DeepBLEU

Austin Narcomey, Andrew Narcomey, Khalid Ahmad
{aon2, aon1, kahmad}@stanford.edu

Abstract

BLEU has been a key tool in the evaluation of machine translation that has allowed for rapid development of new neural network models. However, its ease of computation comes at the expense of true language understanding. This problem motivates our project to test a variety of neural network models, based upon LSTMs, using a collection of human-evaluated machine translations. We aimed to exploit the technique of attention, which has proven successful in NLP tasks, such as machine translation and natural language inference, to develop a better model for machine translation evaluation. We show that using an LSTM layer with a Multi-layer Perceptron outperforms BLEU with respect to correlation with human judgment, as does a model with self attention to encode inputs, thus providing more useful metrics that do not rely on numerous external databases and do not require extensive training.

1 Introduction

The focus of this project is to explore neural-network-based machine translation metrics that escape the pitfalls of traditional automated metrics such as BLEU, while remaining more efficient, cheap, and scalable than human evaluation. Evaluation is a critical aspect of developing more powerful NLP systems, which have to experimentally demonstrate stronger performance. Human evaluation is regarded as the gold standard for rating machine translation, discussed in Papineni *et al.* [8], therefore the aim of this project is to develop a model that can more closely match human ratings than other automated metrics such as BLEU, while retaining similar scalability and efficiency, which is something that human evaluation lacks.

Our objective is to construct a model that maximizes the correlation between a model's machine translation scores and the scores given by human raters in datasets from WMT (http://statmt.org). We will refer to this task as MT Eval: given a series of English reference translations and corresponding model translations into English, evaluate the quality the translation with maximum correlation to human evaluation. We test the performance of many different networks built upon an LSTM architecture with self-attention and inter-sequence attention variations, and other types of layers and architecture that map from sentence encodings to translation scores. We draw inspiration in our implementations of attention from transformer models as they are known to produce promising results when applied to NLP challenges, Domhan (2018) [2]. We have included non-neural-network-based baseline models to give context to the performance of our models in machine translation evaluation.

2 Related Work

The most recent work in the use of neural networks for evaluation of machine translation models was done in Guzmán *et al.* [4], published in 2017. The paper explores the use of an LSTM that takes as input generated translations, a reference translation, and BLEU, NIST, TER, and Meteor scores between each translation and the reference. While the results of this study demonstrated that the use of these metrics shows promise in improving machine translation evaluation models, the aim of our project is more in line with that of Gupta *et al.* [3], a paper published in 2015 that provides a novel

approach, called ReVal, to MT Eval purely using recurrent neural networks for an evaluation metric. The paper explores new evaluation metrics for machine translation models, which are demonstrated to have better or at least comparable correlation with human raters than popular state-of-the-art non-neural methods such as BLEU, METEOR, TERp, and DISCOTK-PARTY-TUNED, when used on their own. Their aim is to demonstrate a simpler, more compact evaluation model, as many of their comparisons make use of complex external resources for information. They experiment with a standard LSTM (Mikolov *et al.* [7], Hochreiter *et al.* [13]) and a Tree-LSTM model (Tai *et al.* [14]) to do this. The task for the non-neural baseline models and for the experimental models in ReVal is to evaluate the quality of a translation, given a reference translation and an MT model translation as inputs, and generating a similarity score from 1 to 5 as output. The goal is to generate machine translation ratings yielding maximum correlation with human raters.

We explore the aspects of Gupta *et al.* [3] that are responsible for its strengths and weaknesses in building such a model while also experimenting with techniques that have emerged since its publication in 2015, such as inter-sequence attention, as explored by Chen *et al.* [12][11], and self attention, as explored by Vaswani *et al.* [10] from 2018. Furthermore, concepts from CS 224N were employed for analyzing and improving LSTM models as used in Gupta *et al.* [3].

We referred to model architectures in Natural Language Inference (NLI) tasks, such as Chen *et al.* [12][11], as NLI similarly involves an input of two sequences and a scalar classification output. Rocktäschel *et al.* [9] demonstarted that attention is a valuable technique in elevating NLI models above the performance of simple bag-of-words baselines, which makes attention promising for our MT Eval task. State-of-the-art models in NLI are capable of interpreting and comparing two input sequences and then making a classification decision based upon that comparison, which we repurpose for the MT Eval task by comparing reference and translation sequences and rating the translation based on qualities like fluency and semantic similarity to the input.

