Improving Mental Health using Machine Learning to Assist Humans in the Moderation of Forum Posts

Dong Wang[1], Julie Weeds[1] and Ian Comley[2]
[1]*Department of Informatics, University of Sussex, Brighton, BN1 9RH, U.K.*
[2]*MeeTwo Education Ltd, 17 Princelet Street, London, E1 6QH, U.K.*

Keywords: Machine Learning, Natural Language Processing, Mental Health, Online Forum Moderation, Data Augmentation, BERT, LSTM.

Abstract: This work investigates the potential for the application of machine learning and natural language processing technology in an online application designed to help teenagers talk about their mental health issues. Specifically, we investigate whether automatic classification methods can be applied with sufficient accuracy to assist humans in the moderation of posts and replies to an online forum. Using real data from an existing application, we outline the specific problems of lack of data, class imbalance and multiple rejection reasons. We investigate a number of machine learning architectures including a state-of-the-art transfer learning architecture, BERT, which has performed well elsewhere despite limited training data, due to its use of pre-training on a very large general corpus. Evaluating on real data, we demonstrate that further large performance gains can be made through the use of automatic data augmentation techniques (synonym replacement, synonym insertion, random swap and random deletion). Using a combination of data augmentation and transfer learning, performance of the automatic classification rivals human performance at the task, thus demonstrating the feasibility of deploying these techniques in a live system.

1 INTRODUCTION

Mental health problems are now very common in the UK. Approximately 1 in 4 people in the UK will experience a mental health problem each year (McManus et al., 2009). Further, in England alone, 1 in 6 people report experiencing a common mental health problem (such as anxiety and depression) in any given week (McManus et al., 2016). This pressing need to help people with mental health problems has given rise to the growing number of initiatives and organizations working in the area. MeeTwo Education is a social enterprise which has, since 2016, been working to reduce the number of mental health issues experienced by young people. Their strategy is to provide an online app where young people can safely share problems and receive advice and support from both professionals and peers.

The users in the MeeTwo scenario can be, due to age and personal nature of posts, very vulnerable. Thus it is essential that they are protected from the potential negative impacts of un-moderated posts to this online forum. For example, the forum prohibits posts containing offensive language or cyberbullying as well as posts containing personal information, since these could let other users identify an individual, posing dangers to them in the non-virtual world. Furthermore, posts which indicate that a user is in danger, for example potentially suicidal, are redirected to a trained professional rather than being left to peers. Currently, all posts to the MeeTwo forum are moderated by trained human moderators. But, as the number of users and associated posts increases, so does the workload for moderators. This increases the cost to run the service and potentially, the delay between a post being made by a user and it appearing online, which has a negative impact on the experience of users. We posit that whilst a fully automated moderation system is unlikely to be able to deal with complex edge cases, there is scope for employing a semi-automated moderation system which pre-labels posts before they are presented to human moderators. Such a semi-automated system could greatly reduce the workload of human moderators, accelerate the moderation process, avoid some low-level human errors and ultimately enhance the experience of users.

The potential benefits of semi-automated moderation are not limited to the specific use-case consid-

Wang, D., Weeds, J. and Comley, I.
Improving Mental Health using Machine Learning to Assist Humans in the Moderation of Forum Posts.
DOI: 10.5220/0008988401870197
In *Proceedings of the 13th International Joint Conference on Biomedical Engineering Systems and Technologies (BIOSTEC 2020) - Volume 5: HEALTHINF*, pages 187-197
ISBN: 978-989-758-398-8; ISSN: 2184-4305

ered so far. More generally, most of the posts and comments on any websites, online forums or social media have the chance of containing some improper content. Following on from earlier research with similar results, a survey of over 4000 Americans in 2017 found that roughly 40% of their respondent had personally experienced some form of online harassment (from offensive name-calling to stalking and sexual harassment) and over 60% of them consider it to be a serious problem (Duggan, 2017). Many companies and non-profit organisations only have a limited budget for hiring human moderators and, thus, a semi-automated moderation model may be useful. With an automated system pre-reviewing the content, a human is only required to check cases which the system cannot place with high probability in either "accepted" or "rejected" categories. In the long-term, the accuracy of such a system may become equal or even better than that of a human moderator, who may be prone to slips of concentration or inconsistency. In this case, the system might be viewed as an automated moderation model which is able to directly 'accept' to 'reject' the posts. In any case, the workload of human moderators is decreased and user experience is increased.

Here, specifically, we explore the feasibility of building a semi-automated moderation model that is able to reduce the moderation workload and accelerate the moderation speed in the MeeTwo scenario. To ensure high accuracy, the aim is to assist human moderators, rather them replace them, in moderating the posts and replies on the MeeTwo platform. Working in a real-world scenario means that there are challenges not always present in academic studies using large datasets which are carefully controlled and balanced. The dataset provided by MeeTwo has been collected over two years of operation and is of medium-size, containing around 22K labelled posts, where the average length of post is 45 words. The dataset is also very unbalanced with regard to the numbers of accepted and rejected posts. Only a minority of posts, less than 10%, have the label "reject". The lack of "reject" data makes it more challenging to reliably train machine learning models which can generalise to unseen data.

