HumorHunter at SemEval-2021 Task 7: Humor and Offense Recognition with Disentangled Attention

Yubo Xie, Junze Li, and Pearl Pu
School of Computer and Communication Sciences
École Polytechnique Fédérale de Lausanne, Switzerland
{yubo.xie, junze.li, pearl.pu}@epfl.ch

Abstract

In this paper, we describe our system submitted to SemEval 2021 Task 7: HaHackathon: Detecting and Rating Humor and Offense. The task aims at predicting whether the given text is humorous, the average humor rating given by the annotators, and whether the humor rating is controversial. In addition, the task also involves predicting how offensive the text is. Our approach adopts the DeBERTa architecture with disentangled attention mechanism, where the attention scores between words are calculated based on their content vectors and relative position vectors. We also took advantage of the pre-trained language models and fine-tuned the DeBERTa model on all the four subtasks. We experimented with several BERT-like structures and found that the large DeBERTa model generally performs better. During the evaluation phase, our system achieved an F-score of 0.9480 on subtask 1a, an RMSE of 0.5510 on subtask 1b, an F-score of 0.4764 on subtask 1c, and an RMSE of 0.4230 on subtask 2a (rank 3 on the leaderboard).

1 Introduction

Humor, appreciated by people with almost any age or cultural background, is perhaps one of the most fascinating human behaviors. Besides providing entertainment, humor can also be beneficial to mental health by serving as a moderator of life stress (Lefcourt and Martin, 2012), and plays an important role in regulating human-human interaction. As Reeves and Nass (1996) have pointed out, people respond to computers in the same way as they do to real people, which indicates that modeling humor computationally could bring positive effects in human-computer interaction (Nijholt et al., 2003). Despite being universal to human beings, the extent to which people find something humorous varies according to one's age, gender, or socio-economic status, making humor a highly subjective experience. This poses many challenges to the field of computational humor. Abundant research has been done to enable computers to automatically decide whether humor is entailed in a given piece of text. Early work (Mihalcea and Strapparava, 2005; Mihalcea et al., 2010) uses manually engineered features to recognize humor in text, while more recent work (Chen and Soo, 2018; Weller and Seppi, 2019) adopts deep learning approaches and pre-trained language models.

SemEval 2021 Task 7: HaHackathon: Detecting and Rating Humor and Offense (Meaney et al., 2021) aims at detecting and rating humor as well as offense in short English text. There are four subtasks involved. Subtask 1a is a binary classification task, predicting if the text would be considered humorous for an average user. Subtask 1b is a regression task and predicts the humor rating of the text if it is considered humorours. Subtask 1c is again a binary classification task and predicts whether the humor rating is controversial, whose ground-truth label is decided based on the variance of the annotators' ratings. This task also involves offense detection. Subtask 2a predicts how offensive the text is for a general user. All the regression subtasks have scores ranging from 0 to 5.

In this paper, we present our system submitted to SemEval 2021 Task 7. We followed the architecture of DeBERTa (He et al., 2020), an improved version of BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) by using two novel techniques: disentangled attention and decoding enhanced masking. We mainly relied on the disentangled attention mechanism, where the attention weights of the input words are calculated based on their content vectors and relative position vectors. For the four subtasks, we used the same base structure and the only difference is at the output layer, where the classification tasks have two out-

Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021), pages 275–280
Bangkok, Thailand (online), August 5–6, 2021. ©2021 Association for Computational Linguistics

put units and the regression tasks only have one. The pre-trained DeBERTa model has two variants that differ in size. During the evaluation phase, the large version achieved an F-score of 0.9480 on subtask 1a, an RMSE of 0.5510 on subtask 1b, an F-score of 0.4764 on subtask 1c, and an RMSE of 0.4230 on subtask 2a (rank 3 on the leaderboard). In addition, we also experimented with the BERT and RoBERTa models as our baselines, and found them generally under-performed by DeBERTa. Our code has been made publicly available.[1]

2 Related Work

Mihalcea and Strapparava (2005) used several human-centric features such as alliteration and synonym to recognize humor in one-liners. Mihalcea et al. (2010) approached the problem by calculating the semantic relatedness between the set-up and the punchline. Morales and Zhai (2017) proposed a generative language model and leveraged background text sources to identify humor in Yelp reviews. Liu et al. (2018) proposed to model sentiment association between elementary discourse units and designed features based on discourse relations. Xie et al. (2020) calculated the uncertainty and surprisal of the set-up and the punchline according to the incongruity humor theory, which were found useful in humor recognition. Recent work also developed neural network based models to recognize humor in text. Chen and Lee (2017) and Chen and Soo (2018) adopted convolutional neural networks, while Weller and Seppi (2019) used a Transformer architecture.

