Hasyarasa at SemEval-2020 Task 7: Quantifying Humor as departure from Expectedness

Ravi Theja Desetty
TCS Research, India
rt.desetty@tcs.com

Ranit Chatterjee
TCS Research, India
ranit.chatterjee@tcs.com

Smita Ghaisas
TCS Research, India
smita.ghaisas@tcs.com

Abstract

This paper describes our system submission Hasyarasa for the SemEval-2020 Task-7: Assessing Humor in Edited News Headlines. This task has two subtasks. The goal of Subtask 1 is to predict the mean funniness of the edited headline given the original and the edited headline. In Subtask 2, given two edits on the original headline, the goal is to predict the funnier of the two. We observed that the departure from expected state/ actions of situations/ individuals is the cause of humor in the edited headlines. We propose two novel features: Contextual Semantic Distance and Contextual Neighborhood Distance to estimate this departure and thus capture the contextual absurdity and hence the humor in the edited headlines. We have used these features together with a Bi-LSTM Attention based model and have achieved 0.53310 RMSE for Subtask 1 and 60.19% accuracy for Subtask 2.

1 Introduction

Humor is a highly intellectual and communicative activity that promotes laughter and provides amusement. A great sense of humor refers to the ability to perceive things as funny. With the advances in Natural Language Processing (NLP) techniques, there has been considerable research effort in developing artificial sense of humor in machines. Much of the existing work on humor recognition is done in evaluating whether a given text is humorous or non-humorous using datasets such as twitter, yelp reviews, reddit jokes and other sources (Barbieri and Saggion, 2014; Weller and Seppi, 2019).

However, the problem dealt in SemEval-2020 Task 7: Assessing the Funniness of Edited News Headlines is not about classifying a given text as humorous or non-humorous. It is about how a micro-edit on a news headline can *change* it from non-humorous to humorous (Hossain et al., 2020a; Hossain et al., 2019). There are two subtasks in this task. Subtask 1: Given the original and edited headline, the task is to predict the mean funniness of the edited headline. Subtask 2: Given the original headline and the two edited versions of the original headline, the task is to predict which edited version is the funnier of the two edited versions. The evaluation criteria, as set by the organizers for Subtask 1 is Root Mean Squared Error (RMSE) and for Subtask 2 is accuracy.

In this work, our contribution to the challenge is to model two novel features namely Contextual Semantic Distance (CSD), and Contextual Neighborhood Distance (CND) which are based on our observation that the departure from expected state/ actions of the situations/ individuals is the cause of humor in the edited news headlines. The remainder of the paper is organized as follows: In Section 2, we present related work. In Section 3, we give a brief description of Hasyrasa - our system for quantifying humor, discuss the proposed features and compare our results with a model that incorporates similarity features. Results and Analysis are presented in Section 4 and Section 5 respectively. Section 6 concludes the paper.

Proceedings of the 14th International Workshop on Semantic Evaluation, pages 833–842
Barcelona, Spain (Online), December 12, 2020.

2 Related Work

The task of assessing humor in micro-edited text such as news headlines is a less explored research area. Most of the previous work is focused on humor recognition which is formulated as a binary classification problem i.e., determining whether a given text is humorous or non-humorous. Various datasets like Pun of the Day (Yang et al., 2015), Reddit Jokes (Weller and Seppi, 2019), 16000 One-liners (Mihalcea and Strapparava, 2005), Short Jokes[1] have been used or created for recognizing humor. Researchers have adopted traditional machine learning algorithms like SVM and Naive Bayes (Mihalcea and Strapparava, 2005), deep learning architectures like Convolution Neural Network (CNN) (Chen and Lee, 2017; Chen and Soo, 2018), Long Short-Term Memory (LSTM) (Bertero and Fung, 2016), Bidirectional Long-Short Term Memory (Bi-LSTM) (Sane et al., 2019), Transformer architecture (Weller and Seppi, 2019) for humor detection. Morreall (2016) discussed about theories of humor and Hossain et al. (2019) argued humor is directly correlated with incongruity.

