Evaluating Machine Translation Systems using Weighted Vote of Different Scoring Metrics

Anmol Sharma
School of Computing Science
Simon Fraser University
Bunarby, BC, Canada
anmol_sharma@sfu.ca

Amirali Sharifian
School of Computing Science
Simon Fraser University
Bunarby, BC, Canada
amiralis@sfu.ca

Shreeasish Kumar
School of Computing Science
Simon Fraser University
Bunarby, BC, Canada
ska196@sfu.ca

Abstract

Automatic evaluation metrics are fundamentally important for Machine Translation, allowing comparison of systems performance and efficient training. Current evaluation metrics fall into two classes: heuristic approaches, like BLEU or METEOR, and those using supervised learning trained on human judgment data. Evaluating a machine translation system using such heuristic metrics is faster, easier and cheaper as compared to human evaluations, which require trained bilingual evaluators.

In this report, we present our approach to combine multiple automatic evaluation heuristics using machine learning and then compare the quality of translation of two different MT system. Our results show that while each heuristic approach can provide a valid comparison between two system, combining the techniques using our machine learning approach provides higher accuracy.

1 Introduction

Human judgments of translation quality are very expensive. For this reason, automatic MT evaluation metrics are used as an approximation by comparing predicted translations to human authored references. Automatic metrics are useful for comparing the performance of different systems on a common translation task, and can be applied on a frequent and ongoing basis during MT system development. In recent years, there have been different automatic MT evaluation metrics proposed, each having their own strengths and weaknesses. The most commonly used MT evaluation metric in recent years has been IBMs Bleu metric (Papineni et al., 2002). BLEU in comparison to other MT evaluation techniques is fast and easy to run. It can also be used as a target function in parameter optimization training procedures that are commonly used in statistical MT systems (Och, 2003). METEOR (Lavie and Agarwal, 2005) is another technique that evaluates a translation by computing a score based on explicit word-to-word matches between the translation and a given reference translation. There are many other MT evaluation techniques (Song and Cohn, 2011), (Lodhi et al., 2002) proposed that each of them try to tackle drawbacks of other techniques.

1.1 Motivation

Our motivation in this project is to combine different MT evaluation metrics to leverage strengths of these techniques. We model the problem as a classification problem and then by training our classifier using reference data we build a model that can capture the benefits of different methods while reducing the weight of drawbacks of implemented techniques in the final decision. We implemented three techniques, BLEU, METEOR, and ROUGE-L in the first step. In the next step, we feed our classifier a vector containing results of these three methods. Using reference data we train our classifier and eventually in the last step we run our classifier on evaluation data. Our results in Section 4 show that our final results are improved in comparison to each of these three implementations in isolation.

The report is organized as follows. Section 2 discusses different MT evaluation techniques that we

implemented and our proposed approach. Section 3 shows our evaluations and how each implementation works in isolation. In Section 4 we provide our final results and analysis. Finally in Section 5, we conclude our report.

2 Approach

In this project, we pose the MT evaluation problem of choosing the best hypothesis which is close to the reference translation as a supervised learning problem. We propose to utilize a weighted vote of different scoring metrics proposed in MT evaluation literature to predict the best hypothesis for the given data. The standard scoring metrics that were used and implemented in this study are briefly described in the subsequent subsections. The machine learning based approach is described in subsection 2.4.

2.1 BLEU

The BiLingual Evaluation Understudy is a scoring metric proposed by Kishore et al. (Papineni et al., 2002) from IBM Research in 2002. The metric was one of the first ones to claim high correlation with human judgments on the task of evaluating candidate translations corresponding to a reference translation in a machine translation system. It is a relatively simple and computationally inexpensive algorithm. BLEU uses a modified unigram precision metric defined over the whole test corpus as:

$$p_n = \frac{\sum_{c \in H} \sum_{ngram \in c} Count_{clip}(ngram)}{\sum_{c' \in H} \sum_{ngram' \in c'} Count(ngram')} \quad (1)$$

where $Count_{clip}(ngram)$ is a function that truncates each word's count in a test hypothesis $c \in H$ where H is the set of all hypothesis in test corpus. This is done to prevent it from exceeding the number of times a word appears in the reference, and penalizing the model for generating high frequency words again and again to get a higher score.

