Semantic Textual Similarity

Pranav Goyal
IIIT Hyderabad
pranav.goyal@research.iiit.ac.in

Saransh Rajput
IIIT Hyderabad
saransh.rajput@research.iiit.ac.in

Evaluating the semantic similarity between text data is an open research problem in the field of Natural Language Processing. Paraphrase Identification and Semantic Similarity are two different, yet very well related tasks in NLP. The versatility of natural language makes it very challenging to define rule based methods for the task of paraphrase identification. To address this issue, various methods have been proposed over the years for the task of paraphrase identification. In this report, we will explore, implement and evaluate some of these methods. We also make some modifications to the proposed methods and evaluate the different models on the MSRP (Microsoft Research Paraphrase Corpus) and QQP (Quora Question Pairs) datasets.

1 Introduction

Paraphrase Identification and Semantic Similarity are important tasks that can be used as features to improve many other NLP tasks such as Information Retrieval, Machine Translation Evaluation, Question Answering, Text Classification etc. Paraphrase Identification is a binary classification task. Given a pair of sentences, the system is required to decide whether the sentences carry the same meaning or not and classify them either as paraphrases or not paraphrase of each other.

Firstly, we explore a bilateral multi-perspective matching (BiMPM) model for our task. Given two sentences P and Q, the model first encodes them with a bidirectional LSTM. The two encoded sentences are then matched in both directions $P \rightarrow Q$ and $P \leftarrow Q$. In both the directions, each time step of Q is matched against all the time steps of P from multiple perspectives using three different matching algorithms. Another BiLSTM layer then aggregates the matching results into a matching vector of fixed length. Based on this matching vector, the decision is made through a fully connected layer.

Secondly, we explore a multiway attention network for our task. Multiway Attention Network model (or MwAN) uses multiple attention functions to match the two sentences at the word level. Specifically, it uses concatenation attention function, bilinear attention function, minus attention function and dot attention function. The last two functions are directly applied between words which enables word level relation building. The two sentences are first encoded using a bidirectional RNN to obtain contextual word representation of the words in two sentences. Then the four matching functions mentioned above are applied to match the sentences at word level. The matching information is first aggregated by a bidirectional RNN along with all the words. Then the outputs of all the attention functions are combined by applying attention mechanism. Another bidirectional RNN is then applied to pass through the combined representation to aggregate mixture matching information. Finally, attention pooling mechanism is applied to the matching information for a fix length vector and is fed to the multilayer perceptron for the final classification decision.

Lastly to test the state of the art transformers architecture, we also fine-tuned pretrained BERT model for the task of paraphrase detection. The final model is identical to the original BERT model except the last layer of dense output to predict the label based on the encoding of the [cls] token. [1]

2 Related Work

Paraphrase identification has proven useful for a wide variety of natural language processing applications (Madnani and Dorr, 2010) [Madnani and Dorr, 2010]. The ACL Wiki gives an excellent summary of the state-of-the-art paraphrase identification techniques; this shows how much effort researchers did to automatically detecting paraphrases. The different approaches can be categorized into supervised methods, i.e. (Madnani et al., 2012) [Madnani et al., 2012], (Socher et al., 2011) and (Wan et al., 2006), and unsupervised methods, i.e. (Fernando and Stevenson, 2008), (Hassan and Adviser-Mihalcea, 2011) and (Islam and Inkpen, 2009). Previous works use the Microsoft Research Paraphrase Corpus (MSRP) dataset (Dolan et al., 2004)

[1] Training notebooks can be found at `https://github.com/Pranav174/ParaphraseDetection`