However, the challenge in the present work is to detect humor created when a headline is edited. For this purpose we propose the use of two novel features - Contextual Semantic Distance (CSD) and Contextual Neighborhood Distance (CND), and compare our results with a model that incorporates Similarity Features (SF). We employ CNN (LeCun et al., 1998), LSTM (Hochreiter and Schmidhuber, 1997) and Attention mechanism (Bahdanau et al., 2015) to (1) Predict intensity of humor in edited news headline and, (2) Predict which edited news headline is more humorous of the two edited versions. From among the existing research, work done by Hossain et al. (2019) is semantically closest to our position on unexpectedness as a source of humor.

3 System Description

In this section we describe our system: 'Hasyarasa'. The word Hasyarasa in Sanskrit means (creating) a mood of laughter. We employ Hasyarasa to quantify the contextual unexpectedness, which is the cause of humor in the edited news headlines. In the following subsections, we start with the discussion of novel features in our approach. Next, we give a detailed description of our model architecture and baselines. The data preprocessing and training details are described in the Appendix.

3.1 Features that contribute to Humor

In a societal and social context, we regard some situations and/or actions of individuals as *normal* and therefore *expected*. For example, one might come across a headline in a newspaper stating: *'Indian IT firms mandate working from home amidst fear of COVID-19'*. This is quite an expected scenario. However, it would be totally unexpected if one comes across this headline instead: *'Indian IT firms mandate working from Mars amidst fear of COVID-19'*. Similarly, certain actions by a particular individual seem expected in a context, while others seem totally unexpected. It is quite expected to come across news such as *'The US President fires the FBI Director'* but not at all expected to see *'The US President kisses the FBI Director'*.

On carefully analyzing the above edited headlines, one can observe that the humor created by editing a particular word in the original headline is because of the absurdity due to subsequent departure from the expected state / action of situations / individuals. This unexpectedness of occurrence of the edited word in the context of the original headline therefore can be a measure of intensity of humor resulting from absurdity due to the edit. Generally, the more unexpected the replacing word, the more would be the intensity of the humor. We compute this departure from expected state / action of situation/ individual as (1) Contextual Semantic Distance (CSD) and (2) Contextual Neighborhood Distance(CND).

The CSD computes the semantic distance between the target word/ edited word and certain thematic words of the original headline, which play a central role in establishing the theme of the sentence. In this context, semantic distance refers to the cosine similarity between the vector representation of the words obtained from GloVe (Pennington et al., 2014) embedding model.

The CND aims at estimating the departure from the neighborhood of words or phrases associated with the thematic words of the original headline. We have used Word Embeddings and a Knowledge Graph

[1] https://www.kaggle.com/abhinavmoudgil95/short-jokes

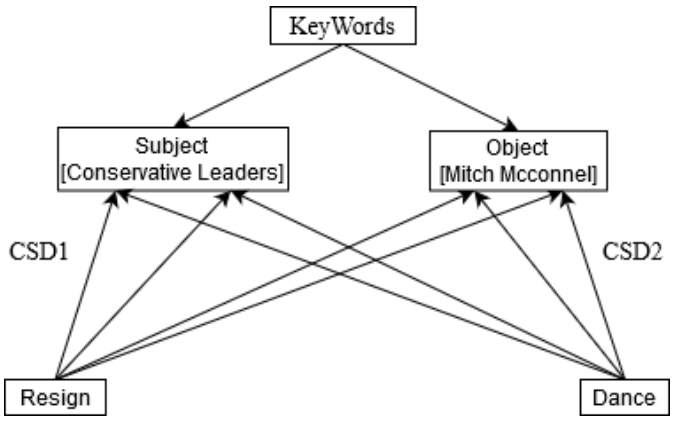


Figure 1: Contextual Semantic Distance

based approach to capture this concept of neighborhood. We post it that the average semantic distance (cosine similarity) of the edited word with the neighborhood words would help us to quantify the departure from expectedness and thus the humor in the edited news headlines.

In addition to these two features, we experimented with Similarity features (SF). SFs help us in comparing the similarity between two sentences. In the following sub-sections we further elaborate these features.

Contextual Semantic Distance (CSD): The CSD captures the semantic distance between certain thematic words in the original headline and both the edit and the target words individually. This feature helps to estimate the contextual relevance of the edited and the target words in the original headline and hence the unexpectedness of the edited word in the given context.

