Kernel Regression Framework for Machine Translation: UCL System Description for WMT 2008 Shared Translation Task

Zhuoran Wang
University College London
Dept. of Computer Science
Gower Street, London, WC1E 6BT
United Kingdom
z.wang@cs.ucl.ac.uk

John Shawe-Taylor
University College London
Dept. of Computer Science
Gower Street, London, WC1E 6BT
United Kingdom
jst@cs.ucl.ac.uk

Abstract

The novel kernel regression model for SMT only demonstrated encouraging results on small-scale toy data sets in previous works due to the complexities of kernel methods. It is the first time results based on the real-world data from the shared translation task will be reported at ACL 2008 Workshop on Statistical Machine Translation. This paper presents the key modules of our system, including the kernel ridge regression model, retrieval-based sparse approximation, the decoding algorithm, as well as language modeling issues under this framework.

1 Introduction

This paper follows the work in (Wang et al., 2007; Wang and Shawe-Taylor, 2008) which applied the kernel regression method with high-dimensional outputs proposed originally in (Cortes et al., 2005) to statistical machine translation (SMT) tasks. In our approach, the machine translation problem is viewed as a string-to-string mapping, where both the source and the target strings are embedded into their respective kernel induced feature spaces. Then kernel ridge regression is employed to learn the mapping from the input feature space to the output one. As a kernel method, this model offers the potential advantages of capturing very high-dimensional correspondences among the features of the source and target languages as well as easy integration of additional linguistic knowledge via selecting particular kernels. However, unlike the sequence labeling tasks such as optical character recognition in (Cortes et al., 2005), the complexity of the SMT problem itself together with the computational complexities of kernel methods significantly complicate the implementation of the regression technique in this field.

Our system is actually designed as a hybrid of the classic phrase-based SMT model (Koehn et al., 2003) and the kernel regression model as follows: First, for each source sentence a small relevant set of sentence pairs are retrieved from the large-scale parallel corpus. Then, the regression model is trained on this small relevant set only as a sparse approximation of the regression hyperplane trained on the entire training set, as proposed in (Wang and Shawe-Taylor, 2008). Finally, a beam search algorithm is utilized to decode the target sentence from the very noisy output feature vector we predicted, with the support of a pre-trained phrase table to generate possible hypotheses (candidate translations). In addition, a language model trained on a monolingual corpus can be integrated either directly into the regression model or during the decoding procedure as an extra scoring function.

Before describing each key component of our system in detail, we give a block diagram overview in Figure 1.

2 Problem Formulation

Concretely, the machine translation problem in our method is formulated as follows. If we define a feature space $\mathcal{H}_x$ of our source language $\mathcal{X}$, and define the mapping $\Phi : \mathcal{X} \rightarrow \mathcal{H}_x$, then a sentence $\mathbf{x} \in \mathcal{X}$ can be expressed by its feature vector $\Phi(\mathbf{x}) \in \mathcal{H}_x$. The definition of the feature space $\mathcal{H}_y$ of our target language $\mathcal{Y}$ can be made in a similar way, with cor-

Proceedings of the Third Workshop on Statistical Machine Translation, pages 155–158,
Columbus, June 2008. ©2008 Association for Computational Linguistics

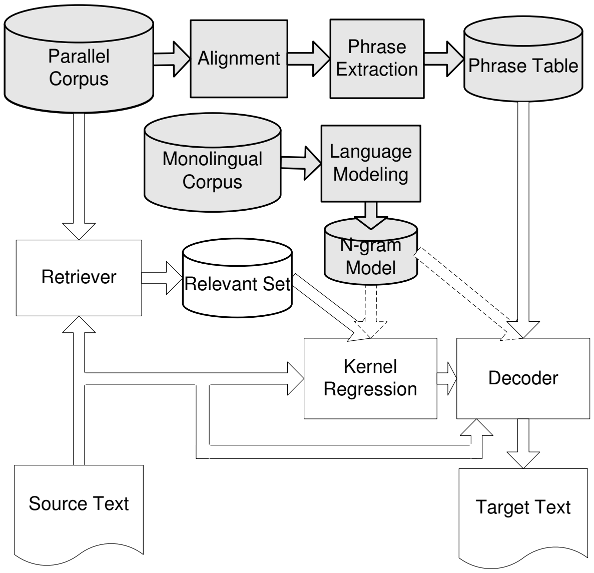

Figure 1: System overview. The processes in gray blocks are pre-performed for the whole system, while the white blocks are online processes for each input sentence. The two dash-line arrows represent two possible ways of language model integration in our system described in Section 6.