Another challenge in studying negative online behaviour is the myriad of forms it can take and the lack of a clear, common definition (Saleem et al., 2017). Some previous work has explored different forms of online behaviour including hate speech (Gagliardone et al., 2015), online harassment (Cheng et al., 2015), and cyber-bullying (Pieschl et al., 2015). In the real-world dataset explored here, there are multiple reasons for posts being labelled with "reject". This makes it much harder to build a general model which can successfully make a binary classification decision. Here, we investigate building multiple classifiers for the most commonly occurring rejection categories. However, breaking down the categories in this way further exacerbates the problems caused by a lack of data.

Various machine learning (ML) researchers (Shrivastava et al., 2017; Park et al., 2019) have demonstrated the effectiveness of data augmentation techniques in increasing the amount of training data available and, thus, the generalizability of the models learnt. As will be discussed further in Section 2, researchers in the area of Natural Language Processing (NLP) have also recently started looking for ways to augment linguistic datasets. Most recently, transfer learning models, e.g., BERT (Devlin et al., 2018), have been developed where a general language model can be trained on very large unlabelled corpus and then leveraged in domain-specific scenarios with small amounts of labelled training data. Here, we investigate the applicability of both data augmentation methods and transfer learning methods in a real-world scenario where accuracy and user trust are paramount. We evaluate the effectiveness of these techniques, comparing with more conventional NLP technology: 1) Logistic Regression (LR) classifier applied to Term Frequency - Inverse Document Frequency (TF-IDF) document representations, and 2) Long Short Term Memory (LSTM) networks applied to general-purpose words embeddings.

The contributions of this research are as follows. We demonstrate that NLP and ML technology has come of age and can now be successfully deployed in challenging real-world scenarios in the area of Health Informatics. More specifically, we show that a combination of training data augmentation and transfer learning methods can yield highly accurate classification models despite small and unbalanced datasets. Furthermore, we show that data augmentation techniques that insert and replace *synonyms* which have been discovered automatically from corpora outperform dictionary-based techniques. Accuracy of our models in classifying previously unseen posts, across 4 different rejection reason categories, is close to human performance. Thus, a live system, which uses these models to pre-label posts, will be effective in increasing the consistency of moderation and in reducing human-time to moderate.

2 RELATED WORK

Wulczyn et al. (2017) demonstrate that the performance of a machine learning model can be close to

humans on comments moderation. They use an N-gram word representation to represent a large dataset of 115K Wikipedia comments which have been labelled as personal attack ("reject") or not ("accept"). The classification architectures compared are linear regression (LR) and a simple feed-forward neural network or multiple layer perceptron (MLP). They show that MLP performs better on detecting the personal attack comments and that it achieves 1% absolute less than humans.

Other researchers (Pavlopoulos et al., 2017) have investigated applying more complex neural networks or deep learning models to the moderation of comments. Here, experiments based on the 115K Wikipedia comments and a Gazzetta dataset (1.45M training comments) show that Recurrent Neural Network (RNN) models outperform Convolutional Neural Network (CNN) models as well as a word-list baseline. Furthermore, more complex RNN models using an attention mechanism lead to further performance increases. Despite near human performance, they also conclude that it is more realistic to build a semi-automated system to assist human moderators rather than replace them.

However, the complex neural network models investigated in the aforementioned research require large labelled datasets in order to achieve such impressive results. These datasets also contain a reasonable balance of examples for both classes and only a single rejection reason. Thus, we cannot necessarily expect such good results on a real world dataset which is small, unbalanced and containing multiple rejection reasons.

Data augmentation as a method to increase the number of training examples and thus boost model performance has been extensively researched in computer vision (Shrivastava et al., 2017) and more recently in NLP (Park et al., 2019). In computer vision, it is now standard to rotate, reflect and crop images as these transformations are unlikely to affect a classification decision as to whether an image contains, say, a face or not. In NLP, the synonyms replacement method (Kobayashi, 2018) has been shown to be an effective method for data augmentation. Other possible data augmentation methods for NLP, explored by Wei et al. (2018) include synonyms insertion, random swap and random deletion. They combined synonyms replacement, synonyms insertion, random swap and random deletion together and find them effective on 5 NLP tasks.

Many of these aforementioned data augmentation techniques rely on a source of linguistic knowledge about the semantics of words. For example, both Kobayashi et al. (2018) and Wei et al.(2019) use synonyms which are randomly selected from WordNet (Fellbaum, 1998). However, there is an extensive literature in NLP on the automatic discovery of semantically related words, stemming from the distributional hypothesis (Harris, 1954) which states that words which have similar meanings are used in similar contexts. Automatic methods for discovering synonyms have an advantage over dictionary-based methods in that they can be tailored to a specific domain. They can therefore be expected to have better coverage of the vocabulary and domain-specific meanings (McCarthy et al., 2004). Currently, two of the most popular methods for constructing general purpose or domain-specific word representations are Word2Vec (Mikolov et al., 2013) and GloVe (Pennington, 2014). These methods owe their popularity due to their ability to represent word meaning in a low-dimensional space (typically around 300 dimensions). However, like all word representation methods based on the distributional hypothesis, it is well known (Weeds et al., 2004) that they conflate different semantic relationships (e.g., synonymy, hyponymy, antonymy, meronymy and topicality) and whether they can be used successfully in data augmentation remains an empirical question, which we explore here.

