The SDL Language Weaver Systems in the WMT12 Quality Estimation Shared Task

Radu Soricut **Nguyen Bach** **Ziyuan Wang**
SDL Language Weaver
6060 Center Drive, Suite 101
Los Angeles, CA, USA
{rsoricut,nbach,zwang}@sdl.com

Abstract

We present in this paper the system submissions of the SDL Language Weaver team in the WMT 2012 Quality Estimation shared-task. Our MT quality-prediction systems use machine learning techniques (M5P regression-tree and SVM-regression models) and a feature-selection algorithm that has been designed to directly optimize towards the official metrics used in this shared-task. The resulting submissions placed 1st (the M5P model) and 2nd (the SVM model), respectively, on both the Ranking task and the Scoring task, out of 11 participating teams.

1 Introduction

The WMT 2012 Quality Estimation shared-task focused on automatic methods for estimating machine translation output quality at run-time (sentence-level estimation). Different from MT evaluation metrics, quality prediction (QP) systems do not rely on reference translations and are generally built using machine learning techniques to estimate quality scores (Specia et al., 2009; Soricut and Echihabi, 2010; Bach et al., 2011; Specia, 2011).

Some interesting uses of sentence-level MT quality prediction are the following: decide whether a given translation is good enough for publishing as-is (Soricut and Echihabi, 2010), or inform monolingual (target-language) readers whether or not they can rely on a translation; filter out sentences that are not good enough for post-editing by professional translators (Specia, 2011); select the best translation among options from multiple MT systems (Soricut and Narsale, 2012), etc.

This shared-task focused on estimating the quality of English to Spanish automatic translations. The training set distributed for the shared task comprised of 1, 832 English sentences taken from the news domain and their Spanish translations. The translations were produced by the Moses SMT system (Koehn et al., 2007) trained on Europarl data. Translations also had a quality score derived from an average of three human judgements of Post-Editing effort using a 1-5 scale (1 for worse-quality/most-effort, and 5 for best-quality/least-effort). Submissions were evaluated using a blind official test set of 422 sentences produced in the same fashion as the training set. Two sub-tasks were considered: (i) scoring translations using the 1-5 quality scores (Scoring), and (ii) ranking translations from best to worse (Ranking). The official metrics used for the Ranking task were DeltaAvg (measuring how valuable a proposed ranking is from the perspective of extrinsic values associated with the test entries, in this case post-editing effort on a 1-5 scale; for instance, a DeltaAvg of 0.5 means that the top-ranked quantiles have +0.5 better quality on average compared to the entire set), as well as the Spearman ranking correlation. For the Scoring task the metrics were Mean-Absolute-Error (MAE) and Root Mean Squared Error (RMSE). The interested reader is referred to (Callison-Burch et al., 2012) for detailed descriptions of both the data and the evaluation metrics used in the shared-task.

The SDL Language Weaver team participated with two submissions based on M5P and SVM regression models in both the Ranking and the Scoring

Proceedings of the 7th Workshop on Statistical Machine Translation, pages 145–151,
Montréal, Canada, June 7-8, 2012.

tasks. The models were trained and used to predict Post-Editing–effort scores. These scores were used as-such for the Scoring task, and also used to generate sentence rankings for the Ranking task by simply (reverse) sorting the predicted scores. The submissions of the SDL Language Weaver team placed 1st (the M5P model) and 2nd (the SVM model) on both the Ranking task (out of 17 entries) and the Scoring task (out of 19 entries).

2 The Feature Set

Both SDLLW system submissions were created starting from 3 distinct sets of features: the baseline feature set (here called BFs), the internal features available in the decoder logs of Moses (here called MFs), and an additional set of features that we developed internally (called LFs). We are presenting each of these sets in what follows.