[Dolan and Brockett, 2004] that is obtained by extracting sentences from news sources on the web. A few recent studies have highlighted the potentiality and importance of developing paraphrase (Zanzotto et al., 2011) and (Xu et al., 2013) and semantic similarity techniques (Guo and Diab, 2012) [Agirre et al., 2013].
Recently, deep learning and neural network models achieve promising results in modeling sentence pairs. Basically, two sentences are encoded into sentence vectors with a neural network encoder, and then the relationship between two sentences was decided solely based on the two sentence vectors [Bowman et al., 2015; Yang et al., 2015; Tan et al., 2016]. Rocktaschel ¨ et al. [2015] use the attention-based technique to improve the performance of LSTM-based recurrent neural network. Wang and Jiang [2016b] [Wang and Jiang, 2016] propose match-LSTM for the natural language inference that tries to match the current word in the hypothesis with an attention-weighted representation of the premise calculated by the word-by-word attention. Wang et al. [2017b] propose BiMPM that matches two sentences by a bilateral matching with attention mechanism in multiple perspectives. Yin et al. [Yin et al., 2016] [2016] propose the attention-based CNN for the paraphrase identification, natural language inference, and answer sentence selection. Wang and Jiang [2016a] use the CNN to aggregate the matching information for the answer sentence selection.

3 Task Definition

Both the datasets we tested on are related to paraphrase detection which is a binary classification problem. Two sentences are paraphrase, if they entail the same proposition. Each entry of the dataset contains a pair of sentences and binary label representing they are paraphrase or not. Formally this is represented by the triple (P, Q, y) where $P = (p_1, p_2, ..., p_m)$ and $Q = (q_1, q_2, ..., q_n)$ are the two sentences where p_i and q_j are the words of the two sentences. $y \in \{0, 1\}$ marks if P and Q are paraphrase or not. The task is to predict y^* given P, Q represented as $y^* = \text{argmax}_{y \in \{0,1\}} Pr(y|P, Q)$. The methods in the following section differ in how they measure this probability.

4 Methods

For the project, we have implemented three methods for the paraphrase detection classification model. Section 4.1 describes the implementation of Bilateral Multi Perspective Matching model [Wang et al., 2017]. Section 4.2 describes the implementation of Multiway attention mechanism [Tan et al., 2018]. Finally we also fine-tuned BERT [Devlin et al., 2018] pretrained model based on the transformer architecture, which is described in Section 4.3.

4.1 Bilateral Multi-perspective matching

The model is similar to Siamese networks in generating the contextual representations of the input sentences. The model however varies in the prediction layers as it employs unique matching algorithm and aggregation to make the final prediction.

4.1.1 Matching Function

The Bilateral Multi-perspective matching is unique in the way it compares contextual representation of the two sentences. While traditional algorithms relied on naive cosine similarity, the model proposes comparing using the following function f_m:

$$m = f_m(v_1, v_2; W)$$

where v_1, v_2 are d-dimensional vectors, $W \in l * d$ is a trainable parameter. The resulting matching vector $m = m_1, m_2, ..., m_k$ is computed as

$$m_i = cosine(W_i \circ v_1, W_i \circ v_2)$$

Here the $\circ$ symbol represents the element multiplication and W_i represents the row i of W. This matching function essentially fir converts each d-dimensional vectors to be compared to the $l * d$ space while the row wise cosine functions results in the final l-dimensional matching vector.

4.1.2 Model Architecture

The input to the model are two sentence P ans Q to be compared. Using pretrained embeddings [Pennington et al., 2014] the words are represented with fixed size vectors from the embeddings space. The two sentences are thus represented as $[p_1, p_2, ..., p_M]$ and $[q_1, q_2, ..., q_N]$ where p_i, q_j are embeddings vector of word i, j of the two sentences.

Then Bidirectional LSTM is used to encode contextual representation of each timestamp for both the sentences. $h^p = [\overrightarrow{h^p}, \overleftarrow{h^p}]$ and $h^q = [\overrightarrow{h^q}, \overleftarrow{h^q}]$

$$\overrightarrow{h_i^p} = \overrightarrow{LSTM}(\overrightarrow{h_{i-1}^p}, p_i) \quad i \in [1, M]$$
$$\overleftarrow{h_i^p} = \overleftarrow{LSTM}(\overleftarrow{h_{i+1}^p}, p_i) \quad i \in [1, M]$$

Similarly the sentence Q is encoded with the same LSTM.

