L_1 Regularized Regression for Reranking and System Combination in Machine Translation

Ergun Biçici
Koç University
34450 Sariyer, Istanbul, Turkey
ebicici@ku.edu.tr

Deniz Yuret
Koç University
34450 Sariyer, Istanbul, Turkey
dyuret@ku.edu.tr

Abstract

We use L_1 regularized transductive regression to learn mappings between source and target features of the training sets derived for each test sentence and use these mappings to rerank translation outputs. We compare the effectiveness of L_1 regularization techniques for regression to learn mappings between features given in a sparse feature matrix. The results show the effectiveness of using L_1 regularization versus L_2 used in ridge regression. We show that regression mapping is effective in reranking translation outputs and in selecting the best system combinations with encouraging results on different language pairs.

1 Introduction

Regression can be used to find mappings between the source and target feature sets derived from given parallel corpora. Transduction learning uses a subset of the training examples that are closely related to the test set without using the model induced by the full training set. In the context of SMT, we select a few training instances for each test instance to guide the translation process. This also gives us a computational advantage when considering the high dimensionality of the problem. The goal in transductive regression based machine translation (TRegMT) is both reducing the computational burden of the regression approach by reducing the dimensionality of the training set and the feature set and also improving the translation quality by using transduction. Transductive regression is shown to achieve higher accuracy than L_2 regularized ridge regression on some machine learning benchmark datasets (Chapelle et al., 1999).

In an idealized feature mapping matrix where features are word sequences, we would like to observe few target features for each source feature derived from a source sentence. In this setting, we can think of feature mappings being close to permutation matrices with one nonzero item for each column. L_1 regularization helps us achieve solutions close to the permutation matrices by increasing sparsity.

We show that L_1 regularized regression mapping is effective in reranking translation outputs and present encouraging results on different language pairs in the translation task of WMT10. In the system combination task, different translation outputs of different translation systems are combined to find a better translation. We model system combination task as a reranking problem among the competing translation models and present encouraging results with the TRegMT system.

Related Work: Regression techniques can be used to model the relationship between strings (Cortes et al., 2007). Wang et al. (2007) applies a string-to-string mapping approach to machine translation by using ordinary least squares regression and n-gram string kernels to a small dataset. Later they use L_2 regularized least squares regression (Wang and Shawe-Taylor, 2008). Although the translation quality they achieve is not better than Moses (Koehn et al., 2007), which is accepted to be the state-of-the-art, they show the feasibility of the approach. Serrano et al. (2009) use kernel regression to find translation mappings from source to target feature vectors and experiment with translating hotel front desk requests. Ueffing (2007) approaches the transductive learning problem for SMT by bootstrapping the training using the translations produced by the SMT system that have a scoring performance above some threshold as estimated by the SMT system itself.

Proceedings of the Joint 5th Workshop on Statistical Machine Translation and MetricsMATR, pages 282–289,
Uppsala, Sweden, 15-16 July 2010.

Outline: Section 2 gives an overview of regression based machine translation, which is used to find the mappings between the source and target features of the training set. In section 3 we present L_1 regularized transductive regression for alignment learning. Section 4 presents our experiments, instance selection techniques, and results on the translation task for WMT10. In section 5, we present the results on the system combination task using reranking. The last section concludes.

2 An Overview of Regression Based Machine Translation

Let X and Y correspond to the token sets used to represent source and target strings, then a training sample of m inputs can be represented as $(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_m, \mathbf{y}_m) \in X^* \times Y^*$, where $(\mathbf{x}_i, \mathbf{y}_i)$ corresponds to a pair of source and target language token sequences. Our goal is to find a mapping $f : X^* \rightarrow Y^*$ that can convert a given set of source tokens to a set of target tokens that share the same meaning in the target language.

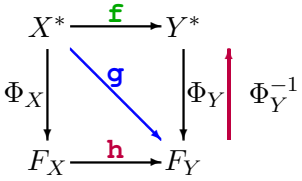

Figure 1: String-to-string mapping.

Figure 1 depicts the mappings between different representations. $\Phi_X : X^* \rightarrow F_X = \mathbb{R}^{N_X}$ and $\Phi_Y : Y^* \rightarrow F_Y = \mathbb{R}^{N_Y}$ map each string sequence to a point in high dimensional real number space where $dim(F_X) = N_X$ and $dim(F_Y) = N_Y$.