3 Approach

3.1 Implementations

3.1.1 NIST and BLEU

We used two untrained baseline models common in automated MT Eval, BLEU and NIST. BLEU and NIST are not trained and so our BLEU and NIST models do not rely on any hand-crafted external resources.

3.1.2 Baseline LSTM

For our baseline neural implementation, we trained a bidirectional LSTM neural model without attention, similar to the one outlined in ReVal. The base neural model, henceforth referred to as Vanilla LSTM, is structured as a pair of single-layer LSTMs which separately encode the reference translation and machine translation inputs, followed by a series of weight matrix operations to combine the two encodings into a scalar similarity score. Using the two sentence encodings h_{ref} and h_{tran}, two new encodings are made, $h_+ \in \mathbb{R}^{2h\times 1}$ and $h_\times \in \mathbb{R}^{2h\times 1}$. $h_\times$ is a representation that uses the angle between the encodings of the reference translation and the machine translation, and h_+ is a representation using the absolute distance between the encodings of the inputs. Two weight matrices are trained, $W^{(+)}$ and $W^{(\times)}$, such that

$$h_s = \sigma(W^{(\times)}h_\times + W^{(+)}h_+ + b^{(h)}),\ h_s \in \mathbb{R}^{2h\times 1}. \tag{1}$$

A third weight matrix is also trained, $W^{(p)}$, such that

$$\hat{p}_\theta = softmax(W^{(p)}h_s + b^{(p)}),\ \hat{p}_\theta \in \mathbb{R}^{K\times 1}. \tag{2}$$

Lastly, $\hat{y}$ is calculated as

$$\hat{y} = r^T\hat{p}_\theta,\ r^T = [1, 2, ..., K],\ \hat{y} \in \mathbb{R}^{K\times 1}. \tag{3}$$

For this problem, $K = 5$, since the ratings are between 1 and 5. This method of calculating $\hat{y}$ from encodings was chosen to match the method used in Gupta *et al.* [3]. This method of producing $\hat{y}$ from $r^T\hat{p}_\theta$ was used in all model implementations for better comparison of results to those reported for the ReVal implementation [3].

Further implementations of variations on Vanilla LSTM used multi-layer LSTMs as encoders and were referred to as LSTM Stacked*l*, where l is the number of layers used per LSTM encoder.

3.1.3 Multi-Layer Perceptron

For more complex implementations, a Multi-Layer Perceptron (MLP) was implemented to take the LSTM encoded hidden states for the machine translation and reference sentence as inputs, and extract a series of features with each layer that culminates in a translation score distribution. A similar approach is used in some modern NLI models Chen *et al.* [12], in which an MLP, following Pooled LSTM hidden states, generates a probability distribution over entailment labels. We use mean pooling and max pooling over the two sequences as described in Section 3.1.5 so that we can propagate information from the entire sequence into the MLP. Various configurations of MLPs were implemented, with each non-output layer consisting of a linear layer with tanh activation and optional dropout and batch normalization layers. The output layer consisted only of a final linear layer of output dimension $K = 5$. MLP output, $h_{output} \in \mathbb{R}^{K\times 1}$ is then used to produce $\hat{p}_\theta$ as

$$\hat{p}_\theta = softmax(h_{output}). \tag{4}$$

Finally, $\hat{y}$ was produced using (3), with r^T the same as used for the standard model. Implementations of MLPs of different layers were referred to as MLPd, were d is the number of layers in the MLP implementation excluding the input layer.

3.1.4 Attention

Two different intermediary Attention layers were tested for processing of machine translation encoded hidden states, h_{tran}, and reference translation encoded hidden states, h_{ref}, before being passed into the MLP. These layers implemented two different types of attention: inter-sequence attention and self attention. Attention has been found to be effective at characterizing the relationships between components of sequences, identifying which states are most closely related and what other states must be paid "attention" to when examining a given state.

Inter-Sequence Attention: Used in NLI applications to attempt to predict how much the meaning of one sentence implies the meaning of the other, as explored in Chen *et al.* [12].

