Adaptive Model Weighting and Transductive Regression for Predicting Best System Combinations

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

S. Serdar Kozat
Koç University
34450 Sariyer, Istanbul, Turkey
skozat@ku.edu.tr

Abstract

We analyze adaptive model weighting techniques for reranking using instance scores obtained by L_1 regularized transductive regression. Competitive statistical machine translation is an on-line learning technique for sequential translation tasks where we try to select the best among competing statistical machine translators. The competitive predictor assigns a probability per model weighted by the sequential performance. We define additive, multiplicative, and loss-based weight updates with exponential loss functions for competitive statistical machine translation. Without any pre-knowledge of the performance of the translation models, we succeed in achieving the performance of the best model in all systems and surpass their performance in most of the language pairs we considered.

1 Introduction

When seen as independent instances, system combination task can be solved with a sequential learning algorithm. Online learning algorithms enable us to benefit from previous good model choices to estimate the next best model. We use transductive regression based machine translation model to estimate the scores for each sentence.

We analyze adaptive model weighting techniques for system combination when the competing translators are SMT models. We use separate model weights weighted by the sequential performance. We use additive, multiplicative, or loss based weight updates to update model weights. Without any pre-knowledge of the performance of the translation models, we are able to achieve the performance of the best model in all systems and we can surpass its performance as well as the regression based machine translation's performance.

The next section reviews the transductive regression approach for machine translation, which we use to obtain instance scores. In section 3 we present competitive statistical machine translation model for solving sequential translation tasks with competing translation models. Section 4 presents our results and experiments and the last section gives a summary of our contributions.

2 Transductive Regression Based Machine Translation

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 since we are not using the full training set and a smaller set of constraints exist to satisfy. Transductive regression based machine translation (TRegMT) aims to reduce the computational burden of the regression approach by reducing the dimensionality of the training set and the feature set and also improve the translation quality by using transduction.

Regression Based Machine Translation: Let n training instances be represented as $(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n) \in X^* \times Y^*$, 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.

We use feature mappers $\Phi_X : X^* \to F_X = \mathbb{R}^{N_X}$ and $\Phi_Y : Y^* \to F_Y = \mathbb{R}^{N_Y}$ to represent the training set. Then, $\mathbf{M}_X \in \mathbb{R}^{N_X \times n}$ and $\mathbf{M}_Y \in \mathbb{R}^{N_Y \times n}$ such that $\mathbf{M}_X = [\Phi_X(\mathbf{x}_1), \ldots, \Phi_X(\mathbf{x}_n)]$ and $\mathbf{M}_Y = [\Phi_Y(\mathbf{y}_1), \ldots, \Phi_Y(\mathbf{y}_n)]$. The ridge regression solution using L_2 regularization is found as:

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

Two main challenges of the regression based machine translation (RegMT) approach are learning the regression function, $g : X^* \to F_Y$, 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:

$$f(\mathbf{x}) = \underset{\mathbf{y} \in Y^*}{\arg\min} \parallel \Phi_Y(\mathbf{y}) - \mathbf{H}\Phi_X(\mathbf{x}) \parallel^2 . \quad (2)$$

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

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

$$\mathbf{H}_{L_1} = \underset{\mathbf{H} \in \mathbb{R}^{N_Y \times N_X}}{\arg\min} \parallel \mathbf{M}_Y - \mathbf{H}\mathbf{M}_X \parallel_F^2 + \lambda \parallel \mathbf{H} \parallel_1 . (3)$$

Equation 3 presents the *lasso* (least absolute shrinkage and selection operator) (Tibshirani, 1996) solution where the regularization term is defined as $\|\mathbf{H}\|_1 = \sum_{i,j} |H_{i,j}|$. We use forward stagewise regression (FSR) (Hastie et al., 2006) and quadratic programming (QP) to find $\mathbf{H}_{L_1}$. The details of the TRegMT model can be read in a separate submission to the translation task (Bicici and Yuret, 2010).

3 Competitive Statistical Machine Translation

We develop the Competitive Statistical Machine Translation (CSMT) framework for sequential translation tasks when the competing models are statistical machine translators. CSMT uses the output of different translation models to achieve a translation performance that surpasses the translation performance of all of the component models or achieves the performance of the best.

