Duluth at SemEval-2020 Task 7:
Using Surprise as a Key to Unlock Humorous Headlines

Shuning Jin[1]   **Yue Yin**[2]   **XianE Tang**[2]   **Ted Pedersen**[2]

[1]Rutgers University   [2]University of Minnesota Duluth

shuning.jin@rutgers.edu, {yinxx325,tang0611,tpederse}@d.umn.edu

Abstract

We use pretrained transformer-based language models in SemEval-2020 Task 7: Assessing the Funniness of Edited News Headlines. Inspired by the incongruity theory of humor, we use a contrastive approach to capture the surprise in the edited headlines. In the official evaluation, our system gets 0.531 RMSE in Subtask 1, 11th among 49 submissions. In Subtask 2, our system gets 0.632 accuracy, 9th among 32 submissions.

1 Introduction

Humor detection is a challenging problem in natural language processing. SemEval-2020 Task 7 (Hossain et al., 2020a)[1] focuses on detecting humor in English news headlines with *micro-edits*. Specifically, the edited headlines have one selected word or entity that is replaced by editors, which are then graded by the degree of funniness. Accurate scoring of the funniness from micro-edits can serve as a footstone of humorous text generation (Hossain et al., 2020a).

Inspired by the incongruity theory (Veale, 2004; Morreall, 2016), we believe that contrast and surprise is a key ingredient of humor. We instantiate this intuition with a contrastive framework. We then systematically compare three widely used models: CBOW, BERT (Devlin et al., 2019), and RoBERTa (Liu et al., 2019), providing a benchmark for this task. Our best system, based on RoBERTa, achieves compelling performance for both subtasks. Our code is available on GitHub.[2]

2 Related Work

Early humor recognition systems are mostly based on traditional machine learning methods, such as support vector machine, decision tree, Naive Bayes, and k-nearest neighbors (Castro et al., 2016). Besides, an n-gram language model shows good performance (Yan and Pedersen, 2017) in learning a sense of humor from tweets. Yet n-gram models are limited to a small number of context words.

Pretrained language models based on Transformer (Vaswani et al., 2017) can obtain contextual information of a whole sentence. Among this family, BERT has been used to assess the humor in tweets and jokes (Mao and Liu, 2019; Weller and Seppi, 2019). Enlightened by these recent advances, we use BERT to judge the funniness of edited news headlines. We additionally experiment with RoBERTa, a robustly optimized variant of BERT.

Lastly, several works also attempt to explicitly model incongruity and surprise of humourous text, focusing on homophonic puns. Kao et al. (2016) formalizes incongruity as a mixed effect of ambiguity and distinctiveness, quantified by entropy and Kullback-Leibler divergence. He et al. (2019) proposes a local-global surprisal measure based on the log-likelihood ratio, to assess whether a sentence is a pun. However, we focus on a broader definition of humor, and formulate incongruity as an input pair to a dual encoder framework.

---

[1]Competition page: https://competitions.codalab.org/competitions/20970
[2]Code: https://github.com/dora-tang/SemEval-2020-Task-7

Task	ID	Original Headline	Edit	Score	Label
Subtask 1	33210	California and **President Trump** are going to war with each other	monkeys	1.8	/
	1664	What if **sociologist** had as much influence as economists?	donkeys	2.8	/
Subtask 2	9934	Chibok girls reunited with **families**	smartphones	1.8	1
	14279	Chibok girls reunited with **families**	cats	0.0	
	10920	Gene Cernan, last **astronaut** on the Moon, dies at 82	dancer	1.2	1
	9866	Gene Cernan, last astronaut on the Moon, **dies** at 82	impregnated	0.8	

Table 1: Examples in train data of both subtasks. Underlined words in original headlines are to be substituted by the edit words.

3 Task Data

The Humicroedit dataset (Hossain et al., 2019) provides the training, development, and test data for this task. We also use additional training data from the FunLines dataset (Hossain et al., 2020b). The dataset statistics are summarized in Appendix A. In Subtask 1, the goal is predicting the funniness score of an edited headline. The score z ranges from 0 to 3, where 0 means not funny and 3 means very funny. In Subtask 2, the goal is to predict the funnier between an edited sentence pair. For labels $y \in \{0, 1, 2\}$, 0 implies two headlines are equally funny, 1 implies the first is the funnier, and 2 implies the second is the funnier. Examples of the two subtasks are in Table 1.

