AI based Context and Sentiment analysis for Messaging

Nikhil Singh, Johanna Wallner
Fraunhofer-FOKUS, Technische Universität Berlin
Berlin, Germany
nikhil.singh@campus.tu-berlin.de
johanna.wallner@campus.tu-berlin.de

***Abstract*—Cyberbullying is a growing problem among children and teenagers in a world where more and more communication happens via messengers and social media. Since the cyberspace is secluded and supervision by adults virtually impossible, this issue requires an automatic approach. Using sentiment analysis, natural language processing and deep learning we tackled this task to implement an automatic detection of cyberbullying in text messages. We built an LSTM model using TensorFlow and trained it on a MySpace conversational-style dataset and Twitter dataset for cyberbullying. We achieved an accuracy in the range from 75% to 94% on our test data based on hyperparameters tuning and embeddings used.**

***Index Terms*—Cyberbullying, Machine Learning, Deep Learning, Sentiment Analysis**

I. Introduction

Modern, digital media have become an integral part of everyday life for children and young people. They use computers, tablets and smartphones as access to information sources, for learning, for entertainment and to stay in touch with friends. The mobile Internet enables even faster, spontaneous appointments and the digital exchange of photos or videos. Social contacts are increasingly maintained via messenger services or social networks. A problem that is connected to social networking is cyberbullying which is defined as "the collective label used to define forms of bullying that use electronic means such as the internet and mobile phones to aggressively and intentionally harm someone." [1, p. 51].
According to a Pew Research Center survey 59% of U.S. teens have personally experienced some kind of abusive online behavior. [2] In Germany, around 1.4 Million children and teenagers are affected. [3] Victims experience a broad range of negative psychological effects like anger, powerlessness, sadness, and fear which leads to reclusiveness or agressions towards others. In the worst case, cyberbullying could even lead to self-harm and suicidal thoughts. Since the cyberspace is a very secluded space, adults who would normally be supervising the teenagers are left outside and cannot intervene. [4]
Sentiment Analysis is a field within Natural Language Processing. NLP combines artificial intelligence, computational linguistics, cognitive science and computer science to allow a machine to understand natural languages like humans do. Natural Language Processing must capture language in the form of sound or text strings and extract meaning. For this, it uses different methods and techniques, which have to be gone through step by step until the meaning of a text is fully understood. The following parts of Natural Language Processing are used for this purpose:

- Speech Recognition
- Segmentation of previously captured language into individual words and sentences
- Recognizing the basic forms of words and capturing grammatical information
- Recognizing the functions of individual words in a sentence (subject, verb, object, article, etc.)
- Extraction of the meaning of sentences and parts of sentences
- Recognition of sentence correlations and sentence relationships

Since human language is often ambiguous, even a complete run through the various steps described above cannot always provide a clear result when stylistic devices such as the rhetorical question, irony or paradox are used in the language to be analysed. [5, p. 1 ff.]

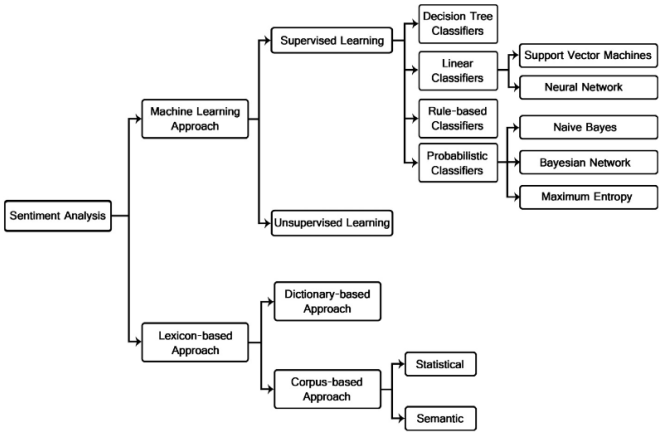

Figure 1: Sentiment Analysis Techniques [6]