2.1 The Baseline Features

The WMT Quality Estimation shared-task defined a set of 17 features to be used as "baseline" features. In addition to that, all participants had access to software that extracted the corresponding feature values from the inputs and necessary resources (such as the SMT-system's training data, henceforth called SMT$_{src}$ and SMT$_{trg}$). For completeness, we are providing here a brief description of these 17 baseline features (BFs):

BF1 number of tokens in the source sentence

BF2 number of tokens in the target sentence

BF3 average source token length

BF4 LM probability of source sentence

BF5 LM probability of the target sentence

BF6 average number of occurrences of the target word within the target translation

BF7 average number of translations per source word in the sentence (as given by IBM 1 table thresholded so that $Prob(t|s) > 0.2$)

BF8 average number of translations per source word in the sentence (as given by IBM 1 table thresholded so that $Prob(t|s) > 0.01$) weighted by the inverse frequency of each word in the source corpus

BF9 percentage of unigrams in quartile 1 of frequency (lower frequency words) in SMT$_{src}$

BF10 percentage of unigrams in quartile 4 of frequency (higher frequency words) in SMT$_{src}$

BF11 percentage of bigrams in quartile 1 of frequency of source words in SMT$_{src}$

BF12 percentage of bigrams in quartile 4 of frequency of source words in SMT$_{src}$

BF13 percentage of trigrams in quartile 1 of frequency of source words in SMT$_{src}$

BF14 percentage of trigrams in quartile 4 of frequency of source words in SMT$_{src}$

BF15 percentage of unigrams in the source sentence seen in SMT$_{src}$

BF16 number of punctuation marks in source sentence

BF17 number of punctuation marks in target sentence

These features, together with the other ones we present here, are entered into a feature-selection component that decides which feature set to use for optimum performance (Section 3.2).

In Table 1, we are presenting the performance on the official test set of M5P and SVM-regression (SVR) models using only the BF features. The M5P model is trained using the Weka package [1] and the default settings for M5P decision-trees (weka.classifiers.trees.M5P). The SVR model is trained using the LIBSVM toolkit [2]. The following options are used: "-s 3" (ϵ-SVR) and "-t 2" (radial basis function). The following parameters were optimized via 5-fold cross-validation on the training data: "-c cost", the parameter C of ϵ-SVR; "-g gamma", the γ parameter of the kernel function; "-p epsilon", the ϵ for the loss-function of ϵ-SVR.

[1] http://www.cs.waikato.ac.nz/ml/weka/

[2] http://www.csie.ntu.edu.tw/∼cjlin/libsvm/

Systems	**Ranking**		**Scoring**		
	DeltaAvg	Spearman	MAE	RMSE	Predict. Interval
17 BFs with M5P	0.53	0.56	0.69	0.83	[2.3-4.9]
17 BFs with SVR	0.55	0.58	0.69	0.82	[2.0-5.0]
best-system	0.63	0.64	0.61	0.75	[1.7-5.0]

Table 1: Performance of the Baseline Features using M5P and SVR models on the test set.

The results in Table 1 are compared against the "best-system" submission, in order to offer a comparison point. The "17 BFs with SVM" system actually participated as an entry in the shared-task, representing the current state-of-the-art in MT quality-prediction. This system has been ranked 6th (out of 17 entries) in the Ranking task, and 8th (out of 19 entries) in the Scoring task.

2.2 The Decoder Features

The current Quality Estimation task has been defined as a glass-box task. That is, the prediction component has access to everything related to the internal workings of the MT system for which the quality prediction is made. As such, we have chosen to use the internal scores of the Moses [3] decoder (available to all the participants in the shared-task) as a distinct set of features. These features are the following:

MF1 Distortion cost

MF2 Word penalty cost

MF3 Language-model cost

MF4 Cost of the phrase-probability of source given target $\Phi(s|t)$

MF5 Cost of the word-probability of source given target $\Phi_{lex}(s|t)$

MF6 Cost of the phrase-probability of target given source $\Phi(t|s)$

MF7 Cost of the word-probability of target given source $\Phi_{lex}(t|s)$

MF8 Phrase penalty cost

These features are then entered into a feature-selection component that decides which feature set to use for achieving optimal performance.

The results in Table 2 present the performance on the test set of the Moses features (with an M5P model), presented against the "best-system" submission. These numbers indicate that the Moses-internal features, by themselves, are fueling a QP system that surpasses the performance of the strong "baseline" system. We note here that the "8 MFs with M5P" system would have been ranked 4th (out of 17 entries) in the Ranking task, and 5th (out of 19 entries) in the Scoring task.