4 Methods

What are the important characteristics of humor? The incongruity theory, a dominant theory of humor, states that "it is the perception of something incongruous—something that violates our mental patterns and expectations" (Morreall, 2016). Therefore, we hypothesize an edited headline is funny if the edit words are semantically distant from the context words or the original words. This can be exemplified by the first two examples in Table 1. We start by looking at Headline 33210 (the MONKEY EXAMPLE), also shown below. The context sentence is extracted from the edit sentence by replacing the edit words with a single [MASK] token, motivated by masked language models. Humans are likely to predict a place or a character for the masked token, while the edit token is *"monkeys"*.

Original	CALIFORNIA AND PRESIDENT TRUMP ARE GOING TO WAR	$T_o = 8$
Edit	CALIFORNIA AND MONKEYS ARE GOING TO WAR	$T_e = 7$
Context	CALIFORNIA AND [MASK] ARE GOING TO WAR	$T_c = 7$

Similarly for Headline 1664, given the context WHAT IF [MASK] HAD AS MUCH INFLUENCE AS ECONOMISTS, humans might fill in occupation-related words like *"scientist"* or *"sociologist"* (as in the original headline). However, the edit word is *"donkeys"*, which is a surprising prediction and is considered very funny (scored 2.8 out of 3). In this section, we describe a concrete architecture that models the strength of contrast and surprise, which translates into the funniness score.

4.1 Span Representation

Let $\mathbf{x} = (x_1 \ldots x_{T_o})$, $\tilde{\mathbf{x}} = (\tilde{x}_1 \ldots \tilde{x}_{T_e})$, $\mathbf{x}^\star = (x_1^\star \ldots x_{T_c}^\star)$ denote the original, edit, context token sequences. A pretrained word embedding or pretrained encoder maps the tokens into vector sequences $\mathbf{e}_{1:T_o}, \tilde{\mathbf{e}}_{1:T_e}, \mathbf{e}_{1:T_c}^\star$. The goal is to encode edit sentence, original sentence, context sentence into fixed-length vector representations $\mathbf{u}, \mathbf{v}', \mathbf{v} \in \mathbb{R}^d$. Importantly, we use span (a.k.a. sub-sentence) representation rather than whole sentences, which corresponds to the underlined ranges in the above MONKEY EXAMPLE. Denote a span as a tuple of start and end position of contiguous tokens: $s = [i, j]$ for edit, $\tilde{s} = [i, k]$ for original, and $s^\star = [i, i]$ for the [MASK] token.

CBOW We first explore context-independent word representations. We use pretrained GloVe (Pennington et al., 2014) vectors with $d = 300$ and a vocabulary of 2.2 million words. We use word averaging to
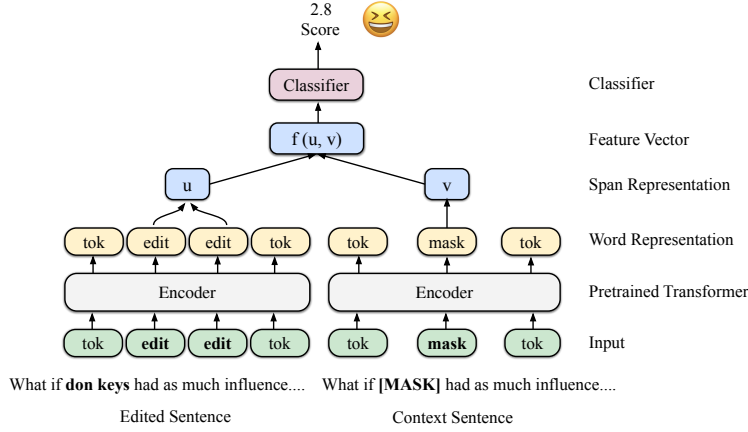
Figure 1: Transformer architecture to predict the funniness score of an edited headline using edit-context sentence pair. The full edited sentence is *"What if donkeys had as much influence as economists"*. *"donkeys"* is the edited word and is tokenized into two subwords in this example.

get edit word vector $\mathbf{u} = \mathrm{MeanPool}(\tilde{\mathbf{e}}_i \ldots \tilde{\mathbf{e}}_k)$ and original word vector $\mathbf{v}' = \mathrm{MeanPool}(\mathbf{e}_i \ldots \mathbf{e}_j)$. We max pool all context words to extract the most salient features. The context vector is $\mathbf{v} = \mathrm{MaxPool}(\mathbf{e}_1^\star \ldots \mathbf{e}_{i-1}^\star, \mathbf{e}_{i+1}^\star \ldots \mathbf{e}_{T_c}^\star)$.