Since early 2000, sentiment analysis has grown to be one of the most active research areas in NLP. Sentiment Analysis identifies and extracts subjective information from text which means it doesn't analyse the facts of the given data but rather the attitudes, emotions and opinions of the author. The sentiment analysis techniques can be divided into 2 main approaches: Machine Learning and Lexicon-based (see Figure 1). For the lexicon-based method there exist special so-called sentiment lexicons which contain language-specific expressions that have already been assigned a positive or negative meaning due to their meaning. Indicators must be identified in the text which allow conclusions to be drawn about the sentiment. In doing so, lexicon-based methods rarely take into account the context or domain dependency of evaluative expressions that can reverse their polarity context sensitively. Since sentiment analysis is the solution to a classification problem, almost any (un)monitored or semi-monitored machine learning approach can be considered that has already been successfully used as a so-called text classifier. In sentiment analysis, machine learning methods are usually more effective than lexicon-based approaches in terms of the accuracy and precision of classification. [6]
With the rise of deep learning caused by growing computing power

and amount of data in the last several years, the ability of algorithms to analyse text has improved considerably and made the results of tasks like speech recognition and machine translation far more accurate. [5, p. 1 ff.] Simultaneously the use of social media (e.g., reviews, forum discussions, comments, and postings in social network sites) grew explosively which resulted in the availability of large-scale datasets to train the algorithms. [1, p. 5]

With the help of deep learning, cyberbullying can be detected automatically and in real time so supervisors and adults can be notified to act accordingly. In general, experts on cyberbullying support such automatic monitoring when effective follow-up strategies are available. These strategies should prevent future cyberbullying and empower the involved parties. [7]

II. Related work

For sentiment analysis in general, early publications used machine learning methods, such as Naïves Bayes and Support Vector Machines extensively. Since the emergence of deep learning techniques that produced state-of-the-art results in other domains, these approaches were also used in sentiment analysis. [8]

Due to the growing prevalence of cyberbullying the body of work on its automatic detection is constantly growing. Publications that use deep neural networks for this task specifically are rare. Dadvar et al. used deep neural networks on social media datasets to detect cyberbullying. They used four different models (CNN, LSTM, BLSTM, and BLSTM with attention) together with three different word embedding methods (random, GloVe, and SSWE) that were trained on Twitter, Formspring, and AskFM datasets. Afterwards they used transfer learning, where the model that you trained on one task is used on another similiar task, for a YouTube datset. This method performed better than all the previous machine learning approaches used on this dataset. [9]

Zhang et al. propose a novel pronounciation based CNN (PCNN), which shower better results than two other baseline CNN models. They implemented three different ways to tackle class imbalance. For the model training they used two different datasets: one from Twitter and one from Formspring. For preprocessing, they used eSpeak, an open source speech synthesizer software, to create phonetic representations of the words. That way misspelled words were not a problem for the analysis ("you" would be the same as "u"). Their model outperformed the other methods CNN with pre-trained Google word vectors and CNN with random word embeddings. It is noteworthy that CNN Random performed better than CNN Google which could be explained by the fact that the Google corpus was not specific to cyberbullying detection. [10]

Rafiq et al. were the first to classify cyberbullying in a multimedial context, namely on Vine, a video-based mobile social network. They used 4 machine learning classifiers: Naive Bayes, AdaBoost, DecisionTree and RandomForest with 10-fold cross-validation. They achieved a lower accuracy than a comparable study on Instagram data which may be explained by the fact that their Vine dataset had more conflicting opinions concerning whether a session contained in fact cyberbullying or not. [11]