3 Dataset

SemEval 2021 Task 7 provides three datasets: the training set (8,000), the validation set (1,000), and the final test set (1,000). Table 1 summarizes the statistics of the three datasets, and lists the respective information of humorous (positive) and non-humorous (negative) examples. Each example is a piece of English text accompanied by four features: `is_humor` (subtask 1a), `humor_rating` (subtask 1b), `humor_controversy` (subtask 1c), and `offense_rating` (subtask 2a). For subtask 1b and 2a, the labels range from 0 to 5. Table 2 gives two samples, one being humorous and the other non-humorous.

[1] https://github.com/yuboxie/semeval-2021-task-7

	Train	Validation	Test
# positive	4,932	632	615
Avg # tokens	24.48	22.04	26.14
# negative	3,068	368	385
Avg # tokens	25.95	26.12	29.36
# total	8,000	1,000	1,000
Avg # tokens	25.05	23.54	27.38

Table 1: Statistics of the provided datasets. Here the respective information of humorous (positive) and non-humorous (negative) examples are also listed.

For subtask 2a, whose goal is to predict the offense rating of the input text, we also visualize top 200 frequent unigrams for examples with offense rating ≥ 2 and < 2, respectively, illustrated as two word clouds (Figure 1a and Figure 1b). As we can observe, Figure 1a contains words that are expected to appear in offensive text, usually targeting at a specific group of people (e.g., "black", "gay", "chinese", "muslim", etc.), while Figure 1b contains more ordinary words, which generally do not imply offense.

4 System Overview

With the increasingly powerful neural networks such as the Transformer (Vaswani et al., 2017), the performance on many downstream NLP tasks has been greatly improved by fine-tuning large pre-trained language models on smaller but task-specific datasets. Traditional Transformer-based language models such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) use absolute positional embeddings in the input layer, which are added up with the word embeddings and serve as the input to the following Transformer layers. The self attention weights between the tokens are calculated solely based on their hidden representations. However, recent work (Shaw et al., 2018; Dai et al., 2019) has shown that relative position representations are more effective for NLP tasks.

Our system leverages the disentangled attention mechanism from the DeBERTa model (He et al., 2020), where the attention weights between input tokens are calculated based on their content vectors as well as their relative positions. As shown in Figure 2, for each Transformer layer, $\boldsymbol{H}_i$'s are the input representations from last layer, and $\boldsymbol{H}_i^o$'s are the output representations after applying the self attention. Instead of using absolute positional

text	is_humor	humor_rating	humor_controversy	offense_rating
Here's a FedEx joke - actually, you'll get it tomorrow.	1	3.21	0	0
When humans make mistakes, it doesn't mean they're evil, it means they're human.	0	-	-	0.1

Table 2: Two samples from the training set.

(a) Word cloud of examples with offense rating ≥ 2

(b) Word cloud of examples with offense rating < 2

Figure 1: Word clouds of the data according to the offense rating.

embeddings at the input layer, we create a relative positional embedding table, which is shared across all layers, to represent the relative position between token i and token j. More specifically, the index of the relative position between token i and j is defined as

$$\delta(i,j) = \begin{cases} 0 & \text{if } i-j \leq -k, \\ 2k-1 & \text{if } i-j \geq k, \\ i-j+k & \text{otherwise,} \end{cases} \quad (1)$$

where k is the maximum distance we consider. Similar to normal Transformer attention mechanism, the content representations $\boldsymbol{H}$ and the relative position representations $\boldsymbol{P} \in \mathbb{R}^{2k \times d}$ are transformed to queries, keys, and values:

$$\begin{aligned} \boldsymbol{Q}^c &= \boldsymbol{H}\boldsymbol{W}_q^c, \boldsymbol{K}^c = \boldsymbol{H}\boldsymbol{W}_k^c, \boldsymbol{V}^c = \boldsymbol{H}\boldsymbol{W}_v^c, \\ \boldsymbol{Q}^p &= \boldsymbol{P}\boldsymbol{W}_q^p, \boldsymbol{K}^p = \boldsymbol{P}\boldsymbol{W}_k^p. \end{aligned} \quad (2)$$

Then, the attention weight $\boldsymbol{A}_{ij}$ between token i and token j are calculated as follow:

$$\boldsymbol{A}_{ij} = \boldsymbol{Q}_i^c \boldsymbol{K}_j^{cT} + \boldsymbol{Q}_i^c \boldsymbol{K}_{\delta(i,j)}^{p\,T} + \boldsymbol{K}_j^c \boldsymbol{Q}_{\delta(j,i)}^{p\,T}. \quad (3)$$

