Toxic Comment Detection

Xiaoyue Lin, Yuling Zheng, Xiyan Zhang
xiaoyuel4, yulingz3, xiyanz@student.unimelb.edu.au

Abstract

This report presents the task of classifying toxic comments using Machine Learning models, which can be applied in censoring inappropriate comments on social media platforms. The dataset used for training and testing is obtained from Jigsaw Unintended Bias in Toxicity Classification competition on Kaggle (Jigsaw, 2019a). The models we explore and experiment in this work are Multinomial Naive Bayes (or NB), Support Vector Machine (or SVM), Long Short-Term Memory network (or LSTM) and Bidirectional Encoder Representations from Transformers (or BERT). Furthermore, the evaluation of the classification problem has also performed on comments related to different identities. Apart from common evaluation methods for binary classification problems, the false alarm rate is also adopted for a more comprehensive analysis of the models. Overall, LSTM and BERT are the two most effective models for the task with similar accuracy and F1 scores. In practice, the choice of the best model can depend on the level of censorship. Further improvement in model performance can be expected if more computational resources are available in the future.

1 Introduction

Online discussion allows people to share their opinions openly at any time, which is usually constructive for human life (Ozoh, Adigun, & Olayiwola, 2019). However, divergences in opinions can result in debates. While some can remain civil, others emboldened by online anonymity may comment in aggressive or offensive languages (Rainie, Anderson, & Albright, 2017). These are toxic comments which are usually defined as messages left by another Internet user, which can involve profanity, racism or harassment, and with the purpose to slow down or interrupt an online discussion (Brassard-Gourdeau & Khoury, 2020; Tomaiuolo, Lombardo, Mordonini, Cagnoni, & Poggi, 2020). Studies have shown that 17% of internet users have received toxic messages with a proportion of 19% targeting women, 24% targeting the poor and 34% targeting the homosexual community. Unfortunately, 10% of people develop depression or suicidal thoughts as a result of toxic messages.

We are motivated to use Machine learning algorithms to transform online discussion platforms to be more respectful. The aim of this project is to identify toxic comments from a vast amount of comment texts so that they can be filtered out based on the censorship of the platforms.

2 Related work

The challenge of automatically detecting the toxicity in online comments has recently been the subject of many Natural Language Processing studies (Brassard-Gourdeau & Khoury, 2020). The tasks of cyberbully detection and hate speech detection have been covered by various studies and approached successfully by Support Vector Machine (or SVM), logistic regression and neural networks.

The Conversation AI team have built models to protect voice from toxicity (Jigsaw, 2019a). They have noticed that the models incorrectly predicted a higher score of toxicity for comments containing

frequently attacked identities, such as gay or female. The performance of models on different identities of previous studies is not clear.

In this work, we used two models that we learned from previous subjects, NB and SVM, and then switched to explore the power of deep learning models, LSTM and BERT. Our goal is to classify comments into toxic and non-toxic. We are inspired to fit models for better performance in recognizing toxicity towards vulnerable identities and adapt models for different online communities in real-world applications. We focus on identifying a more effective model for our goal, which is able to avoid letting the guilty ones slip as well as accusing the innocent ones.

3 Dataset

The dataset 'Multilingual Toxic Comment Classification' from Kaggle is used, which contains 1804874 comments from Wikipedia's talk page (Jigsaw, 2019b). Each comment is given a toxicity score between 0 and 1. A score that is greater than or equal to 0.5 is considered toxic. As shown in Figure 1, the data is unbalanced, with 8% being toxic. In addition, each comment is labelled with various identity attributes, such as "female", "male" and "black". These are the identities mentioned in the comments. In this work, we investigate The six frequently attacked identities including "psychiatric or mental illness", "homosexual", "black" and "female". The counts of toxic and non-toxic comments of each identity are shown in Figure 2. The counts and the distributions of toxicity are similar.

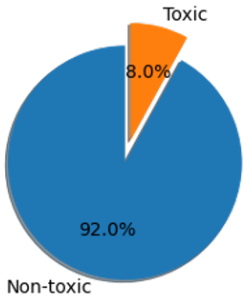

Figure 1: Class distribution

The comments extracted from the dataset are pre-processed before training. The steps include converting all text to lowercase, replacing special characters and numbers with whitespace, and removing duplicates. The text data are lemmatized and tokenized. Comments that are shorter than 0 or longer than 242 which is the 3 times interquartile range higher than the third quartile.