2.3 Language Weaver Features

In addition to the features presented until this point, we have created and tested additional features that helped our systems achieve improved performance. In addition to the SMT training corpus, these features also use the SMT tuning dev set (henceforth called Dev$_{src}$ and Dev$_{trg}$). These features are the following:

LF1 number of out-of-vocabulary tokens in the source sentence

LF2 LM perplexity for the source sentence

LF3 LM perplexity for the target sentence

LF4 geometric mean (λ-smoothed) of 1-to-4–gram precision scores (i.e., BLEU score without brevity-penalty) of source sentence against the sentences of SMT$_{src}$ used as "references"

LF5 geometric mean (λ-smoothed) of 1-to-4–gram precision scores of target translation against the sentences of SMT$_{trg}$ used as "references"

[3] http://www.statmt.org/moses/

Systems	**Ranking**		**Scoring**		
	DeltaAvg	Spearman-Corr	**MAE**	RMSE	Predict. Interval
8 MFs with M5P	0.58	0.58	0.65	0.81	[1.8-5.0]
best-system	0.63	0.64	0.61	0.75	[1.7-5.0]

Table 2: Performance of the Moses-based Features with an M5P model on the test set.

LF6 geometric mean (λ-smoothed) of 1-to-4–gram precision scores of source sentence against the top BLEU-scoring quartile of Dev$_{src}$

LF7 geometric mean (λ-smoothed) of 1-to-4–gram precision scores of target translation against the top BLEU-scoring quartile of Dev$_{trg}$

LF8 geometric mean (λ-smoothed) of 1-to-4–gram precision scores of source sentence against the bottom BLEU-scoring quartile of Dev$_{src}$

LF9 geometric mean (λ-smoothed) of 1-to-4–gram precision scores of target translation against the bottom BLEU-scoring quartile Dev$_{trg}$

LF10 geometric mean (λ-smoothed) of 1-to-4–gram precision scores of target translation against a pseudo-reference produced by a second MT Eng-Spa system

LF11 count of one-to-one (O2O) word alignments between source and target translation

LF12 ratio of O2O alignments over source sentence

LF13 ratio of O2O alignments over target translation

LF14 count of O2O alignments with Part-of-Speech–agreement

LF15 ratio of O2O alignments with Part-of-Speech–agreement over O2O alignments

LF16 ratio of O2O alignments with Part-of-Speech–agreement over source

LF17 ratio of O2O alignments with Part-of-Speech–agreement over target

Most of these features have been shown to help Quality Prediction performance, see (Soricut and Echihabi, 2010) and (Bach et al., 2011). Some of them are inspired from word-based confidence estimation, in which the alignment consensus between the source words and target-translation words are informative indicators for gauging the quality of a translation hypothesis. The one-to-one (O2O) word alignments are obtained from the decoding logs of Moses. We use the TreeTagger to obtain Spanish POS tags[4] and a maximum-entropy POS tagger for English. Since Spanish and English POS tag sets are different, we normalize their fine-grained POS tag sets into a coarser tag set by mapping the original POS tags into more general linguistic concepts such as noun, verb, adjective, adverb, preposition, determiner, number, and punctuation.

3 The Models

3.1 The M5P Prediction Model

Regression-trees built using the M5P algorithm (Wang and Witten, 1997) have been previously shown to give good QP performance (Soricut and Echihabi, 2010). For these models, the number of linear equations used can provide a good indication whether the model overfits the training data. In Table 3, we compare the performance of several M5P models: one trained on all 42 features presented in Section 2, and two others trained on only 15 and 14 features, respectively (selected using the method described in Section 3.2). We also present the number of linear equations (L.Eq.) used by each model. Aside from the number of features they employ, these models were trained under identical conditions: default parameters of the Weka implementation, and 1527 training instances (305 instances were held-out for the feature-selection step, from the total 1832 labeled instances available for the shared-task).

