LMML at SemEval-2020 Task 7: Siamese Transformers for Rating Humor in Edited News Headlines

Pramodith Ballapuram
Independent Researcher, Atlanta GA, USA
pramodith1@gmail.com

Abstract

This paper contains a description of my solution to the problem statement of SemEval 2020: Assessing the Funniness of Edited News Headlines. I propose a Siamese Transformer based approach, coupled with a Global Attention mechanism that makes use of contextual embeddings and focus words, to generate important features that are fed to a 2 layer perceptron to rate the funniness of the edited headline. I detail various experiments to show the performance of the system. The proposed approach outperforms a baseline Bi-LSTM architecture and finished 5th (out of 49 teams) in sub-task 1 and 4th (out of 32 teams) in sub-task 2 of the competition and was the best non-ensemble model in both tasks.

1 Introduction

Machines that can recognize and understand humor can prove to be invaluable in applications like chat bots, personal digital assistants in order to make communication more fun and humane, story and script generation to provide comical relief or even in recommendation engines that can provide better recommendations to people on what Netflix stand-up show they can watch next. At the end of the day we all live to laugh don't we? Surprisingly enough, there hasn't been much work in the field of AI along these lines. The organizers of SemEval-2020 Task 7 (Hossain et al., 2020a) released a new dataset in the English language and created a couple of sub tasks that can hopefully take us a step forward in creating machines that better understand humor.

HAHA - Humor Analysis based on Human Annotation (Castro et al.,), (Chiruzzo et al., 2019) started in 2018 was a similar task, where the dataset comprised of Spanish Tweets and participants were asked to classify the tweets as either a joke or not and also rate the jokes on a scale of 0-5. In their overview paper, the authors describe that teams that used Transformer Based Models such as BERT (Devlin et al., 2019) and ULMFit (Howard and Ruder, 2018), along with techniques like slanted learning rates, domain specific language modeling etc. proved valuable. My approach takes inspiration from this and focuses on Transformer Models. SemEval 2017 Task-6 (Potash et al., 2017) consisted of sub-tasks asking participants to rank the funniness of tweets that had a specific HashTag. They mention that some of the top teams used Siamese Networks (Bromley et al., 1994), (Koch, 2015) based approaches.

He et. al., (2019) introduce the concepts of *local surprisal* and *global surprisal* their work suggests that for a pun to be considered good, the pun-word must have high agreeableness in the local context of where it occurs in the sentence but a lower level of agreeableness in the global context of the entire sentence. I take inspiration from this idea as well. My hypothesis is that an edit in a sentence can be funny if the edited sentence continues to make sense and also have a bit of a twist. My model attempts to model the global agreeableness of the replaced and original word in the news headline. The remainder of the paper will be as follows. In section 2, I will briefly describe the task. Section 3 will detail the System design and architecture. Section 4 will state the implementation details. Section 5 will cover details of all experiments, results and interesting findings. It will be followed by a small section containing my final thoughts.

Proceedings of the 14th International Workshop on Semantic Evaluation, pages 1026–1032
Barcelona, Spain (Online), December 12, 2020.

Original News Headline	Edit
EU says summit with Turkey provides no answers to <concerns/ >	stuffing
The GOP just ca n't <escape/ > the 80s	remember

Table 1: Sample Data from sub-task 1.

2 Task Review

Table 1 gives an example of entries in the provided dataset(Hossain et al., 2019). The words within the angular brackets are substituted with the corresponding word in the edit column. The edited headline would read *The GOP just can't remember the 80s.* The score of the edited headline is a value between 0-3. It's worthy to note that there is only one edit per sample in the dataset. Sub-task 1 dealt with predicting the funniness score given the original headline and the edit. In Sub-task 2 we were given the same original headline, but there would be 2 different edited headlines and the edit could be made anywhere in the original headline. Participants were asked to classify which of the two edits was funnier. In my approach I create one model that can predict the funniness score of a given sample and I use the same model to compute the results of Sub-Task 2, by finding the difference between the funniness scores of the two edited headlines.

The organizers of the task were kind enough to provide additional training samples a few months into the competition (Hossain et al., 2020b). I make use of both the initial dataset and the additional dataset provided, to train my models.

3 System Design and Architecture

3.1 Dataset and Preprocessing

From hereon the original headline will be referred to as X_{org} and the edited headline as X_{edit}. I will refer to the word being replaced from the original headline and the edit word as **focus words**. My approach centers around the idea that a model should learn features that are conditioned on the focus words, or make use of the focus words either directly or indirectly. Wu and He (2019) show that in the task of relationship classification between two entities adding special tokens between the span of the entities leads to an improved performance. Similarly in this approach the token $<$ is added before and after the word to be replaced in X_{org} and ^ (symbol for exponent) is added before and after the edit word in X_{edit}.