The matching layer is then used to generate l-dimensional matching vectors by comparing the d-dimensional contextual representations results from the bidirectional LSTM. The paper [Tan et al., 2018] implemented four distinct matching algorithm, however in our experiment we found one three of them to help improve the results. Hence our final architecture uses three matching layers, Full-matching, Max-pooling Matching and Max-attentive Matching to generate three matching vectors m_i^{full}, m_i^{max}, m_i^{att} for each timestamp of a sentence by comparing it to the timestamps of other sentence.

Full-Matching computes the matching by comparing the current timestamp to the final output of the other sentence. For the sentence P for example, the matching vector m_i^{full} : $[\overrightarrow{m_i^{full}}, \overleftarrow{m_i^{full}}]$ will be computed as:

$$\overrightarrow{m_i^{full}} = f_m(\overrightarrow{h_i^p}, \overrightarrow{h_N^q}; W^1)$$
$$\overleftarrow{m_i^{full}} = f_m(\overleftarrow{h_i^p}, \overleftarrow{h_1^q}; W^2)$$

Max pooling matching compares the current timestamp with all the timestamps of the other sentence and only chooses the max entry for each perspective from each matching. The *max* in the following operation defines the element wise maximum.

$$\overrightarrow{m_i^{max}} = \max_{1 \le j \le N} f_m(\overrightarrow{h_i^p}, \overrightarrow{h_j^q}; W^3)$$
$$\overleftarrow{m_i^{max}} = \max_{1 \le j \le N} f_m(\overleftarrow{h_i^p}, \overleftarrow{h_j^q}; W^4)$$

Max-attentive matching first calculates cosine similarity among every timestamp, and computes the matching between the vector with the highest cosine similarity:

$$\overrightarrow{h_i^{best}} = \underset{j \in \{0,N\}}{\operatorname{argmax}} \, cosine(\overrightarrow{h_i^p}, \overrightarrow{h_j^q})$$
$$\overleftarrow{h_i^{best}} = \underset{j \in \{0,N\}}{\operatorname{argmax}} \, cosine(\overleftarrow{h_i^p}, \overleftarrow{h_j^q})$$
$$\overrightarrow{m_i^{att}} = f_m(\overrightarrow{h_i^p}, \overrightarrow{h_i^{best}}; W^5)$$
$$\overleftarrow{m_i^{att}} = f_m(\overleftarrow{h_i^p}, \overleftarrow{h_i^{best}}; W^6)$$

The matching vectors are concatenation for each timestamp $m_i : [m_i^{full}, m_i^{max}, m_i^{att}]$. These matching vectors of both the sentences are again passed through bidirectional LSTMs to result in fixed length vector from the final output of LSTMs in both directions for both the sentences.

This fixed sized vector is then passed through a 2-layer feed-forward network to make the final prediction. The intermediate layers of the feed-forward network have ReLu activation while the final output of size 2 (for the binary labels of paraphrase detection) has the softmax activation.

4.2 Multiway Attention

Firstly, we obtain contextual word representation for the two sentences P and Q using a bi directional RNN. Then, for every word pair from P and Q, we obtain four matching scores using different attention functions. Then, we match the two sentences inside each attention function and then combine the matching information from all the functions. We use a bi directional RNN for aggregation. Finally, we use attention pooling to aggregate the matching information and apply multilayer perceptron classifier for the final decision.

4.2.1 Model Architecture

Using pertained embeddings, the input sentences $P = [W_t^p]_{t=1}^M$ and $Q = [W_t^q]_{t=1}^N$ are represented as vectors of fixed size. A bi directional GRU is then used encode contextual representation for both the sentences and we get the new representations $h_1^p, h_2^p, ...h_m^p$ and $h_1^q, h_2^q, ...h_n^q$.

$$\overrightarrow{h_t^q} = GRU(\overrightarrow{h_{t-1}^q}, w_t^q)$$
$$\overleftarrow{h_t^q} = GRU(\overleftarrow{h_{t+1}^q}, w_t^q)$$
$$h_t^q = [\overrightarrow{h_t^q}, \overleftarrow{h_t^q}]$$

Meanwhile, another GRU is used to get generate representations for P similarly. Then, we use multiple attention functions to compare the two vectors.