When aggregating the input representations $\boldsymbol{H}$, we apply a scaling factor $1/\sqrt{3d}$ to obtain the output representations $\boldsymbol{H}^o$:

$$\boldsymbol{H}^o = \text{softmax}\left(\frac{\boldsymbol{A}}{\sqrt{3d}}\right)\boldsymbol{V}^c. \quad (4)$$

For subtask 1a and 1c, which are binary classification tasks, we use softmax output layer and cross entropy loss. For subtask 1b and 2a, which are regression tasks, we use mean square error as the loss function. Otherwise, the base structure is the same, and we initialize the model with the pre-trained DeBERTa weights.

5 Experimental Setup

We evaluated and compared our system with several baselines on the provided dataset, whose statistics are provided in Section 3. In this section, we are going to elaborate the setup of our experiment.

5.1 Baselines

In our experiment, we consider the following approaches as our baselines:

- **Bag of words (BoW)**. In this approach, we neglect the order of the input tokens, and simply add up the word embeddings of the tokens to form the vector representation of the input text. We implemented logistic regression for subtask 1a and 1c, and linear regression for subtask 1b and 2a, using the 300d GloVe word embeddings (Pennington et al., 2014).

- **Convolutional neural network (CNN)**. Convolutional neural networks have been widely adopted in computer vision and image recognition. When applied to NLP tasks, the input is a 2D matrix with each row being the word embeddings of the respective token, and the convolution is operated along the rows, with a

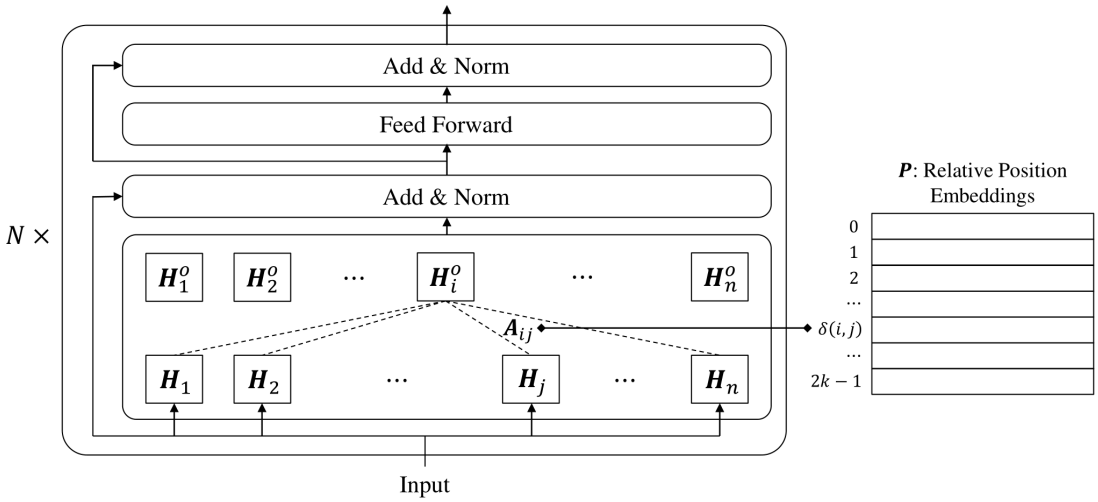

Figure 2: An illustration of the model architecture.

	Subtask 1a				Subtask 1b
	Precision	Recall	F-Score	Accuracy	RMSE
BoW	0.7884 / 0.8141	0.7712 / 0.8067	0.7778 / 0.8099	0.7990 / 0.8220	0.5433 / 0.5617
CNN	0.8289 / 0.8524	0.8221 / 0.8485	0.8252 / 0.8503	0.8390 / 0.8590	0.6661 / 0.6399
Bi-LSTM	0.8340 / 0.8620	0.8438 / 0.8610	0.8381 / 0.8615	0.8470 / 0.8690	0.5645 / 0.5504
BERT (base)	0.9061 / 0.9119	**0.9462 / 0.9593**	0.9257 / 0.9350	0.9040 / 0.9180	0.4994 / 0.5402
BERT (large)	0.9246 / 0.9442	0.9320 / 0.9350	0.9283 / 0.9395	0.9090 / 0.9260	0.5099 / 0.5500
RoBERTa (base)	0.9398 / 0.9469	0.9383 / 0.9577	0.9390 / 0.9523	0.9230 / 0.9410	0.5259 / 0.6320
RoBERTa (large)	0.9597 / 0.9515	0.9415 / 0.9561	0.9505 / **0.9538**	0.9380 / **0.9430**	0.4994 / **0.5326**
Our system (base)	0.9463 / 0.9521	0.9209 / 0.9382	0.9334 / 0.9451	0.9170 / 0.9330	0.4978 / 0.5456
Our system (large)	**0.9707 / 0.9604**	0.9446 / 0.9463	**0.9575** / 0.9533	**0.9470 / 0.9430**	**0.4923** / 0.5538

