CLASSIFICATION OF IMBALANCED DATASET USING BERT EMBEDDINGS

Jeniffer David
Master of Computer Science
Dalhousie University
jd@dal.ca

Jiarong Cui
Master of Computer Science
Dalhousie University
jr333501@dal.ca

Fatemeh Rahimi
Master of Computer Science
Dalhousie University
fatemeh.rahimi@dal.ca

September 4, 2020

ABSTRACT

Online harassment is becoming prevalent as a specific type of communication on Twitter. Considering the huge amount of user-generated tweets each day, the problem of detecting and possibly limiting these contents automatically in real-time is becoming a fundamental problem. But often real-world datasets are imbalanced, comprising predominantly of "normal" examples and less number of "abnormal" ones which causes the learning algorithm to simply generate a trivial classifier that classifies every example as the majority class. To tackle this problem, we use SMOTE to oversample the embeddings of the minority classes where the embeddings are obtained from the BERT pre-trained language model. Finally, we use these oversampled embeddings to train our bi-directional LSTM classifier model to categorize the tweets into four classes: non-harassment, sexual harassment, physical harassment and indirect harassment. Our experiments show that using SMOTE on the top layer representations of BERT significantly improves the F1 score than merely adjusting the class weights.

1 Introduction

A main area of focus in machine learning is classifier models that can identify toxicity in online conversations, especially in Social Media where online harassment is becoming prevalent as a specific communication type in Twitter. Considering the huge amount of user-generated tweets each day, the problem of detecting and possibly limiting these contents automatically in real time is becoming a fundamental problem specifically for female figures who have been harassed for a long time and Twitter was incapable of helping them. In our project, we focus on the detection of "Sexually harassing", "Physically harassing", "Indirectly harassing" or "Non-harassing" tweets in the data set. We implement supervised learning algorithm where the given tweet is analyzed by a model and then the appropriate tags are applied based on its content.

One of the requirements for using Deep Learning would be having enough amount of data for training. As we have only 10,622 tweets in total, it would be a solution for us to use transfer learning. Transfer Learning address the problem of not having enough data for training. Using this idea, a model is trained as a starting point of many downstream NLP tasks. Transfer learning would speed up training and also the performance of the model.

A breakthrough on NLP pretrained models was developed by the Google AI group, a model called BERT [3]. BERT stands for "Bidirectional Encoder Representations from Transformers" which was able to achieve state-of-the-art results on different NLP tasks. It is an autoencoding (AE) language model which uses denoising approach for recovering masked words.

Transformer is a model architecture that uses attention [8]. This model is auto-regressive (AR) and at each time step only looks at the previously generated symbols. Following attention structure, transformers are using stacked self-attention with fully-connected layers for both encoder and decoder. Transformer comes up with Multi-Head attention which consists of several attention layers running in parallel. As mentioned, Building on Transformers, Bert was proposed. Key technical innovation of BERT is applying the **bidirectional** training of transformer [3].

Attention addresses the problem of having a bottleneck vector when passing information from encoder to decoder [1]. It tries to improve the performance by passing all the hidden layers of encoder to the decoder and then calculate weights to assign to these different layers. Summing up these layers attention would help each layer of decoder to focus on the most related layers.

Transformers is built on attention architecture. And BERT is using Transformers architecture. Where all of them tries to find better word representation and more accurate language models. Further in our work we'll be working with both the BERT pretrained model and also using it to create contextual word embeddings for our classification problem.

2 Related work

It is required that the inputs of a Neural network be numbers, so that model be able to perform mathematical operations. Using vectors to represent words is what we call embeddings and was first developed by Mikolov et al. [5]. They call it Word2vec which provide us with non-contextualized word embeddings. Mearning, Same word would have the same representation in different context. On the other hand there are word representations that are different in various contexts and BERT is one of them. BERT learns contextual relations and also long term dependencies between words in a text [3]. The word bank in "river bank" and "bank deposits" would have different representations.

3 Methodology

3.1 Using BERT models

Pre-trained models save us a lot of time, energy and resources. Since our training dataset is small, we chose to use the BERT-base pre-trained model for its size and time complexity. The Uncased-Base version has 12 encoder layers (also called Transformer Blocks), 12 attention heads, 768 hidden units in the feedforward network layers and 110M parameters that were pre-trained.

3.2 Obtaining pre-trained contextual embeddings

The input to the pre-trained model are token representations where each sentence is tokenized by the BERT tokenizer using its corresponding vocabulary ID. BERT uses wordPiece for tokenization, which splits out-of-vocabulary words until it finds a match in the vocabulary (i.e tokenizing "playing" to "play" and "##ing") [9]. These tokens are then converted to a format that the model understands called as Input Features. For each example, the first token is always [CLS] which stands for Classification and the last token denotes the separation of two sequences [SEP]. For classification tasks, the first vector (corresponding to [CLS]) is used as the "sentence vector". And since there is only a single sequence, the type ids of each token is represented as 0. Along with the input token ids and the type ids, BERT also requires mask ids (1 for real tokens and 0 for padding tokens), so that only the real tokens are attended to in the attention layers. The maximum length of each sequence is set to 128 tokens.