Another potential data augmentation technique, not explored here, exploits machine translation technology (Yu et al., 2018): alternative training examples can be generated by translating a text from English to some other language (e.g., French) and then translating it back into English. However, this method relies on an external service (e.g., the Google Translate API) or a fully implemented machine translation model, making it considerably more time-consuming to produce a similar sized augmented dataset than when using the simpler methods described above.

Most recently in NLP, there has been a lot of interest in transfer learning models (Howard and Ruder, 2018; Devlin et al., 2018; Peters et al., 2018; Radford et al., 2013). Rather than augmenting a small domain-specific training set, these models rely on building a large general-purpose language model (through pre-training on unlabelled data) and then *transferring* this knowledge to a domain-specific task (through fine-tuning on small amounts of labelled data). Universal Language Model Fine-tuning (ULMFiT) (Howard and Ruder, 2018) was shown to outperform the state-of-art on six classification tasks. Furthermore, performance with 100 labelled examples matched the performance of training on 10K from scratch (Howard and Ruder, 2018), thus making it particularly useful in scenarios with limited amounts of labelled data. Subsequently, Bidirectional Encoder Representations

from Transformers (BERT) (Devlin et al., 2018) have been shown to beat ULMFiT and achieve a new state of art in many NLP tasks. The architecture of BERT differs from other recent deep learning architectures (Peters et al., 2018; Radford et al., 2013) in its use of transformers (rather than LSTMs) and in its inherent bidirectionality. It also uses a fully connected layer for both the encoder and decoder networks. Stacked self-attention and point-wise architecture are used in the encoder and decoder parts (Vaswani et al., 2017). The encoder has 6 identical layers which take the input embeddings and produce a 512-dimensional output. The decoder takes that vector and finally outputs the probabilities. An attention function is used to map a set of key-value and query pairs to an output vector. The network linearly projects the queries, keys and values in order to make a multi-head attention. Thus, the embedding of each word in a sequence captures contextual information about words in other positions in the sequence.

Pre-training of the BERT model also differs from pre-existing models, which have tended to use a continuous bag-of-words (CBOW) model in training, through its use of masked language models (MLM). The MLM model randomly masks some percentage (15%) of input tokens, replacing them with 80% probability of [MASK], 10% probability of a random word and 10% probability of unchanging; the model must then predict those masked tokens (Devlin et al., 2018). The difference between MLM and CBOW is that the MLM model randomly *masks* some percentage of input tokens while the CBOW model continuously *masks* tokens withion a fixed window. This means that in every round of training, a MLM model is able to consider the information of the whole input while a CBOW model is only able to consider the information of that fixed window.

After pre-training, a BERT model can be fine-tuned for a variety of NLP tasks with the simple addition of a single output layer. Recently, pre-trained BERT has been released making it simple and cheap to deploy in real-world scenarios. However, its performance on a real-world problem such as ours, rather than standard NLP tasks from the academic literature, has yet to be seen.

3 DATASET

The data for this research was supplied free-of-charge by MeeTwo Education Ltd. At the time of the study, the dataset contained around 22487 labelled posts made by over 1000 users. Each post is associated with a user profile, which includes a user's id, gender, birth month, birth year and general location. User ids were anonymised by MeeTwo in the dataset so that they are meaningless strings and cannot be used to identify individuals. Further, personal details from rejected posts were also removed at source and data encryption was used to futher safe-guard the dataset. Any rejected posts are labelled with "reject" together with the reasons for rejection.

The dataset is a heavily imbalanced dataset; 1654 out of 22487 posts are labelled "reject" and the remainder are labelled "accept". There are also 37 different categories or reasons for rejection observed in the dataset. Most of the categories of rejected posts contain very few posts. Here we focus on the 4 categories which contain more than 100 posts, which are *Suicidal Ideation*, *Not Right for MeeTwo*, *Unclear* and *Offensive* posts. The corresponding frequencies of each of these categories is shown in Table 1.

Table 1: Rejection reasons occurring more than 100 times in the MeeTwo dataset.

Rejection Reasons	Frequency
Suicidal Ideation	453
Not right for MeeTwo	353
Unclear	225
Offensive	109
One of 33 other reasons	514
Total Rejected posts	1654

Figure 1: WordClouds for different rejection reasons. Top left: *Suicidal*, top right: *Unclear*, bottom left: *NotRight*, bottom right: *Offensive*.

The posts in the dataset range in length from 1 word to 140 words, with an average length of 45 words. Figure 1 shows the most frequently occurring words (ignoring stop words) in each of the 4 chosen categories of rejected posts. We see that two of the rejection categories (*Suicidal Ideation* and *Offensive*) appear to have clearly related words associated with them. For example, in *Suicidal Ideation* posts, the most frequent content words include 'life', 'kill', 'end', 'die' and 'hate'. Looking at *Offensive* posts, many of the most frequently occurring words

are well-known swear words. In contrast, the other two rejection categories (*Unclear* and *NotRight*) do not have such obvious clearly related words. For example, the most frequent words in both of these categories include 'still', 'help', 'girl' and 'good', which are likely to also occur in other categories and in accepted posts. Consequently, these categories of rejected posts are likely to be harder to identify.

4 METHOD

There are two main parts to our method. First, the training dataset is automatically augmented. Second, binary classification models are trained for each of the rejection categories. Due to the small number of posts in each category, it is not possible to create separate training, validation and testing sets for each category. Therefore, hyper-parameter optimisation is carried out on a single category *Suicidal Ideation* and the other three categories are reserved for testing.