responding mapping $\Psi : \mathcal{Y} \to \mathcal{H}_y$. Now in the machine translation task, we are trying to seek a matrix represented linear operator $\mathbf{W}$, such that:

$$\Psi(\mathbf{y}) = \mathbf{W}\Phi(\mathbf{x}) \tag{1}$$

to predict the translation $\mathbf{y}$ for an arbitrary source sentence $\mathbf{x}$.

3 Kernel Ridge Regression

Based on a set of training samples, i.e. bilingual sentence pairs $S = \{(\mathbf{x}_i, \mathbf{y}_i) : \mathbf{x}_i \in \mathcal{X}, \mathbf{y}_i \in \mathcal{Y}, i = 1, \dots, m.\}$, we use ridge regression to learn the $\mathbf{W}$ in Equation (1), as:

$$\min \quad \|\mathbf{W}\mathbf{M}_\Phi - \mathbf{M}_\Psi\|_F^2 + \nu\|\mathbf{W}\|_F^2 \tag{2}$$

where $\mathbf{M}_\Phi = [\Phi(\mathbf{x}_1), ..., \Phi(\mathbf{x}_m)]$, $\mathbf{M}_\Psi = [\Psi(\mathbf{y}_1), ..., \Psi(\mathbf{y}_m)]$, $\|\cdot\|_F$ denotes the Frobenius norm that is a matrix norm defined as the square root of the sum of the absolute squares of the elements in that matrix, and ν is a regularization coefficient.

Differentiating the expression and setting it to zero gives the explicit solution of the ridge regression problem:

$$\mathbf{W} = \mathbf{M}_\Psi(\mathbf{K}_\Phi + \nu\mathbf{I})^{-1}\mathbf{M}_\Phi^\top \tag{3}$$

where $\mathbf{I}$ is the identity matrix, and $\mathbf{K}_\Phi = \mathbf{M}_\Phi^\top\mathbf{M}_\Phi = (\kappa_\Phi(\mathbf{x}_i, \mathbf{x}_j)_{1\le i,j\le m})$. Note here, we use the kernel function:

$$\kappa_\Phi(\mathbf{x}_i, \mathbf{x}_j) = \langle\Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j)\rangle = \Phi(\mathbf{x}_i)^\top\Phi(\mathbf{x}_j) \tag{4}$$

to denote the inner product between two feature vectors. If the feature spaces are properly defined, the 'kernel trick' will allow us to avoid dealing with the very high-dimensional feature vectors explicitly (Shawe-Taylor and Cristianini, 2004).

Inserting Equation (3) into Equation (1), we obtain our prediction as:

$$\Psi(\mathbf{y}) = \mathbf{M}_\Psi(\mathbf{K}_\Phi + \nu\mathbf{I})^{-1}\mathbf{k}_\Phi(\mathbf{x}) \tag{5}$$

where $\mathbf{k}_\Phi(\mathbf{x}) = (\kappa_\Phi(\mathbf{x}, \mathbf{x}_i)_{1\le i\le m})$ is an $m \times 1$ column matrix. Note here, we will use the exact matrix inversion instead of iterative approximations.

3.1 N-gram String Kernel

In the practical learning and prediction processes, only the inner products of feature vectors are required, which can be computed with the kernel function implicitly without evaluating the explicit coordinates of points in the feature spaces. Here, we define our features of a sentence as its word n-gram counts, so that a blended n-gram string kernel can be used. That is, if we denote by $\mathbf{x}^{i:j}$ a substring of sentence $\mathbf{x}$ starting with the ith word and ending with the jth, then for two sentences $\mathbf{x}$ and $\mathbf{z}$, the blended n-gram string kernel is computed as:

$$\kappa(\mathbf{x}, \mathbf{z}) = \sum_{p=1}^{n} \sum_{i=1}^{|\mathbf{x}|-p+1} \sum_{j=1}^{|\mathbf{z}|-p+1} [\![\mathbf{x}^{i:i+p-1} = \mathbf{z}^{j:j+p-1}]\!] \tag{6}$$

Here, $|\cdot|$ denotes the length of the sentence, and $[\![\cdot]\!]$ is the indicator function for the predicate. In our system, the blended tri-gram kernel is used, which means we count the n-grams of length up to 3.