Concat Attention:

$$s_j^t = v_c^T tanh(W_c^1 h_j^q + W_c^2 h_t^p)$$
$$a_i^t = exp(s_i^t) / \Sigma_{j=1}^N exp(s_j^t)$$
$$q_t^c = \Sigma_{i=1}^N a_i^t h_i^q$$

Bilinear Attention:

$$s_j^t = h_j^{q^T} W_b h_t^p$$
$$a_i^t = exp(s_i^t) / \Sigma_{j=1}^N exp(s_j^t)$$
$$q_t^b = \Sigma_{i=1}^N a_i^t h_i^q$$

Minus Attention:

$$s_j^t = v_m^T tanh(W_m(h_j^q - h_t^p))$$
$$a_i^t = exp(s_i^t) / \Sigma_{j=1}^N exp(s_j^t)$$
$$q_t^m = \Sigma_{i=1}^N a_i^t h_i^q$$

Dot Attention:

$$s_j^t = v_d^T tanh(W_d(h_j^q \cdot h_t^p))$$
$$a_i^t = exp(s_i^t) / \Sigma_{j=1}^N exp(s_j^t)$$
$$q_t^d = \Sigma_{i=1}^N a_i^t h_i^q$$

Where, h_t^p represents P at position t, h^q represents Q. Therefore, using the above four functions, at each position t of P, the word can match Q to obtain the corresponding weighted sum representations. We then aggregate the matching information from multiple attention functions. For each position t, we concatenate the word representation, h_t^p in P with corresponding attention q_t^k of Q. Below is an example of concat attention:

$$\begin{aligned}
x_t^c &= [q_t^c, h_t^p] \\
g_t &= sigmoid(W_g x_t^c) \\
x_t^{c*} &= g_t \cdot x_t^c \\
\overrightarrow{h_t^c} &= GRU(\overrightarrow{h_{t-1}^c}, x_t^{c*}) \\
\overleftarrow{h_t^c} &= GRU(\overrightarrow{h_{t+1}^c}, x_t^{c*}) \\
h_t^c &= [\overrightarrow{h_t^c}, \overleftarrow{h_t^c}]
\end{aligned}$$

Mixed aggregation is then used to combine the matching information:

$$\begin{aligned}
s_j &= v^T tanh(W^1 h_t^j + W^2 v^a)(j = c, d, m, b) \\
a_i &= exp(s_i)/\Sigma^{(j=c,d,m,b)} exp(s_j) \\
x_t &= \Sigma(i = c, d, m, b) a_i h_i^t
\end{aligned}$$

x_t is then fed to the bi directional GRU to aggregate the matching information from the different attention functions.

$$\begin{aligned}
\overrightarrow{h_t^o} &= GRU(\overrightarrow{h_{t-1}^o}, x_t) \\
\overleftarrow{h_t^o} &= GRU(\overleftarrow{h_{t+1}^o}, x_t) \\
h_t^o &= [\overrightarrow{h_t^o}, \overleftarrow{h_t^o}]
\end{aligned}$$

After aggregating the information from multiway matching,the resulting representations of all positions in P are converted to fixed length vectors.

$$\begin{aligned}
s_j &= v^T tanh(W_q^1 h_j^q + W_q^2 v^q) \\
a_i &= exp(s_i)/\Sigma_{j=1}^N exp(s_j) \\
r^q &= \Sigma_{i=1}^N a_i h_i^q
\end{aligned}$$

This representation r^q is then used to select the information in the matching vectors.

$$\begin{aligned}
s_j &= v^T tanh(W_p^1 h_j^o + W_p^2 r^q) \\
a_i &= exp(s_i)/\Sigma_{j=1}^M exp(s_j) \\
r^p &= \Sigma_{i=1}^M a_i h_i^o
\end{aligned}$$

Finally, r^p is fed to the MLP classifier that makes the final decision.