3.2 Siamese Networks

Siamese Networks (Bromley et al., 1994), (Koch, 2015) are twin networks that share the same parameters but each of the twins receive distinct inputs. Given that, for this task, we have an original and edited headline, I hypothesized that extracting features from both the headlines would be beneficial since humans require context of what the original sentence is, to deem an altered sentence to be funny. Each of X_{org} and X_{edit} is passed to one of the twins in the Siamese Network. Both of them comprise the tokens $\mathbf{X}_{org/edit}$=[$x_{org/edit_0}$,$x_{org/edit_1}$, ...$x_{org/edit_i}$,...$x_{org/edit_n}$].

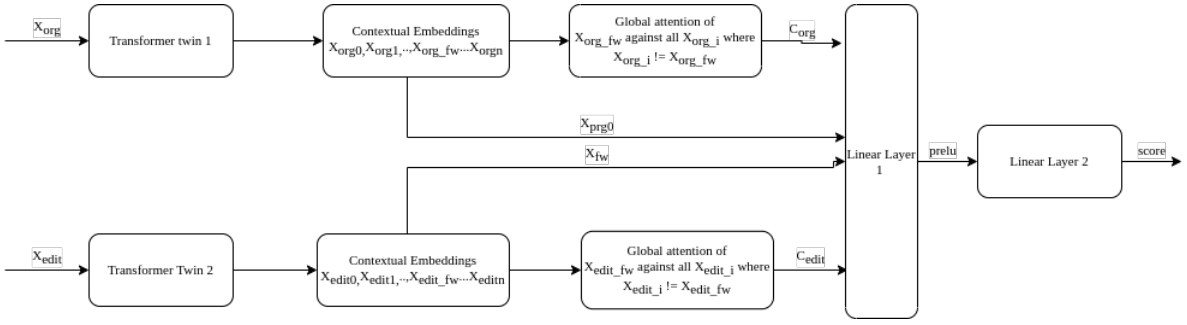

Figure 1. Model architecture. In the figure U = **concat([**C_{edit},C_{org},S_{org},E_{edit}**])**

3.3 Model Architecture

Over the last couple of years Transformer based Architectures (Vaswani et al., 2017) such as BERT (Devlin et al., 2019), XLNet (Yang et al., 2019), Roberta (Liu et al., 2019) etc. have become extremely popular. These models are pre-trained on language modeling tasks using large amounts of data and as a result are capable of providing contextual token embeddings that can be fine-tuned to achieve state of the art results in various downstream NLP tasks such as Question Answering, Sentiment Analysis, Natural Language Inference tasks etc. I experiment with different transformer models that act as the Siamese twins in order to obtain contextual token embeddings and choose the best one.

The token embeddings are used to create a set of useful features that are passed to a two layer perceptron to predict the funniness score of the edited news headline. The features extracted are described in the following subsection. Figure 1 depicts the model architecture.

3.4 Features

It's been observed that different layers in a Neural Network capture different kinds of syntactic and semantic information (Yosinski et al., 2014) . Sun et al., (2019) observed an improved performance on classification tasks by concatenating the token embeddings of BERT with the embeddings from the penultimate layer i.e. layer 11. I experimented with concatenating the outputs of different layers and observed the same. I concatenate the final token embeddings with those from the 11^{th} layer of the Transformer. Please note that from hereon all features are concatenations of the Transformer's token embedding and its penultimate layer.

Most BERT (Devlin et al., 2019) based architectures that are fine tuned towards a downstream task tend to use the first token i.e the [CLS] token as a vector that summarizes the entire input sequence in essence. I make use of the first token from the Transformer twins for both X_{org} and X_{edit}, these two vectors will be referred to as S_{org} and S_{edit}. I also extract the vectors that correspond to the focus words. This is done easily thanks to the special tokens $<$ and $\hat{}$ that demarcate these words. In the event that the word spans more than one token the mean of all the tokens between the special tokens is computed to obtain a single vector. These two word vectors will be referred by E_{org} and E_{edit} from hereon.

$$E_{org/edit} = \textbf{mean}([E_{org/edit_1}, E_{org/edit_i} ..., E_{org/edit_n}])$$

For each of X_{org} and X_{edit} I compute context vectors C_{org} and C_{edit}. $C_{org/edit}$ is computed as the result of a Global Attention Mechanism (Bahdanau et al., 2015), (Luong et al., 2015). The idea is that the $C_{org/edit}$[1] would contain information about how well the replaced word and the edited word fit into the headline. Below $\cdot$ is the dot product operation.