Agrawal et al. identified the reliance on manual feature extraction e.g. swear word list and POS tagging as a potential bottleneck in past works on cyberbullying detection. Other two common bottlenecks are the limitation of the algorithm on only one social media platform and on only one form of cyberbullying (for example sexism or racism). They tackled these problems by using deep learning based models along with transfer learning on three different types of social media datasets (Formspring: a Q&A forum, Twitter: microblogging, and Wikipedia: collaborative knowledge repository) for three topics of cyberbullying (personal attack, racism, and sexism). Their DNN models beat state of the art results for all three datasets but the results could be improved by more fine-grained datasets. [12]

Abdullah et al. proposed their own fully automated algorithm CNN-CB which is based on a convolutional neural network (CNN) and considers semantics through the use of word embedding. They created their own dataset of 39.000 Tweets which were manually labelled by two human contributors. The problem of class imbalance was solved by querying Twitter for swear words to extract more bullying tweets. Compared to the traditional cyberbullying approach cont-SVM, CNN-CB reached better results in accuracy, precision and recall. [13]

In another publication by Abdullah et al. CNN was used as well, this time together with a GloVe word embedding to detect cyberbullying on Twitter. They also used a metaheuristic optimization algorithm to find the optimal hyperparameters for the CNN. [14]

III. conceptualization

A. Framework

As an open source tool for distributed database systems, Google's TensorFlow forms an innovative basis for neural networks in the field of speech and image processing tasks. Its scalability, clean design, flexibility, and extensive documentation made it to the top deep learning libraries. [15, p. 368 f.] TensorFlow enables neural networks to be represented by computational graphs. While the edges represent the inputs and outputs of the individual calculation steps, the nodes are responsible for processing all inputs into outputs. This means that the mathematical operations such as addition, multiplication or various variations of functions, are performed at the nodes of such a graph. The graph edges represent the multidimensional data arrays (tensors) that communicate between the individual nodes. [16, p. 32 ff.] For our project we are using Python with the Tensorflow, Pandas and Numpy libraries as well as other libraries for data extraction.

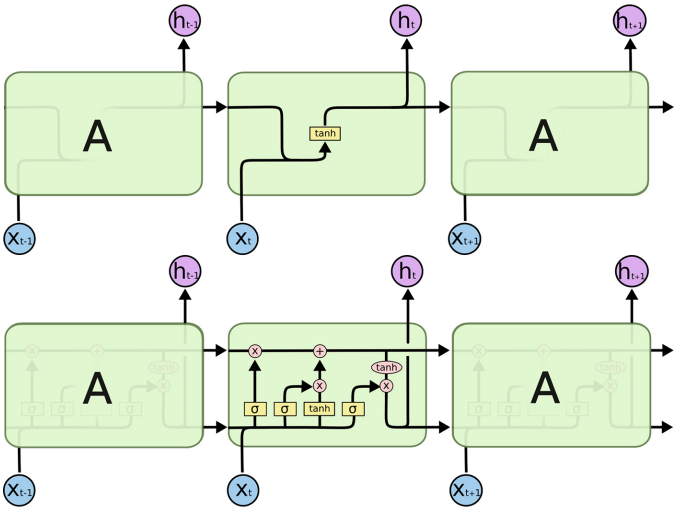

Figure 2: RNN cell structure (top) vs. LSTM cell structure (bottom) [17]

B. Deep Neural Networks

Deep neural networks are multilayered artificial neural networks which are inspired by biological neural networks. They started to

gain attention in the 90s but the great benefits appeared with the rise of Big Data and high computational power. One of the biggest advantages of deep learning algorithms in comparison to traditional machine learning techniques is the automated feature extraction. Deep learning enables multi-layered automatic feature representation while traditional machine learning algorithms rely heavily on hand-crafted features. [5, p. 6 f.]

Recurrent Neural Networks (RNN) are a strong and robust variant of neural networks that are among the most important algorithms today. RNNs can remember important information about the received information and predict the next step due to their internal memory. This is why they are considered the preferred algorithm for sequential data such as time series, speech, text, financial data, audio, video, weather and much more. You can gain a deeper understanding of a sequence and context compared to other algorithms. [18, p. 373 ff.]