Transformer We use pretrained transformer-based language models to obtain contextual word representations. The architecture is shown in Figure 1, a Siamese network (Bromley et al., 1994) where the two encoders have identical structures and shared parameters.[3] With the self-attention mechanism, each word attends to all other words in the sentence and aggregates contextual information. The edit and original vectors are obtained by averaging: $\mathbf{u} = \mathrm{MeanPool}(\tilde{\mathbf{e}}_i \ldots \tilde{\mathbf{e}}_k)$, $\mathbf{v}' = \mathrm{MeanPool}(\mathbf{e}_i \ldots \mathbf{e}_j)$. The context vector is simply the contextual embedding of the masked token: $\mathbf{v} = \mathbf{e}_i^\star$. We experiment with BERT and RoBERTa, using the PyTorch (Paszke et al., 2019) implementation from HuggingFace Transformers library (Wolf et al., 2019).[4] We use `bert-base-uncased` ($L = 12$, $d = 768$, lower-cased) and `roberta-base` ($L = 12$, $d = 768$).

Transfer Paradigm When using those pretrained word representations, we consider two transfer paradigms: finetuning (FINETUNE) and not finetuning (FREEZE). In the case of FREEZE, we use fixed word embedding directly as the feature for CBOW. For transformers, we use a weighted average of hidden layers from the frozen encoder, with trainable mixing scalars. This approach follows ELMo (Peters et al., 2018) and the edge probing model (Tenney et al., 2019). Specifically, the final aggregated embedding for i^{th} position is $\mathbf{e}_i = \gamma \sum_{l=0}^{L} \alpha_l \, \mathbf{e}_{l,i}$, where γ is a scaling factor, α_l is the weight of the lth layer, L is the total number of layers, and $l = 0$ corresponds to the embedding layer.

4.2 Task Specific

Regression As mentioned at the beginning of the section, contrast and surprise is the key to humor. To represent the pairwise relationship between two vectors, we derive feature from $\mathbf{h} = f(\mathbf{x}, \mathbf{y}) = [\mathbf{x}; \mathbf{y}; |\mathbf{x} - \mathbf{y}|; \mathbf{x} * \mathbf{y}] \in \mathbb{R}^{4d}$, where ; denotes concatenation and $*$ denotes element-wise multiplication. This feature has been used as the input to the classifier in the sentence pair tasks of SentEval (Conneau and Kiela, 2018). To formulate the contrast pair, we either use edit sentence and its context $f(\mathbf{u}, \mathbf{v})$, or edit sentence and original sentence $f(\mathbf{u}, \mathbf{v}')$. We denote the two scenarios as CONTEXT and ORIGINAL respectively. Finally, we use a classifier to predict the funniness score of the edited headline. $\hat{z} = \mathrm{Classifier}(\mathbf{h}) \in \mathbb{R}$. The classifier is a two-layer MLP with 256 hidden dimensions. When finetuning transformers we use single-layer linear projection instead, since its large number of parameters have already given us sufficient flexibility. The optimization objective is mean squared error $\mathcal{L} = \|\mathbf{z} - \hat{\mathbf{z}}\|^2$.

---

[3]An alternative architecture is to use a single transformer to encode the concatenation of a sentence pair, which enforces cross-sentence attention. However, since the paired sentences here are almost identical, cross-sentence attention seems unnecessary.

