Signal Processing*, vol. 52, no. 8, pp. 2165–2176, 2004.

[15] G. M. Korpelevich, “The extragradient method for finding saddle points and other problems,” *Ekonomika i Matematicheskie Metody*, vol. 12, pp. 747–756, 1976.

[16] B. Santorini, “Part–of–speech tagging guidelines for the penn treebank project,” in *Technical report MS-CIS-90-47, Department of Computer and Information Science, University of Pennsylvania*, 1990.

[17] T. Joachims, “Making large–scale svm learning pratical,” in *Advances in Kernel Methods – Support Vector Learning*, B. Schölkopf, C. Burges, , and A. Smola, Eds. 1999, MIT Press.

[18] P. Koehn, “Pharaoh: A beam search decoder for phrase–based statistical machine translation models,” in *Proc. of the 6th Conf. of the AMTA*, October 2004, pp. 115–124.