Long Short-Term Memory (LSTM) - Networks are an extension for Recurrent Neural Networks that improves their performance by essentially expanding their memory. The units of an LSTM are used as building blocks for the layers of an RNN, often referred to as an LSTM network. In LSTM, there are three gates: Input, Forget and Output Gate. They determine whether new inputs are to be admitted or not, whether the information is deleted because it is not important (forgetting the gate), or whether it affects the output in the current time step. [18, p. 410 ff.] GRU (Gated Recurrent Units) are based on the same ideas of gates, but have a simpler architecture and no separate storage compartments.

Both gated RNN architectures outperform the traditional RNNs on sequential data. Convergence is often faster, and the final solutions tend to be better. For LSTM vs. GRU, there is no clear winner which suggests that the choice of the type of gated recurrent unit may depend heavily on the dataset and corresponding task. [19] For our use case we decided to use LSTM since it is the older variant and there are more ressources available to help with the configuration.

Department

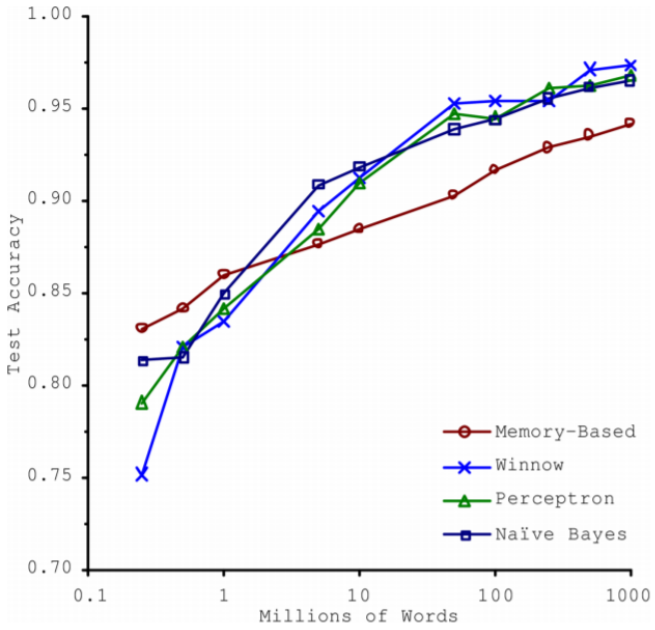

Figure 3: Large datasets [15, p. 25]

C. Dataset

Banko et al. showed that for a NLP task a larger datasets benefits the performance of machine learning algorithms (see Figure 3). They suggested that the development of very large training datasets may be more effective for progress in Natural Language Processing than improving methods that use existing smaller training corpora. [20] As [21] stated, a key challenge for machine learning tasks working on cyberbullying is to find a suitable dataset. In recent years only few datasets in this particular field have been created of which some are not publicly available. Especially messaging-focused social media platforms like Instagram, Snapchat and WhatsApp are underrepresented in these datasets. [22]

Unfortunately, datasets that contain these verbal exchanges are very rare, due to their private nature. Something fitting would have been the WhatsApp data set by [23] but it is in Italian. Reference [21] created their own dataset consisting of 85.485 social media posts that were randomly crawled from AskFM. The posts were manually annotated following a fine-grained annotation scheme that covers roles, typology of harassment and level of harmfulness. Unfortunately, our request to obtain the dataset was denied.

For our use case we chose a MySpace dataset which was provided as part of the CAW 2.0 workshop. It was crawled from thread-style forums where many users can participate and lots of different subjects are discussed. The dataset consists of windows where each windows contains 10 posts. Three undergraduate research assistants from the Media and Communications Studies reviewed each windows and searched for signs of cyberbullying. If at least 2 assistants regarded the window as cyberbullying it was labelled as such. [24] We chose this dataset because of the conversational style, though we don't regard it as ideal.

