Adversarial Perturbations Augmented Language Models for Euphemism Identification

Guneet Singh Kohli
Thapar University, Patiala, India
guneetsk99@gmail.com

Prabsimran Kaur
Thapar University, Patiala, India
pkaur_be18@thapar.edu

Jatin Bedi
Thapar University, Patiala, India
jatin.bedi@thapar.edu

Abstract

Euphemisms are mild words or expressions used instead of harsh or direct words while talking to someone to avoid discussing something unpleasant, embarrassing, or offensive. However, they are often ambiguous, thus making it a challenging task. The Third Workshop on Figurative Language Processing colocated with EMNLP 2022 organized a shared task on Euphemism Detection to better understand euphemisms. We have used the adversarial augmentation technique to construct new data. This augmented data was then trained using two language models, namely, BERT and Longformer. To further enhance the overall performance, various combinations of the results obtained using Longformer and BERT were passed through a voting ensembler. We were able to achieve an F1 score of 71.5 using the combination of two adversarial Longformers, two ad- versarial BERT, 1 non adversarial BERT.

1 Introduction

Euphemisms are mild words or expressions used instead of harsh or direct words while talking to someone to avoid discussing something unpleasant, embarrassing, or offensive. They are often used as a sign of politeness while discussing sensitive or taboo topics (Bakhriddionova, 2021), for instance, using the term "Let go" instead of the word "Fired," using "Put down" instead of "euthanized," or any similar phrase that would make it sound less unappealing or unpleasant (Karam, 2011). Euphemism can also be employed to disguise the truth (Rababah, 2014) to minimize a threatening situation to create a favorable image. For instance, when the phrase "enhanced interrogation techniques" is used, they mean "torture" or use "armed conflict" instead of "war". This figurative behavior of euphemisms makes it ambiguous and challenging for natural language processing techniques to handle these words since they can be interpreted literally in some situations. Moreover, humans might disagree with what constitutes a euphemism (Gavidia et al., 2022).

In the past, many computational approaches for processing have been employed. A sentiment analysis-based approach was used by (Felt and Riloff, 2020) to handle x-phemisms (a term used to refer to both euphemisms and dysphemisms). In their work, they found synonym pairs and used a weakly supervised bootstrapping algorithm to generate semantic lexicon. These lexicons were then used to classify phrases as euphemistic, dysphemistic, or neutral. (Zhu et al., 2021) worked on detecting euphemisms used for dug names on the internet and identifying the terms these euphemisms refer to. Similarly, (Magu and Luo, 2018) also worked on a similar problem statement. However, (Zhu et al., 2021) and (Magu and Luo, 2018) interpreted euphemisms as code words, which is different from those of the shared task organizers. Both (Zhu and Bhat, 2021) and (Zhu et al., 2021) considered the detection and identification of euphemism as a masked language model (MLM) problem where they filtered out words that did not fit their list of euphemisms.

This paper defines our participation in the Euphemism Detection shared task (Lee et al., 2022) organized for the Third Workshop on Figurative Language Processing co-located with EMNLP 2022. We have used the adversarial augmentation technique in combination with transformers to detect euphemisms. A detailed explanation of our proposed approach is given in Section 3.

2 Task & Data Description

Euphemism Detection is a binary classification shared task that focuses on detecting euphemisms in a given input statement. This shared task aims to study the performance of Natural Language Models on euphemisms. The data provided in the shared task comprised a Euphemism Corpus created by

Proceedings of the 3rd Workshop on Figurative Language Processing (FLP), pages 154 - 159