The whole dataset is split into 60% train, 20% validation and 20% test. To obtain a clearer comparison among models, the same training, validation and test datasets are used across all models. The validation set is used for hyperparameter tuning and the test set is used for evaluating and comparing model performance.

4 Methods

4.1 Models

4.1.1 NB

An NB model is set as the baseline model for the project, which is intuitive and "suitable for classification with discrete features" (Scikit Learn, n.d.). Before fitting with NB, comments are transformed using Count Vectoriser into vector form. This assumes the independence of each word since the comments are transformed to a matrix. The NB is also a probabilistic model which calculates the individual probabilities of different conditions. For example, sentences with many swear words will result in higher probabilities to be classified as toxic comments since any single swear word would have a higher marginal probability. Furthermore, NB has the advantages of requiring less computational resources and not requiring hyperparameter tuning, which can be a good baseline

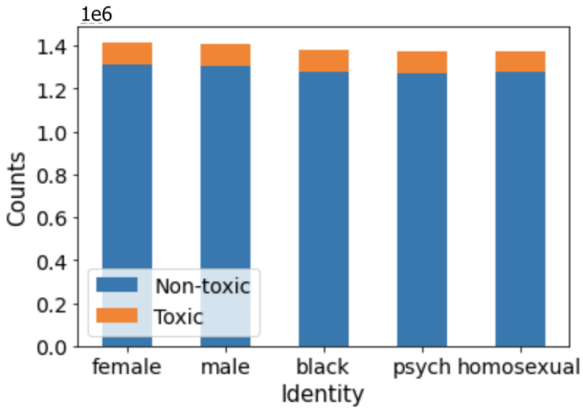

Figure 2: Counts of toxic and non-toxic comments by identity

model. However, a limitation of NB can be its assumption of independence between each word, which may not be true in comments. Hence, more advanced models will be introduced and examined in the following sections.

4.1.2 SVM

The second model used is Support Vector Machine (or SVM), it is a model that determines the best decision boundary between vectors belonging to different groups. In Figure 3, the black solid line separating the circles and triangles, which maximises the margin is considered as the best hyperplane and the margins indicate distances between each vector and the hyperplane. This means that the texts need to be converted to vector form as well before fitting into the model and Term Frequency-Inverse Document Frequency (or TF-IDF) is used to transform comments here. TF-IDF is a method to score the importance of words in a document based on how frequently they appear across multiple documents (Gupta, 2019). SVM is suitable when performing with linearly separable data or datasets with many features and high dimensions. However, the performance of plain SVM is often limited when the dataset is enormous or not linearly separable.

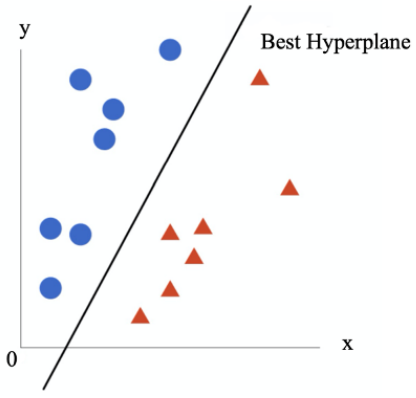

Figure 3: An example of SVM

4.1.3 LSTM

Long Short-Term Memory (LSTM) network is a specific kind of Recurrent Neural Network (RNN) (Ghosh, et al., 2016). An LSTM model learns each word based on its understanding of previous words similar to humans rather than developing understandings from scratch of the text whenever it reads the next word (Olah, 2015). Also, unlike NB, LSTM assumes the dependencies in the data, which makes it popular in learning sequential data like text (Tao & Liu, 2018). In previous studies,

LSTM networks have achieved impressive performance in capturing the long-distance contextual dependencies in text (Wang, Liu, Luo, & Wang, 2018). Therefore, LSTM can be well suited to our task.