Let $\mathbf{M}_X \in \mathbb{R}^{N_X \times m}$ and $\mathbf{M}_Y \in \mathbb{R}^{N_Y \times m}$ such that $\mathbf{M}_X = [\Phi_X(\mathbf{x}_1), \ldots, \Phi_X(\mathbf{x}_m)]$ and $\mathbf{M}_Y = [\Phi_Y(\mathbf{y}_1), \ldots, \Phi_Y(\mathbf{y}_m)]$. The ridge regression solution using L_2 regularization is found as:

$$\mathbf{H}_{L_2} = \underset{\mathbf{H} \in \mathbb{R}^{N_Y \times N_X}}{\arg\min} \| \mathbf{M}_Y - \mathbf{H}\mathbf{M}_X \|_F^2 + \lambda \| \mathbf{H} \|_F^2 \quad (1)$$

Proposition 1 *Solution to the cost function given in Equation 1 is found by the following identities:*

$$\begin{aligned} \boldsymbol{H} &= \boldsymbol{M}_Y \boldsymbol{M}_X^T (\boldsymbol{M}_X \boldsymbol{M}_X^T + \lambda \boldsymbol{I}_{N_X})^{-1} \quad \textit{(primal)} \\ \boldsymbol{H} &= \boldsymbol{M}_Y (\boldsymbol{K}_X + \lambda \boldsymbol{I}_m)^{-1} \boldsymbol{M}_X^T \quad \textit{(dual)} \end{aligned} \quad (2)$$

where $\boldsymbol{K}_X = \boldsymbol{M}_X^T \boldsymbol{M}_X$ is the Gram matrix with $\boldsymbol{K}_X(i,j) = k_X(\boldsymbol{x}_i, \boldsymbol{x}_j)$ and $k_X(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is the kernel function defined as $k_X(\boldsymbol{x}_i, \boldsymbol{x}_j) = \phi(\boldsymbol{x}_i)^T \phi(\boldsymbol{x}_j)$.

The primal solution involves the inversion of the covariance matrix in the feature space ($O(N_X^3)$) and the dual solution involves the inversion of the kernel matrix in the instance space ($O(m^3)$) and L_2 regularization term prevents the normal equations to be singular. We use the dual solution when computing $\mathbf{H}_{L_2}$.

Two main challenges of the RegMT approach are learning the regression function, $g : X^* \rightarrow F_Y$, and solving the *pre-image problem*, which, given the features of the estimated target string sequence, $g(\mathbf{x}) = \Phi_Y(\hat{\mathbf{y}})$, attempts to find $\mathbf{y} \in Y^*$: $f(\mathbf{x}) = \arg\min_{\mathbf{y} \in Y^*} ||g(\mathbf{x}) - \Phi_Y(\mathbf{y})||^2$. Pre-image calculation involves a search over possible translations minimizing the cost function:

$$\begin{aligned} f(\mathbf{x}) &= \underset{\mathbf{y} \in Y^*}{\arg\min} \| \Phi_Y(\mathbf{y}) - \mathbf{H}\Phi_X(\mathbf{x}) \|^2 \\ &= \underset{\mathbf{y} \in Y^*}{\arg\min}\, k_Y(\mathbf{y}, \mathbf{y}) - 2(\mathbf{K}_Y^{\mathbf{y}})^T (\mathbf{K}_X + \lambda \mathbf{I}_m)^{-1} \mathbf{K}_X^{\mathbf{x}}, \end{aligned} \quad (3)$$

where $\mathbf{K}_Y^{\mathbf{y}} = [k_Y(\mathbf{y}, \mathbf{y}_1), \ldots, k_Y(\mathbf{y}, \mathbf{y}_m)]^T \in \mathbb{R}^{m \times 1}$ and $\mathbf{K}_X^{\mathbf{x}} \in \mathbb{R}^{m \times 1}$ is defined similarly.

We use n-spectrum weighted word kernel (Shawe-Taylor and Cristianini, 2004) as feature mappers which consider all word sequences up to order n:

$$k(\mathbf{x}, \mathbf{x}') = \sum_{p=1}^{n} \sum_{i=1}^{|x|-p+1} \sum_{j=1}^{|x'|-p+1} p\, I(\mathbf{x}[i:i+p-1] = \mathbf{x}'[j:j+p-1]) \quad (4)$$

where $\mathbf{x}[i:j]$ denotes a substring of $\mathbf{x}$ with the words in the range $[i, j]$, $I(.)$ is the indicator function, and p is the number of words in the feature.