Table 3: Performance of subtask 1a and 1b on the validation / test set.

fixed window size. We follow the CNN model in the work of Chen and Lee (2017), which includes an extra highway layer before the final fully connected layer, allowing shortcut connections with gate functions.

- **Bidirectional long short-term memory (Bi-LSTM)**. LSTM (Hochreiter and Schmidhuber, 1997) has shown to perform quite well in handling sequential inputs, making it suitable for many NLP tasks. Bidirectional LSTM incorporates two LSTMs, one in the forward direction and the other in the backward direction, thus better modeling the context. In this approach, we use a Bi-LSTM with hidden size 200 and one hidden layer.

- **BERT**. BERT (Devlin et al., 2019) is a deep bidirectional Transformer pre-trained on BooksCorpus and English Wikipedia, with two training objectives: (1) masked language model, where some of the input tokens are randomly masked and are to be recovered by the model; (2) next sentence prediction, where the goal is to predict if the input second sentence follows the first one. By fine-tuning the pre-trained BERT, the performance of a wide range of NLP tasks can be largely improved, compared with previous models such as LSTMs.

- **RoBERTa**. RoBERTa (Liu et al., 2019) is an optimized version of BERT, which was trained on bigger datasets and longer sequences. In addition, the next sentence prediction objective was removed, which was found to slightly improve the performance of downstream tasks. RoBERTa reportedly achieved better results than BERT on benchmarks such as GLUE, RACE and SQuAD.

5.2 Implementation

All the Transformer-based models in the experiment have two variants that differ in model size. The base version has 12 Transformer layers, 768 hidden units, and 12 multiheads. The large version has 24 Transformer layers, 1024 hidden units,

	Subtask 1c				Subtask 2a
	Precision	Recall	F-Score	Accuracy	RMSE
BoW	**0.5539 / 0.5585**	0.5539 / 0.5584	0.5538 / 0.5584	0.5538 / **0.5626**	0.9418 / 0.7207
CNN	0.5052 / 0.5084	0.5051 / 0.5084	0.5012 / 0.5055	0.5032 / 0.5057	0.8238 / 0.6913
Bi-LSTM	0.4907 / 0.4908	0.4907 / 0.4919	0.4905 / 0.4817	0.4905 / 0.5089	0.7825 / 0.6666
BERT (base)	0.5455 / 0.4924	0.5649 / 0.4659	0.5550 / 0.4788	**0.5585** / 0.5398	0.5681 / 0.5228
BERT (large)	0.5013 / 0.4891	0.6071 / 0.5627	0.5492 / 0.5233	0.5142 / 0.5350	0.5550 / 0.5022
RoBERTa (base)	0.4873 / 0.4537	**1.0000 / 1.0000**	0.6553 / **0.6242**	0.4873 / 0.4537	0.5634 / 0.5310
RoBERTa (large)	0.5027 / 0.4695	0.9221 / 0.9104	0.6506 / 0.6195	0.5174 / 0.4927	0.5013 / 0.4566
Our system (base)	0.4873 / 0.4537	**1.0000 / 1.0000**	0.6553 / **0.6242**	0.4873 / 0.4537	0.5484 / 0.4653
Our system (large)	0.4943 / 0.4574	0.9903 / 0.9032	**0.6595** / 0.6072	0.5016 / 0.4699	**0.4794 / 0.4516**

Table 4: Performance of subtask 1c and 2a on the validation / test set.

and 16 multiheads. We used the Adam optimizer (Kingma and Ba, 2015) with learning rate 5×10^{-6}, and a batch size of 16. All the models were trained until the minimum loss value is reached on the validation set.

5.3 Evaluation Metrics

For classification tasks 1a and 1c, we use precision, recall, F-score, and accuracy as the evaluation metrics. For regression tasks 1b and 2a, we use the root mean square error as the evaluation metric:

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{n=1}^{N}(\hat{y}_n - y_n)^2}, \qquad (5)$$

where $\hat{y}_n$ is the predicted value, and y_n is the ground-truth value.