In Section 4.1, we discuss augmentation techniques in more detail: introducing 4 potential augmentation techniques which we use in our experiments. In Section 4.2, we discuss machine learning architectures for classification in more detail: introducing 3 potential models of increasing complexity (Logistic Regression, LSTM, BERT).

4.1 Data Augmentation

The point of data augmentation is to improve a model's performance on unseen data. In general, the number of training examples is increased by taking existing examples and carrying out simple transformations which we do not expect to affect the label. All of the techniques described below have two parameters α and n: α controls how similar a transformed example will be to the original example, and n controls how many times the transformation is applied to a single example and, thus, also the size of the resulting augmented dataset, which will be $n+1$ times the size of the original training set. We will now describe each technique in detail. Table 2 shows examples of using these 4 methods.

- **Synonyms Replacement(SR):** Randomly select a proportion (α) of the words in each sentence and replace them by their closest synonyms i.e., most similar words. For example, if $\alpha = 0.2$ we will replace 20% of the words in each post. If $n = 2$, then we will do this twice to each post resulting in 2 new posts for each of the original posts.
- **Synonyms Insertion(SI):** Randomly select a proportion (α) of the words in each sentence and insert their most similar words at a random position in the same sentence.
- **Random Deletion(RD):** Randomly delete a proportion (α) of the words in each sentence.
- **Random Swap(RS):** Randomly select a proportion (α) of words in each sentence and swap their position with another randomly selected word in the sentence.

We experiment with two ways of generating synonyms for words: WordNet and word2vec. WordNet (Fellbaum, 1998) is a lexical database which groups words into sets of synonyms called synsets. It is straightforward to lookup the synonyms of a word in WordNet, but these will be based on lexicographers' knowledge of general usage and will not reflect the dominant sense within the domain. If a word has multiple synonyms, one of them is chosen at random. In the first example in Table 2, we see that the word 'school' is replaced by the WordNet synonym 'civilise'. This is unlikely to be the intended sense of 'school' in the MeeTwo dataset. In fact, the word 'civilise' is not used in any of the MeeTwo posts and therefore this training example is of very limited use.

Word2vec (Mikolov et al., 2013) is a continuous space model based on neural networks which generates distributed word embeddings that can be used in downstream NLP tasks. In order to use word2vec to generate synonyms we first prepare the in-domain training corpus (using the `nltk` library to carry out case normalisation, tokenisation and stopword & punctuation removal). We then use the `gensim` library to build a word2vec model with default parameters (sg=0, window=5, size=100) and also to find the most similar word to a given word, according to cosine similarity between embeddings.

Crucially, the word2vec model is trained on the dataset, so all the words it generates must be in that vocabulary. In the second example in Table 2, we see that 'go' is replaced by 'bring', which occurs 546 times in the MeeTwo dataset.

4.2 Classification

Machine learning classifiers for document classification take a numerical representation of the text as input and output the most likely label for the document. In its simplest form, the numerical representation of a post might be a vector which associates each word in the vocabulary with a weight according to its perceived importance in the post (e.g., based on frequency). For more sophisticated machine learning approaches, the numerical representation might be a sequence of word embeddings. Here, we investigate

Table 2: Examples of different data augmentation techniques applied to the text "I cant push myself to go to school".

Method	alpha	num	synonyms	text
Original Text				"I cant push myself to go to school"
SR	0.15	1	word2vec	"i cant push myself to bring to school"
SR	0.15	1	WordNet	"i cant push myself to bring to civilise"
SI	0.15	1	word2vec	"i cant push myself bring to go to school"
SI	0.15	1	WordNet	"i cant tug push myself to go to school"
RD	0.15	1		"i cant myself to go to school"
RS	0.15	1		"school cant push myself to go to i"

three different classifiers: Logistic Regression(LR), LSTM and BERT. Specifically, the input to the LR is a TF-IDF document representation; the input to the LSTM is a sequence of GloVe word embeddings; and the input to BERT is a sequence of words, since this model handles both the representation and the classification internally. We now describe each classification technique in more detail.

4.2.1 Architecture 1: TF-IDF and Logistic Regression

In this architecture, a post is represented as a vector of its TF-IDF values. For a given post, term frequency (TF) is the frequency of a word in that post. IDF is the inverse document frequency of the word i.e., the log reciprocal of the number of posts containing that word. Thus, TF-IDF, which is the product of TF and IDF, assigns higher weights to words which occur more frequently in an individual post and less frequently in other posts.

Logistic regression (LR) is a simple classification method, widely used in statistics and machine learning. It uses a logistic function to model a binary variable. Having less parameters, it is not as sensitive to the amount of training data as more complex machine learning methods. Here, we use the TF-IDF embedding as the input to LR. The output of the classifier is the probability of each class label.

In our implementation, we first use python's `nltk` library to pre-process the data, carrying out tokenisation, case normalisation, stopword and punctuation removal and lemmatisation. These standard preprocessing steps reduce the size of the vocabulary and remove tokens / distinctions which are unlikely to have an effect on classification. We then use python's `scikit-learn` library to construct the TF-IDF representation of each post and to realise the LR classifer.