3 L_1 Regularized Regression

In statistical machine translation, parallel corpora, which contain translations of the same documents in source and target languages, are used to estimate a likely target translation for a given source sentence based on the observed translations. String kernels lead to very sparse representations of the feature space and we examine the effectiveness of L_1 regularized regression to find the mappings between sparsely observed feature sets.

3.1 Sparsity in Translation Mappings

We would like to observe only a few nonzero target feature coefficients corresponding to a source feature in the coefficient matrix. An example solution matrix representing a possible alignment between unigram source and target features could be the following:

$\mathbf{H}$	e_1	e_2	e_3
f_1	1	1	
f_2		1	
f_3			1

Here e_i represents unigram source features and f_i represent unigram target features. e_1 and e_3 have unambiguous translations whereas e_2 is ambiguous. Even if unigram features lead to ambiguity, we expect higher order features like bigrams and trigrams to help us resolve the ambiguity. Typical $\mathbf{H}$ matrices have thousands of features. L_1 regularization helps us achieve solutions close to permutation matrices by increasing sparsity (Bishop, 2006). In contrast, L_2 solutions give us dense matrices.

3.2 L_1 Regularized Regression for Learning

$\mathbf{H}_{L_2}$ does not give us a sparse solution and most of the coefficients remain non-zero. L_1 norm behaves both as a feature selection technique and a method for reducing coefficient values.

$$\mathbf{H}_{L_1} = \underset{\mathbf{H}\in\mathbb{R}^{N_Y\times N_X}}{\arg\min} \| \mathbf{M}_Y - \mathbf{H}\mathbf{M}_X \|_F^2 + \lambda \| \mathbf{H} \|_1 \quad (5)$$

Equation 5 presents the *lasso* (least absolute shrinkage and selection operator) (Tibshirani, 1996) solution where the regularization term is now the L_1 matrix norm defined as $\| \mathbf{H} \|_1 = \sum_{i,j} |H_{i,j}|$. Since L_1 regularization cost is not differentiable, $\mathbf{H}_{L_1}$ is found by optimization or approximation techniques. We briefly describe three techniques to obtain L_1 regularized regression coefficients.

Forward Stagewise Regression (FSR): We experiment with forward stagewise regression (FSR) (Hastie et al., 2006), which approximates the *lasso*. The incremental forward stagewise regression algorithm increases the weight of the predictor variable that is most correlated with the residual by a small amount, ϵ, multiplied with the sign of the correlation at each step. As $\epsilon \to 0$, the profile of the coefficients resemble the *lasso* (Hastie et al., 2009).

Quadratic Programming (QP): We also use quadratic programming (QP) to find $\mathbf{H}_{L_1}$. We can pose *lasso* as a QP problem as follows (Mørup and Clemmensen, 2007). We assume that the rows of $\mathbf{M}_Y$ are independent and solve for each row i, $\mathbf{M}_{y_i} \in \mathbb{R}^{1\times m}$, using non-negative variables $\mathbf{h}_i^+, \mathbf{h}_i^- \in \mathbb{R}^{N_X\times 1}$ such that $\mathbf{h}_i = \mathbf{h}_i^+ - \mathbf{h}_i^-$:

$$\mathbf{h}_i = \underset{\mathbf{h}}{\arg\min} \ \| \mathbf{M}_{y_i} - \mathbf{h}\mathbf{M}_X \|_F^2 + \lambda \sum_{k=1}^{N_X} |h_k|, \quad (6)$$

$$\mathbf{h}_i = \underset{\tilde{\mathbf{h}_i}}{\arg\min} \frac{1}{2}\tilde{\mathbf{h}_i}\widetilde{\mathbf{M}_X}\widetilde{\mathbf{M}_X}^T\tilde{\mathbf{h}_i}^T - \tilde{\mathbf{h}_i}(\widetilde{\mathbf{M}_X}\mathbf{M}_{y_i}^T - \lambda\mathbf{1}), \quad (7)$$

$$\text{s.t. } \tilde{\mathbf{h}_i} > 0, \ \widetilde{\mathbf{M}_X} = \begin{bmatrix} \mathbf{M}_X \\ -\mathbf{M}_X \end{bmatrix}, \ \tilde{\mathbf{h}_i} = \begin{bmatrix} \mathbf{h}_i^+ & \mathbf{h}_i^- \end{bmatrix}.$$

Linear Programming (LP): L_1 minimization can also be posed as a linear programming (LP) problem by interpreting the error term as the constraint (Chen et al., 1998) and solving for each row i:

$$\mathbf{h}_i = \underset{\mathbf{h}}{\arg\min} \| \mathbf{h} \|_1 \ \text{ subject to } \mathbf{M}_{y_i} = \mathbf{h}\mathbf{M}_X, \quad (8)$$

which can again be solved using non-negative variables. This is a slightly different optimization and the results can be different but linear programming solvers offer computational advantages.