4 Retrieval-based Sparse Approximation

For SMT, we are not able to use the entire training set that contains millions of sentences to train our regression model. Fortunately, it is not necessary either. Wang and Shawe-Taylor (2008) suggested that a small set of sentences whose source is relevant to the input can be retrieved, and the regression model can be trained on this small-scale relevant set only.

Src	*n' y a-t-il pas ici deux poids , deux mesures*
Rlv	pourquoi *y a-t-il deux poids , deux mesures*
	pourquoi *deux poids* et *deux mesures*
	peut-être *n' y a-t-il pas* d' épidémie non plus
	pourquoi *n' y a-t-il pas* urgence
	cette directive doit exister d' *ici deux* mois

Table 1: A sample input (Src) and some of the retrieved relevant examples (Rlv).

In our system, we take each sentence as a document and use the *tf-idf* metric that is frequently used in information retrieval tasks to retrieve the relevant set. Preliminary experiments show that the size of the relevant set should be properly controlled, as if many sentences that are not very close to the source text are involved, they will correspond to adding noise. Hence, we use a threshold of the *tf-idf* score to filter the relevant set. On average, around 1500 sentence pairs are extracted for each source sentence. Table 1 shows a sample input and some of its top relevant sentences retrieved.

5 Decoding

After the regression, we have a prediction of the target feature vector as in Equation (1). To obtain the target sentence, a decoding algorithm is still required to solve the pre-image problem. This is achieved in our system by seeking the sentence $\hat{\mathbf{y}}$ whose feature vector has the minimum Euclidean distance to the prediction, as:

$$\hat{\mathbf{y}} = \arg\min_{\mathbf{y}\in\mathcal{Y}(\mathbf{x})} \|\mathbf{W}\Phi(\mathbf{x}) - \Psi(\mathbf{y})\| \tag{7}$$

where $\mathcal{Y}(\mathbf{x}) \subset \mathcal{Y}$ denotes a finite set covering all potential translations for the given source sentence $\mathbf{x}$. To obtain a smaller search space and more reliable translations, $\mathcal{Y}(\mathbf{x})$ is generated with the support of a phrase table extracted from the whole training set. Then a modified beam search algorithm is employed, in which we restricted the distortion of the phrases by only allowing adjacent phrases to exchange their positions, and rank the search states in the beams according to Equation (7) but applied directly to the partial translations and their corresponding source parts. A more detailed explanation of the decoding algorithm can be found in (Wang et al., 2007). In addition, Wang and Shawe-Taylor (2008) further showed that the search error rate of this algorithm is acceptable.

6 Language Model Integration

In previous works (Wang et al., 2007; Wang and Shawe-Taylor, 2008), there was no language model utilized in the regression framework for SMT, as similar function can be achieved by the correspondences among the n-gram features. It was demonstrated to work well on small-scale toy data, however, real-world data are much more sparse and noisy, where a language model will help significantly.

There are two ways to integrate a language model in our framework. First, the most straightforward solution is to add a weight to adjust the strength of the regression based translation scores and the language model score during the decoding procedure. Alternatively, as language model is n-gram-based which matches the definition of our feature space, we can add a langauge model loss to the objective function of our regression model as follows. We define our language score for a target sentence $\mathbf{y}$ as:

$$\mathrm{LM}(\mathbf{y}) = \mathbf{V}^\top \Psi(\mathbf{y}) \tag{8}$$

where $\mathbf{V}$ is a vector whose components $\mathbf{V}_{y''y'y}$ will typically be log-probabilities $\log P(y|y''y')$, and y, y' and y'' are arbitrary words. Note here, in order to match our blended tri-gram induced feature space, we can make $\mathbf{V}$ of the same dimension as $\Psi(\mathbf{y})$, while zero the components corresponding to uni-grams and bi-grams. Then the regression problem can be defined as:

$$\min \|\mathbf{W}\mathbf{M}_\Phi - \mathbf{M}_\Psi\|_F^2 + \nu_1\|\mathbf{W}\|_F^2 - \nu_2\mathbf{V}^\top\mathbf{W}\mathbf{M}_\Phi\mathbf{1} \tag{9}$$

where ν_2 is a coefficient balancing between the prediction being close to the target feature vector and being a fluent target sentence, and $\mathbf{1}$ denotes a vector with components 1. By differentiating the expression with respect to $\mathbf{W}$ and setting the result to zero, we can obtain the explicit solution as:

$$\mathbf{W} = (\mathbf{M}_\Psi + \nu_2\mathbf{V}\mathbf{1}^\top)(\mathbf{K}_\Phi + \nu_1\mathbf{I})^{-1}\mathbf{M}_\Phi^\top \tag{10}$$

7 Experimental Results

Preliminary experiments are carried out on the French-English portion of the Europarl corpus. We

System	BLEU (%)	NIST	METEOR (%)	TER (%)	WER (%)	PER (%)
Kernel Regression	26.59	7.00	52.63	55.98	60.52	43.20
Moses	31.15	7.48	56.80	55.14	59.85	42.79

Table 3: Evaluations based on different metrics with comparison to Moses.

train our regression model on the training set, and test the effects of different language models on the development set (test2007). The results evaluated by BLEU score (Papineni et al., 2002) is shown in Table 2.

It can be found that integrating the language model into the regression framework works slightly better than just using it as an additional score component during decoding. But language models of higher-order than the n-gram kernel cannot be formulated to the regression problem, which would be a drawback of our system. Furthermore, the BLEU score performance suggests that our model is not very powerful, but some interesting hints can be found in Table 3 when we compare our method with a 5-gram language model to a state-of-the-art system Moses (Koehn and Hoang, 2007) based on various evaluation metrics, including BLEU score, NIST score (Doddington, 2002), METEOR (Banerjee and Lavie, 2005), TER (Snover et al., 2006), WER and PER. It is shown that our system's TER, WER and PER scores are very close to Moses, though the gaps in BLEU, NIST and METEOR are significant, which suggests that we would be able to produce accurate translations but might not be good at making fluent sentences.

8 Conclusion

This work is a novel attempt to apply the advanced kernel method to SMT tasks. The contribution at this stage is still preliminary. When applied to real-world data, this approach is not as powerful as the state-of-the-art phrase-based log-linear model. However, interesting prospects can be expected from the shared translation task.

Acknowledgements

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

	no-LM	$\text{LM}^1_{\text{3gram}}$	$\text{LM}^2_{\text{3gram}}$	$\text{LM}^1_{\text{5gram}}$
BLEU	23.27	25.19	25.66	26.59

Table 2: BLEU score performance of different language models. LM^1 denotes adding the language model during decoding process, while LM^2 represents integrating the language model into the regression framework as described in Problem (9).

References

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72.

Corinna Cortes, Mehryar Mohri, and Jason Weston. 2005. A general regression technique for learning transductions. In *Proc. of ICML'05*.

George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proc. of HLT'02*, pages 138–145.

Philipp Koehn and Hieu Hoang. 2007. Factored translation models. In *Proc. of EMNLP-CoNLL'07*.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proc. of HAACL-HLT'03*, pages 48–54.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proc. of ACL'02*.

John Shawe-Taylor and Nello Cristianini. 2004. *Kernel Methods for Pattern Analysis.* Cambridge University Press.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proc. of AMTA'06*.

Zhuoran Wang and John Shawe-Taylor. 2008. Kernel-based machine translation. In Cyril Goutte, Nicola Cancedda, Marc Dymetman, and George Foster, editors, *Learning Machine Translation*. MIT Press, to appear.

Zhuoran Wang, John Shawe-Taylor, and Sandor Szedmak. 2007. Kernel regression based machine translation. In *Proc. of NAACL-HLT'07, Short Paper Volume*, pages 185–188.