In section 5, I describe experiments where different combinations of the features which will be referred to as U are passed as input to a two layer perceptron. **prelu** (He et al., 2015) is used as the activation function after the first linear layer.

$$attention\_scores_{org/edit} = softmax(V \cdot (WE_{org/edit})) \tag{1}$$

$$C_{org/edit} = attention\_scores_{edit/org} \cdot V \quad where\, V = (X_{org/edit_i} \notin E_{org/edit_i}) \tag{2}$$

Symbol	Meaning
X	Input Sequence
S	Sequence Representation Token
E	Focus word Representation Token
C	Global Attention Vector

Table 2: Summary of symbols used.

A final list of notations used and what they stand for is summarized in Table 2.

[1] In (1) W is a learned parameter

3.5 Baseline Architecture

I present two baseline architectures; In the first one I make use of a 2-layered Bidirectional LSTM (Hochreiter and Schmidhuber, 1997), initialized with Glove (Pennington et al., 2014) Embeddings of size 300 followed by a self-attention layer. The mean of the output from the self-attention layer is then passed as input to a 2 layer perceptron with **tanh** as the activation function. The input to this model is only the **edited news headline** i.e. X_{edit}. The first linear layer projects a vector of size 600 to 128.

The second baseline is a Siamese 2-layered Bidirectional LSTM with a single head self-attention layer on top of it. I pass U to the 2 layer perceptron where U is defined as below. The output from the self-attention layer are treated as the token embeddings. Both of these models perform similarly with a RMSE of 0.581 on the validation set and 0.577 on the test set.

$$U = \textbf{concat}([C_{edit}, C_{org}, S_{edit}, E_{edit}])$$

4 Implementation Details

All experiments mentioned below were conducted making use of Pytorch[2], the HuggingFace[3] (Wolf et al., 2019) library was used for transformer architectures. Spacy[4] for Glove (Pennington et al., 2014). Batch size is fixed at 64, I use Adam (Kingma and Ba, 2014) with a learning rate of 2e-5 as my optimizer. For all the experiments the best validation score and the test score of the model corresponding to the best validation score are reported. I use a linear warm-up scheduler (Howard and Ruder, 2018) with the warm-up period equal to 10% of the total number of steps. The model is trained for 5 epochs and the model with the best validation score is used at test time. A dropout of 0.1 is applied to all the transformer architectures and 0.3 to both the linear layers. I clip the norm of the gradients to 1.0. Unless stated otherwise all of the transformer models were *uncased* apart from Roberta, for Roberta I use the *roberta-base* model. All the transformer models are pre-trained models and are not trained from scratch. For all experiments reported in this the random seed is fixed to be 12. The max sequence length is fixed to 50. I make my code publicly available[5] in the form of a jupyter notebook.

5 Experiments and Results

5.1 Identifying the best token embeddings

Neural Architecture	Best RMSE on Validation Set	RMSE on Test Set
Bi-LSTM + Self Attn.	0.581	0.577
DistilBERT	0.531	0.530
BERT	0.525	0.525
Roberta	**0.516**	**0.516**

Table 3: RMSE using different models to generate Token Embeddings.

I experiment with using DistilBERT (Sanh et al., 2019), BERT, Roberta for producing the token embeddings along with the baseline Bi-LSTM model (non-siamese) mentioned above. The embeddings are used to create U=**concat**([C_{edit},C_{org},S_{edit},E_{edit}]) which is passed to the 2 layer perceptron.

The results are shown in the table 2. The LSTM based model has the poorest performance. Roberta shows the best performance with 0.516 on both the validation set and the test set. All experiments listed from hereon make use of the Roberta model to obtain the token embeddings.

5.2 Experiments for finding the best set of features

. The next set of experiments show the impact of using different combinations of the features explained in section 3.4 to obtain the best U vector which is passed to the 2 layer perceptron. The model with the

[2] https://pytorch.org/
[3] https://huggingface.co/
[4] https://spacy.io/
[5] https://github.com/pramodith/Humor

Composition of U	Best RMSE on Validation Set	RMSE on Test Set
C_{edit},C_{org}	0.5256	0.5285
S_{edit},S_{org}	0.5250	0.5239
C_{edit},C_{org},S_{edit}	0.5245	0.5204
C_{edit},C_{org},E_{edit}	0.5175	0.5210
C_{edit},C_{org},S_{edit},E_{edit}	**0.5166**	0.5169
C_{edit},C_{org},S_{org},E_{edit}	0.5218	0.5218
C_{edit},C_{org},E_{org},E_{edit}	0.5224	**0.5160**
S_{edit},S_{org},E_{org},E_{edit}	0.5225	0.5237

Table 4: RMSE using different features.

best RMSE on the test set is with the features C_{edit},C_{org},E_{org},E_{edit}. Since the model with the features C_{edit},C_{org},S_{edit},E_{edit} gives the most consistent results on the test and validation set all experiments following this section use the concatenation of these features as U. From these results it's not too clear that one feature or one set of features is more important than the other.