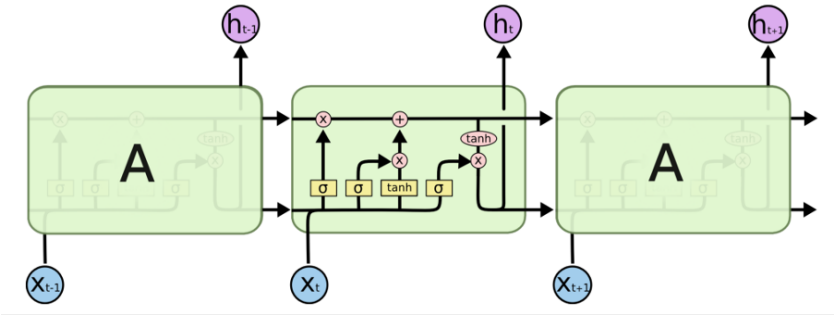

Figure 4: LSTM networks. Aparted from "Understanding LSTM Networks" by C. Olah, 2015 (https://colah.github.io/posts/2015-08-Understanding-LSTMs/).

A LSTM network has the form of a chain of repeating neural network modules (Olah, 2015). In Figure 4, A represents a chunk of neural network, which takes inputs $x_t$and outputs h_t. The yellow boxes are neural network layers with the activation function labelled inside. Pointwise operations are represented by pink circles. The horizontal line running through the top of the diagram is the cell state in a LSTM network, which carries information throughout the entire chain of the network, enabling long-term memory. Three different gates in the network are neural net layers which remove or add relevant information to the cell state. A forget gate is a sigmoid layer (σ), which is represented by the first yellow box on the left and decides what information to be discarded. An input gate is represented by the sigmoid and tanh boxes in the middle, which decides how the cell state is updated. The last gate is the output gate represented by the sigmoid layer on the right and the pink tanh circle, which decides the output value. The cell state passes through a tanh function and is multiplied by the output of the sigmoid layer.

Before training an LSTM network, the input text data are vectorised and converted into a word embedding matrix created by Crawl and GloVe word embedding methods, which contains 2 million and 2.2 million vocabulary respectively (Mikolov, Grave, Bojanowski, Puhrsch, & Joulin, 2018; Pennington, Socher, & Manning, 2014). Since the dependencies in sequential data like text are usually not only in one direction, bidirectional layers are used in the LSTM model. A binary cross-entropy loss function defined in Equation 1 is used to monitor the performance of classifying data into two groups C_1 and C_2at the end of each epoch. To reduce overfitting, layer outputs are randomly dropped out with a dropout rate of p (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014). In addition, early stopping is applied to the model to halt the training when no further improvement in the validation loss is obtained.

$$Binary \quad Cross-Entropy \quad = \quad -t_1 \log(s_1) - (1-t_1)\log(1-s_1) \tag{1}$$

4.1.4 BERT

BERT is a state-of-art language model that has achieved high performance through only fine-tuning the last layer (Horev, 2018). It is also a bidirectional model where the transformer's encoder reads the entire sequence of the words at once. Unlike LSTM, the Transformers use attention mechanisms, which allows them to understand the context in which the word is being used (Vaswani, et al., 2017).

The input of the BERT model is tokenized with the WordPiece tokenizer (Shulga, 2019). The tokenizer consists of a vocabulary of 28,996 commonly used words. For words that are not available in the vocabulary, BERT segments words into sub-word levels (Sennrich, Haddow & Birch, 2016). This means subwords are in the vocabulary and rare words are not. The training data is shuffled to ensure each training batch contains both toxic and non-toxic comments.

BERT can be described as a trained Transformer Encode, as shown in Figure 5. In the Transformer, it takes a vector (a sequence of words) as the input, and flows through from the first encoder layer up to the last layer in the stack. "Distillbert-based-cased" model is used in this project, which consists of

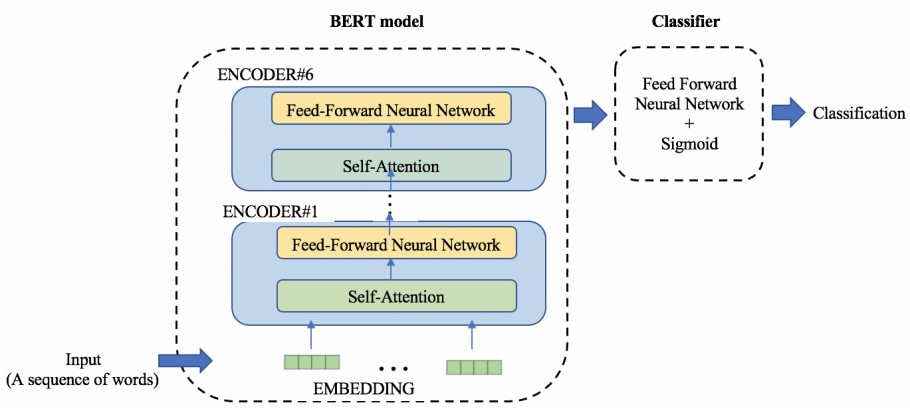

Figure 5: BERT model

six encoder layers. Each layer in the stack uses a self-attention method. It allows the encoder to learn the meaning and the intention of the word based on its entire input sequence as it encodes a specific word. Then, the layer is fed to a feed-forward neural network. There are 768 hidden units and 12 attention heads for the model (pretrained models, n.d.). The output of the model is a fully connected dense layer. Lastly, a layer with sigmoid function is used to compute the label probabilities.