Takes in $h_{tran} \in \mathbb{R}^{2h\times T}$, with hidden size h and machine translation sequence length T, and $h_{ref} \in \mathbb{R}^{2h\times R}$, with reference translation sequence length R, and calculates e as

$$e = h_{tran} \times h_{ref}, \;\; e \in \mathbb{R}^{T\times R}, \tag{5}$$

which is meant to represent the relationship between each pair of hidden states between h_{tran} and h_{ref}. From there a_{tran}, attention for h_{tran} over h_{ref}, and a_{ref}, attention for h_{ref} over h_{tran}, are calculated as

$$a_{tran} = \beta h_{tran}, \;\; \beta = softmax(e, dim = R), \;\; a_{tran} \in \mathbb{R}^{2h\times R}, \tag{6}$$

$$a_{ref} = \alpha h_{ref}, \;\; \alpha = softmax(e, dim = T), \;\; a_{ref} \in \mathbb{R}^{2h\times T}. \tag{7}$$

Original sequence h_{tran} is then concatenated with a_{tran} to produce $h_{tran\_with\_att} \in \mathbb{R}^{2h\times T+R}$. Original sequence h_{ref} is then concatenated with a_{ref} to produce $h_{ref\_with\_att} \in \mathbb{R}^{2h\times T+R}$.

Inter-sequence attention was explored as a technique for characterizing how states in a machine translation sequence relate to the states in a reference translation sequence, and vice versa, creating a representation of the relationship between two sentences which could then be analyzed to approximate their semantic similarity.

Self Attention: Used in Sentiment Analysis applications to attempt to better characterize the meaning of a given sentence by taking into account how much different words in a sentence influence the meanings of each word in that sentence. The best example of the use of self attention to accomplish this can be found in Vaswani *et al.* [10], where self attention is used to encode the meanings of sentences and then use those encodings to produce decoded output translations.

Takes in $h_{tran} \in \mathbb{R}^{2h\times T}$, with hidden size h and machine translation sequence length T, and $h_{ref} \in \mathbb{R}^{2h\times R}$, with reference translation sequence length R, and calculates e_{tran} and e_{ref} where

$$e_{tran} = h_{tran} \times h_{tran}, \;\; e_{tran} \in \mathbb{R}^{2h\times T}, \tag{8}$$

$$e_{ref} = h_{ref} \times h_{ref}, \;\; e_{ref} \in \mathbb{R}^{2h \times R}, \tag{9}$$

which is meant to represent the relationship between each pair of hidden states in h_{tran} and in h_{ref}, respectively. From there, a_{tran}, attention for h_{tran} over h_{tran}, and a_{ref}, attention for h_{ref} over h_{ref}, are calculated as

$$a_{tran} = \alpha h_{tran}, \;\; \alpha = softmax(e_{tran}), \;\; a_{tran} \in \mathbb{R}^{2h \times T}, \tag{10}$$

$$a_{ref} = \beta h_{ref}, \;\; \beta = softmax(e_{ref}), \;\; a_{ref} \in \mathbb{R}^{2h \times R}. \tag{11}$$

Original sequence h_{tran} is then concatenated with a_{tran} to produce $h_{tran\_with\_att} \in \mathbb{R}^{2h \times 2T}$. Original sequence h_{ref} is then concatenated with a_{ref} to produce $h_{ref\_with\_att} \in \mathbb{R}^{2h \times 2R}$.

Self attention was explored as a technique for improving the encoded representations of two sequences of the same language. Going further, an MLP (section 3.1.3) was then used as a feed-forward network aimed at taking in these revised encodings and extracting from them a representation of their semantic similarity. This was done in an attempt to approximate the capabilities of a Transformer [10] that decodes concatenated encoded translation and reference translations into a representation of their similarity.

3.1.5 Pooling

An additional Pooling layer was implemented to process input before being passed into the MLP, incorporating information from the entire sequence and standardizing dimensions of the input to the MLP by eliminating the dimension dependent on sequence lengths. As discussed in CS224N lecture material, RNN architectures output a final hidden state that is predominantly determined by the end of the sequence, so this Pooling layer aims to counteract this deficiency of recurrent networks. The Pooling layer takes in $seq \in \mathbb{R}^{H \times T+R}$, with number of hidden states H, corresponding to concatenated h_{tran} and h_{ref} in models without Attention or concatenated $h_{tran\_with\_att}$ and $h_{ref\_with\_att}$ in models with Attention, and creates two sequences, seq_max_pooled and seq_mean_pooled, where

$$seq\_max\_pooled = maxPool(seq, dim = H), \;\; seq\_max\_pooled \in \mathbb{R}^{H \times 1}, \tag{12}$$

$$seq\_avg\_pooled = avgPool(seq, dim = H), \;\; seq\_avg\_pooled \in \mathbb{R}^{H \times 1}. \tag{13}$$

Both seq_max_pooled and seq_avg_pooled are then concatenated together to produce $seq\_pooled \in \mathbb{R}^{2H \times 1}$, which is then returned to be passed into the MLP.