[4]HuggingFace Transformers library: https://github.com/huggingface/transformers

Model	Subtask 1			Subtask 2			
	RMSE†	Gain	Spearman	Accuracy†	Gain	Reward	RMSE
Baseline	0.575	-0.033	/	0.490	-0.109	-0.020	/
CBOW							
with CONTEXT+FREEZE	**0.542**	**0.000**	**0.319**	0.599	0.000	0.184	0.546
+ORIGINAL	0.559	-0.017	0.243	0.599	0.000	0.169	0.760
+EXTRA	0.544	-0.002	0.308	0.605	0.006	**0.191**	**0.545**
+ORIGINAL+EXTRA	0.558	-0.016	0.250	0.601	0.002	0.173	0.574
+FT	0.544	-0.002	0.311	0.604	0.005	0.178	**0.545**
+FT+ORIGINAL	0.561	-0.019	0.247	0.592	-0.007	0.165	0.604
+FT+EXTRA	0.548	-0.006	0.313	**0.606**	**0.007**	0.188	0.547
+FT+ORIGINAL+EXTRA	0.563	-0.021	0.261	0.589	-0.010	0.161	0.581
BERT							
with CONTEXT+FREEZE	0.531	0.011	0.384	0.616	0.017	0.207	0.546
+ORIGINAL	0.534	0.008	0.366	0.603	0.004	0.186	0.547
+EXTRA	**0.530**	**0.012**	0.382	0.615	0.016	0.207	**0.530**
+ORIGINAL+EXTRA	0.541	0.001	0.346	0.615	0.016	0.204	0.542
+FT	0.536	0.006	0.366	**0.635**	**0.036**	0.234	0.552
+FT+ORIGINAL	0.536	0.006	0.365	0.628	0.029	0.231	0.555
+FT+EXTRA	0.541	0.001	0.373	0.630	0.031	0.232	0.554
+FT+ORIGINAL+EXTRA	0.533	0.009	**0.387**	0.629	0.030	**0.236**	0.550
RoBERTa							
with CONTEXT+FREEZE	0.528	0.014	0.388	0.635	0.036	0.246	**<u>0.529</u>**
+ORIGINAL	0.536	0.006	0.366	0.625	0.026	0.224	0.540
+EXTRA	0.528	0.014	0.390	0.640	0.041	0.252	0.536
+ORIGINAL+EXTRA	0.533	0.009	0.368	0.618	0.019	0.207	0.543
+FT	0.534	0.008	0.382	0.649	0.050	<u>0.254</u>	0.535
+FT+ORIGINAL	0.527	0.015	<u>0.425</u>	<u>0.650</u>	<u>0.051</u>	<u>0.254</u>	0.538
+FT+EXTRA	0.526	0.016	0.407	0.638	0.039	0.233	0.594
+FT+ORIGINAL+EXTRA	<u>0.522</u>	<u>0.020</u>	0.410	0.626	0.027	0.216	0.625

Table 2: Post-evaluation test results. FREEZE: no finetuning; FT: finetuning; CONTEXT: use context of edit headline; ORIGINAL: use original headline; EXTRA: use additional training data from the FunLines dataset. **RMSE**† and **Accuracy**† are primary metrics. The best within each model type is **bolded**, and the overall best is <u>underlined</u>. Gain measures the improvement over CBOW-CONTEXT-FREEZE. In Subtask 1 this is w.r.t. to RMSE and in Subtask 2 w.r.t. accuracy. Baseline 1 uses the average score and Baseline 2 uses the majority label.

Classification In Subtask 2, we use the same method to predict the scores of two edited versions $\hat{z}^{(1)}$ and $\hat{z}^{(2)}$. By comparing the scores, the funnier version is found during evaluation and testing time: $\hat{y} = \arg\max_{i \in \{1,2\}} \hat{z}^{(i)}$. The loss function is $\mathcal{L} = \|\mathbf{z}^{(1)} - \hat{\mathbf{z}}^{(1)}\|^2 + \|\mathbf{z}^{(2)} - \hat{\mathbf{z}}^{(2)}\|^2$.

5 Experiments and Results

5.1 Metrics

For Subtask 1, the primary metric for official ranking is Root Mean Squared Error (RMSE). In addition, we calculate Spearman's rank correlation coefficient which measures the monotonic relationship between predicted scores and true scores. In the evaluation of Subtask 2, instances with label 0 are ignored. The primary metric for official ranking is accuracy. As an auxiliary metric, reward takes pairwise score differences into account: $r = \frac{1}{N} \sum_{i=1}^{N} (\mathbb{1}_{\hat{y}_i = y_i} - \mathbb{1}_{\hat{y}_i \neq y_i}) |z_i^{(1)} - z_i^{(2)}|$, where y_i and $\hat{y}_i$ are true labels and predicted labels respectively, and $z_i^{(1)}$ and $z_i^{(2)}$ are true scores.