4.2 Undersampling

Class imbalance and redundant training samples can affect classification accuracy (Hernandez, Carrasco-Ochoa, & Martínez-Trinidad, 2013). This problem has been handled by applying undersampling in previous studies, which randomly removes some data of the major class from the training set. This technique is usually preferred when a large amount of data is available. A downside may be causing underfitting as some valuable data are possibly removed when undersampled. Since we have a fairly large training set of 1411932 rows after pre-processing and splitting, we resample 111411 rows, the same amount as the toxic comments from the non-toxic comments to handle the class imbalance.

4.3 Evaluation Metrics

The performance of models on the binary classification tasks is compared through accuracy, precision, recall and F1 score. Performance on detecting toxic comments associated with several frequently attacked identities is measured separately by recall and false alarm rate which is defined in Equation 2. Recall can measure the model capacity in recognizing toxic comments while false alarm rate can measure the probability of the model wrongly accusing the comments related to the identity.

$$False\ alarm\ rate \quad = \quad \frac{False\ positive}{False\ positive + True\ negative} \tag{2}$$

5 Experiments

The experiments in SVM, LSTM and BERT are described in this section. The NB model as the baseline model, does not include any hyperparameter tuning.

5.1 SVM

Hyperparameter tuning is used to optimise the performance of SVM. This includes the maximum features number used (“max_features”), ngram ranges (“ngram_range”) in TF-IDF, C values and maximum iteration times (“max_iter”) for SVM. Furthermore, the method of pipeline and grid search is utilised here to tune the hyper-parameter. Additionally, english stop words are filtered out before training and a linear kernel is adopted.

Table 1: Model performance in detecting toxic comments

Model	Accuracy	Precision	Recall	F1 score
NB	0.9076	0.4633	0.5520	0.5038
SVM (C=10, max_iter=1000, "TF-IDF": max_features = 50000, ngram_range = (1, 2))	0.8698	0.2585	0.2849	0.2710
LSTM (batch size = 5000)	0.9452	0.6789	0.6733	0.6761
LSTM (batch size = 512)	0.9495	0.7399	0.6253	0.6778
LSTM (batch size = 512, Undersampling)	0.9020	0.8644	0.4593	0.5999
BERT (batch size = 64)	0.9408	0.6719	0.5440	0.6012
BERT (batch size = 32)	0.9416	0.7393	0.4836	0.5847

5.2 LSTM

In our LSTM models, the value of batch size is tuned to seek a balance between shorter computational time and better generalisation (Hoffer, Hubara, & Soudry, 2017). It has been found that using a large batch size can easily exceed the limit of CPU memory. Batch size values of 512 and 5000 are experimented. Additionally, the under-sampling technique is experimented in a LSTM model to seek further improvement. The LSTM networks are instantiated with 2 hidden layers. It has been shown that 2 layers were enough for learning text data and conducting a binary classification task text (Tao & Liu, 2018). While stacking LSTM layers can increase the modelling power, it dramatically increases the training time. The dropout rate is set to 0.5 since the probability of retaining a unit is typically chosen in the range 0.5 to 0.8 (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014). Choosing a large dropout rate can lead to underfitting and a small one may not be sufficient to reduce overfitting. The LSTM models are fitted for 20 epochs with early stopping.

5.3 BERT

Due to the limitation of time, the BERT model is only trained with one epoch. One epoch can avoid overfitting and considering that we have more than one million of data, it might be already promising to yield a good result as BERT is known to be more efficient than many other simpler models suggested in a study by Komatsuzaki (2019). Batch size values of 32 and 64 are experimented. Adam is used as the optimizer as it is suggested that it learns fast and is more stable than other optimers (Mack, 2018).