4.3 Bert Fine-tuning

The previous models both relied on recurrent neural network, but most of the state of the art results have been given by transformers based models which their much faster processing can be pretrained on large datasets. For this part we used pretrained BERT (Bidirectional Encoder Representations from Transformers) from tensorflow hub repository. The model takes in input the sentence pair in the format: $[cls], [p_1], [p_2]...[p_M], [sep], [q_1], [q_2]...[q_N]$, where $[p_i]$ and $[q_j]$ are tokens from the two sentences P and Q, while $[sep]$ token is used to seperate the two sentences. $[cls]$ is the token for classification task and in the original model, it was trained to predict whether the second sentence follows the first.

As BERT is an auto-encoding model, we use the output of the $[cls]$ to make predictions of paraphrase detection. At the output of $[cls]$ we add a dropout layer for regularization followed by a single output layer of size out with softmax activation to make the binary predictions. The input to the model is the input sequence as describe above along with the mask and type sequence which distinguishes the padding and sentence class of tokens respectively.

5 Experiment

The following section describes the environment for each of the model implementation described above. While the section 5.2 and 5.3 state our analysis of the results for both the datasets.

5.1 Experiment Settings

For the embedding layer in BiMPM and multiway attention model, we used glove 300-dimensional pretrained embeddings trained on 6 Billion tokens. [2]. Out of vocabulary words were replaced with the 'unk' token. Due to memory limits, for some of the models with lots of parameters had sentence length threshold to allow batch processing. All the models were trained for the binary classification task of paraphrase detection using the cross-entropy loss function. Both the datasets, QQP and MSRP, were randomly divided into training and validation sets with the ratio 4:1

For the BiMPM model, similar to the [Wang et al., 2017], we used 20 perspectives for the matching vector as it gave the best results. Both the bidirectional LSTMs had a hidden state size of 64. The hidden layers in the feed-forward network to make the final predictions were of size 128 and 64 giving a total of about 330,000 trainable parameters. Dropouts of keep rate 0.9 were used between the dense layers and in LSTM. Adam optimizer was used during the training phase and batch size of 16 and 32 was used for MSRP and QQP dataset respectively to account for larger sentences length in MSRP.

For the Multiway Attention, 300 dimensional pre trained Glove embeddings were used without update for training. The out of vocabulary words (OOV) were represented using zero vectores. Hidden vector length was set to 150 and dropout value = 0.3 was used between layers. AdaDelta optimizer was used with step size = 10 and initial learning rate = 1. Both the datasets, QQP and MSRP, were randomly divided into training and validation sets with the ratio 4:1.

[2]http://nlp.stanford.edu/data/glove.6B.zip

Model	Validation Accuracy	
	QQP	MSRP
BiMPM	85.28%	75.52%
Multiway Attention	83.47%	72.56%
Bert	90.10%	84.12%

Table 1. Final results

Batch size was set to 64 and the model was trained for 12 epochs.

The Bert-finetuning model used the pretrained BERT-base model with 12 stacked transformers with representations of 768 dimensions.[3]. This model had been trained for English on the Wikipedia and BookCorpus. The final output layer was initialized normal distribution of standard deviation of 0.02 and a custom adam optimizer was used for the training with the initial learning rate 2e-5.

5.2 Results

Table 1 lists the final validation accuracy attained by all the implemented models for the two datasets. The Quara Question Pairs dataset, QQP, had just over 400,000 sentence pairs with an average of 10 words per sentence. While the Microsoft Research Paraphrase, MSRP, dataset being much smaller had only about 5800 sentence pairs with an average of 20 words per sentence.