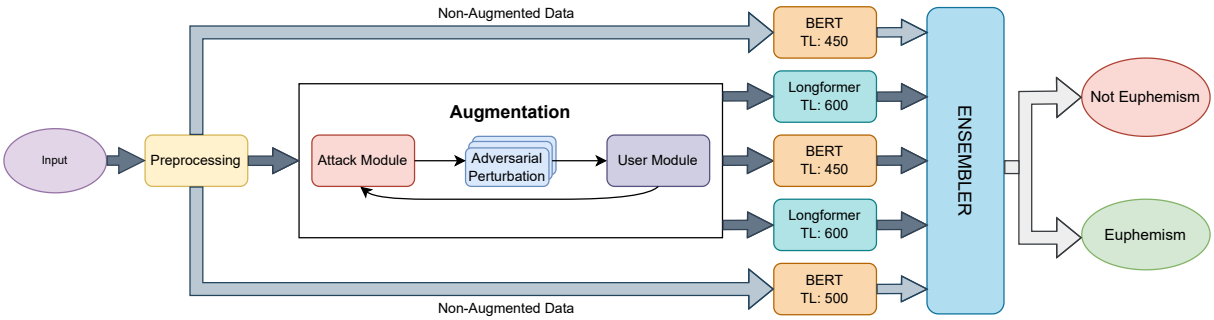

Figure 1: Architecture of the proposed pipeline. Here TL indicates the Token length used for training.

(Gavidia et al., 2022). The raw data used for the creation of this dataset is extracted from the Corpus of Global Web-Based English (GloWbE) (Davies and Fuchs, 2015), which contains text data from websites, blogs, and forums of twenty different English-speaking countries. The training data provided to the participants comprised 1572 sentences, of which 466 were labeled "0," and the rest were labeled "1". The testing data consists of 393 data points. The participants were expected to classify the sentences into "0" (not euphemistic) and "1" (is euphemistic) classes.

3 Methodology

This section gives a detailed description of the pipeline proposed. Section 3.1 and Section 3.2 provides a detailed overview of the preprocessing performed on the data and the augmentation technique used before passing the data through the models. Section 3.4 details the models used for the training.

3.1 Data Pre-Processing

The data in its raw form is often unstructured and comprises punctuations, unusual text, and symbols, which make it unfit for the distillation of correct features causing the model to underperform. Thus, it is essential to preprocess the data before using it. In this paper, we have performed basic preprocessing involving tokenization (splitting the sentences into words), conversion of words into lowercase, removal of stopwords (such as a, the, an), and removal of punctuation and emojis using the NLTK library (Loper and Bird, 2002).

3.2 Augmentation

Deep learning models require a large dataset to produce higher accuracy. However, the training data for the task comprised merely 1572 data points. Moreover, the data was imbalanced (as mentioned in Section 2, which could cause the model to overfit on the predominant label ("1"). Thus, data augmentation was performed for the label "0" of the dataset using the Adversarial attack technique available in TextAttack[1] (Morris et al., 2020). TextAttack iterates through the dataset and generates an adversarial perturbation (changes in the input that causes the model to misclassify) for each correct prediction that the model makes. There are two ways to generate adversarial perturbation:

1. **Visual**: is the method in which a text sequence similar to the original sequence is generated by changing a few characters or introducing realistic "typos" that humans would make.

2. **Semantic**: is the method in which the generated sentence is semantically similar to the original. This is done by paraphrasing the sentence or using synonyms.

TextAttack supports both of these adversarial perturbation techniques. Each attack by TextAttack is built using these four components. The first component is Goal Function that determines whether the attack was successful. Constraints component checks whether the perturbation made preserves the semantics or not.Transformations component generates a set of potential perturbations through deletion, insertion, or substitution of words, characters, and phrases.The Search Method component explores the transformation space to select the best perturbation. The augmentation of the data was done in two steps:

1. **Class Imbalance Removal** : We augment 466 instances of '0' labels with their adversarial representation, which brought the final

[1]https://github.com/QData/TextAttackaugmenting-text-textattack-augment

instance of non-euphemism instances to 932 and total data instances to 2038

2. **Adversarial Augmentation**: We test the dual context training setup discussed in section 3.3. and generate the adversarial version of all the 2038 instances.