We also considered to use the Bullying Traces Data Set by [25]. This dataset consists of tweets that were crawled by the public Twitter stream API. Each tweet contains at least one of the following keywords: bully, bullied, bullying. The result were approximately 7.000 tweets that were manually labeled out of which 4.354 were retrieved using tweet ids. Bullying traces are defined as response to the bullying experience, which include but far exceed the incidences of cyberbullying. Because of this fact, we decided against using this dataset.

Other datasets that were developed by cyberbullying researchers were crawled from websites such as Wikipedia [26], Twitter [27] or Formspring [28] were found not to be a good fit because of their lack of verbal exchange.

D. Data Preparation

A big challenge in this part of the project was to load the data into the model. The 10-post windows were stored in 2.048 XML files while 11 Excel files contain the corresponding labels. The Excel files consist of two columns: windows id and cyberullying (which can be either yes or no). To extract the labels we used the glob module which implements globbing of directory contents. This means we could load all the files from the folder "BayzickBullyingData/Human Concensus/" using a wildcard ("*"). For the XML files we used the XML processing module and XML DOM to parse the other folder "BayzickBullyingData/packet-all/". We then joined the posts into conversations and matched them with the labels. Afterwards we had to convert the textual labels from "yes/no" into numbers ("1/0"). Also the conversation length needed to be euqalised. We calculated the average length of all the conversations and fed them into Tensorflow's vocabulary_processor, which pads the shorter and truncates the longer conversations. We shuffled the data and split it

into training and test, with an about 75% to 25% ratio. We also tried to build separate model using Twitter dataset to check if we find a significant improvement in the performance. Twitter dataset was not directly available because of privacy reasons. We had to fetch the tweets via tweet-ids using our own twitter credentials.

E. Word Embedding

Word embedding is one of the most popular methods to transform text into number so it can be processed by a neural network. The easiest approach is Bag-Of-Words (BOW) where after an initial cleanup of the text corpus, a dictionary of all occurring words is created and each word is assigned a number which sorts the words by occurrence in descending order. With the help of this dictionary a number is assigned to each word existing in the text corpus. This can already be sufficient for certain simple tasks. However, the Bag-Of-Words approach has the following disadvantage: The size of the number is in no way related to the meaning of the word. [29, p. 69 ff.]

But there are approaches, like Word2Vec, that are capable of capturing the context of a word in a document as well as semantic and syntactic similarity in relation to other words. This is achieved by having an algorithm arrange all words present in the text corpus by their occurrence and the words surrounding them in a multidimensional space so that words that frequently appear in similar contexts also have a similar vector (which means that they are close to each other in this space). The result is a vector representation for each word contained in the text corpus. Above all, each word vector represents at least the semantic meaning of the word in context. So you can use pre-trained models to implement e.g. arithmetic with words: KING - MAN + WOMAN = QUEEN. Thus, the models actually contain a relatively meaningful representation of the meaning of words. Of course this meaning is not comprehensive and refers only to single words and not to word roles or whole passages. However, the practical vector format can be well processed by neural networks of all kinds. [16, p 77 ff.]

The disadvantage of this method Word Vectors is that you need a very large and "clean" text corpus to generate a meaningful vector representation. In addition, the algorithms are very computationally intensive. For individual projects this is usually not feasible, as the data volume required for meaningful training is simply missing. When using pre-trained models, there is the problem that these models also only know words from their corpus - all words that were not present during the training receive a null vector. This can be especially difficult for special areas with their own jargon. [16, p 77 ff.]

In 2014, the Stanford NLP Group developed GloVe as a probabilistic successor to Word2Vec. Even though the developers of GloVe stated it to perform significantly better than Word2Vec [30], other researchers found that both algorithms it depends on the used language and model. [31] There exist pretrained embeddings for both approaches and for our use case we chose the GloVe Wikipedia 50d embedding[1] which consists of vocabulary size of 400k words.