5.2 Official Evaluation

For the official evaluation, our submitted system is RoBERTa-FREEZE-CONTEXT. We use Adam (Kingma and Ba, 2015) optimizer with a learning rate of 1e-3 and use the 10th epoch. Our system gets 0.531 RMSE for Subtask 1 (11th among 49 submissions) and 0.632 accuracy for Subtask 2 (9th among 32 submissions)

Feature	Subtask 1 RMSE	Subtask 2 Accuracy
CONTEXT	0.526	0.649
ORIGINAL	0.522	0.650
EDIT	0.526	0.651

Table 3: Comparison between non-contrastive and contrastive approaches, based on RoBERTa. CONTEXT and ORIGINAL are contrastive, using the edit-context pair and the edit-original pair as input respectively. EDIT is non-contrastive, using the edit sentence only. The results are from test sets of the two subtasks. We tune other hyperparameters on the validation sets and select the best model for each cell.

on the test set.[5]

5.3 Post-Evaluation

In the post-evaluation phase, we conduct a more extensive search on hyperparameters and select the best models based on validation performance. Experiment details and hyperparameters are in Appendix B. We systematically compare CBOW, BERT, and RoBERTa, and perform an ablation study to understand the effects of various factors: extra training data, finetuning or freezing the pretrained embeddings, and using CONTEXT or ORIGINAL feature. The post-evaluation results on the test set are in Table 2.

Contextual Representation Despite its simplicity, CBOW is surprisingly effective. Its best result is significantly better than the baseline and is comparable to Subtask 1 #19 (0.547) and Subtask 2 #17 (0.605) on the leaderboard. By comparing the three models, we see that pretrained language models have better performance than context-independent word embedding. While results for BERT and RoBERTa are similar, both of them outperform CBOW, evidencing that contextual information is essential for humor detection.

CONTEXT vs. ORIGINAL In the ablation study, we first notice that neither finetuning nor using extra data from the FunLines dataset make much difference for all models. Interestingly, using different contrast pairs as the feature has different effects on models. CONTEXT is better than ORIGINAL for CBOW, yet they are similar for pretrained language models. Why does this happen? CBOW with ORIGINAL only uses the information of edited word and original word while completely neglecting the contextual relation. Pairing CBOW with CONTEXT can alleviate this limitation. On the other hand, pretrained language models exploit the contextual relation between edit words and context words in both cases.

6 Analysis and Discussion

6.1 Non-contrastive Approach

In our main experiment, we focus on the contrastive approach using a sentence pair (i.e., CONTEXT and ORIGINAL) and show its effectiveness. The remaining question is, can we predict humor using the edited sentence as the only input? Thus, we investigate a non-contrastive approach, with a single encoder to obtain the span representation of the edit sentence. We refer to this as EDIT. This is equivalent to only using the left part in Figure 1. We conduct an experiment with RoBERTa on both subtasks. The results are in Table 3. We see that EDIT has similar performance as contrastive approaches. We conjecture that EDIT captures contrast *implicitly*, while CONTEXT and ORIGINAL capture contrast *explicitly* by design.

6.2 Error Analysis

To understand the relationship between human judgment and model predictions, we calculate the correlation matrix between true funniness scores and predicted scores from RoBERTa with different features (CONTEXT, ORIGINAL, and EDIT) for Subtask 1. From Table 4, we see that the models correlate poorly with human judgment (correlations ≈ 0.4), while correlating well with each other (correlation ≈ 0.8).

---

[5]Task leaderboard: https://competitions.codalab.org/competitions/20970#results. "Evaluation-Task-1" is for Subtask 1 and "Evaluation-Task-2" is for Subtask 2.

	Human	**ORIGINAL**	**CONTEXT**	**EDIT**
Human	/	0.41	0.41	0.42
ORIGINAL	0.42	/	0.84	0.77
CONTEXT	0.41	0.84	/	0.80
EDIT	0.43	0.79	0.81	/

Table 4: The correlation matrix of human scores and predicted scores from RoBERTa with different features: EDIT, CONTEXT, and ORIGINAL. Lower triangle: Pearson correlation coefficients. Upper triangle: Spearman correlation coefficients. Results are from the test set of Subtask 1.

ID	Original Headline	Edit	Human	Prediction		
				ORIGINAL	CONTEXT	EDIT
9100	WSJ: Trump's top national security adviser is being investigated for his communications with **Russia**	children	0.0	1.35	1.50	0.94
9406	Man Sets Off Explosive Device at L.A.-Area Cheesecake Factory, No **Injuries**	complaints	2.4	0.76	0.52	0.72
12271	Turkey tells **citizens** to reconsider travelling to US	poultry	2.4	0.85	0.83	0.65
12685	CBS Poll: Americans lack confidence in Trump's ability to **handle** North Korea	locate	2.4	0.97	0.94	0.96

Table 5: Error analysis on the test set of Subtask 1. The examples show disagreements between human ratings and model predictions.