3.3 Transductive Regression

Transduction uses test instances, which can sometimes be accessible at training time, to learn specific models tailored towards the test set. Transduction has computational advantages by not using the full training set and by having to satisfy a smaller set of constraints. For each test sentence, we pick a limited number of training instances designed to improve the coverage of correct features to build a regression model. Section 4.2 details our instance selection methods.

4 Translation Experiments

We perform experiments on the translation task of the English-German, German-English, English-French, English-Spanish, and English-Czech language pairs using the training corpus provided in WMT10.

4.1 Datasets and Baseline

We developed separate SMT models using Moses (Koehn et al., 2007) with default settings with maximum sentence length set to 80 using 5-gram language model and obtained distinct 100-best lists for the test sets. All systems were tuned with 2051 sentences and tested with 2525 sentences. We have randomly picked 100 instances from the development set to be used in tuning the regression experiments (*dev.100*). The translation challenge test set contains 2489 sentences. Number of sentences in the training set of each system

and baseline performances for uncased output (test set BLEU, challenge test set BLEU) are given in Table 1.

Corpus	# sent	BLEU	BLEU Challenge
en-de	1609988	.1471	.1309
de-en	1609988	.1943	.1556
en-fr	1728965	.2281	.2049
en-es	1715158	.2237	.2106
en-cz	7320238	.1452	.1145

Table 1: Initial uncased performances of the translation systems.

Feature mappers used are 3-spectrum counting word kernels, which consider all N-grams up to order 3 weighted by the number of tokens in the feature. We segment sentences using some of the punctuation for managing the feature set better and do not consider N-grams that cross segments.

We use BLEU (Papineni et al., 2001) and NIST (Doddington, 2002) evaluation metrics for measuring the performance of translations automatically.

4.2 Instance Selection

Proper selection of training instances plays an important role to learn feature mappings with limited computational resources accurately. In previous work (Wang and Shawe-Taylor, 2008), sentence based training instances were selected using *tf-idf* retrieval. We transform test sentences to feature sets obtained by the kernel mapping before measuring their similarities and index the sentences based on the features. Given a source sentence of length 20, its feature representation would have a total of 57 uni/bi/tri-gram features. If we select closest sentences from the training set, we may not have translations for all the features in this representation. But if we search for translations of each feature, then we have a higher chance of covering all the features found in the sentence we are trying to translate. The index acts as a dictionary of source phrases storing training set entries whose source sentence match the given source phrase.

The number of instances per feature is chosen inversely proportional to the frequency of the feature determined by the following formula:

$$\#instance(f) = n/\ln(1 + \text{idfScore}(f)/9.0), \quad (9)$$

where idfScore(f) sums the *idf* (inverse document frequency) of the tokens in feature f and n is a small number.

4.3 Addition of Brevity Penalty

Detailed analysis of the results shows TRegMT score achieves better N-gram match percentages than Moses translation but suffers from the brevity penalty due to selecting shorter translations. Due to using a cost function that minimizes the squared loss, TRegMT score tends to select shorter translations when the coverage is low. We also observe that we are able to achieve higher scores for NIST, which suggests the addition of a brevity penalty to the score.

Precision based BLEU scoring divides N-gram match counts to N-gram counts found in the translation and this gives an advantage to shorter translations. Therefore, a brevity penalty (BP) is added to penalize short translations:

$$BP = \min(1 - \frac{\text{ref-length}}{\text{trans-length}}, 0) \quad (10)$$

$$BLEU = e^{\left(\log(\texttt{ngram}_{prec}) + BP\right)} \quad (11)$$

where $\texttt{ngram}_{prec}$ represent the sum of n-gram precisions. Moses rarely incurs BP as it has a word penalty parameter optimized against BLEU which penalizes translations that are too long or too short. For instance, Moses 1-best translation for *en-de* system achieves .1309 BLEU versus .1320 BLEU without BP.