It has been observed that certain words in the sentential structure carry more weight than the other words in determining the central idea (or theme) of the sentence. These are the *subject, object* and the *action* words. These words are essentially the thematic words of a sentence. The following example illustrates this observation.

Original: Conservative Leaders urge Mitch McConnell to ***resign***.
Edited: Conservative Leaders urge Mitch McConnell to ***dance***.
Subject: Conservative Leaders
Object: Mitch McConnell
Original Action: resign
New Action: dance

In the above original headline, the word resign (Target Word) has been replaced with the word *dance* (Edited Word) and the edited headline is created.

We extract the *subject, object and action* words using the spaCy dependency parser[2]. Hence forth, we refer to these words as *Keywords*. We compute Contextual Semantic Distance 1 (CSD1) and Contextual Semantic Distance 2 (CSD2) using these *Keywords*. The detailed calculations of CSD1 and CSD2 are described in the Appendix.

Contextual Neighborhood Distance (CND): The neighborhood of a word is defined as cluster of words or phrases which are contextually relevant to that particular word. Such words usually co-occur together on numerous occasions. For example, the word *'Trump'* would be associated with the neighborhood words, which would mainly comprise of those surrounding US politics. Words such as *'Republican'*, *'America'*, *'Hilton'*, *'Bernie'*, *'Ivanka'*, *'cabinet meetings'*, *'trade deals'*,*'tweets'* are some of the words which are a part of this neighborhood. It is important to capture the neighborhood of the *Keywords* of the

[2]https://spacy.io/.

original headline. Such a neighborhood highlights the main theme or the context of the headline. We have explored two ways to capture this neighborhood by using (1) Word Embeddings, (2) Knowledge Graphs.

The first step to compute CND is the same as that of the CSD feature, where we extract the Keywords from the original headline using the dependency parser. For each of these Keywords (*w*), we find the top *n* similar words from the word embedding by arranging the words in ascending order of their cosine similarity with *w*. The only problem with this approach is that word embeddings are implicit representations of context. It is impossible to visualize the similar words just by observing the word vectors and so the whole implementation becomes an incomprehensible black box.

An alternate way to capture this neighborhood is by using explicit context representation like Knowledge Graphs (Pujara et al., 2013). Knowledge Graphs were introduced by Google to enhance their search engine results. Knowledge Graphs are used to present information in a connected fashion. Information is stored as a graph with two entities and an edge between them describes the relationship between the two entities. Now, in order to compute the CND feature, we have created the knowledge graph on Kaggle's *All the News* dataset[3], which contains around 150000 news headlines from a variety of sources mostly biased towards American politics which is also the case for our dataset.

After constructing the knowledge graph, we query the graph with each of the Keywords for each headline in our dataset to find the related words and phrases. Table 1 shows some of the entity relationships obtained by querying the knowledge graph for the word *'Trump'*.

Source	Edge	Target
Trump	nominates	Supreme Court
Trump	chooses	CIA Director
Trump	hires	White House Team
Trump	overrules	Deputy State
Trump	requests	Military US Parade

Table 1: Examples of Querying Results on the Knowledge Graph for the word *'Trump'*

We form two clusters of words and denote them as circles of neighborhood. Each of these clusters capture a different aspect of the contextual neighborhood. The first cluster consists of the target words associated with a particular source and hence gives an idea as to which nodes (entities) are linked with the source. This cluster forms the *Circle of Associated Entities*. The second cluster consists of all the edges linked with the source and gives an idea of the associated relationships (edges). These relationship words are usually verbs and mainly highlight the expected set of action words surrounding a particular entity. We call this cluster the *Circle of Associated Relationships*. Figure 2 shows the visualization of the neighborhood circles associated with the word *'Trump'*, which has been created by choosing a few entity relationships by querying our knowledge graph.

There can be cases when querying the knowledge graph might result in no hits i.e no such node is present corresponding to a word. In such cases we fall back to the word embedding approach to capture the neighborhood. Thus, we implement a system of word embeddings and knowledge graphs to obtain the contextual neighborhood. Finally, we compute the CND by finding the average cosine similarity between the edited word and the neighborhood words for each of the Keywords of the original headline. We believe that it would capture the departure from expectedness in the edited headline which results in absurdity and would therefore correlate with humor.