CSMT uses online learning to update the weights used for estimating the best performing translation model. Competitive predictor assigns a weight per model estimated by their sequential performance. At each step, m component translation models are executed in parallel over the input source sentence sequence and the loss $l_p[n]$ of model p at observation n is calculated by comparing the desired data $y[n]$ with the output of model p, $\hat{y_p}[n]$. CSMT model selects a model based on the weights and the performance of the selected model as well as the remaining models to adaptively update the weights given for each model. This corresponds to learning in *full information setting* where we have access to the loss for each action (Blum and Mansour, 2007). CSMT learning involves two main steps: *estimation* and *weight update*:

$$\begin{aligned} \hat{y_c}[n] &= E(\mathbf{w}[n], \mathbf{x}[n]), && (\texttt{estimation}) \\ l_p[n] &= y[n] - \hat{y_p}[n], && (\texttt{instance loss}) \\ \mathcal{L}_p[n] &= \textstyle\sum_{i=1}^n l_p[i]^2, && (\texttt{cumulative loss}) \\ \mathbf{w}[n+1] &= U(\mathbf{w}[n], \hat{y_c}[n], \boldsymbol{\mathcal{L}}[n]), && (\texttt{update}) \end{aligned} \quad (4)$$

where $\mathbf{w}[n] = (w_1[n], \ldots, w_m[n])$ for m models, $\mathcal{L}_p$ is the cumulative squared loss of model p, $\boldsymbol{\mathcal{L}}[n]$ stores cumulative and instance losses, and $\hat{y_c}[n]$ is the competitive model estimated for instance n. The learning problem is finding an adaptive $\mathbf{w}$ that minimizes the cumulative squared error with appropriate estimation and update methods.

Related Work: Multistage adaptive filtering (Kozat and Singer, 2002) combines the output of multiple adaptive filters to outperform the best among them where the first stage executes models in parallel and the second stage updates parameters using the performance of the combined prediction, $\hat{y_c}[n]$. Macherey and Och (2007) investigate different approaches for system combination including candidate selection that maximize a weighted combination of BLEU scores among different system outputs. Their system uses a fixed weight vector trained on the development set to be multiplied with instance BLEU scores.

3.1 Estimating the Best Performing Translation Model

We use additive, multiplicative, or loss based updates to estimate model weights. We measure instance loss with $\texttt{trLoss}(y[i], \hat{y_p}[i])$, which is a function that returns the translation performance of the output translation of model p with respect to the reference translation at instance i. 1-BLEU (Papineni et al., 2001) is one such function with outputs in the range $[0, 1]$. Cumulative squared loss of the p-th translation model is defined as:

$$\mathcal{L}_p[n] = \sum_{i=1}^{n} \texttt{trLoss}(y[i], \hat{y_p}[i])^2. \qquad (5)$$

We use *exponentially re-weighted prediction* to estimate model performances, which uses exponentially re-weighted losses based on the outputs of the m different translation models.

We define the *additive* exponential weight update as follows:

$$w_p[n+1] = \frac{w_p[n] + e^{-\eta\, l_p[n]}}{\sum_{k=1}^{m} \left(w_k[n] + e^{-\eta\, l_k[n]}\right)}, \qquad (6)$$

where $\eta > 0$ is the learning rate and the denominator is used for normalization. The update amount, $e^{-\eta\, l_p[n]}$ is 1 when $l_p[n] = 0$ and it approaches zero with increasing instance loss. Perceptrons, gradient descent, and Widrow-Huff learning have additive weight updates.

We define the *multiplicative* exponential weight update as follows:

$$w_p[n+1] = w_p[n] \times \frac{e^{-\eta\, l_p[n]^2}}{\sum_{k=1}^{m} w_k[n]\; e^{-\eta\, l_k[n]^2}}, \qquad (7)$$

where we use the squared instance loss. Equation 7 is similar to the update of Weighted Majority Algorithm (Littlestone and Warmuth, 1992) where the weights of the models that make a mistake are multiplied by a fixed β such that $0 \le \beta < 1$.

We use *Bayesian Information Criterion (BIC)* as a *loss based* re-weighting technique. Assuming that instance losses are normally distributed with variance σ^2, BIC score is obtained as (Hastie et al., 2009):

$$\text{BIC}_p[n] = \frac{\mathcal{L}_p[n]}{\sigma^2} + d_p \log(n), \qquad (8)$$

where σ^2 is estimated by the average of model sample variances of squared instance loss and d_p is the number of parameters used in model p which we assume to be the same for all models; therefore we can discard the second term. The model with the minimum BIC value becomes the one with the highest posterior probability where the posterior probability of model p can be estimated as (Hastie et al., 2009):

$$w_p[n+1] = \frac{e^{-\frac{1}{2}\text{BIC}_p[n]}}{\sum_{k=1}^{m} e^{-\frac{1}{2}\text{BIC}_k[n]}}. \qquad (9)$$

The posterior probabilities become model weights and we basically forget about the previous weights, whose information is presumably contained in the cumulative loss, $\mathcal{L}_p$. We define multiplicative re-weighting with BIC scores as follows:

$$w_p[n+1] = w_p[n] \times \frac{e^{-\frac{1}{2}\text{BIC}_p}}{\sum_{k=1}^{m} w_k[n]\; e^{-\frac{1}{2}\text{BIC}_k}}. \qquad (10)$$

Model selection: We use *stochastic* or *deterministic* selection to choose the competitive model for each instance. Deterministic choice randomly selects among the maximum scoring models with minimum translation length whereas stochastic choice draws model p with probability proportional to $w_p[n]$. Randomization with the stochastic model selection decreases expected mistake bounds in the weighted majority algorithm (Littlestone and Warmuth, 1992; Blum, 1996).