5.3 Importance of Siamese Architecture

In order to verify that using a Siamese architecture is advantageous. I train a non-siamese[6] network in which X_{org} and X_{edit} are concatenated together. In order for Roberta to recognize X_{org} and X_{edit} as a text pair. They're concatenated as follows:

$$X_{concat} = < s > + X_{org} + < /s > + < /s > + X_{edit} + < /s >$$

X_{concat} is passed to the network to obtain the features mentioned above in section 3.4. The best RMSE of this model on the validation set is 0.5194 and the corresponding test set RMSE is 0.5247. Despite the Siamese architecture doing moderately better than this model it's not too convincing that the Siamese architecture is helping the model improve its performance. Here S_{edit} is the first token from Roberta.

5.4 Fine-Tuning the Language Model

Sun et al., (2019) and Chiruzzo et al., (2019) mention that fine-tuning the language model of transformer based architectures against the task specific data improves performance of the models. I fine-tune the Roberta model for masked language modeling against the original news headlines. The language model is trained for 2 epochs with a batch size of 32. The model yields an RMSE of 0.5212 on the validation set and 0.5194 on the test set. It's observed that there is no notable improvement.

6 Conclusion

My final submissions to the competition for Sub-task 1 and Sub-task 2 resulted in a 5th and 4th place finish in the competition and was the best non-ensemble model in both tasks amongst the final submissions. The final submissions actually corresponded to a Siamese BERT based architecture, where U = **concat**([C_{edit},C_{org},S_{edit},E_{edit}] which I obtained by performing a search on random seeds. I didn't experiment with the Roberta model at that point of time after the release of the extra dataset. The submitted model achieved a validation score of 0.5186. and a test score of 0.5202. For Sub-task 2 I observed that the model with the best accuracy did not necessarily need to be the same as the one that had the lowest RMSE for Sub-task 1, which is quite surprising. In Sub-task 2 my official submission obtained an accuracy of 0.6465 on the validation set and 0.6468 on the test set.

In conclusion, this paper presents a Siamese Transformer based approach, that makes use of features that center around the focus words and their impact against other tokens. From the experiments shown above its tough to conclude if the Siamese architecture or if any of the features in particular are responsible for an improved performance of the model, it looks like just following the best practices of training Transformer networks can yield very good results. In the future I would like to probe the model to better

[6]this experiment was conducted on a different GPU from that of 5.2 to accommodate longer sequence lengths.

understand why it deems one sentence to be funnier than another, it would also be interesting to study if a model that can generate jokes can also grade how funny a joke is and vice versa.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. 1994. Signature verification using a" siamese" time delay neural network. In *Advances in neural information processing systems*, pages 737–744.

Santiago Castro, Luis Chiruzzo, and Aiala Rosá. Overview of the haha task: Humor analysis based on human annotation at ibereval 2018.

Luis Chiruzzo, S Castro, Mathias Etcheverry, Diego Garat, Juan José Prada, and Aiala Rosá. 2019. Overview of haha at iberlef 2019: Humor analysis based on human annotation. In *Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2019). CEUR Workshop Proceedings, CEUR-WS, Bilbao, Spain (9 2019)*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June. Association for Computational Linguistics.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Nabil Hossain, John Krumm, and Michael Gamon. 2019. "president vows to cut <taxes> hair": Dataset and analysis of creative text editing for humorous headlines. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 133–142, Minneapolis, Minnesota, June. Association for Computational Linguistics.

Nabil Hossain, John Krumm, Michael Gamon, and Henry Kautz. 2020a. Semeval-2020 Task 7: Assessing humor in edited news headlines. In *Proceedings of International Workshop on Semantic Evaluation (SemEval-2020)*, Barcelona, Spain.

Nabil Hossain, John Krumm, Tanvir Sajed, and Henry Kautz. 2020b. Stimulating creativity with funlines: A case study of humor generation in headlines. In *Proceedings of ACL 2020, System Demonstrations*, Seattle, Washington, July. Association for Computational Linguistics.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *ACL*. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.

Gregory Koch. 2015. Siamese neural networks for one-shot image recognition.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal, September. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October. Association for Computational Linguistics.

Peter Potash, Alexey Romanov, and Anna Rumshisky. 2017. SemEval-2017 task 6: #HashtagWars: Learning a sense of humor. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 49–57, Vancouver, Canada, August. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5754–5764.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. How transferable are features in deep neural networks? In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 3320–3328, Cambridge, MA, USA. MIT Press.