F. Hyperparameters

A crucial step in the development of a machine learning model is usually to find the optimal settings for the model architecture used in order to achieve the best possible prediction accuracy. During the training process, weightings within the model are automatically changed so that it is increasingly able to establish the desired

[1]https://nlp.stanford.edu/projects/glove/

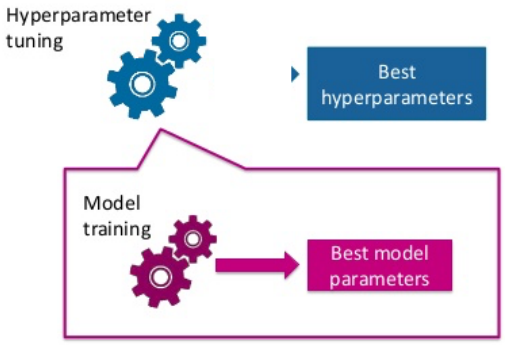

Figure 4: Hyperparametertuning [32]

relationship between features and prediction. However, what remains unchanged during the training process is the basic architecture of the model: the structure of the model function. These adjustment screws, which remain the same throughout the training process, are called hyperparameters. The set of all possible combinations of the these is also called the parameter space. Since the hyperparameters have a large influence on the accuracy of the model, their optimization is an important step in model development and can significantly improve prediction accuracy. [33, p. 271 ff.]

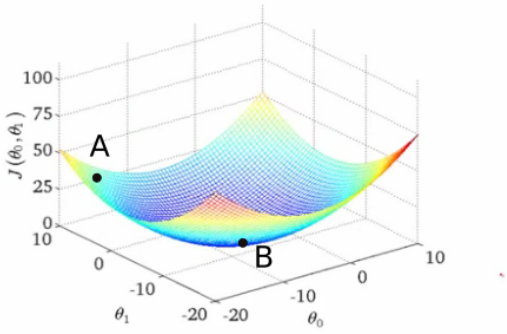

Figure 5: Gradient Descent: Point A has a high loss while at Point B the loss function has a minima [34]

The actual training of the neural network happens via Gradient Descent Optimization. It allows us to find those parameters of our model where the loss between the predicted and the actual output is minimized. In Figure 5 you see the curve where the Gradient Descent will converge on the smallest value of loss (point B) starting from the initial value of loss (point A). In general, a neural network are trained by using the back-propagation algorithm which consists of two steps: First, the error is calculated by comparing the target values with the predictions in the output layer and by error feedback to the neurons of the hidden layers. Then the weights are being adjusted against the gradient increase of the error function (loss function). [18, p. 203 ff.] While back-propagation refers to the method for computing the gradient, another algorithm is used to perform learning using this gradient. This Optimizer has the task of adjusting the weights and bias values of the net during training on the basis of the calculated model deviations of the cost function. For this, cost function gradients that indicate the direction in which weights and bias values need to be adjusted to minimize the cost function of the model. The

development of fast and stable optimizers is a major area of research in neural networks and deep learning. For this model we used the Adam Optimizer which is currently one of the most frequently used optimizers. Adam stands for adaptive moment estimation and combines the benefits of two other optimization techniques: AdaGrad and RMSProp. It was specifically developed for deep learning and since it is the state-of-the-art optimizer for this task, we are using this approach. [35]

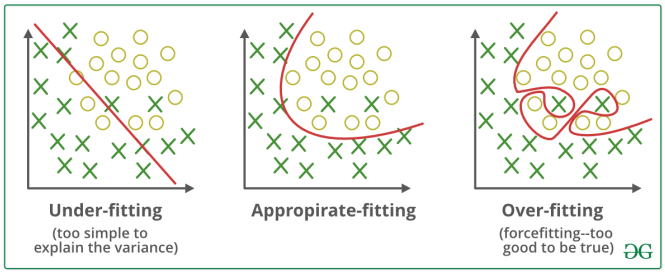

Figure 6: Underfitting vs. Overfitting [36]