As the numbers in Table 3 clearly show, the set of

[4] http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/

Systems	#L.Eq.	**Dev Set**		**Test Set**	
		DeltaAvg	MAE	DeltaAvg	MAE
42 FFs with M5P	10	0.60	0.58	0.56	0.64
(best-system) 15 FFs with M5P	**2**	**0.63**	**0.52**	**0.63**	**0.61**
14 FFs with M5P	6	0.62	**0.50**	0.61	0.62

Table 3: M5P-model performance for different feature-function sets (15-FFs ∈ 42-FFs; 14-FFs ∈ 42-FFs).

feature-functions that an M5P model is trained with matters considerably. On both our development set and the official test set, the 15-FF M5P model outperforms the 42-FF model (even if 15-FF ∈ 42-FF). The 42-FF model would have been ranked 5th (out of 17 entries) in the Ranking task, and also 5th (out of 19 entries) in the Scoring task. In comparison, the 15-FF model (feature set optimized for best performance under the DeltaAvg metric) was our official M5P submission (SDLLW_M5PBestDeltaAvg), and ranked 1st in the Ranking task and also 1st in the Scoring task. The 14-FF model (also a subset of the 42-FF set, optimized for best performance under the MAE metric) was not part of our submission, but would have been ranked 2nd on both the Ranking and Scoring tasks.

The number of linear equations used (see #L.Eq. in Table 3) is indicative for our results. When using 42 FFs, the M5P model seems to overfit the training data (10 linear equations). In contrast, the model trained on a subset of 15 features has only 2 linear equations. This latter model is less prone to overfitting, and performs well given unseen test data. The same number for the 14-FF model indicates slight overfit on the training and dev data: with 6 equations, this model has the best MAE numbers on the Dev set, but slightly worse MAE numbers on the Test score compared to the 15-FF model.

3.2 Feature Selection

As we already pointed out, some of the features of the entire 42-FF set are highly overlapping and capture roughly the same amount of information. To achieve maximum performance given this featureset, we applied a computationally-intensive featureselection method. We have used the two official metrics, DeltaAvg and MAE, and a development set of 305 instances to perform an extensive featureselection procedure that directly optimizes the two official metrics using M5P regression-trees.

The overall space that needs to be explored for 42 features is huge, on the order of 2^{42} possible combinations. We performed the search in this space in several steps. In a first step, we eliminated the obviously overlapping features (e.g., BF5 and MF3 are both LM costs of the target translation), and also excluded the POS-based features (LF14-LF17, see Section 2.3). This step reduced the overall number of features to 24, and therefore left us with an order of 2^{24} possible combinations. Next, we exhaustively searched all these combinations by building and evaluating M5P models. This operation is computationally-intensive and takes approximatively 60 hours on a cluster of 800 machines. At the conclusion of this step, we ranked the results and considered the top 64 combinations. The performance of these top combinations was very similar, and a set of 15 features was selected as the superset of active feature-functions present in most of the top 64 combinations.

DeltaAvg optim.	BF1 BF3 BF4 BF6 BF12 BF13 BF14 MF3 MF4 MF6 LF1 LF10 LF14 LF15 LF16
MAE optim.	BF1 BF3 BF4 BF6 BF12 BF14 BF16 MF3 MF4 MF6 LF1 LF10 LF14 LF17

Table 4: Feature selection results.

The second round of feature selection considers these 15 feature-functions plus the 4 POS-based feature-functions, for a total of 19 features and therefore a space of 2^{19} possible combinations (2^{15} of these already covered by the first search pass). A second search procedure was executed exhaustively

		Dev Set		Test Set	
SVR Model (C;γ;ϵ)	#S.V.	DeltaAvg	MAE	DeltaAvg	MAE
1.0 ; 0.00781; 0.50	695	0.62	0.52	0.60	0.66
1.74; 0.00258; 0.3299	**952**	**0.63**	**0.51**	**0.61**	**0.64**
8.0 ; 0.00195; 0.0078	1509	0.64	0.50	0.60	0.68
16.0; 0.00138; 0.0884	1359	0.63	0.51	0.59	0.70

Table 5: SVR-model performance for dev and test sets.

over the set of all the new possible combinations. In the end, we selected the winning feature-function combination as our final feature-function sets: 15 features for DeltaAvg optimization and 14 features for MAE optimization. They are given in Table 4, using the feature id-s given in Section 2. The performance of these two feature-function sets using M5P models can be found in Table 3.

3.3 The SVM Prediction Model

The second submission of our team consists of rankings and scores produced by a system using an ϵ-SVM regression model (ϵ-SVR) and a subset of 19 features. This model is trained on 1,527 training examples by the LIBSVM package using radial basis function (RBF) kernel. We have found that the feature-set obtained by the feature-selection optimization for M5P models described in Section 3.2 does not achieve the same performance for SVR models on our development set. Therefore, we have performed our SVR experiments using a hand-selected set of features: 9 features from the BF family (BF1 BF3 BF4 BF6 BF10 BF11 BF12 BF14 BF16); all 8 features from the MF family; and 2 features from the LF family (LF1 LF10).