6 Results

The performance of our system and the baselines is shown in Table 3 (subtask 1a and 1b) and Table 4 (subtask 1c and 2a). We show the performance scores on both the validation and the test set. Generally speaking, the large version of our system performs quite well on all the four subtasks, compared with the other models. It can also be observed that, Transformer-based models always outperform the traditional methods by a large margin, except for subtask 1c, where all the models perform poorly and similarly. We conjecture this is because humor controversy is itself a highly subjective task, which is difficult even for humans. We also observe that large version of BERT-like models are generally better than their base counterparts, which is natural since larger models with more parameters usually bring better performance.

Table 5 gives the confusion matrix of our system on the test set in subtask 1a. We can see that in both positive and negative cases, the system performs quite well and makes only few errors. We manually examined some cases where our system makes a false prediction, and found that when our system predicts humorous but the ground-truth is non-humorous, the input text usually contains a question, e.g.,

		Ground-truth		
		P	N	Total
Predicted	P	582	24	606
	N	33	361	394
	Total	615	385	1,000

Table 5: The confusion matrix of our system (large) on the test set (subtask 1a). P: Positive, N: Negative.

> *There are 2 kinds of families on Thanksgiving. Which one are you?*

We infer this is because most of the humorous examples in the training set contains a question, usually followed by a short answer serving as the punchline.

7 Conclusion

In this paper, we describe our system submitted to SemEval 2021 Task 7. We adopted the disentangled attention mechanism from the DeBERTa model, and participated in all the four subtasks. During the evaluation phase, we got a rank of 3 on the leaderboard for subtask 2a. For future work, we would like to combine human-centric features with the current architecture using the disentangled attention mechanism, and develop a hybrid model. In addition, we plan to expand the provided dataset with extra jokes from various sources such as Reddit forums, hoping to further improve the performance of our system.

References

Lei Chen and Chong Min Lee. 2017. Convolutional neural network for humor recognition. *CoRR*, abs/1702.02584.

Peng-Yu Chen and Von-Wun Soo. 2018. Humor recognition using deep learning. In *Proceedings of NAACL-HLT 2018, Volume 2 (Short Papers)*, pages 113–117.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G. Carbonell, Quoc Viet Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings ACL 2019*, pages 2978–2988.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT 2019*, pages 4171–4186.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. DeBERTa: Decoding-enhanced BERT with disentangled attention. *CoRR*, abs/2006.03654.

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

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of ICLR 2015*.

Herbert M Lefcourt and Rod A Martin. 2012. *Humor and life stress: Antidote to adversity*. Springer Science & Business Media.

Lizhen Liu, Donghai Zhang, and Wei Song. 2018. Modeling sentiment association in discourse for humor recognition. In *Proceedings of ACL 2018, Volume 2 (Short Papers)*, pages 586–591.

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.

J.A. Meaney, Steven R. Wilson, Luis Chiruzzo, Adam Lopez, and Walid Magdy. 2021. SemEval 2021 task 7, HaHackathon, detecting and rating humor and offense. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*.

Rada Mihalcea and Carlo Strapparava. 2005. Making computers laugh: Investigations in automatic humor recognition. In *Proceedings of HLT/EMNLP 2005*, pages 531–538.

Rada Mihalcea, Carlo Strapparava, and Stephen G. Pulman. 2010. Computational models for incongruity detection in humour. In *Proceedings of CICLing 2010*, volume 6008 of *Lecture Notes in Computer Science*, pages 364–374.

Alex Morales and Chengxiang Zhai. 2017. Identifying humor in reviews using background text sources. In *Proceedings of EMNLP 2017*, pages 492–501.

Anton Nijholt, Oliviero Stock, Alan J. Dix, and John Morkes. 2003. Humor modeling in the interface. In *CHI 2003 Extended Abstracts on Human Factors in Computing Systems*, pages 1050–1051.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of EMNLP 2014*, pages 1532–1543.

Byron Reeves and Clifford Nass. 1996. *The media equation: How people treat computers, television, and new media like real people and places*. Cambridge University Press.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *Proceedings of NAACL-HLT 2018*, pages 464–468.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of NeurIPS 2017*, pages 5998–6008.

Orion Weller and Kevin D. Seppi. 2019. Humor detection: A transformer gets the last laugh. In *Proceedings of EMNLP-IJCNLP 2019*, pages 3619–3623.

Yubo Xie, Junze Li, and Pearl Pu. 2020. Uncertainty and surprisal jointly deliver the punchline: Exploiting incongruity-based features for humor recognition. *CoRR*, abs/2012.12007.