To further learn when the models make an erroneous judgment, we look at model predictions on the test set of Subtask 1. We see the models generally capture the incongruity phenomenon. While being key to many examples, incongruity does not account for others. We summarize some typical examples in Table 5:

- For Headline 9100, the edit word *"children"* is incongruous with the context of national security. While humans consider it not funny at all, models assign a high funniness score. The fallacy is that incongruity is not a sufficient condition for humor.

- In other cases, the edit words are congruous with the context. While humans consider them very funny, models predict the opposite. That is, incongruity is not a necessary condition for humor. Humor has diverse underlying causes. For instance, Headline 12685 shows *sarcasm*, taunting Trump's lack of geography knowledge and common sense. Headline 12271 uses *pun* based on polysemy: *"turkey"* can either mean a country (when capitalized) or a bird. Also, humor can require an understanding of *cultural commentary*, exemplified by Headline 9406. Since the Cheesecake Factory is a large chain of restaurants that some may look down upon, they are happy to see it blown up with *"no complaints"*.

7 Conclusions

We use incongruity as the key to assessing funniness in edited news headlines. Specifically, we use pretrained transformer-based language models to encode contrastive pairs. Our best performing model is RoBERTa, which is submitted for the official evaluation and achieves competitive performance in both subtasks. The additional experiment shows that a non-contrastive approach may also encode incongruity implicitly. While incongruity is a common ingredient of humor, error analysis indicates it is neither sufficient nor necessary. This invites future research to take other factors (e.g., sarcasm, pun, or world knowledge) into account to better tackle humor, an intricate phenomenon rooted in human creativity.

Acknowledgements

References

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, Matías Cubero, Diego Garat, and Guillermo Moncecchi. 2016. Is this a joke? Detecting humor in Spanish tweets. In *Ibero-American Conference on Artificial Intelligence (IBERAMIA 2016)*, pages 139–150. Springer.

Alexis Conneau and Douwe Kiela. 2018. SentEval: An evaluation toolkit for universal sentence representations. In *Proceedings of the 11th International Conference on Language Resources and Evaluation (LREC 2018)*.

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.

He He, Nanyun Peng, and Percy Liang. 2019. Pun generation with surprise. 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 1734–1744.

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.

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)*.

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 the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 256–262.

Justine T. Kao, Roger Levy, and Noah D. Goodman. 2016. A computational model of linguistic humor in puns. *Cognitive Science*, 40(5):1270–1285.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of International Conference on Learning Representations (ICLR)*.

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. *arXiv preprint arXiv:1907.11692*.

Jihang Mao and Wanli Liu. 2019. A BERT-based approach for automatic humor detection and scoring. In *Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2019)*.

John Morreall. 2016. Philosophy of humor. In *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, winter 2016 edition.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8026–8037.

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.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? Probing for sentence structure in contextualized word representations. In *Proceedings of International Conference on Learning Representations (ICLR)*.

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.

Tony Veale. 2004. Incongruity in humor: Root cause or epiphenomenon? *Humor*, 17(4):419–428.

Orion Weller and Kevin Seppi. 2019. Humor detection: A transformer gets the last laugh. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3621–3625.

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 preprint arXiv:1910.03771*.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Xinru Yan and Ted Pedersen. 2017. Duluth at SemEval-2017 task 6: Language models in humor detection. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 385–389.

A   Task Summary

Task	Type	Metric	Train	Train Extra	Dev	Test
Subtask 1	Regression	RMSE	9653	8248	2420	3025
Subtask 2	Classification	Accuracy	9382	1959	2356	2961

Table 6: Summary of subtasks. Train, Dev, Test are from the Humicroedit dataset. Train Extra is from the FunLines dataset.

B   Experiment Details

Preprocessing   We use spaCy word tokenizer for CBOW. The pretrained transformers use byte-pair encoding (Sennrich et al., 2016, BPE) to convert text into subword units. BERT uses WordPiece (Wu et al., 2016) tokenization, a character-level BPE, with a vocabulary size of 30K. RoBERTa preserves cases and uses a byte-level BPE with a vocabulary size of 50K.

Training   For training, we use a batch size of 32 in Subtask 1 and 16 in Subtask 2. We use Adam optimizer and perform gradient clipping with a max ℓ_2 norm of 5. For most experiments, we train for 10 epochs with a learning rate in {1e-3, 3e-4}. However, when finetuning transformers, we choose max epochs in {3, 10}, and use either a constant learning rate or a linear decreasing schedule with an initial learning rate in {2e-5, 5e-5}. We perform validation on the development set every 1/3 epoch and save the best checkpoint.