3.2 Code Base

All model implementations were coded by our team. Vanilla LSTM model was implemented in pytorch based on specifications outlined in Gupta *et al.* [3]. The MLP layer was implemented based on concepts outlined in Chen *et al.* [12].

Inter-sequence attention was implemented based on concept described in Chen *et al.* [12], modified to exclude knowledge-enriched co-attention which had no application to MT Eval. Self attention was implemented based on concept described in Vaswani *et al.* [10], modified to serve as single-head instead of multi-head attention without positional encoding. Pooling was implemented based on concept described in CS224N Assignment 5 (A5), modified to include both max and mean pooling.

ReVal code base was used only in further processing of preprocessed WMT-13 and SICK datasets, to ensure matching of their dataset. `Utils.py` and `Vocab.py` were modified from CS224N Assignment 4 (A4) to operate on a single vocabulary, `translations_vocab`, rather than a source and target vocab. Run.py was adapted to train neural modesl and evaluate performance on the validation set using pearson and spearman correlation, rather than perplexity.

Our code base can be found at https://github.com/ANarcomey/DeepBLEU.

4 Experiments

4.1 Data

We make use of 4 different datasets extracted from WMT-13, 14, and 15, and the SICK dataset from Marelli *et al.* [6].

WMT-BASE Our standard dataset, referred to as WMT-BASE, consists of training, validation, and testing segments extracted from WMT-13 to match dataset used in Gupta *et al.* [3]. The WMT-BASE corpus contains $9,559$ training examples, in which each example contains a human-created reference translation, a model-generated translation to evaluate, a score given by human raters, and the number of human raters. For this corpus, the model-generated outputs are translations into English and the references are corresponding sentences in English. For the WMT data, human ratings were derived by giving each human rater a correct human-generated source sentence along with 5 translations from different models. Each rater was told to rank the translations from 1 to 5 with ties allowed, where 1 indicates most similar sentence meaning and 5 indicates least similar meaning. Allowing ties means 5 very poor translations could all have rank 5, for example. Each translation received an integer score from 1 to 5 from each worker and its final score was the average of all its ratings, so scores are not strictly integer values. The development and test sets for the WMT-BASE corpus each contain $1,000$ similarly structured examples extracted from WMT-13. The WMT-BASE data as provided in Gupta *et al.*[3] inverts the conventional WMT scale, resulting in a scale in which accurate translations receive a score of 5 and poor translations receive a score of 1. We converted all of the data obtained for WMT-14 and WMT-15 into this inverted scale, for consistency with WMT-BASE.

WMT+SICK Our second corpus, referred to as WMT+SICK, was made by combining WMT-BASE with the SICK dataset. The SICK dataset contains $4,500$ sentence pairs with scores of semantic relatedness and entailment. Relatedness was scored on a continuous scale from 1 to 5 (with 5 indicating that both sentences conveyed the exact same idea and 1 indicating both sentences conveyed entirely separate ideas). Entailment was removed from dataset through processing. The resulting data set contains $14,059$ training examples along with the same validation and test sets used in WMT-BASE.

WMT-LARGE Our largest corpus, referred to as WMT-LARGE, extends the training set WMT-BASE with data from WMT-15 and consists of similarly structured training, validation, and testing segments. We pulled raw WMT-15 data from WMT (http://www.statmt.org/wmt15/results.html) and aggregated individual ratings into a structured dataset. The added WMT-15 data consists of English references and translations into English from source languages Czech, French, German, Finnish, and Russian. This augmented training set gives models more data as well as access to a wider range of translation errors due to the varied selection of source languages.

WMT-14 We pulled raw data from WMT (http://www.statmt.org/wmt14/results.html) and aggregated individual ratings into a dataset structured like WMT-BASE. The data extracted from WMT-14 consists of English references and translations into English from Czech, French, German, Hindi, and Russian. We use this WMT-14 data to match reporting in Gupta *et al.* [3], but it is not stated exactly how the raw data was aggregated and pre-processed so our results cannot be precisely compared. We average ratings for the same reference and model translation, and filter out any translation-reference pairs with fewer than 3 ratings.