4.2.2 Archictecture 2: General Purpose Word Embeddings and LSTM

Recurrent neural networks (RNNs) are typically used to model sequences because the hidden state of an RNN at any given time depends both on the current input and the previous state of the network. Typically, in language modelling, the input to an RNN is an embedding of a word (a high dimensional representation which captures similarities between words) and the network is trained to predict the next word in the observed sequence, given the current word and state of the network (which represents the context). Vanilla RNNs, however, have been shown to struggle with long range dependencies between words (Hochreiter et al., 2001). LSTMs attempt to overcome this problem by using 4 interacting layers in each repeated neuron: a cell state for long term memory; a forget gate to forget information; an input gate decides which values to update and what with; and an output gate that controls what to output. A classification layer can be put on top of any RNN, to make a prediction for a document label based on the hidden state of the network: either after the last token has been read or by pooling hidden states after each token is read.

We use the `pytorch` library to realise an LSTM which takes a sequence of general purpose (pre-trained) word embeddings as input. Specifically, we use GloVe (Pennington, 2014) embeddings with a dimensionality of 300 and a context window size of 8, trained on a large, general corpus of English text. The model has 2 LSTM and 2 linear hidden layers. The final classification output is decided using the final linear layer.

4.2.3 Architecture 3: BERT Pre-trained Embedding, BERT Fine-tuning and Training

BERT (Devlin et al., 2018) is a deep neural network architecture which can be used to generate contextualised word embeddings and carry out classification tasks. Using BERT typically has two steps. The first is pre-training on a very large general corpus; the second is fine-tuning on the specific task. Pre-training BERT has a huge computational cost. Fortunately, a number of pre-trained BERT models have been re-

leased as open source by the developers[1]. The models for English have been pre-trained on the concatenation of BooksCorpus (800M words) and English Wikipedia (2,500M words). There are different versions for cased and uncased text as well as two different sizes of model: BERT-base and BERT-large. BERT-base has 12 layers, 768 hidden units, 12 heads and a total of 110M parameters. BERT-large has 24 layers, 1024 hidden units, 16 heads and a total of 340M parameters.

In our implementation, the BERT-base uncased pre-trained model is employed. We then directly build a downstream model by fine-tuning this pre-trained BERT using our own labelled training data.

The fine-tuning part of BERT for sequence-level classification tasks is straightforward. To get an embedding of the input sequence, the final hidden state is taken for the first token in the input by identifying the special [CLS] word embedding and outputting a vector as $C \in R^H$. This vector is the input of a classification layer $W \in R^{K \times H}$ where K is the number of classes. The final label probability is computed with a softmax function. The parameters W are then trained in order to maximise the probability of the correct label. The hyper-parameters for fine-tuning are batch size, learning rate and number of epochs. We use a training batch size of 24 and a learning rate of $2e-5$. Convergence was achieved after a single epoch of training, with no benefit seen from continued training.

5 EXPERIMENTS

We carried out two sets of experiments. First, we optimised the hyper-parameters for data augmentation (see Section 5.1). Second, we compared the three different machine learning architectures, with and without data augmentation, on the four different categories of rejection reason (see Section 5.2).

5.1 Data Augmentation Hyperparameter Tuning

As discussed in Section 4.1, each data augmentation technique has 2 parameters: α and n. We experiment with these hyperparameters on just the *Suicidal Ideation* category of posts. We prepare the training and testing data in the following way.

- Select all of the *Suicidal Ideation* posts labelled "reject" (453 posts).
- Randomly select the same number of posts (453) from posts not rejected for *Suicidal Ideation*.
- Merge the posts selected in above 2 steps to make a new dataset: the *Suicidal Ideation* dataset.
- Split the dataset into 75% training and 25% testing data.

After testing each data augmentation technique individually to find the best hyperparameters, all of the techniques were used together with their best hyperparameters to augment the training dataset for the subsequent experiments.

5.2 Post Classification Experiments

Here, we investigate the effect of data augmentation on each of the machine learning architectures introduced previously (LR, GloVe + LSTM, BERT) on each of the categories for possible post rejection (*Suicidal Ideation*, *Offensive*, *Not Right for MeeTwo* and *Unclear*). The process of carrying out these experiments is as follows.

- Prepare each the data for each category of rejection reason. First, select those rejected posts in that category. Second, randomly select the same number of other posts (accepted or other rejection category). Third, merge the above 2 datasets together to form the original dataset for that category, which is split into training (75%) and testing (25%) sets..
- Augment the training dataset prepared using the best parameters of data augmentation methods.
- Use the original prepared dataset and augmented dataset separately to train each model and test on the same held-out testing data.
- Compare the accuracy of the different models on the test data.

The results of model comparison are the accuracy of each model on each of four different categories of posts. The average accuracy over the four categories is also considered.

6 RESULTS

In this section, we present our results on each set of experiments.

6.1 Data Augmentation Hyperparameter Tuning

We conducted experiments using a range of α and n values ($\alpha \in \{0, 0.1, 0.2, 0.3, 0.4, 0.5\}$ and $n \in$

[1] https://github.com/google-research/bert

$\{0,4,8,12,16,20\}$).

When testing on *Suicidal Ideation* data, all 3 machine learning architectures make gains in performance through the use of the data augmentation techniques. Figure 2 shows the effect of tuning α in data augmentation methods combined with an LSTM (on *Suicidal Ideation* posts). Here n is kept constant ($n = 3$). We see that optimal performance is achieved with all of the data augmentation methods with a relatively low value of α. This means that the best augmented posts are relatively similar to the original posts (containing, say, 80-90% of the same tokens). In general, we observed the same patterns for the other classification architectures. The only significant difference being that random swap has no effect on the LR classifier (since the document representation is based on a bag-of-words rather than a sequence).