Similarity Features (SF): Similarity Features (SF) will estimate the humor by computing the similarity between original and edited headlines. These features are inspired from Infersent (Conneau et al., 2017) and Sentence-bert (Reimers and Gurevych, 2019). If u and v are the representations of original and edited headlines, the absolute element wise difference $|u - v|$ and element-wise dot product $u * v$ will capture the similarity between the original and edited headlines. The calculations for computing the features in the above sections are with respect to Subtask 1. The features for Subtask 2 are computed in similar manner.

[3]https://www.kaggle.com/snapcrack/all-the-news

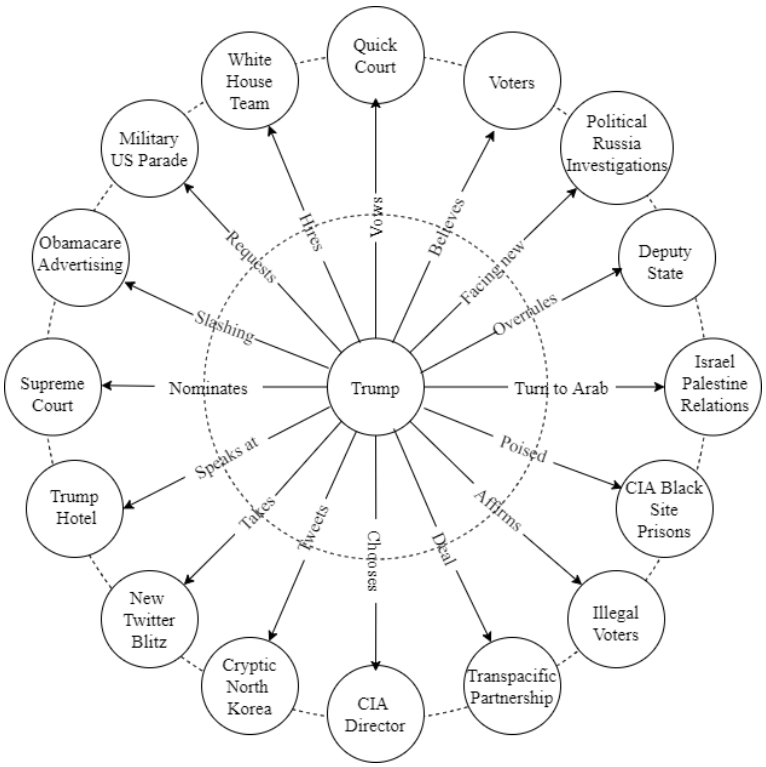


Figure 2: Circles of Neighborhood for the word *Trump*. The outer dashed circle denotes the *Circle of Associated Entities* and the inner dashed circle denotes the *Circle of Associated Relationships*.