4.2 Experimental Details

Multiple different model configurations were explored combining LSTM depths, MLP depths, and use of different Attention schemes. Different configurations ultimately narrowed down to 7 model implementations, each utilizing Embedding and Hidden Size of 256, Learning Rate of 0.001, and Dropout Rate of 0.3. Defining model features are shown in Table 1 below. We recorded all of our experimental results in a spreadsheet at https://goo.gl/CHHBXz.

	Vanilla	Stacked2	MLP2	Inter-Seq + MLP2	Self + MLP2	Inter-Seq + MLP3	Self + MLP3
LSTM Layers	1	2	1	1	1	1	1
MLP Layers	N/A	N/A	2	2	2	3	3
Attention	N/A	N/A	N/A	Inter-Seq	Self	Inter-Seq	Self

Table 1: Parameters/Architecture Choices Used in Implementation of Models

4.3 Evaluation Method

The goal of the MT Eval task is to provide a metric with maximum correlation to human judgement on machine translation datasets. Given that human raters are currently considered the most accurate judges of translation quality, performance of models were judged based on rating similarity to human raters Papineni *et al.* [7]. Average Pearson and Spearman correlation were used to evaluate rating similarity over all language pairs, as done in Gupta *et al.* [3].

		NIST	BLEU	Vanilla	Stacked2	MLP2	Inter-Seq + MLP2	Self + MLP2	Inter-Seq MLP3	Self + MLP3
Train	Pearson	22.07	30.57	**72.23**	71.34	63.74	63.86	50.55	64.89	49.90
	Spearman	21.36	30.00	**71.19**	70.60	62.81	63.10	51.21	64.03	50.24
Dev	Pearson	18.89	28.94	34.61	**35.38**	34.76	34.47	35.09	35.15	35.19
	Spearman	17.69	27.30	34.51	**35.00**	34.03	33.73	34.50	34.74	34.57
Test	Pearson	19.14	26.76	37.14	37.98	**40.59**	39.80	40.17	39.18	40.36
	Spearman	18.48	25.40	36.84	37.88	**39.85**	39.03	39.62	38.33	39.74

Table 2: Pearson & Spearman Correlations For Models Trained On WMT-BASE Data

		NIST	BLEU	Vanilla	Stacked2	MLP2	Inter-Seq + MLP2	Self + MLP2	Inter-Seq + MLP3	Self + MLP3
Train	Pearson	31.11	37.32	50.16	**53.67**	49.19	50.10	51.8	48.66	48.92
	Spearman	30.36	38.02	48.77	**53.08**	50.33	50.99	52.47	49.73	50.11
Dev	Pearson	18.89	28.94	34.60	34.50	34.24	34.62	34.56	34.10	**35.14**
	Spearman	17.69	27.30	33.76	33.27	33.46	33.80	33.38	33.49	**34.33**
Test	Pearson	19.14	26.76	38.66	38.28	39.31	39.47	39.78	38.75	**40.22**
	Spearman	18.48	25.40	38.02	37.45	38.33	38.51	38.23	37.65	**39.19**

Table 3: Pearson & Spearman Correlations For Models Trained On WMT+SICK Data

		NIST	BLEU	Vanilla	Stacked2	MLP2	Inter-Seq + MLP2	Self + MLP2	Inter-Seq + MLP3	Self + MLP3
Train	Pearson	16.37	28.87	**49.39**	49.25	39.15	41.40	37.93	43.58	43.48
	Spearman	15.97	27.25	**48.55**	48.33	38.14	40.60	36.93	42.82	42.54
Dev	Pearson	18.90	28.94	35.12	**37.19**	37.07	35.30	36.73	36.60	36.71
	Spearman	17.69	27.30	33.64	**35.48**	35.82	33.70	35.40	34.99	35.32
Test	Pearson	19.14	26.76	37.92	36.60	**38.03**	37.96	37.85	36.77	36.82
	Spearman	18.48	25.40	36.86	35.88	**37.11**	36.50	36.18	35.63	35.82

Table 4: Pearson & Spearman Correlations For Models Trained On WMT-LARGE Data

		NIST	BLEU	Vanilla	Stacked2	MLP2	Inter-Seq + MLP2	Self + MLP2	Inter-Seq + MLP3	Self + MLP3
Test	Pearson	19.84	27.31	23.91	26.78	26.77	27.31	**27.83**	27.43	26.03
	Spearman	19.23	24.37	24.83	25.90	26.52	27.08	**27.71**	27.06	25.35

Table 5: Pearson & Spearman Correlations For WMT-LARGE Models Tested On WMT-14

5 Analysis

We see that for all trained models, scores on WMT-14 are significantly lower than both validation and test performance on WMT-LARGE. This is most likely because WMT-14 data, in the way we aggregated and minimally processed it, comes from a different distribution than the training data derived from WMT-13. Untrained metrics NIST and BLEU perform similarly, but trained models suffer because they were trained on a different data distribution. We also see that the trained models largely do not benefit from the larger training set of WMT-LARGE, which may be a result of early-stopping criteria ending training too early, since the much larger dataset requires more time to fit than WMT-BASE.