6 Results

The results of fitted models are presented in this section for model comparison. The performance is measured and compared in terms of accuracy, precision, recall and F1 score. For SVM, only the result of the best hyperparameter combination produced by Grid Search is presented, which has the highest accuracy among all fitted SVM models. Based on the accuracy, the best models of each algorithm are further compared by evaluating the performance in detecting toxic comments related to frequently attacked identities. “Psychiatric or mental illness”, “homosexual”, “black” and “female” are chosen as examples.

Table 1 presents the performance of fitted models in detecting toxic comments regardless of identities. For the toxic comment detection task, the LSTM model with a batch size of 512 has achieved the highest accuracy (0.9495) and F1 score (0.6778). In terms of precision, the LSTM model with undersampled majority class has outperformed others (with a precision score of 0.8644). Additionally, the LSTM model with a batch size of 5000 is able to achieve the highest recall (0.6733). Overall, the LSTM models have outperformed NB, SVM and BERT. Both LSTM and BERT models have been tuned by varying the batch size. It has been found that reducing the batch size can lead to a higher accuracy, a higher precision but also lower recall. Due to class imbalance, accuracy score may be a misleading indicator for measuring performance. The goal of our task is to recall toxic comments as much as possible and also avoid wrongly blocking non-toxic comments which can cause unsatisfying user experience, therefore, F1 score which is calculated by weighted precision and recall, is also considered when determining the best model of each algorithm and overall. Thus, we can conclude

Table 2: Model evaluation on selected identities

Identity	Model	Recall	False Alarm Rate
psychiatric or mental illness	NB	0.5368	0.0493
	SVM	0.2947	0.0769
	LSTM (batch size = 512)	0.6996	0.1100
	BERT (batch size = 64)	0.5049	0.0623
homosexual gay or lesbian	NB	0.5614	0.0646
	SVM	0.2865	0.0854
	LSTM (batch size = 512)	0.4906	0.0887
	BERT (batch size = 64)	0.5049	0.0623
black	NB	0.5387	0.0599
	SVM	0.3063	0.0777
	LSTM (batch size = 512)	0.6136	0.1328
	BERT (batch size = 64)	0.5014	0.1191
female	NB	0.5433	0.0612
	SVM	0.2722	0.0769
	LSTM (batch size = 512)	0.5688	0.0450
	BERT (batch size = 64)	0.4509	0.0313

that the LSTM model with a batch size of 512 has performed best regardless of identities. Also, a batch size of 64 is considered to be a better option for the BERT model relatively.

Table 2 presents the model performance in detecting toxic comments related to different identities. The results show that NB has the lowest false alarm rate when detecting toxicity related to psychiatric or mental illness and black people. The BERT model has achieved the lowest false alarm when detecting toxic comments related to the homosexual and female. In terms of recall, it can be noticed that the LSTM model has the highest score when recognizing toxicity related to several identities except for the homosexual.

7 Discussion

7.1 Model comparison

It can be unexpected that NB, our baseline model has outperformed SVM. The possible reason for this might be the large size and high dimensionality of the dataset. As a result, the dataset can be noisy with overlapping classes. Even by limiting the maximal features, the dataset may still be too large for SVM to train and find the best hyperplane. Also, the dataset may be not linearly separable as we has assumed. However, the NB baseline model treats vectorised form of words as multinomial features and calculates the probability of different scenarios individually, which is found to be less expensive and more robust than our SVM models.

The best model among LSTM models and the best one among BERT models have achieved similar performance. While it is claimed that BERT can be better at capturing context by learning words in all positions, our LSTM model slightly outperformed the BERT model. We have concluded that the LSTM model is the best model for our task, however, we cannot conclude that LSTM networks are more successful in detecting toxic comments than BERT. Apart from different batch sizes chosen in models, this is because LSTM has used a larger word embedding yielded by Crawl and GloVe, which can potentially capture more useful words for the training. In addition, we trained the BERT model with only one epoch. This may be underfitted compared with the LSTM model. We can expect that longer training time and more computational resources can improve the performance of BERT.

Both of the deep learning models have performed better compared to NB and SVM. The improvement in training capacity can be explained by that deep learning learns multiple levels of representation of the text. Since words and phrases that compose sentences are usually interrelated in a context, LSTM and BERT can do a better job in learning the sequential data (Di, Bhardwaj, & Wei, 2018). Furthermore, the lower level of representation in deep learning often can be shared across tasks, which is a great way to boost the training efficiency.