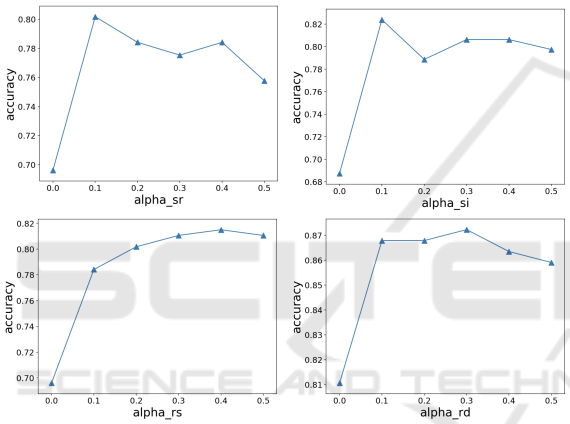

Figure 2: Tuning α for LSTM on *Suicidal Ideation* posts. Top left: *SR*, top right: *SI*, bottom left: *RS*, bottom right: *RD*.

Figure 3 shows the effect of tuning n in the data augmentation methods for the LSTM and for BERT (on *Suicidal Ideation* posts). Both architectures show substantial improvements with data augmentation (over 20%). We observe that the LSTM requires a greater amount of augmentation ($n > 20$) to achieve results in the same ballpark as BERT ($n = 4$). The LR classifier, on the other hand, benefited less from data augmentation, with performance only improving by around 4%. This peak performance occurred at around $n = 8$.

6.2 Post Classification Experiments

Here, we present the performance of each model on four categories of posts and the average performance over the four categories.

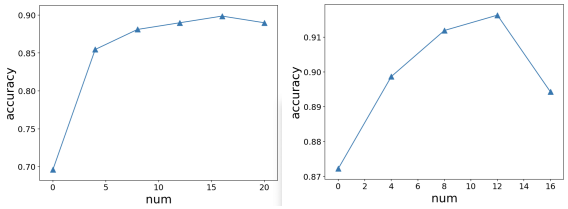

Figure 3: Tuning n for LSTM and BERT on *Suicidal Ideation* Posts. left: *LSTM*, right: *BERT*.

Figure 4 shows our results for the different categories. The blue bars represent the accuracy of models which are trained on original training data, while orange bars represent the accuracy of models which are trained on data augmented by data augmentation methods with best parameters.

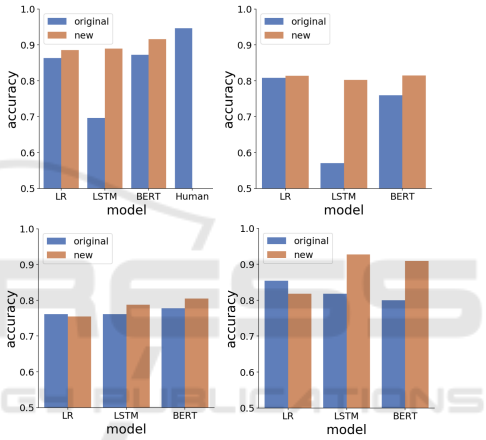

Figure 4: Testing on different categories. Top left: *Suicidal Ideation*, top right: *Not Right for MeeTwo*, bottom left: *Unclear*, bottom right: *Offensive*.

From the results we make the following observations. For the *Suicidal Ideation* category, the TF-IDF+LR architecture gains 3 absolute points with data augmentation; GloVe+LSTM gains 21 absolute points; BERT gains 5 absolute points and reaches the highest accuracy among 3 models. The best accuracy is 0.916 which is close to humans accuracy which is 0.946[2].

For the *Not Right* category, TF-IDF+LR increases 1 absolute point; GloVe+LSTM gains 22 absolute points; BERT gains 3 absolute points and reaches the highest accuracy among 3 models. The best accuracy is 0.82 which is much less than the accuracy on *Suicidal Ideation* category. One likely reason for this is that *Suicidal Ideation* posts normally contain some obvious words such as kill, life, sad and so on. *Not*

[2]Human accuracy is using the same test data and testing by professional posts moderation staff of MeeTwo

Right posts are those posts that are not in line with the MeeTwo rules. These are more difficult to classify.

For the *unclear* category, TF-IDF+LR decreases 1 absolute point; GloVe+LSTM gains 3 absolute points; BERT gains 3 absolute points and reaches the highest accuracy among 3 models. The best accuracy is 0.81 which is quite close to the accuracy on *notRight* category. *Unclear* posts may include posts that are essentially meaningless. Those kind of posts are also more difficult for the machine learning models to classify.

LSTM achieves highest accuracy on the *Offensive* category of posts. The accuracy is 0.93 which is much higher than LR. BERT also achieves a good accuracy which is 0.91. *Offensive* posts are quite similar with *Suicidal Ideation* posts. Both of them have some obvious words, which makes classification easier for all of the models.

More generally we see that data augmentation is much more beneficial to the more complex, data hungry architectures such as LSTM and BERT. These architectures need a large amount of data to achieve peak performance. The accuracy of 4 different categories in Table 3 all show that when using augmented training data, the test accuracy of LSTM and BERT are greatly improved. For the TF-IDF+LR model, the increase in data amount does not have the same impact. In fact, in some categories, the accuracy of LR decreases with the augmented data.