2.2 ROGUE-L

Recall-Oriented Understudy for Gisting Evaluation (ROUGE) (Lin, 2004) is a set of metric used for evaluating automatic summarization and machine translation software in natural language processing. ROGUE-L computes the length of the longest common subsequence which can be matched between the hypothesis and reference sentences. Once the length is computed, precision and recall metrics are computed as:

$$P = \frac{lcs}{t_h}, \;\; R = \frac{lcs}{t_r} \quad (2)$$

where t_h = number of tokens in hypothesis, and t_r = number of tokens in the reference sentence. Then, the final score is calculated as follows:

$$\text{ROGUE-L} = \frac{(1+\beta^2)PR}{R + \beta^2 P} \quad (3)$$

2.3 METEOR

Metric for Evaluation of Translation with Explicit ORdering (METEOR) (Lavie and Agarwal, 2005) is another machine translation system evaluation metric which uses a harmonic mean of unigram precision and recall, with a higher weight value assigned to recall than precision. METEOR also makes use of stemming and synonym matching following the exact word matching step.

METEOR initializes by computing word alignments between hypothesis translation and reference translation. The alignments are computed first using exact word matching then stemming (stripping each unaligned word to its stem and matching with an identical unaligned stemmed word in reference), and finally by looking up synonyms of unaligned words. The order of matching in our implementation is exactly how its described in the paper. Once the alignment is computed, various quantities are calculated, like unigram precision:

$$P = \frac{m}{w_t} \quad (4)$$

where m = number of unigrams in hypothesis translation matched to reference translation, and w_t = total number of unigrams in hypothesis translation.

Also, unigram recall R is calculated as:

$$R = \frac{m}{w_r} \quad (5)$$

where w_r = number of unigrams in reference translation. The harmonic mean of P and R is calculated by weighting recall 9 times more than precision, given by:

$$F_{mean} = \frac{10PR}{R + 9P} \tag{6}$$

Also, longer ngram matches are used to compute a penalty ρ for the alignment, given as

$$\rho = 0.5 \left(\frac{c}{u_m} \right)^3 \tag{7}$$

which is finally used to compute the METEOR score M as:

$$M = F_{mean} * (1 - \rho) \tag{8}$$

2.4 Proposed Method

In this paper, we propose to use a weighted vote of different scoring schemes described above. It is known that different scoring methods tend to capture different characteristics of the hypotheses with respect to reference translations. We aim to investigate whether a combination of these metrics can perform better than how they perform individually.

Instead of manually finding weights for each score, we pose this as a classification problem in a supervised learning setting, and employ different machine learning based methods for classification. We implement the above standard scoring schemes for MT system evaluation to obtain scores for each hypothesis translation with respect to the reference translation. We generate BLEU, ROGUE and METEOR scores for the whole training corpora, and append the scores into a vector corresponding to each training instance as $[B_{h1n}, B_{h2n}, R_{h1n}, R_{h2n}, M_{h1n}, M_{h2n}]$ where n = training instance. The input is a matrix $X \in \mathbb{R}^{n\text{x}6}$ where n = number of training examples (25568 in our case), and 6 = number of total scores (one for each hypothesis, two for each metric, and total six for all metrics). The target vector $Y \in \mathbb{R}^{n\text{x}1}$ has three distinct classes, 1 = if hypothesis 1 is chosen, -1 = if hypothesis 2 is chosen, and 0 = both hypothesis equally good. Once the data is generated, we employ Logistic Regression (one-vs-all setting), Support Vector Machine (in a one-vs-all setting), and Artificial Neural Network classifiers to test our proposed strategy. We found that our proposed method of using a weighted vote of the scores perform much better than using the scores individually. A detailed description of the experimental strategy is given in subsection 3.2.

3 Evaluation

3.1 Data

The data used in this study consists of a total 51135 training instances, with two hypothesis translations h_1 and h_2, and a reference translation ref from a source language to a target language (English). However the given data only has ground truth values for the first 25568 instances, hence to train our machine learning models, we use the first 25568 instances of the given training data. The data suitable for training the models was generated after calculating scores from each scoring metric and concatenating them as a vector corresponding to each training instance. We do not employ any outside data other than the provided data.