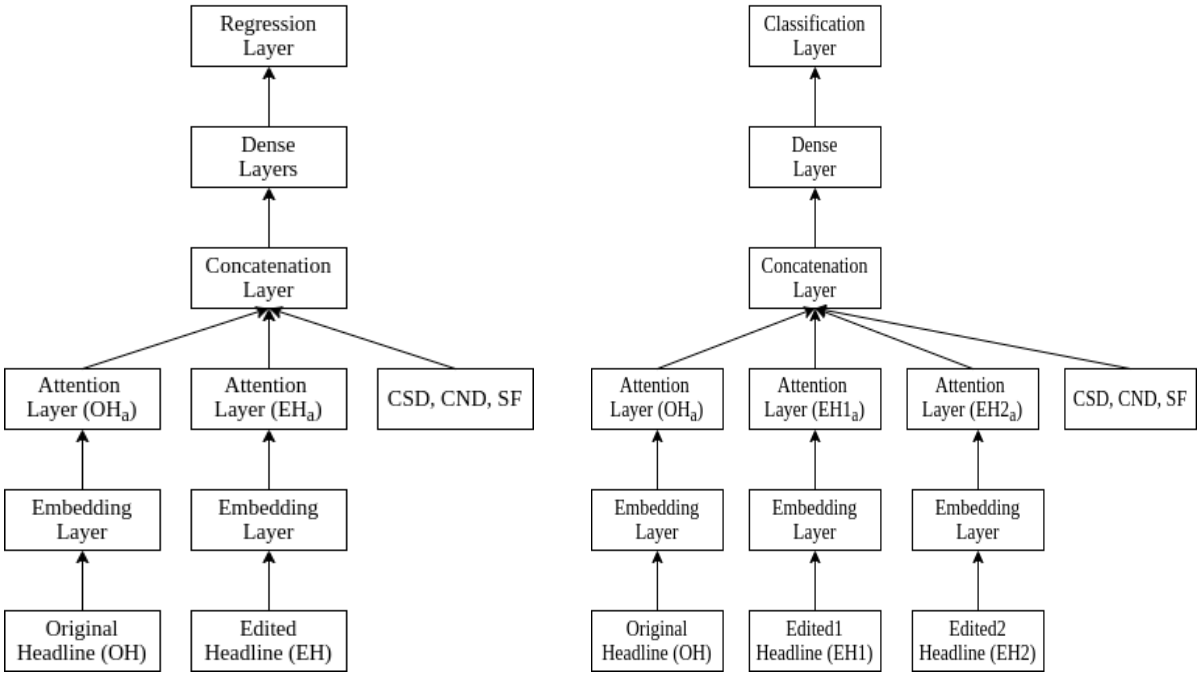
3.2 Model Architecture

In this subsection we discuss about our models for Subtask 1 and Subtask 2 which consists of six layers namely embedding, Bi-LSTM, attention, concatenation, dense and regression/ classification layers and also describe in detail about usage of these layers in Subtask 1 and Subtask 2. The model architecture for Subtask 1 and Subtask 2 is illustrated in Figure 3.

Subtask 1: We passed the Original Headline (OH) and Edited Headline (EH) to the embedding layer which is a lookup table that encodes each word in the input sentence to a d-dimensional feature space (R_d). These word vector representation of OH and EH are passed to Bi-directional LSTM (Bi-LSTM layer) to get hidden representations $oh_1, oh_2, oh_3, ..., oh_i, ...oh_T$ and $eh_1, eh_2, eh_3, ..., eh_i, ...eh_T$ for each word in the sentences. These hidden representations are used by the attention layer to obtain a fixed-length representation for OH (OH_a) and EH (EH_a). These fixed-length representations are calculated as described in Luong et al. (2015). The reason behind using an attention layer is that it helps the model to focus on the important parts of the sentence in detecting humor. These fixed-length representations OH_a and EH_a are concatenated with CSD, CND and SF features and are fed to dense layers. The dense layers are followed by a regression layer which gives the mean funniness score for the micro-edit in the original headline.

Subtask 2: Similar to Subtask 1, the Original Headline (OH), Edited1 Headline (EH1), Edited2 Headline (EH2) are passed to embedding layer followed by Bi-LSTM layer and attention layer. The fixed-length representations from attention layers $OH_a, EH1_a$ and $EH2_a$ are concatenated with CSD, CND and SF features and then fed to a dense layer which is followed by a softmax layer that would give the probability distribution over the output classes.

Baselines: To ensure the effectiveness of our model, we have built 3 baseline models. As described by Chen and Soo (2018), we have used CNN with highway layers as baseline – 1. For baseline – 2, instead of using the representation of original/ edited headlines from attention layer, we have used the final hidden state from Bi-LSTM layer as the representation of original and edited headlines followed by concatenation layer, dense layers, regression/ classification layer. For baseline – 3, we have removed CSD,



(a) Architecture for Subtask 1 (b) Architecture for Subtask 2

Figure 3: Model Architecture

CND and SF features in the concatenation layer of our model.

4 Results

Our results for Subtask 1 and Subtask 2 are reported in Table 2 and Table 3 respectively. The reported results differ from the official leader board as we worked on new features afterward. The reported results in Table 2 and Table 3 are trained on training dataset and evaluated on the development and test datasets provided by the task organizers. The results indicates that our BiLSTM attention based model with CSD, CND, SF features out performs the baseline - 1 (CNN + Highway Layers), baseline -2 (BiLSTM) and baseline - 3 (BiLSTM + Attention) models.