3.3 Dual Context training setup

Inspired by the Siamese BERT(Reimers and Gurevych, 2019) , we tried using a dual context setup in which the input given to the language model was as follows:
Input text: *Original instance* $[SEP]$ *Adversarial augmented instance*

Here the input to the language model is the original instance from the training data and the adversarial augmented text instance from the text attack separated by a token $[SEP]$. The following setup aims at leveraging two different perspectives of the same instance to make the model more robust to the other contextual representations of Euphemism. The following structure increased the input length of the system.

In the case of unaugmented data the input text can be understood as follows:
Input text: *Original instance* $[SEP]$ *Original instance*

3.4 Modeling

In this paper, we have used BERT(Devlin et al., 2018) and Longformer (Beltagy et al., 2020) language models to detect euphemisms. This section gives a detailed explanation of their respective architectures.

3.4.1 BERT

Bidirectional Encoder Representations from Transformers (BERT) is used for pre-training deep bidirectional transformers on unlabeled data to develop a language understanding. The sentences are passed through BERT as a sequence of tokens. Before feeding the word sequences, 15% of the words are replaced by [MASK] in each sequence. A [CLS] is appended at the beginning of the first line, and a [SEP] is appended at the end of each sentence. A token, sentence, and positional embedding are added to each token, as shown in Figure 2. The truncation or padding of the sequence is done based on the maximum sequence length used.The maximum sequence length used for each case was determined by finding the average length of the

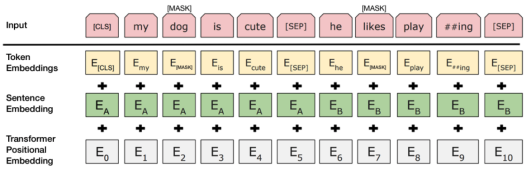

Figure 2: BERT input representations

text in the dataset. These encoded sentences are then passed through the transformer model. This pre-trained can then be fine-tuned d by adding the output layer depending on the task at hand.

3.4.2 Longformer

The major drawback of transformer models like BERT is that they cannot attend to sequence lengths longer than 512. This is because the memory and computational requirements of self-attention grow quadratically. Thus Longformer: The Long-Document Transformer, a transformer whose attention pattern rises linearly with the input sequence, was proposed. To achieve this reduced complexity Longformer combines several attention patterns:

1. **Sliding Window**: Each token in the sequence will only attend to tokens that fall under an arbitrary window whose size is assumed to be *"w"* (w/2 tokens on the right and w/2 tokens on the left). If this is done for *"l"* layers, each token would have attended to $(l * w)$ adjacent tokens. This is the reach of the attention for a given token and is known as the receptive field.
2. **Dilated Sliding Window**: To further improve the performance of the sliding window attention a dilatation of size *"d"* is taken. Here "d" represents the number of gaps between each token in the window. The reception field of this dilated sliding window will be $(l * w * d)$.
3. **Global Attention (full self-attention)**: To ensure support for long-term dependencies, the model utilizes global self-attention where, instead of using three different hidden vectors query (Q), key (K), and value (V), two separate sets of vectors *Q_s, K_s, V_s* (for sliding window), and *Q_g, K_g, V_g* (for global attention) are used.

3.5 Ensembler

To enhance the overall performance of the proposed pipeline, the results obtained by training:

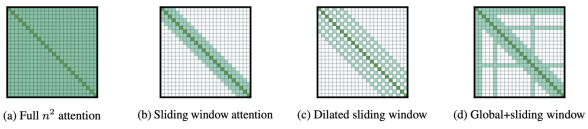

Figure 3: Comparison of transformer self attention and Longformer attention patterns

- Longformer on augmented data with the maximum sequence length of 600
- Longformer on augmented data with the maximum sequence length of 650
- BERT on augmented data with the maximum sequence length of 450
- BERT on the preprocessed data with a maximum sequence length of 450
- BERT on the preprocessed data with a maximum sequence length of 500

are passed through a voting ensembler, and the label with highest frequency is selected as the final label for that sentence, as depicted in Figure 1.