Once the input features are generated, it is fed to the Transformer blocks of the model which resembles the vanilla stacked encoder layers of the Transformer. Each encoder consists of two sublayers - self-attention layer and feedforward network layer. The output of the attention layer is passed onto the fully connected feedforward network whose output is then passed onto the next encoder.

Each output of the 12 hidden layer, of each token would be considered as word embeddings. Each layer of BERT captures different features of the input text [2]. For our classifier, we experimented with the effectiveness of features from different layers. Fig.1 shows one such setting where we sum the output representations of the last 4 encoder layers to get the token embeddings. We fix the resultant token embeddings as the output features of all the samples.

3.3 Oversampling with SMOTE

SMOTE stands for *Synthetic Minority Oversampling Technique*. This is a statistical technique for increasing the number of cases in our dataset in a balanced way. The module works by generating new instances from existing minority cases that you supply as input. This implementation of SMOTE does not change the number of majority cases.

The new instances are not just copies of existing minority cases; instead, the algorithm takes samples of the feature space for each target class and its nearest neighbors, and generates new examples that combine features of the target case with features of its neighbors. This approach increases the features available to each class and makes the samples more general.

SMOTE takes the feature embeddings of our entire training dataset as an input, but it increases the percentage of only the minority cases. In our training dataset, the majority class is the "Non-harassment" class and all other classes are minority classes. The counts of the "SexualH", "IndirectH" and "PhysicalH" are increased by synthetically generated samples as shown in Fig.2.

3.4 Classifier model

In the BERT paper [3], the model achieves state-of-the-art results by just using a single-layer neural network as the classifier. The logits of the final feedforward neural network are softmax-ed to calculate the output probabilities. But in order to encode some positional information of the input sequences, we replace the feedforward neural network layer by a bi-directional LSTM layer followed by a couple of dense hidden layers before calculating the softmax probabilities. The input to the LSTM layers are the embeddings extracted from BERT after oversampling. Spatial dropout after the input layer allows the model to dropout entire embedding channels for the purpose of regularization. Our classifier model uses "categorical cross-entropy" as the loss function since we are dealing with multi-class classification. The optimizer "Adam" is used with default parameters because it is currently recommended as the default algorithm to use, and often works slightly better than RMSProp.

4 Experiments

4.1 Dataset

For the project, we use the data set provided by the SIMAH (SocIaL Media And Harassment) competition [7, 6, 4]. This was the first competition on categorizing different types of online harassment language in social media. This dataset has two main classes and three subclasses with labels indicating if a tweet is harassment or non-harassment and if it's harassment, then it also indicates the type of harassment: indirect, physical or sexual. We redefine the dataset into a multi-class classification problem by associating each tweet with one of the four classes: non-harassment, physical harassment, indirect harassment and physical harassment. Training, validation and testing datasets consist of 6374, 2125 and 2123 examples respectively and it is hugely unbalanced. The details of the number of each class is indicated below.

Attributes	Count	Percentage
Non-harassment	3661	57.43%
Harassment	2713	42.5%
IndirectH	55	0.86%
PhysicalH	76	1.91%
SexualH	2582	40.50%
Training Total	6374	100%

Table 1: Training: Highly imbalanced with more than 50% of the examples belonging to one class and less than 1% in another class

4.2 Preprocessing

Tweets usually contain abbreviations, emojis, misspellings, etc. In order to have a classifier that emphasizes on the most important texts some preprocessing has to be done. Here are the steps that were taken to reach that goal:

1. Sometimes people start their tweet with "RT" to indicate that they are reposting someone else's content. As this is not giving us any intuition about the sentiment of the tweets we just delete this word from the dataset.
2. Mentioning starts with co in tweets, we make sure to delete both co and the next word which is the person who is being mentioned.

Attributes	Count	Percentage
Non-harassment	1493	70.25%
Harassment	632	29.79%
IndirectH	71	3.34%
PhysicalH	36	1.69%
SexualH	525	24.70%
Validation Total	2125	100%

Table 2: Validation: Similar to training data, majority class is "Non-harassment" and minority class is "PhysicalH"

Attributes	Count	Percentage
Non-harassment	1512	71.21%
Harassment	611	28.78%
IndirectH	197	9.27%
PhysicalH	100	4.71%
SexualH	312	14.69%
Testing Total	2123	100%

Table 3: Testing: Slightly better than training and validation datasets, but still heavily imbalanced

3. Finding emojis with regular expression and delete them.
4. Verbs are expanded to their full verb form (i.e. the verb "i ve" is expanded to "i have").
5. Replacing abbreviated words such as "idk" to "i do not know".
6. Removing stop words.

4.3 Results

4.3.1 Baseline model

We set the vanilla BERT combined with a feedforward network (the classifier) as our baseline model. We do not make any attempts to address the imbalance problem of the datasets and use the uncased base version of the pre-trained model in all our experiments. As expected, the model makes poor generalization of the training examples and performs badly.