Table 1: Hyperparameters for different machine learning models

Model	Hyperparameter	Value
Artificial Neural Network	weights	Xavier
	bias	Xavier
	λ	0.2
	iterations	50
	optimizer	ADADELTA
	batch_size	2
	learning rate l_r	1.0
	ρ	0.95
	ϵ	$1e^{-08}$
Logistic Regression	penalty	ℓ_2
	ϵ	$1e^{-04}$
	C	1.0
	solver	*saga*
	multi-class	*multinomial*
	max-iter	1000
Support Vector Machine	C	1.0
	kernel	RBF
	γ	*auto*
	degree	3
	max-iter	*auto*
	ϵ	$1e^{-03}$

3.2 Experiments

To judge the performance gain of weighing different metrics, as compared to using them individually,

Table 2: Weight coefficients for different scores learnt by logistic regression classifier.

Class Label	B_{h_1}	B_{h_2}	R_{h_1}	R_{h_2}	M_{h_1}	M_{h_2}
Class 1 (h_1)	0.88951787	-0.85079232	0.81369278	-1.20302491	1.16166655	-0.96624235
Class 0 ($h_1 = h_2$)	-0.03558901	-0.09759929	0.10346993	0.5692253	0.0562018	-0.3993358
Class -1 (h_2)	-0.85392886	0.94839161	-0.9171627	0.6337996	-1.21786835	1.36557814

we first perform experiments comparing the three metrics independently on the given dataset, which are given in Table 3. Further, we use three machine learning classifiers - Logistic Regression (LR), Support Vector Machine (SVM) and Aritifical Neural Networks (ANN) to learn the underlying pattern of scores and predicting the best hypothesis given a vector of these scores. The models were implemented using scikit-learn (Pedregosa et al., 2011). The hyperparameters for each of the classifier were chosen using a heuristic process, and are shown in the Table 1. The weights of our ANN model were initialized using Xavier Uniform (Glorot and Bengio, 2010) method.

The comparison of all the classifiers in terms of performance is given in Table 4. We evaluate our method using accuracy scores observed on the leaderboard[1].

4 Results and Analysis

In our experiments we found that Logistic Regression performs the best in terms of classifying the best hypothesis with respect to closeness with the reference translation. Since logistic regression assigns weights to individual features of the input (compared to SVM, where it assigns weights to the individual training instances), we dig further and investigate the weights assigned to each feature.

Table 3: Comparing different MT evaluation methods

Metric	**Accuracy**
BLEU	0.525
ROGUE-L	0.524
METEOR	**0.537**

We found that for class 1 (given hypothesis 1 is the better one), BLEU and METEOR scores get the highest weights. For class 0, we get high weights for ROGUE and METEOR scores, and for class -1, BLEU and METEOR again get the highest weights assigned to the scores. Our experiments prove our assumption that exploiting different metrics in a weighted vote setting may lead to an increased performance since different metrics capture different characteristics of the translations.

[1] http://anoopsarkar.github.io/nlp-class/leaderboard.html

Table 4: Comparing different machine learning classifier performances.

Classifier	**Accuracy**
LR	**0.552**
SVM	0.551
ANN	0.550

5 Conclusion

Application of machine learning towards finding weighted combination of MT evaluation score metrics has shown promising prospects. As future work, it would be interesting to experiment with more scoring metrics, preferably with the ones which compliment other metrics in terms of the characteristics of the hypothesis they capture. This would allow the classifiers to better find the combinations given a larger search space and options. Also, it would be interesting to apply deep learning methods, by first converting the hypothesis and reference to word embeddings (using word2vec) and then comparing their distance with the matching ngrams.

References

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *International conference on artificial intelligence and statistics*, pages 249–256.

Alon Lavie and Abhaya Agarwal. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the EMNLP 2011 Workshop on Statistical Machine Translation*, pages 65–72.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out: Proceedings of the ACL-04 workshop*, volume 8. Barcelona, Spain.

Huma Lodhi, Craig Saunders, John Shawe-Taylor, Nello Cristianini, and Chris Watkins. 2002. Text classification using string kernels. *Journal of Machine Learning Research*, 2(Feb):419–444.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 160–167. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Xingyi Song and Trevor Cohn. 2011. Regression and ranking based optimisation for sentence level machine translation evaluation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 123–129. Association for Computational Linguistics.