We optimize the three hyper parameters C, γ, and ϵ of the SVR method using a grid-search method and measure their performance on our development set of 305 instances. The C parameter is a penalty factor: if C is too high, we have a high penalty for non-separable points and may store many support vectors and therefore overfit the training data; if C is too low, we may end up with a model that is poorly fit. The ϵ parameter determines the level of accuracy of the approximated function; however, getting too close to zero may again overfit the training data. The γ parameter relates to the RBF kernel: large γ values give the model steeper and more flexible kernel functions, while small gamma values give the model smoother functions. In general, C, ϵ, and γ are all sensitive parameters and instantiate ϵ-SVR models that may behave very differently.

In order to cope with the overfitting issue given a small amount of training data and grid search optimization, we train our models with 10-fold cross validation and restart the tuning process several times using different starting points and step sizes. We select the best model parameters based on a couple of indicators: the performance on the development set and the number of support vectors of the model. In Table5 we present the performance of different model parameters on both the development set and the official test set. Our second submission (SDLLW_SVM), which placed 2nd in both the Ranking and the Scoring tasks, is the entry in bold font. It was chosen based on good performance on the Dev set and also a setting of the (C, γ, ϵ) parameters that provides a number of support vectors that is neither too high nor too low. As a contrastive point, the model on the row below it uses 1,509 support vectors extracted from 1,527 training vectors, which represents a clear case of overfitting. Indeed, the performance of this model is marginally better on the Dev set, but ends up underperforming on the Test data.

4 Conclusions

The WMT 2012 Quality Estimation shared-task provided the opportunity for the comparing different QP systems using shared datasets and standardized evaluation metrics. Our participation in this shared-task revealed two important aspects of Quality Prediction for MT that we regard as important for the future. First, our experiments indicated that the

Moses-internal features, by themselves, can fuel a QP-system that surpasses the performance of the strong "baseline" system used in this shared task to represent state-of-the-art performance in MT quality prediction. This is a surprising finding, considering that these decoder-internal features have been primarily designed to gauge differences in translation quality when starting from the same source sentence. In contrast, for quality-prediction tasks like ranking one needs to gauge differences in quality of translations of different source sentences.

The second aspect relates to the importance of feature selection. Given the availability and good scalability of Machine Learning toolkits today, it is tempting to throw as much features as possible at this problem and let the built-in mechanisms of these learning algorithms deal with issues relating to feature overlapping, training-data overfitting, etc. However, these learning algorithms have their own limitations in these regards, and, in conjunction with the limited availability of the labeled data, can easily produce models that are underperforming on blind tests. There is a need for careful engineering of the models and evaluation of the resulting performance in order to achieve optimal performance using the current state-of-the-art supervised learning techniques.

References

Nguyen Bach, Fei Huang, and Yaser Al-Onaizan. 2011. Goodness: A method for measuring machine translation confidence. In *Proceedings of the ACL/HLT*, Portland, Oregon, USA.

Chris Callison-Burch, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2012. Findings of the 2012 Workshop on Statistical Machine Translation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, Montreal, Canada, June. Association for Computational Linguistics.

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 *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*.

Radu Soricut and Abdessamad Echihabi. 2010. Trustrank: Inducing trust in automatic translations via ranking. In *Proceedings of ACL*.

Radu Soricut and Sushant Narsale. 2012. Combining quality prediction and system selection for improved automatic translation output. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, Montreal, Canada, June. Association for Computational Linguistics.

Lucia Specia, Nicola Cancedda, Marc Dymetman, Marcho Turchi, and Nello Cristianini. 2009. Estimating the sentence-level quality of machine translation. In *Proceedings of EAMT*.

Lucia Specia. 2011. Exploiting objective annotations for measuring translation post-editing effort. In *Proceedings of EAMT*.

Y. Wang and I. H. Witten. 1997. Induction of model trees for predicting continuous classes. In *Proceedings of the 9th European Conference on Machine Learning*.