4.3.2 Intermediate model

Next, we replace the feedforward neural network with a bi-directional LSTM classifier for two reasons:

1. LSTM introduces some positional information about the inputs to the classifier.
2. Bi-directional LSTM allows the model to look forwards and backwards. For example, consider the two sentences "I hate this bitch" and "I hate it when someone calls her a bitch" belonging to two different classes. The representations of "hate" must be conditioned on both left and right context, otherwise the model may make false assumptions.

In order to tackle the imbalance problem, we decide to assign different class weights to each class while training. For instance, consider a dataset where majority of examples belong to class A and a minority class B. Then assigning a class weight of A=1 and B=50 means that the model should treat every instance of class B as 50 instances of class A and the loss becomes a weighted average. We compute the class weights using sklearn's compute_class_weight method and then use it to train the classifier model. The results were better than the base model, but we still could not beat the competition top scores **f1=0.41, prec=0.44** and **recall=0.39**.

4.3.3 Final model

Our final model combines BERT, SMOTE and LSTM to provide better results than the previous models. We first extract features from BERT's encoder layers and obtain the embeddings of the training dataset. After getting the embeddings the train, validation and test dataset would have the size of 6374x128x768, 2125x128x768, and 2123x128x768 respectively . Before any changes the train dataset classes which are Non-Harassment, Physical, Indirect, sexual harassment contains 3661, 76, 55, and 2582 examples respectevly. the test dataset contains 1512, 100, 197, and 312 examples respectevly. These embeddings are oversampled using SMOTE to increase the size of minority classes to create a balanced dataset. After oversampling the training dataset each of the classes would have each 3661 examples, which would be 14644 examples intotal.

These new embeddings are fed to the classifier model as multidimensional input sequences and trained for 5 epochs. The classifier model consists of 2 bidirectional LSTM layers with 128 LSTM units and also two dense layers are added on top of that with 512 hidden units. We observe that there is a significant improvement in the F1 score after combining these three techniques for this type of dataset. We also achieve the top F1 score based on the competition results. Table 4 summarizes the results of all 3 models.

Model	F1 score	Precision	Recall
BERT + FNN	0.33	0.42	0.33
BERT + LSTM	0.38	0.35	0.42
BERT + SMOTE + LSTM	**0.47**	**0.55**	**0.47**

Table 4: F1-macro, precision and recall scores for the test dataset of all 3 models

Fixing the final model, we experimented with different settings of the model like extracting different combination of features, increasing the epochs and pre-processing the dataset.

Dataset	Features	Epochs	F1 score	Precision	Recall
Original	Last 4 layers (sum)	5	0.42	**0.53**	0.44
		10	0.41	0.40	0.37
	Last 2 layers (sum)	5	0.40	0.43	0.43
		10	0.41	0.44	0.44
	Last layer	5	**0.48**	0.49	**0.48**
		10	0.43	0.52	0.43
Preprocessed	Last 4 layers (sum)	5	0.47	0.46	**0.45**
		10	0.45	0.41	0.41
	Last 2 layers (sum)	5	0.40	0.48	0.43
		10	0.40	**0.48**	0.39
	Last layer	5	**0.50**	0.41	0.41
		10	0.48	0.43	0.41

Table 5: Performance of different settings of the final model

5 Conclusion

The problem of imbalanced datasets cause serious problems even in the most advanced state-of-the-art language models. Generally, we tackle this problem using some sampling techniques like undersampling, oversampling or a combination of both. In this project, we use SMOTE on the extracted BERT embeddings. Here are some experimental findings:

1. The top layer of BERT generates more useful features for text classification.
2. Preprocessing and cleaning up of the training datasets removes noise and improves the results.

3. A complex classifier is much better than a single output layer in terms of fine-tuning BERT.

6 Future work

The reason we chose SMOTE is because it is a very popular technique. In the future, we plan to try more advanced versions that produce less noisy samples like borderline-SMOTE, SVM-SMOTE or ADASYN to approach the problem from a slightly modified perspective. If the training dataset size is large, we can also replace the base version with the large version of BERT which has 24 encoder layers, 1024-hidden, 16-heads and 340M parameters for more stable results. Another fine-tuning we plan to do in the future is to further pre-train BERT with target domain data. In the future, we will produce more insight of BERT on how it works for imbalanced datasets.

References

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[2] Yige Xu Xuanjing Huang Chi Sun, Xipeng Qiu. How to fine-tune bert for text classification? *arXiv preprint arXiv:1905.05583*, 2019.

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

[4] Borna Jafarpour, Stan Matwin, et al. Boosting text classification performance on sexist tweets by text augmentation and text generation using a combination of knowledge graphs. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pages 107–114, 2018.

[5] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[6] Sima Sharifirad, Borna Jafarpour, and Stan Matwin. How is your mood when writing sexist tweets? detecting the emotion type and intensity of emotion using natural language processing techniques. *arXiv preprint arXiv:1902.03089*, 2019.

[7] Sima Sharifirad and Stan Matwin. When a tweet is actually sexist. a more comprehensive classification of different online harassment categories and the challenges in nlp. *arXiv preprint arXiv:1902.10584*, 2019.

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

[9] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.