7.2 Class imbalance remedy

Furthermore, to handle the imbalanced dataset. Undersampling is adopted on non-toxic comments, which transforms the dataset to have an equal number of toxic and non-toxic comments. By comparing the LSTM model fitted with undersampled data with the LSTM model with the same batch size, only the precision score has been improved and the recall score has dropped. That is, the model becomes more conservative in recognizing toxic comments. Such technique can be unfavourable for our task considering the decrease in overall accuracy score, recall and F1 score. After undersampling, the size of training has significantly shrunk from 1411933 to 222822. Although we still have a fairly large amount of training set, this can potentially discard useful information. Thus, undersampling may be less competitive with other models training on an imbalance but much larger dataset.

7.3 Real-world applications

The best model may depend on the expectation of the stakeholders. In general, based on the result of overall accuracy and F1 score, LSTM can be more efficient and effective for the task compared with BERT. If we also consider the balance between recalling and preventing false alarms, stakeholders should be alerted that LSTM which has achieved higher recall but also higher false alarm rates may be too hypercritical for platforms with lower censorship. This may be suitable for stakeholders who are aiming for the least amount of toxic comments. In reality, our stakeholders, social media platforms may have various strategies regarding filtering toxic comments. It is recommended to forums or websites that are more rigorous in censoring toxicity to adopt models with higher recall. Platforms which prefer to allow more freedom in speech are recommended to adopt models with lower false alarm rate. In addition, different models have yielded different results when evaluated by separating identities. Based on our analysis, different advice can be provided to platforms targeting different communities. For example, in practice, a website dedicating to connect homosexual people and build a less restritive environment will be adviced to adopt a BERT model which may cause the least amount of false alarms.

7.4 Future improvement

Most difficulties in hyperparameter tuning, model fitting and surpassing our own results can be attributed to the limitations of time and computational resources when the complexity of models increases. In the future, more hyperparameters such as the kernel type of SVM, number of hidden layers in LSTM, and maximum features used in training can be experimented if more time and resources are available. Additionally, by fitting for more epochs and preventing overfitting by early stopping at the same time, we can expect possible improvements in the BERT's performance. To better compare the results of different models, we can use the same word embedding matrix for training if possible. When handling the imbalanced class distribution in the dataset, more robust methods, such as the oversampling techniques, a weight-loss LSTM or BERT model, can be experimented for further improvement.

8 Conclusion

In conclusion, we have implemented model fitting using NB, SVM, LSTM and BERT. The models have been evaluated on classifying comments and by separating identities. It can be concluded that the LSTM model which is able to achieve an accuracy and F1 score of 0.9495 and 0.6778 respectively, is our best model regardless of identities. Specifically, the BERT model achieves a lower false alarm rate in detecting toxic comments related to the homosexual and female, however, it requires much longer training time and more computational power than other models. Meanwhile, SVM failed to surpass the baseline. In practice, the most suitable model can be dependent on the censorship and population composition of different online platforms. In the future, we aim to further improve the performance and better handle class imbalance by experimenting on more hyperparameters and sophisticated techniques.

References

Brassard-Gourdeau, É., & Khoury, R. (2020, June 17). *Using Sentiment Information for Preemptive Detection of Toxic Comments in Online Conversations [Working paper].* Retrieved from http://arxiv.org/abs/2006.10145

Di, W., Bhardwaj, A., & Wei, J. (2018). *Why Deep Learning is perfect for NLP (Natural Language Processing).* Retrieved October 24, 2020, from https://www.kdnuggets.com/2018/04/why-deep-learning-perfect-nlp-natural-language-processing.html#:~:text=Deep%20learning%20learns%20multiple%20levels,can%20be%20shared%20across%20tasks

Ghosh, S., Vinyals, O., Strope, B., Roy, S., Dean, T., & Heck, L. (2016). *Contextual LSTM (CLSTM) models for Large scale NLP tasks [Working paper].* Retrieved from http://arxiv.org/abs/1602.06291

Gupta, R. K. (2019, November 9). *An Introduction to TF-IDF.* Retrieved from Analytics Vidhya: https://medium.com/analytics-vidhya/an-introduction-to-tf-idf-using-python-5f9d1a343f77

Hernandez, J., Carrasco-Ochoa, J. A., & Martínez-Trinidad, J. F. (2013). An Empirical Study of Oversampling and Undersampling for Instance Selection Methods on Imbalance Datasets. *Iberoamerican Congress on Pattern Recognition* (pp. 262-269). Springer, Berlin, Heidelberg. doi:https://doi.org/10.1007/978-3-642-41822-8_33