We handle short translations in two ways. We optimize the λ parameter of QP, which manages the sparsity of the solution (larger λ values correspond to sparser solutions) against BLEU score rather than the squared loss. Optimization yields λ = 20.744. We alternatively add a BP cost to the squared loss:

$$BP = e^{\left(\min(1 - \frac{|\Phi_Y(\mathbf{y})|}{|\ulcorner \mathbf{H}\Phi_X(\mathbf{x}) + \alpha_{BP} \urcorner|}, 0)\right)} \quad (12)$$

$$f(\mathbf{x}) = \arg\min_{\mathbf{y} \in Y^*} \| \Phi_Y(\mathbf{y}) - \mathbf{H}\Phi_X(\mathbf{x}) \|^2 + \lambda_{BP} BP \quad (13)$$

where $|.|$ denotes the length of the feature vector, $\ulcorner . \urcorner$ rounds feature weights to integers, α_{BP} is a constant weight added to the estimation, and λ_{BP} is the weight given for the BP cost. $|\ulcorner \mathbf{H}\Phi_X(\mathbf{x}) + \alpha_{BP} \urcorner|$ represents an estimate of the length of the reference as found by the TRegMT system. This BP cost estimate is similar to the cost used in (Serrano et al., 2009) normalized by the length of the reference. We found α_{BP} = 0.1316 and λ_{BP} = -13.68 when optimized on the *en-de* system. We add a BP penalty to all of the reranking results given in the next section and QP results also use optimized λ.

	en-de		*de-en*		*en-fr*		*en-es*		*en-cz*	
Score	**BLEU**	**NIST**	**BLEU**	**NIST**	**BLEU**	**NIST**	**BLEU**	**NIST**	**BLEU**	**NIST**
Baseline	.1309	5.1417	.1556	5.4164	.2049	6.3194	.2106	6.3611	.1145	4.5008
Oracle	.1811	6.0252	.2101	6.2103	.2683	7.2409	.2770	7.3190	.1628	5.4501
L2	**.1319**	**5.1680**	.1555	**5.4344**	.2044	**6.3370**	**.2132**	**6.4093**	.1148	**4.5187**
FSR	*.1317**	**5.1639**	.1559	**5.4383**	.2053	**6.3458**	**.2144**	**6.4168**	.1150	**4.5172**
LP	.1317	**5.1695**	.1561	**5.4304**	.2048	6.3245	.2109	**6.4176**	.1124	4.5143
QP	.1309	**5.1664**	.1550	**5.4553**	.2033	*6.3354**	**.2121**	**6.4271**	.1150	**4.5264**

Table 2: Reranking results using TRegMT, TM, and LM scores. We use approximate randomization test (Riezler and Maxwell, 2005) with 1000 repetitions to determine score difference significance: results in **bold** are significant with $p \leq 0.01$ and *italic* results with (*) are significant with $p \leq .05$. The difference of the remaining from the baseline are not statistically significant.

4.4 Reranking Experiments

We rerank N-best lists by using linear combinations of the following scoring functions:

1. TRegMT: Transductive regression based machine translation scores as found by Equation 3.
2. TM: Translation model scores we obtain from the baseline SMT system that is used to generate the N-best lists.
3. LM: 5-gram language model scores that the baseline SMT system uses when calculating the translation model scores.

The training set we obtain may not contain all of the features of the reference target due to low coverage. Therefore, when performing reranking, we also add the cost coming from the features of $\Phi_Y(\mathbf{y})$ that are not represented in the training set to the squared loss as in:

$$\| \Phi_Y(\mathbf{y}) \setminus F_Y \|^2 + \| \Phi_Y(\mathbf{y}) - \mathbf{H}\Phi_X(\mathbf{x}) \|^2, \quad (14)$$

where $\Phi_Y(\mathbf{y}) \setminus F_Y$ represent the features of $\mathbf{y}$ not represented in the training set.

We note that TRegMT score only contains ordering information as present in the bi/tri-gram features in the training set. Therefore, the addition of a 5-gram LM score as well as the TM score, which also incorporates the LM score in itself, improves the performance. We are not able to improve the BLEU score when we use TRegMT score by itself however we are able to achieve improvements in the NIST and 1-WER scores. The performance increase is important for two reasons. First of all, we are able to improve the performance using blended spectrum 3-gram features against translations obtained with 5-gram language model and higher order features. Outperforming higher order n-gram models is known to be a difficult task (Galley and Manning, 2009). Secondly, increasing the performance with reranking itself is a hard task since possible translations are already constrained by the ones observed in N-best lists. Therefore, an increase in the N-best list size may increase the score gaps.