The BERT model achieves highest accuracy on most of the categories. The highest average accuracy is the BERT model. The average accuracy of BERT and LSTM are very close and is around 5 absolute points higher than LR.

6.3 Error Analysis

The incorrect predictions made by the different models and the human are not all the same. Table 4 shows some examples of errors made by each on the *Suicidal Ideation* test set. 0 represents an "accepted" decision for the post while 1 represents a "rejected" decision for the post. The first post shows that human is correct while all other models are wrong. The post is quite short. Some words like *break*, *anymore* may lead the models to a wrong prediction. The second post is the scenario where the human is wrong while all models are correct. The third and fourth posts show only BERT is correct or only BERT is wrong. We note that the gold standard label was also made by some human moderator when the posts were uploaded. Therefore, the gold standard label may not be one hundred percent correct.

Figure 5 shows the proportion of the same predictions made by each pair of models on the *Suicidal Ideation* dataset. We note that LSTM and BERT have the biggest overlap in errors. Therefore, typically, the mistakes made by these two models are very similar. We also see that the smallest overlaps in errors are between LSTM and human and between LR and human. In other words, a greater proportion of the BERT model's errors are the same as the human errors. Consequently, when the results of the LSTM model and the BERT model differ, the BERT is more likely to have made a human-like error.

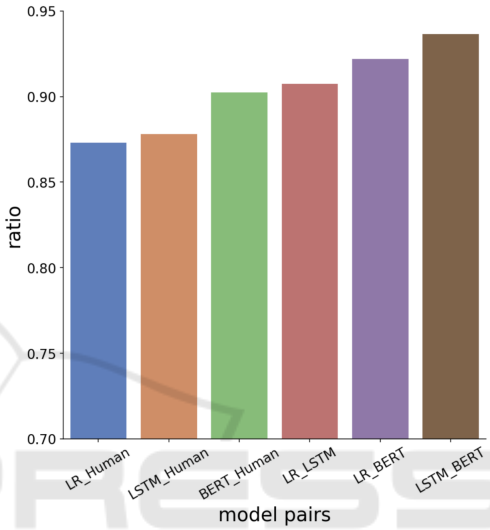

Figure 5: Proportion of predicting the same result for any pair of models or human.

7 CONCLUSION AND FUTURE WORK

In this work, we have explored the feasibility of building a semi-automated moderation model that is able to reduce the moderation workload and accelerate the moderation speed in the MeeTwo scenario. We have demonstrated a practical pipeline which can be used in this and other small dataset classification scenarios. We recommend the use of simple data augmentation methods to build up a much larger training dataset and then use this dataset to train a data hungry model such as BERT.

There are in total 38 different possible reasons for posts being rejected from MeeTwo. Here, we have focused on the top 4 rejection reasons which account for more than half of the rejected posts. These are *Suicidal Ideation*, Not Right for MeeTwo, Offensive and *Unclear*. Using a combination of dataset augmentation and modern machine learning architectures for classification, we have demonstrated that, despite

Table 3: The average performance of 3 models using original data and augmented data.

	Model 1(LR)	Model 2(LSTM)	Model 3(BERT)
Original Data Average Accuracy	0.82	0.70	0.75
Augmented Data Average Accuracy	0.83	0.86	0.87

Table 4: Example errors made by the different architectures on the *Suicidal Ideation* dataset.

No.	Text	Real Label	Human Label	LR Label	LSTM Label	BERT Label
1	my friend doesn't want to be with me at break or lunch anymore	0	0	1	1	1
2	Sometimes I wonder what it's like to not be alive. Does anybody else wonder whether death is better than life? I think about it a little too much I think.	1	0	1	1	1
3	I need to stay happy. I have a bottle of anxiety pills in the bathroom. One pill a day it says. I could just swallow them all if I get upset and suicidal. I need to stay happy.	1	0	0	0	1
4	I'm falling for someone I can't fall for what do I do?	0	1	1	1	0

limited and unbalanced training data, it is possible to build classifiers with near-human accuracy for a number of different rejection reasons. On the *Suicidal Ideation* category, we achieve an accuracy of 91.2% on a balanced dataset (compared to human performance of 94.5%). In the *Offensive* category, we achieve 92% accuracy. Accuracy is lower (around 80%) on less well-defined categories such as *Unclear* and *Not Right for MeeTwo*. However, in the real-world scenario, where a much smaller proportion of posts should actually be rejected than in our testing scenario, these methods will be very valuable in being able to identify and flag posts of potential concern, which a human moderator can then check.

We have demonstrated 4 different augmentation techniques which can increase the amount of training data for machine learning by 20 times or more. Our results show that these methods can massively boost model performance when we only have small labelled dataset. We compared three machine learning architectures (TF-IDF+LR, GloVe+LSTM and BERT) in a practical text classification task. In general, the performance of BERT was the highest. However, without data augmentation, the performance of the LSTM is poor and performance of BERT is on a par or worse than the simple baseline of TF-IDF representations and a LR classifier. Thus, we conclude, that whilst some data sparsity issues may be overcome through the use of a pre-trained BERT model, it is of further benefit to augment the training set used for fine-tuning.