5.1 Qualitative Analysis

In analyzing the errors each model makes, we saw that untrained metrics BLEU and NIST most closely predict human ratings when the reference and translation sentences are almost exactly identical or one segment is identical and the other segment is very incorrect. Here "most closely predict" means minimizing `(human_score - model_score)/human_score`. Given that BLEU and NIST operate on n-gram pattern matching, they assign near perfect scores for near identical translations just like human raters, and will assign middle-range scores for sentences where a segment is identical, with better scores for longer identical segments. BLEU scores are most distant from human ratings when the translation uses a different phrase to indicate the same meaning, such as "You have to pay extremely careful attention, says Wentzler" and "Wentzler adds: vigilance is required." In this common failure case of BLEU and NIST, the reference and translation use very different phrases that carry identical meaning.

The trained models offer no guarantee that identical translations and references will receive a perfect score, but the models are capable of combining information in the sequences in much more complex ways than matching n-grams. The Self+MLP3 model is qualitatively invariant to the order of words: the reference-translation pair which most closely predicted the human rating in the test set was "Similar conclusions are also reached by the joint audit from the Czech and German auditors" and "An audit conducted jointly by inspectors, Czechs, and Germans also suggests similar conclusions." These two sentences convey nearly identical meanings with the same set of words but broken into a completely different set of n-grams. Self-attention, however, allows our model to attend to any part of the sequence and capture the similar semantic meaning, capturing how the words in each sentence relate to each other other regardless of their relative orderings.

An additional strength, general to all of our neural models due to their use of learned word embeddings, is that they can easily capture synonyms and give a fair rating to translations that choose different but synonymous words and phrases.

5.2 Model Comparisons

Vanilla is a simple one layer LSTM network, using the parameters described in section 4.3 and architecture as described in section 3.1.2, which consistently resulted in the highest correlations, both pearson and spearman, seen across Tables 2-4, on training data across the three different WMT datasets that were tested. Comparing this performance to validation and test correlations, it is clear that Vanilla, being the simplest model tested, strongly overfit to the data and was not able to generalize elsewhere.

To improve upon Vanilla, Stacked2 was implemented, which added a second layer to the basic LSTM configuration and was implemented according to Table 1. Stacked2 performed similarly to the vanilla LSTM across datasets, with mixed results on WMT-LARGE. Overall, Stacked2 was one of the strongest performers on validation performance on WMT-BASE data and WMT-LARGE, with correlations shown in bold in Table 2 and Table 4. Stacked2 was able to significantly outperform Vanilla most likely due to its ability to add an extra layer of abstraction. Stacked2 is also, in greater context of our experiments, a low-complexity model which explains why it would be less likely to overfit to the datasets.

As a comparison to Stacked2, MLP2 was designed to be a one layer LSTM that feeds into a two layer MLP, described further in section 3.1.3. MLP2 continued to produce a more generalizable model, seen in its much lower training performance and similar validation correlations across all datasets compared to Stacked2. MLP2 outperformed the previous models on all test correlations. MLP2 was a better generalizer, likely in part due to its integration of dropout between layers for regularization. However, it is worth noting that an early stopping method was used in training that could lead more complex models to stop training before fitting more completely.