Table 2 presents reranking results on all of the language pairs we considered, using TRegMT, TM, and LM scores with the combination weights learned in the development set. We are able to achieve better BLEU and NIST scores on all of the listed systems. We are able to see up to .38 BLEU points increase for the *en-es* pair. Oracle reranking performances are obtained by using BLEU scoring metric.

If we used only the TM and LM scores when reranking with the *en-de* system, then we would obtain .1309 BLEU and 5.1472 NIST scores. We only see a minor increase in the NIST score and no change in the BLEU score with this setting when compared with the baseline given in Table 2.

Due to computational reasons, we do not use the same number of instances to train different models. In our experiments, we used $n = 3$ for L2, $n = 1.5$ for FSR, and $n = 1.2$ for QP and LP solutions to select the number of instances in Equation 9. The average number of instances used per sentence in training corresponding to these choices are approximately 140, 74, and 61. Even with these decreased number of training instances, L_1 regularized regression techniques are able to achieve comparable scores to L_2 regularized regression model in Table 2.

5 System Combination Experiments

We perform experiments on the system combination task for the English-German, German-English, English-French, English-Spanish, and English-Czech language pairs using the training

	en-de		*de-en*		*en-fr*		*en-es*		*en-cz*	
Score	**BLEU**	**NIST**	**BLEU**	**NIST**	**BLEU**	**NIST**	**BLEU**	**NIST**	**BLEU**	**NIST**
Random	.1490	5.6555	.2088	6.4886	.2415	6.8948	.2648	7.2563	.1283	4.9238
Best model	.1658	5.9610	.2408	6.9861	.2864	7.5272	.3047	7.7559	.1576	5.4480
L2	**.1694**	5.9974	.2336	**6.9398**	**.2948**	7.7037	.3036	7.8120	.1657	5.5654
FSR	.1689	5.9638	.2357	6.9254	.2947	7.7107	**.3049**	**7.8156**	.1657	5.5632
LP	**.1694**	5.9954	**.2368**	6.8850	.2928	**7.7157**	.3027	7.7838	.1659	5.5680
QP	.1692	**5.9983**	**.2368**	6.9172	.2913	7.6949	.3040	7.8086	**.1662**	**5.5785**

Table 3: Reranking results using TRegMT, TM, and LM scores. **bold** correspond to the best score in each rectangle of scores.

corpus provided in WMT10.

5.1 Datasets

We use the training set provided in WMT10 to index and select transductive instances from. The challenge split the test set for the translation task of 2489 sentences into a tuning set of 455 sentences and a test set with the remaining 2034 sentences. Translation outputs for each system is given in a separate file and the number of system outputs per translation pair varies. We have tokenized and lowercased each of the system outputs and combined these in a single N-best file per language pair. We also segment sentences using some of the punctuation for managing the feature set better. We use these N-best lists for TRegMT reranking to select the best translation model. Feature mappers used are 3-spectrum counting word kernels, which consider all n-grams up to order 3 weighted by the number of tokens in the feature.

5.2 Experiments

We rerank N-best lists by using combinations of the following scoring functions:

1. TRegMT: Transductive regression based machine translation scores as found by Equation 3.

2. TM': Translation model scores are obtained by measuring the average BLEU performance of each translation relative to the other translations in the N-best list.

3. LM: We calculate 5-gram language model scores for each translation using the language model trained over the target corpus provided in the translation task.

Since we do not have access to the reference translations nor to the translation model scores each system obtained for each sentence, we estimate translation model performance (TM') by measuring the average BLEU performance of each translation relative to the other translations in the N-best list. Thus, each possible translation in the N-best list is BLEU scored against other translations and the average of these scores is selected as the TM score for the sentence. Sentence level BLEU score calculation avoids singularities in n-gram precisions by taking the maximum of the match count and $\frac{1}{2|s_i|}$ for $|s_i|$ denoting the length of the source sentence s_i as used in (Macherey and Och, 2007).

Table 3 presents reranking results on all of the language pairs we considered, using TRegMT, TM, and LM scores with the same combination weights as above. Random model score lists the random model performance selected among the competing translations randomly and it is used as a baseline. Best model score lists the performance of the best model performance. We are able to achieve better BLEU and NIST scores in all of the listed systems except for the *de-en* language pair when compared with the performance of the best competing translation system. The lower performance in the *de-en* language pair may be due to having a single best translation system that outperforms others significantly. The difference between the best model performance and the mean as well as the variance of the scores in the *de-en* language pair is about twice their counterparts in *en-de* language pair.