On the *Not Right for MeeTwo* and the *Unclear* categories, the performance of BERT is higher than LSTM while the performance of LSTM is close to BERT on the *Suicidal Ideation* category and even higher than BERT on the *Offensive* category. These latter categories can be more easily defined in terms of individual words (see Figure 1). This suggests that the BERT model can work well even when the text has no obvious words as indicators of a particular class.

There are a number of potential avenues for further work. First, the data augmentation techniques presented herein might be improved upon or extended. For example, other models or datasets could be used to train the synonyms generation model. A different or larger corpus for discovering semantically similar words might create a more generalizeable model. Techniques for ensuring that only synonyms (rather than antonyms or other related words) are inserted or substituted could also be investigated. Further, part-of-speech tagging or dependency analysis might be used to ensure that the new training examples are linguistically plausible. Finally, we have looked at ways in which to transform posts at the word level. However, we could consider transforming posts at the sentence or discourse level. For example, it might be possible to create new sentences or posts by combining different sentences or parts of sentence which are in a post with a given label.

The second avenue for future work relates to machine learning classification model performance and integrating these models into a live system. The 3 architectures investigated here could be blended (e.g., through a use of a simple voting system), which could

increase the performance. In a practical system which values recall over precision, if any of the models predict that a post should be rejected, then this should be flagged to the human moderators. There are also a number of other hyper-parameters and potential features which could be explored. For example, we have noticed that posts made by boys have a higher rejection rate. Consequently, future work could explore whether and how to incorporate extra-linguistic features of the posts including gender and age of user.

REFERENCES

Cheng, J., Danescu-Niculescu-Mizil, C., and Leskovec, J. (2015). Antisocial behavior in online discussion communities. *In Ninth International AAAI Conference on Web and Social Media*.

Devlin, J., Chang, M., Lee, K., and Toutanova, K. (2018). BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Duggan, M. (2017). Online harrassment. Technical report, Pew Research Center.

Fellbaum, C. (1998). *WordNet: An Electronic Lexical Database*. Bradford Books.

Gagliardone, I., Gal, D., Alves, T., and Martinez, G. (2015). Countering online hate speech. *Unesco Publishing*.

Harris, Z. (1954). Distributional structure. *Word*, 10(23):146 – 162.

Hochreiter, S., Bengio, Y., Frasconi, P., Schmidhuber, J., et al. (2001). Gradient flow in recurrent nets: the difficulty of learning long-term dependencies.

Howard, J. and Ruder, S. (2018). Universal language model fine-tuning for text classification. *arXiv preprint*.

Kobayashi, S. (2018). Contextual augmentation: Data augmentation by words with paradigmatic relations. *CoRR*, abs/1805.06201.

McCarthy, D., Koeling, R., Weeds, J., and Carroll, J. (2004). Finding predominant word senses in untagged text. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pages 279–286, Barcelona, Spain.

McManus, S., Bebbington, P., Jenkins, R., and Brugha, T. (2016). Mental health and wellbeing in england: Adult psychiatric morbidity survey 2014.

McManus, S., Meltzer, H., Brugha, T. S., Bebbington, P. E., and Jenkins, R. (2009). Adult psychiatric morbidity in england, 2007: results of a household survey.

Mikolov, T., Yih, W. T., and Zweig, G. (2013). Linguistic regularities in continuous space word representations. *In Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751.

Park, D. S., Chan, W., Zhang, Y., Chiu, C., Zoph, B., Cubuk, E. D., and Le, Q. V. (2019). Specaugment: A simple data augmentation method for automatic speech recognition. *CoRR*, abs/1904.08779.

Pavlopoulos, J., Malakasiotis, P., and Androutsopoulos, I. (2017). Deep learning for user comment moderation. *CoRR*, abs/1705.09993.

Pennington, J., S. R. . M. C. (2014). Glove: Global vectors for word representation. *In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. *CoRR*, abs/1802.05365.

Pieschl, S., Kuhlmann, C., and Porsch, T. (2015). Beware of publicity! perceived distress of negative cyber incidents and implications for defining cyberbullying. *Journal of School Violence*, 14(1):111–132.

Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. (2013). Improving language understanding by generative pre-training.

Saleem, H. M., Dillon, K. P., Benesch, S., and Ruths, D. (2017). A web of hate: Tackling hateful speech in online social spaces. *CoRR*, abs/1709.10159.

Shrivastava, A., Pfister, T., Tuzel, O., Susskind, J., Wang, W., and Webb, R. (2017). Learning from simulated and unsupervised images through adversarial training. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2242–2251.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez A. N., ., and Polosukhin, I. (2017). Attention is all you need. *In Advances in neural information processing systems*, pages 5998–6008.

Weeds, J., Weir, D., and McCarthy, D. (2004). Characterising measures of lexical distributional similarity. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, pages 1015–1021, Geneva, Switzerland.

Wei, J. W. and Zou, K. (2019). EDA: easy data augmentation techniques for boosting performance on text classification tasks. *CoRR*, abs/1901.11196.

Wulczyn, E., Thain, N., and Dixon, L. (2017). Ex machina: Personal attacks seen at scale. *International World Wide Web Conferences Steering Committee.*, pages 1391–1399.

Yu, A. W., Dohan, D., Luong, M., Zhao, R., Chen, K., Norouzi, M., and Le, Q. V. (2018). Qanet: Combining local convolution with global self-attention for reading comprehension. *CoRR*, abs/1804.09541.