Auer et al. (2002) show that optimal fixed learning rate for the weighted majority algorithm is found as $\eta[n] = \sqrt{m/\mathcal{L}_*[n]}$ where $\mathcal{L}_*[n] = \min_{1 \le i \le m} \mathcal{L}_i[n]$, which requires prior knowledge of the cumulative losses. We use $\eta = \sqrt{m/(0.05n)}$ for constant η.

4 Experiments and Discussion

We perform experiments on the system combination task for the English-German (*en-de*), German-English (*de-en*), English-French (*en-fr*), English-Spanish (*en-es*), and English-Czech (*en-cz*) language pairs using the translation outputs for all the competing systems provided in WMT10. We experiment in a *simulated online learning* setting where only the scores obtained from the TRegMT system are used during both tuning and testing. We do not use reference translations in measuring instance performance in this simulated setting for the results we obtain be in line with system combination challenge's goals.

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

4.2 Reranking Scores

The problem we are solving is online learning with prior information, which comes from the comparative BLEU scores, LM scores, and TRegMT scores at each step n. The scoring functions are explained below:

1. TRegMT: Transductive regression based machine translation scores as found by Equation 2. We use the TRegMT scores obtained by the FSR model.

2. CBLEU: Comparative BLEU scores we obtain by measuring the average BLEU performance of each translation relative to the other systems' 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.

To make things simpler, we use a single prior TRegMT system score linearly combining the three scores mentioned with weights learned on the tuning set. The overall TRegMT system score for instance n, model i is referred as $\texttt{TRegScore}_i[n]$.

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 by measuring the average BLEU performance of each translation relative to 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 CBLEU 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).

4.3 Adaptive Model Weighting

We initialize model weights to $1/m$ for all models, which are updated after each instance according to the losses based on the TRegMT model. Table 1 presents the performance of the algorithms on the *en-de* development set. We have measured their performances with stochastic (stoc.) or deterministic (det.) model selection when using only the weights or mixture weights obtained when instance scores are also considered. Mixture weights are obtained as: $w_i[n] = w_i[n]\ \texttt{TRegScore}_i[n]$, for instance n, model i.

Baseline performance obtained with random selection has .1407 BLEU and 4.9832 NIST scores. TRegMT model obtains a performance of .1661 BLEU and 5.3283 NIST with reranking. The best model performance among the 12 *en-de* translation models has .1644 BLEU and 5.2647 NIST scores. Therefore, by using TRegMT score, we are able to achieve better scores.

Not all of the settings are meaningful. For instance, stochastic model selection is used for algorithms having multiplicative weight updates. This is reflected in the Table 1 by low performance on the additive and BIC models. Similarly, using mixture weights may not result in better scores for algorithms with multiplicative updates, which resulted in decreased

	Additive		Multiplicative		BIC		BIC Weighting	
Setting	BLEU	NIST	BLEU	NIST	BLEU	NIST	BLEU	NIST
Stoc., W	.1419	5.0016 ±.003	.1528	5.1710 ±.001	.1442	5.0468	.1568 ±.001	5.2052 ±.005
Stoc., M	.1415	5.0001	.1525	5.1601 ±.001	.1459	5.0619 ±.004	.1566 ±.001	5.2030 ±.006
Det., W	.1644	5.3208	.1638	5.2571	.1638	5.2542	.1646	5.2535
Det., M	.1643	5.3173	.1536	5.1756	.1530	5.1871	.1507	5.1973

Table 1: Performances of the algorithms on the development set over 100 repetitions. W: Weights, M: Mixture.

performance in Table 1. Decreased performance with BIC hints that we may use other techniques for mixture weights.

Table 2 presents reranking results on all of the language pairs we considered with the random, TRegMT, and CSMT models. Random model score lists the random model performance selected among the competing translations randomly and it can be used as a baseline. Best model score lists the performance of the best model performance. CSMT models are named with the weighting model used (Add for additive, Mul for multiplicative, BICW for BIC weighting), model selection technique (S for stochastic, D for deterministic), and mixtures model (W for using only weights, M for using mixture weights) with hyphens in between. Our challenge submission is given in the last row of Table 2 where we used multiplicative exponential weight updates, deterministic model selection, and only the weights during model selection. For the challenge results, we initialized the weights to the weights obtained in the development set.