4 Results and Discussion

Euphemism in speech is generally difficult to identify semantically for human beings and thus makes it even more challenging task for the AI to map the understanding and undergo right identification. Our submission aims to handle the task of identifying the Euphemism in a given sentence by modelling Longformer and BERT in an adversarial setup.

4.1 Experimental Setting

To train the language models, we used an 80:10:10 split. We use the default hyperparameters to train BERT and Longformers. We use a learning rate of 1e-5 and an LR scheduler with Polynomial Decay and train the model for three epochs. We use the AdamW optimizer and set the batch size to 4. We trained the models on Tesla P100-PCIE-16GB GPU. The experimentations using the dual context setup yielded lower scores than the other submissions to the leaderboard. We aim to focus more on this proposed methodology and refine our approach for further research into the idea of making a model robust through multiple representation learning.

4.2 BERT Results Analysis

In this section we report our evaluations for the Adversarial Bert and Vanilla BERT. The BERT (TL: 450, UA))[2] yielded the lowest F1 score of 0.667 among all the experimentation. The main aim of our evaluation was to highlight the performance improvement using adversarial augmentation. On close observation in Table 1, it can be noticed that use of augmented data improved the F1 score for BERT to 0.671 (TL:450) and 0.681 (TL:500). The improvement in the F1 scores can be attributed to the robustness introduced by fine tuning the language model on the adversarial examples. It is to be noted that with better hyper parameter tuning the results could have been improved.

4.3 Longformer Results Analysis

The introduction of adversarial augmentation, along with dual context input, increases the average token length of the sentences in the given data set. This increase in average token size highlighted the shortcoming of the BERT model, which can only work up to 512 tokens, and brought about the requirement for Longformers. The results in Table 1 highlight the improvement in the performance of the Longformers. We achieved an F1 score of 0.689 with TL:600 and 0.704 with TL:650. The adversarial examples helped create a more dynamic, robust embedding space for the Longformer to exploit and make better predictions than the BERT. Though Longformer has been known to perform less than BERT in many of the Natural Language Inference tasks in our case, they take the lead and leverage the dual context adversarial setup quite well.

4.4 Ensemble Modelling Results Analysis

We leveraged the individual performance of **Longformers**[3] and **BERT**[4] in a combined way by preparing three different variations of Voting Ensemble to report our results.

1. B(TL:450, UA)+ B(TL:450, A)+B(TL:500, A): The following ensemble yielded a F1 Score of 0.709 which was comparable to the individual performance of the Longformer(TL:650). The ensemble of B perform significantly better than individual B variations by minimum of 4%
2. LF (TL:600, A)+ LF(TL:650, A)+B(TL:500, A): Ensemble of 2 LF and the best individual B variation reported the F1 score of 0.713 which was a slight gain from the LF(TL: 650, A).

[2]'TL' refers to token length, 'UA' refers to Unaugmented Data, 'A' refers to adversarial augmented data

[3]Longformer has been referred to as 'LF' in section 4.3

[4]BERT has been referred to as 'B' is section 4.3

Model and Technique	Precision	Recall	F1 Score
BERT (TL: 450, UA)	0.655	0.702	0.667
BERT (TL: 450, A)	0.660	0.701	0.671
BERT (TL : 500, A)	0.674	0.693	0.681
Longformer (TL : 600, A)	0.681	0.701	0.689
Longformer (TL : 650, A)	0.714	0.699	0.704
Ensemble (Longformer (TL : 600, A)+Longformer (TL : 650, A)+ BERT (TL: 450, UA)+BERT (TL: 450, A)+BERT (TL: 500, UA)	**0.716**	**0.714**	**0.715**
Ensemble (Longformer (TL : 600, A)+Longformer (TL : 650, A)+ BERT (TL: 500, A)	0.708	0.719	0.713
Ensemble (Longformer (TL : 600, A)+BERT (TL: 450, A)+ BERT (TL: 500, A)	0.723	0.702	0.709