Hoffer, E., Hubara, I., & Soudry, D. (2017). Train longer, generalize better: closing the generalization gap in large batch training of neural networks. *Advances in Neural Information Processing Systems*, 1729-1739. Retrieved from http://arxiv.org/abs/1705.08741

Horev, R. (2018, November 11). *BERT Explained: State of the art language model for NLP.* Retrieved from towards data science: https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270

Jigsaw. (2019a). *Jigsaw Unintended Bias in Toxicity Classification.* Retrieved from Kaggle: https://www.kaggle.com/c/jigsaw-unintended-bias-in-toxicity-classification/overview

Jigsaw. (2019b). *Jigsaw Unintended Bias in Toxicity Classification* [Data file]. Retrieved from Kaggle: https://www.kaggle.com/c/jigsaw-unintended-bias-in-toxicity-classification/data

Komatsuzaki, A. (2019). *One Epoch Is All You Need* [Working paper]. Retrieved from http://arxiv.org/abs/1906.06669

Mack, D. (2018, April 16). *How to pick the best learning rate for your machine learning project.* Retrieved from freeCodeCamp: https://www.freecodecamp.org/news/how-to-pick-the-best-learning-rate-for-your-machine-learning-project-9c28865039a8/

Mikolov, T., Grave, E., Bojanowski, P., Puhrsch, C., & Joulin, A. (2018). Advances in Pre-Training Distributed Word Representations. *International Conference on Language Resources and Evaluation.*

Olah, C. (2015, August 27). *Understanding LSTM Networks.* Retrieved from colah's blog: https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Ozoh, P. A., Adigun, A. A., & Olayiwola, M. O. (2019). Identification and Classification of Toxic Comments on Social Media using Machine Learning Techniques. *International Journal of Research and Innovation in Applied Science,* 142-147.

Pennington, J., Socher, R., & Manning, C. D. (2014, August). *GloVe: Global Vectors for Word Representation.* Retrieved from The Stanford Natural Language Processing Group: https://nlp.stanford.edu/pubs/glove.pdf

Pretrained models. (n.d.). Retrieved from huggingface: https://huggingface.co/transformers/pretrained_models.html

Rainie, L., Anderson, J., & Albright, J. (2017, March 29). *The Future of Free Speech, Trolls, Anonymity and Fake News Online.* Retrieved from Pew Research Center: https://www.pewresearch.org/internet/2017/03/29/the-future-of-free-speech-trolls-anonymity-and-fake-news-online/

Scikit Learn. (n.d.). Retrieved October 2020, from sklearn.naive_bayes.MultinomialNB: https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html

Sennrich, R., Haddow, B., & Birch, A. (2016). *Neural Machine Translation of Rare Words with Subword Units.* Retrieved from arXiv: https://arxiv.org/abs/1508.07909

Shulga, D. (2019, June 6). *BERT to the rescue.* Retrieved from towards data science: https://towardsdatascience.com/bert-to-the-rescue-17671379687f

Singh, T. (2020). *Deep Learning For NLP: Zero To Transformers & BERT.* Retrieved from Kaggle: https://www.kaggle.com/tanulsingh077/deep-learning-for-nlp-zero-to-transformers-bert

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research, 15*(56), 19291958. Retrieved from http://jmlr.org/papers/v15/srivastava14a.html

Tao, F., & Liu, G. (2018). Advanced LSTM: A Study about Better Time Dependency Modeling in Emotion Recognition. *2018 IEEE International Conference on Acoustics, Speech and Signal Processing* (pp. 2906–2910). Calgary: IEEE.

Tomaiuolo, M., Lombardo, G., Mordonini, M., Cagnoni, S., & Poggi, A. (2020). A Survey on Troll Detection. *Future Internet, 12*(2), 31.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., & Kaiser, L. (2017). *Attention Is All You Need.* Retrieved from arXiv: https://arxiv.org/abs/1706.03762

Wang, J.-H., Liu, T.-W., Luo, X., & Wang, L. (2018). An LSTM Approach to Short Text Sentiment Classification with. *The 2018 Conference on Computational Linguistics and Speech Processing* (pp. 214-223). Hsinchu: The Association for Computational Linguistics and Chinese Language Processing.