In order to understand the effectiveness of the features, we experimented with different permutations of features with attention mechanism. The results are shown in Table 2 and Table 3. We trained our attention based model with extracted features by including Funlines training data (Hossain et al., 2020b) along with training data provided in the competition and results are reported in Table 4 which indicate that inclusion of Funlines data improved the performance of our model for Subtask 1 but not for Subtask 2.

5 Analysis

The features CSD, CND and SF are engineered keeping in mind the characteristics of the dataset. We theorized that quantifying the incongruities in the edited news headlines focusing on the departure from expected state which is along the lines of Morreall (2016) will correlate with humor scores and our experiments validate this theory. While the CSD feature captures the relevance of the edited word keeping in mind just the context of the original headline, the CND feature builds on existing knowledge of the world to capture the contextual relevance. The datasets provided by the organizers of the competition are mostly biased towards American politics. Therefore, we selected Kaggle's *All the News* dataset for building the knowledge graph in order to evaluate the CND feature. The knowledge graph implementation is heavily dependent on the corpus it is built on. Therefore, bigger and more elaborate news headlines datasets will help build a stronger knowledge graph. Such a graph will be more efficient in capturing the contextual neighborhood. Humor can also be estimated by comparing the representations of original and edited headlines. SF which is primarily an architecture specific feature computes the similarity between

Model	Subtask 1 (RMSE)	
	Validation	Test
CNN + Highway Layers	0.57840 ± 0.00103	0.72089 ± 0.00121
BiLSTM	0.55438 ± 0.00131	0.54567 ± 0.00133
BiLSTM + Attention	0.54392 ± 0.00142	0.54015 ± 0.00148
BiLSTM + Attention + CSD	0.54165 ± 0.00122	0.53633 ± 0.00145
BiLSTM + Attention + CND	0.54175 ± 0.00163	0.53682 ± 0.00136
BiLSTM + Attention + SF	0.54315 ± 0.00156	0.53755 ± 0.00172
BiLSTM + Attention + CSD + SF	0.53995 ± 0.00133	0.53565 ± 0.00152
BiLSTM + Attention + CND + SF	0.54068 ± 0.00111	0.53604 ± 0.00143
BiLSTM + Attention + CSD + CND	0.53594 ± 0.00154	0.53452 ± 0.00165
BiLSTM + Attention + CSD + CND + SF	0.53458 ± 0.00161	**0.53310 ± 0.00181**
Submitted	0.53594 ± 0.00134	0.70333 ± 0.00237

Table 2: Experimental Results for Subtask 1. Submitted are the results we officially submitted to the task leaderboard. Each experiment is trained with 10 different random seeds, accordingly the mean and Standard deviation of the results are shown here. CNN + Highway Layers is Baseline-1, BiLSTM is Baseline-2 and BiLSTM + Attention is Baseline-3.

Model	Subtask 2 (Accuracy)	
	Validation	Test
CNN + Highway Layers	51.10 ± 0.092	51.10 ± 0.084
BiLSTM	56.64 ± 0.519	54.26 ± 0.442
BiLSTM + Attention	60.26 ± 0.412	59.89 ± 0.574
BiLSTM + Attention + CSD	61.31 ± 0.238	59.66 ± 0.286
BiLSTM + Attention + CND	61.64 ± 0.358	58.59 ± 0.541
BiLSTM + Attention + SF	61.70 ± 0.347	59.13 ± 0.442
BiLSTM + Attention + CSD + SF	60.98 ± 0.326	**60.19 ± 0.357**
BiLSTM + Attention + CND + SF	61.31 ± 0.412	59.30 ± 0.311
BiLSTM + Attention + CSD + CND	61.10 ± 0.213	58.90 ± 0.317
BiLSTM + Attention + CSD + CND + SF	61.71 ± 0.418	59.12 ± 0.348
Submitted	61.93 ± 0.208	59.70 ± 0.413

Table 3: Experimental Results for Subtask 2. Submitted are the results we officially submitted to the task leaderboard. Each experiment is trained with 10 different random seeds, accordingly the mean and Standard deviation of the results are shown here. CNN + Highway Layers is Baseline-1, BiLSTM is Baseline-2 and BiLSTM + Attention is Baseline-3.