Table 1: Experimentation results of model variations. Here 'TL' is maximum token length. 'A' represents that the model was trained on the adversarial augmented data, and 'UA' indicates the model trained on unaugmented data

3. LF (TL:600, A)+ LF(TL :650, A)+ B(TL:450, UA)+ B(TL:450, A)+ B(TL: 500, UA): This ensemble was the best performing submission in this task for our team. The ensemble reported F1 score of 0.715. The results are comparable to the previous ensemble thus highlighting the dominance of LF in ensemble setup

5 Conclusion

In this paper, we proposed an Adversarial Perturbated (TextAttack) BERT and Longformer model, which aims to create a robust model capable of identifying the Euphemism in text. We experimented with different token lengths and eventually created a voting ensemble model that combined our other experiments into a single encapsulation. The ensemble of two adversarial Longformers, two adversarial BERT, and one non-adversarial produced an F1 score of 0.715, which was our best submission. The use of dual context input to the models falls short of the expected performance boost and motivates us to look further into the concept of using multiple representations. We aim to experiment with different methods to combine these representations into a single exemplary representation that can pass into these language models to solve the downstream tasks.

6 Limitations

In this paper, we propose using dual context input with an adversarial training set up to approach the challenge of Euphemism Detection. The approach currently failed to make a significant impact, as reflected by our system performance on the leaderboard. On further analysis, the lack of high performance can be attributed to a selection non ideal set of hyperparameters while training the system. Combining two different contextual representations requires introducing an Attention module or experimenting with other methods to that can result in a better pair encoding of the input.

References

Dildora Oktamovna Bakhriddionova. 2021. The needs of using euphemisms. *Mental Enlightenment Scientific-Methodological Journal*, 2021(06):55–64.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Mark Davies and Robert Fuchs. 2015. Expanding horizons in the study of world englishes with the 1.9 billion word global web-based english corpus (glowbe). *English World-Wide*, 36(1):1–28.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Christian Felt and Ellen Riloff. 2020. Recognizing euphemisms and dysphemisms using sentiment analysis. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 136–145.

Martha Gavidia, Patrick Lee, Anna Feldman, and Jing Peng. 2022. Cats are fuzzy pets: A corpus and analysis of potentially euphemistic terms. *arXiv preprint arXiv:2205.02728*.

Savo Karam. 2011. Truths and euphemisms: How euphemisms are used in the political arena. *3L, Language, Linguistics, Literature*, 17(1).

Patrick Lee, Martha Gavidia, Anna Feldman, and Jing Peng. 2022. Searching for pets: Using distributional and sentiment-based methods to find potentially euphemistic terms. *arXiv preprint arXiv:2205.10451*.

Edward Loper and Steven Bird. 2002. Nltk: The natural language toolkit. *arXiv preprint cs/0205028*.

Rijul Magu and Jiebo Luo. 2018. Determining code words in euphemistic hate speech using word embedding networks. In *Proceedings of the 2nd workshop on abusive language online (ALW2)*, pages 93–100.

John X Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp. *arXiv preprint arXiv:2005.05909*.

Hussein Abdo Rababah. 2014. The translatability and use of x-phemism expressions (x-phemization): Euphemisms, dysphemisms and orthophemisms) in the medical discourse. *Studies in Literature and Language*, 9(3):229–240.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Wanzheng Zhu and Suma Bhat. 2021. Euphemistic phrase detection by masked language model. *arXiv preprint arXiv:2109.04666*.

Wanzheng Zhu, Hongyu Gong, Rohan Bansal, Zachary Weinberg, Nicolas Christin, Giulia Fanti, and Suma Bhat. 2021. Self-supervised euphemism detection and identification for content moderation. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 229–246. IEEE.