While training on limited memory, we were able to validation accuracy close to the original models which had used ensemble techniques to get the best results. Our BiMPM model achieved 85.28% validation accuracy on QQP dataset while the original paper [Wang et al., 2017] stated a accuracy of 88.69%, however it had only considered 10000 sentence pairs in validation set which is only 2.5% of the whole dataset. The MWaN paper had stated accuracy of 89.60% which also used Elmo contextual embeddings, while we were able to achieve 83.47% with static glove embeddings. Also due to memory limitations, we used glove embeddings of least vocab size, improving which should have definitely improved the results bringing them closer to the originally stated results in the papers. We believe sentence of length will not have any effect on the accuracy of all the models as MWaN and BERT uses attention mechanism, and BiMPM compares all the timestamps making sure proper phrase alignment.

Contrary to what we expected, in our experiment, the BiMPM gave better results compared to the Multiway attention network which we believe doesn't necessarily mean it is a better model, as it is likely a result of hyperparameters and stronger checks like k-fold validation would be more ideal to make any conclusion. The BERT model surely outperforms both the models by a large margin which highlights the advantage of transformers, heavy pretraining and transfer learning. However the bert model did have much much more trainable parameters than the other two, which meant more training time and would have suffered underfitting if not pre-trained.

None of the original papers had tested their models on the MSRP dataset, which does give lower accuracy mainly because of the much smaller amount of data to be trained and longer sentences on average. The finetuned BERT model outperformed our implemented models by large margin because it was pretrained on a large corpus and has a much better understanding of the language as it was simultaneously pretrained on the auto-encoding language modeling task.

6 Conclusion

In this project, we implemented multiple models to tackle the task of semantic textual similarity, specifically the binary classification task of paraphrase detection. The models are unique in their way of comparing sentence pairs at the semantic level. BiMPM uses custom multi-perspective matching function to compare contextual representations of token from the two sentences. MwAN employs attention mechanism to extract useful matching information for a word from all the words in the other sentence. Finally the pre-trained BERT models brings in the improvement of transformers and greatly boosts results with transfer learning. Experiments show that evel while training on limited memory, all the models are able to achieve close to state of the art performance for the paraphrase detection task while the Pre-trained BERT model gets the best result due to better language understanding.

References

[Agirre et al., 2013] Agirre, E., Cer, D., Diab, M., Gonzalez-Agirre, A., and Guo, W. (2013). * sem 2013 shared task: Semantic textual similarity. In *Second joint conference on lexical and computational semantics (* SEM), volume 1: proceedings of the Main conference and the shared task: semantic textual similarity*, pages 32–43.

[Devlin et al., 2018] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

[Dolan and Brockett, 2004] Dolan, W. B. and Brockett, C. (2004). Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

[Madnani and Dorr, 2010] Madnani, N. and Dorr, B. J. (2010). Generating phrasal and sentential paraphrases: A survey of data-driven methods. *Computational Linguistics*.

[Madnani et al., 2012] Madnani, N., Tetreault, J., and Chodorow, M. (2012). Re-examining machine translation metrics for paraphrase identification. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 182–190.

[Pennington et al., 2014] Pennington, J., Socher, R., and

[3] `https://tfhub.dev/tensorflow/bert_en_uncased_L-12_H-768_A-12/3`

Manning, C. D. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

[Tan et al., 2018] Tan, C., Wei, F., Wang, W., Lv, W., and Zhou, M. (2018). Multiway attention networks for modeling sentence pairs. In *IJCAI*, pages 4411–4417.

[Wang and Jiang, 2016] Wang, S. and Jiang, J. (2016). Machine comprehension using match-lstm and answer pointer. *arXiv preprint arXiv:1608.07905*.

[Wang et al., 2017] Wang, Z., Hamza, W., and Florian, R. (2017). Bilateral multi-perspective matching for natural language sentences. *arXiv preprint arXiv:1702.03814*.

[Yin et al., 2016] Yin, W., Ebert, S., and Schütze, H. (2016). Attention-based convolutional neural network for machine comprehension. *arXiv preprint arXiv:1602.04341*.