MLP2 and MLP3, a variation of MLP2 with an extra layer in the MLP (illustrated in Table 1), were both implemented with inter-sequence attention and self attention to experiment with the benefits of each kind of attention and model size. Overall, MLP models implemented with self attention saw lower train correlations compared to those implemented with inter-sequence attention across Tables 2-4, while maintaining similar validation and test correlations. This is most likely an attribute of the early stopping mechanism used. Changing the MLP to utilize three layers from two layers first tested showed mixed results, but resulted in the strongest results on WMT+SICK data in combination with self attention, showing that adding more model flexibility and complexity beyond two layers was mostly not needed, and might be only useful with more data. On WMT-BASE and WMT+SICK data self attention performed better than inter-sequence attention, while there were more mixed results in WMT-LARGE, as both versions were beaten by an MLP with no attention. This suggests that, in our datasets and parameter settings, self-attention generates a more meaningful semantic encoding of each sentence by having each sequence attend to itself, and these, more powerful, encodings yield better performance than the cross-sequence information transfer offered by inter-sequence attention. Inter-sequence attention, in this context, could be providing more noise than information, particularly since these datasets are not extremely large, and could require more model tuning to exhibit stronger performance over models with self attention or without any attention implemented.

Comparing all models, we can conclude that MLP2 outperformed other models on test correlations for WMT datasets not trained with the additional SICK data, while Self+MLP3 outperformed other models on test correlations when trained on SICK data. This could be attributed to SICK data following a different distribution than WMT data sources, and that the simple MLP2 model is not able to learn the features in this data as well, but its simplicity is rewarded when the training and test data follow a more similar distribution as seen in Table 2 and Table 4.

6 Conclusion

We see an interesting result that the strengths and weaknesses of n-gram-based metrics and neural evaluation models are complementary: BLEU performs strongly when reference and translation have extended segments of near identical phrases, while our neural models are better at extracting semantics and are invariant to issues like phrase order and synonyms that would confuse a metric such as BLEU. Additionally, using a two layer Multi-layer Perceptron proved useful in reducing overfitting in simpler one and two layer LSTM networks and led to some of the highest pearson and spearman correlations. Self attention led to better input encodings than an LSTM with Multi-layer Perceptron, posting the highest correlations on WMT+SICK data. Inter-sequence attention also showed comparable test performance to self-attention, but most likely is lagging due to small dataset and model sizes. With more data, we would experiment with longer training and deeper networks as, ultimately, our models were limited by the amount of human rated machine translation data available. Despite this, our results all outperform NIST and BLEU correlations, showing the promise of neural network architectures to provide evaluation that measures closer to human judgment, without needing any external training resources.

Additional Information

We are pursuing the custom project and have been assigned Anand Dhoot (anandd@stanford.edu) as our teaching assistant point of contact. This project is not shared with another class and is unrelated to any of the listed projects proposed by Stanford AI Labs.

References

[1] Chen, Zhiming, et al. "Improving machine translation quality estimation with neural network features." *Proceedings of the Second Conference on Machine Translation.* 2017.

[2] Domhan, Tobias. "How much attention do you need? a granular analysis of neural machine translation architectures." *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers).* Vol. 1. 2018.

[3] Gupta, Rohit, Constantin Orasan, and Josef van Genabith. "Reval: A simple and effective machine translation evaluation metric based on recurrent neural networks." *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing.* 2015.

[4] Guzmán, Francisco, et al. "Machine translation evaluation with neural networks." *Computer Speech & Language 45* (2017): 180-200.

[5] Kim, Hyun, and Jong-Hyeok Lee. "A recurrent neural networks approach for estimating the quality of machine translation output." *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.* 2016.

[6] Marelli, Marco, et al. "Semeval-2014 task 1: Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment." *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014).* 2014.

[7] Mikolov, Tomáš, et al. "Recurrent neural network based language model." *Eleventh annual conference of the international speech communication association.* 2010.

[8] Papineni, Kishore, et al. "BLEU: a method for automatic evaluation of machine translation." *Proceedings of the 40th annual meeting on association for computational linguistics.* Association for Computational Linguistics, 2002.

[9] Rocktäschel, Tim, et al. "Reasoning about entailment with neural attention." *arXiv preprint arXiv:1509.06664* (2015).

[10] Vaswani, Ashish, et al. "Attention is all you need." *Advances in Neural Information Processing Systems.* 2017.

[11] Chen, Qian, et al. "Enhanced LSTM for natural language inference." *Proceedings of the 55th annual meeting on association for computational linguistics.* Association for Computational Linguistics, 2017.

[12] Chen, Qian, et al. "Natural language inference with external knowledge." *6th International Conference on Learning Representations.* International Conference on Learning Representations, 2018.

[13] Hochreiter, Sepp, et al. "Long short-term memory." *Neural computation Volume 9 Issue 8*, 1997.

[14] Tai, Kai Sheng et al. "Improved semantic representations from tree-structured long short-term memory networks." *arXiv preprint arXiv:1503.00075* (2015).