We have presented scores that are better than or close to the best model in **bold**. We observe that the additive model performs the best by achieving the performance of the best competing translation model and performing better than the best in most of the language pairs. For the *en-de* language pair, additive model score achieves even better than the TRegMT model, which is used for evaluating instance scores.

5 Contributions

We have analyzed adaptive model weighting techniques for system combination when the competing translators are statistical machine translation models. We defined additive, multiplicative, and loss-based weight updates with exponential loss functions for the competitive statistical machine translation framework.

Competitive SMT via adaptive weighting of various translators is shown to be a powerful technique for sequential translation tasks. We have demonstrated its use in the system combination task by using the instance scores obtained by the TRegMT model. Without any pre-knowledge of the performance of the translation models, we have been able to achieve the performance of the best model in all systems and we are able to surpass its performance as well as TRegMT's performance with the additive model.

Acknowledgments

The research reported here was supported in part by the Scientific and Technological Research Council of Turkey (TUBITAK). The first author would like to thank Deniz Yuret for helpful discussions and for guidance and support during the term of this research.

References

Auer, Cesa-Bianchi, and Gentile. 2002. Adaptive and self-confident on-line learning algorithms. *JCSS: Journal of Computer and System Sciences*, 64.

Ergun Bicici and Deniz Yuret. 2010. L_1 regularized regression for reranking and system combination in machine translation. In *Proceedings of the ACL 2010 Joint Fifth Workshop on Statistical Machine Translation and Metrics MATR*, Uppsala, Sweden, July. Association for Computational Linguistics.

Avrim Blum and Yishay Mansour. 2007. Learning, regret minimization and equilibria. In Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay V. Vazirani, editors, *Algorithmic Game Theory (Cambridge University Press, 2007)*.

Avrim Blum. 1996. On-line algorithms in machine learning. In *In Proceedings of the Workshop on On-Line Algorithms, Dagstuhl*, pages 306–325. Springer.

	en-de		*de-en*		*en-fr*		*en-es*		*en-cz*	
Model	**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
TRegMT	.1689	5.9638	.2357	6.9254	.2947	7.7107	.3049	7.8156	.1657	5.5632
Add-D-W	**.1697**	**5.9821**	**.2354**	**6.9175**	**.2948**	**7.7094**	**.3043**	**7.8093**	**.1642**	**5.5463**
Add-D-M	**.1698**	**5.9824**	**.2353**	**6.9152**	**.2949**	**7.7103**	**.3044**	**7.8091**	**.1642**	**5.5461**
Mul-S-W	.1574	5.7564	.2161	6.5950	.2805	7.4599	**.2961**	**.7.6870**	**.1572**	**5.4394**
Mul-D-W	.1618	5.8912	**.2408**	**6.9854**	**.2847**	**7.5085**	.2785	7.4133	.1612	5.5119
BIC-D-W	.1614	5.8852	**.2408**	**6.9853**	**.2842**	**7.5022**	.2785	7.4132	**.1623**	**5.5236**
BIC-D-M	.1580	5.7614	.2141	6.5597	.2791	7.4309	.2876	7.5138	**.1577**	**5.4488**
BICW-S-W	.1621	5.8795	.2274	6.8142	.2802	7.4873	.2892	7.5569	**.1565**	**5.4126**
BICW-S-M	.1618	5.8730	.2196	6.6493	.2806	7.4948	.2849	7.4845	.1561	5.4099
BICW-D-W	**.1648**	**5.9298**	**.2355**	**6.9112**	.2807	7.4648	.2785	7.4134	.1534	5.3458
Challenge	.1567	5.73	**.2394**	**6.9627**	.2758	7.4333	**.3047**	**7.7559**	**.1641**	**5.5435**

Table 2: CSMT results where **bold** corresponds to scores better than or close to the best model. Underlined scores are better than both the TregMT model and the best model.

George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Human Language Technology Research*, pages 138–145.

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.

S.S. Kozat and A.C. Singer. 2002. Further results in multistage adaptive filtering. *ICASSP*, 2:1329–1332.

Nick Littlestone and Manfred K. Warmuth. 1992. The Weighted Majority Algorithm. Technical Report UCSC-CRL-91-28, University of California, Santa Cruz, Jack Baskin School of Engineering, October 26,.

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.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2001. Bleu: a method for automatic evaluation of machine translation. In *ACL*, pages 311–318. ACL.

J. Shawe Taylor and N. 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.