Dataset Used For Training	Subtask 1		Subtask 2	
	Validation	Test	Validation	Test
Humicroedit + Funlines	0.5486 ± 0.0015	0.5326 ± 0.0012	61.34 ± 0.32	58.48 ± 0.41

Table 4: Experimental Results with inclusion of Funlines dataset

original and edited headlines representations from attention layer for estimating humor.

The computation of both CSD and CND features depend upon word embeddings (GloVe in our case) for calculating the cosine similarity between the words. Hence the efficiency of these features for this problem is limited by the limitations of word embeddings itself. After conducting elaborate experimentation, we have observed that even though the usage of CSD and CND improves the performance in terms of RMSE (for Subtask 1) and Accuracy (for Subtask 2), the improvement is not significant over that of the standard BiLSTM Attention Model.

One of the primary drawbacks of word embeddings (word vector space models in general) is that words with multiple meanings are collated into a single vector representation in the semantic space. In linguistic terms, figures of speech such as *polysemy* and *homonymy* are not handled properly using word embedding models. For example, in the following news headline taken from the dataset- *'Rex Tillerson : US has direct channels to Pyongyang'*, it is impossible to understand if the word *channel(s)* is related to its word sense implying a *strait* (Geography) or the word sense of a *frequency band* (Telecommunications) or any other sense associated with the word *channel(s)*. Such usages of words with multiple meanings are often found in English texts.

Neologism is defined as a relatively new or isolated term, word, or phrase which is in the process of entering colloquial usage, but has not yet been fully accepted into mainstream language. An example of neologism is the word *webinar* which has its roots in the words *web* and *seminar* and essentially means an online seminar. Neologisms are often driven by changes in culture and technology and can be found very often in daily news headlines. Word embedding models like GloVe and Word2vec have no way to learn the representation of Out-of-Vocabulary (OOV) words, which often causes difficulties for the model to understand such neologistic word usages. In order to tackle this problem, we have also experimented with custom embeddings created over news headlines dataset like Kaggle's *All the News* dataset. However, we did not obtain any better result than what we did with GloVe embeddings, which can be primarily attributed to the amount of vastness and elaborateness of such news datasets. So, we believe a bigger and more elaborate news headlines dataset would surely boost the scores to a higher degree.

6 Conclusion

In this paper, we presented our system on the task of Assessing Humor in Edited News Headlines in SemEval-2020. Our experiments with CSD, CND and SF features combined with attention mechanism proved to be reasonably effective in assessing humor in edited news headlines.

Future work involves collecting more extensive news headlines datasets and building a stronger knowledge graph. Using the datasets provided in the competition, it would be an interesting exercise to approach the problem for generating a humorous headline given a news headline and predicting the target word of the headline that should be micro-edited.

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*.

Francesco Barbieri and Horacio Saggion. 2014. Automatic detection of irony and humour in twitter. In *ICCC*, pages 155–162.

Dario Bertero and Pascale Fung. 2016. A long short-term memory framework for predicting humor in dialogues. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 130–135.

Lei Chen and Chong Min Lee. 2017. Predicting audience's laughter during presentations using convolutional neural network. In Joel R. Tetreault, Jill Burstein, Claudia Leacock, and Helen Yannakoudakis, editors, *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications, BEA@EMNLP 2017, Copenhagen, Denmark, September 8, 2017*, pages 86–90. Association for Computational Linguistics.

Peng-Yu Chen and Von-Wun Soo. 2018. Humor recognition using deep learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 113–117.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel, editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 670–680. Association for Computational Linguistics.

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. *arXiv preprint arXiv:2002.02031*.

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

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In Lluís Màrquez, Chris Callison-Burch, Jian Su, Daniele Pighin, and Yuval Marton, editors, *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1412–1421. The Association for Computational Linguistics.

Rada Mihalcea and Carlo Strapparava. 2005. Making computers laugh: Investigations in automatic humor recognition. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 531–538. Association for Computational Linguistics.

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

Jeffrey Pennington, Richard Socher, and Christopher D 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.