Due to computational reasons, we do not use the same number of instances to train different models. In our experiments, we used $n = 4$ for L2, $n = 1.5$ for FSR, and $n = 1.2$ for QP and LP solutions to select the number of instances in Equation 9. The average number of instances used per sentence in training corresponding to these choices are approximately 189, 78, and 64.

6 Contributions

We use transductive regression to learn mappings between source and target features of given parallel corpora and use these mappings to rerank translation outputs. We compare the effectiveness of L_1 regularization techniques for regression. TRegMT score has a tendency to select shorter translations when the coverage is low. We incorporate a brevity penalty to the squared loss and optimize λ parameter of QP to tackle this problem and further improve the performance of the system.

The results show the effectiveness of using L_1 regularization versus L_2 used in ridge regression. Proper selection of training instances plays an important role to learn correct feature mappings with limited computational resources accurately. We plan to investigate better instance selection methods for improving the translation performance. TRegMT score has a tendency to select shorter translations when the coverage is low. We incorporate a brevity penalty to the score and optimize the λ parameter of QP to tackle this problem.

Acknowledgments

The research reported here was supported in part by the Scientific and Technological Research Council of Turkey (TUBITAK).

References

Christopher M. Bishop. 2006. *Pattern Recognition and Machine Learning.* Springer.

Olivier Chapelle, Vladimir Vapnik, and Jason Weston. 1999. Transductive inference for estimating values of functions. In *NIPS*, pages 421–427.

Scott Shaobing Chen, David L. Donoho, and Michael A. Saunders. 1998. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20(1):33–61.

Corinna Cortes, Mehryar Mohri, and Jason Weston. 2007. A general regression framework for learning string-to-string mappings. In Gokhan H. Bakir, Thomas Hofmann, and Bernhard Sch editors, *Predicting Structured Data*, pages 143–168. The MIT Press, September.

George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the second international conference on Human Language Technology Research*, pages 138–145, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Michel Galley and Christopher D. Manning. 2009. Quadratic-time dependency parsing for machine translation. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 773–781, Suntec, Singapore, August. Association for Computational Linguistics.

Trevor Hastie, Jonathan Taylor, Robert Tibshirani, and Guenther Walther. 2006. Forward stagewise regression and the monotone lasso. *Electronic Journal of Statistics*, 1.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. 2009. *The Elements of Statistical Learning: Data Mining, Inference and Prediction*. Springer-Verlag, 2nd edition.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Annual Meeting of the Assoc. for Computational Linguistics*, pages 177–180, Prague, Czech Republic, June.

Wolfgang Macherey and Franz J. Och. 2007. An empirical study on computing consensus translations from multiple machine translation systems. In *EMNLP-CoNLL*, pages 986–995.

M. Mørup and L. H. Clemmensen. 2007. Multiplicative updates for the lasso. In *Machine Learning for Signal Processing MLSP, IEEE Workshop on*, pages 33 –38, Aug.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2001. BLEU: a method for automatic evaluation of machine translation. In *ACL '02: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 311–318, Morristown, NJ, USA. Association for Computational Linguistics.

Stefan Riezler and John T. Maxwell. 2005. On some pitfalls in automatic evaluation and significance testing for MT. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 57–64, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Nicolas Serrano, Jesus Andres-Ferrer, and Francisco Casacuberta. 2009. On a kernel regression approach to machine translation. In *Iberian Conference on Pattern Recognition and Image Analysis*, pages 394–401.

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

Robert J. Tibshirani. 1996. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58(1):267–288.

Nicola Ueffing, Gholamreza Haffari, and Anoop Sarkar. 2007. Transductive learning for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 25–32, Prague, Czech Republic, June. The Association for Computer Linguistics.

Zhuoran Wang and John Shawe-Taylor. 2008. Kernel regression framework for machine translation: UCL system description for WMT 2008 shared translation task. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 155–158, Columbus, Ohio, June. Association for Computational Linguistics.

Zhuoran Wang, John Shawe-Taylor, and Sandor Szedmak. 2007. Kernel regression based machine translation. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers*, pages 185–188, Rochester, New York, April. Association for Computational Linguistics.