Jay Pujara, Hui Miao, Lise Getoor, and William Cohen. 2013. Knowledge graph identification. In *International Semantic Web Conference*, pages 542–557. Springer.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3980–3990. Association for Computational Linguistics.

Sushmitha Reddy Sane, Suraj Tripathi, Koushik Reddy Sane, and Radhika Mamidi. 2019. Deep learning techniques for humor detection in hindi-english code-mixed tweets. In *Proceedings of the Tenth Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 57–61.

Orion Weller and Kevin D. Seppi. 2019. Humor detection: A transformer gets the last laugh. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3619–3623. Association for Computational Linguistics.

Diyi Yang, Alon Lavie, Chris Dyer, and Eduard Hovy. 2015. Humor recognition and humor anchor extraction. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2367–2376.

A Data Preprocessing

In the provided dataset, the word to be replaced in the original headline is enclosed between '<' and '/>'. For Subtask 1, we created a column named 'edited' by replacing the word in between '<', '/>' with the 'edit' word in the original headline. Similarly, for Subtask 2 we created two columns 'edited1' and 'edited2' by replacing the words in between '<', '/>' with edit1 word and with edit2 word from 'original1' and 'original2' headlines respectively. We replaced '<' and '/>' with white spaces in the original headline column for Subtask 1 and in 'original1' and 'original2' headlines columns for Subtask 2. The dataset provided for Subtask 2 has samples labeled with 0, 1, 2. Since we need to label the test data with 1 or 2, we have excluded samples labeled with 0 from the training dataset.

B CSD Calculations

The following cases arise while computing CSD1 and CSD2:

Case 1: Target word/ edited word is in the subject/ object/ action. In this scenario, there are three sub-cases.

a. If target word/ edited word is in action, we concatenate subject, object Keywords and then compute the semantic distance between target word/ edited word and concatenated subject, object Keywords.

b. If target word/ edited word is in subject, then we concatenate object, action Keywords and then compute the semantic distance between target word/ edited word and concatenated object, action Keywords.

c. If target word/ edited word is in object, then we concatenate subject, action Keywords and then compute the semantic distance between target word/ edited word and concatenated subject, action Keywords.

Case 2: Target word/ edited word is not in the subject/ object/ action.

If the Target word/ edited word is not in Keywords, then we concatenate subject, object and action Keywords and then compute the semantic distance between target word/ edited word and concatenated subject, object and action Keywords.

In the example discussed in CSD, target word/ edited word is in action. As per case-1(a), we compute CSD1 as semantic distance between target word and concatenated subject, object Keywords and CSD2 as semantic distance between edited word and concatenated subject, object Keywords. Therefore,

$$CSD1 = Sim(AWE(Target\,Word),\ AWE(Concatenated\,Keywords))$$

$$CSD2 = Sim(AWE(Edited\,Word),\ AWE(Concatenated\,Keywords))$$

Here, $Sim(Word_i, Word_j)$ denotes the cosine similarity between the two vector representations. AWE denotes Averaged Word Embedding representation of the words which we obtain using GloVe. The CSD calculations are illustrated in Figure-1.

C Training Details

While training our model on the training dataset, we tuned the hyper-parameters using the development set based on RMSE loss for Subtask 1 and cross-entropy loss for Subtask 2. We used keras[4] backend with tensorflow for our experiments. The default Tokenizer class of Keras has been used for tokenization. Other important model parameters include a maximum sequence length of 30 for padding the headlines, spatial dropout of 0.5 after the embedding layers for both original and edited headlines, batch size of 32 and 30 epochs with early stopping criteria having a patience value of 5. In Subtask 1, the dropout and the recurrent dropout have been set to 0.1 and 0.4 respectively in Bi-LSTM layers. We used dropout layers after each dense layer each with 0.2, 0.2 and 0.1 respectively. Sigmoid activation is used for final dense layer and ReLU activation is used for remaining dense layers. In Subtask 2, the dropout and the recurrent dropout have been set to 0.3 and 0.5 respectively in Bi-LSTM layers. A dropout of 0.3 has been used in the dense layers. ReLU and Softmax activation functions have been used for dense and final dense layers respectively. We have used Adam optimizer (Kingma and Ba, 2015) with default parameters available in keras for both the experiments.

[4]https://keras.io/