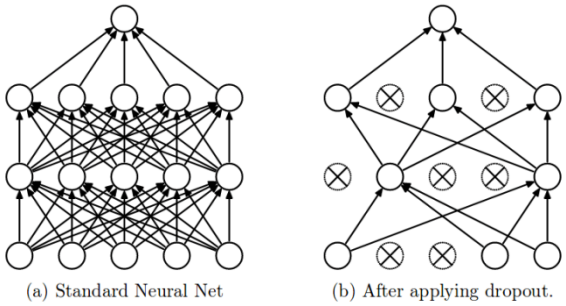

Figure 7: Standard and "thinned" neural network [36]

Overfitting can be a huge problem for deep neural networks with a large number of parameters (see Figure 6). The reason for this is that the neural network adapt to the test data too much. Such models often do not provide good predictions for new observations. Regularization, which is the process of updating a neural network to analyze general data, is an attempt to solve this problem. Researchers have developed numerous methods for regularization. Two of the most important are L1/L2 regularization and dropout. In L1 and L2 regularization, weights are reduced by increasing loss. Dropout is a technique where during training, units and their input and output connections are randomly removed from the network ("drop out"). This prevents the units from approaching each other too much. The units should, if possible, differ from each other individually so that their characteristics come to light. Dropout means that "thinned out" layers of the network are created. The thinned network consists of all units that have survived the dropout. A neural network of n units can be seen as a collection of 2^n possible thinned nets (see Figure 7). [37] We decided to use Dropout because there it is very easy to implement into Tensorflow

G. Chat Application

For our chat application we used Django, an open source Python Web framework. Django offers an extensive template language, an object-relational mapper, an automatically generated admin interface and a customizable URL design with regular expressions. We followed the Django Channels documentation[2]. Django Channels uses chat protocols and WebSockets for real-time communication between the server and client. We built a very simple application where you first enter a chat room and then can write messages. Each message gets fed into the RNN saved model which then decides if cyberbullying is present or not.

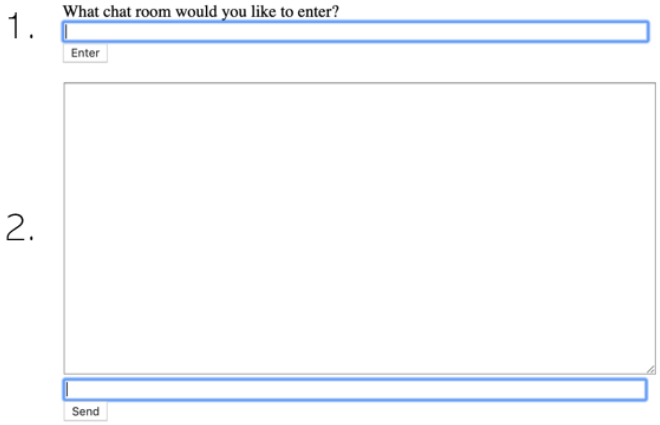

Figure 8: Screenshot of our chat app

IV. Results

We created four different RNN models. Firstly, using MySpace dataset with TensorFlow vocab processor embeddings. Secondly, using MySpace dataset with GloVe word embeddings. Thirdly, Twitter dataset with TensorFlow vocab processor embeddings and lastly, Twitter dataset with GloVe word embeddings. We found TensorFlow vocab processor embedding worked better than GloVe perhaps because both of our datasets were quite small. We got maximum test accuracy of 94%. The range was from 75% to 94% based on hyperparameters tuning and word embeddings. We saved our models after training them and embedded our best model with the chat app. Once the user is inside a chat room and types something, saved model predicts whether it is cyberbullying or not which is really useful in real life to flag the users which bullies and block them. We still need some improvements especially to discover huge authentic well labeled cyberbullying dataset to train our model for real life applications.

V. Conclusions

Deep learning is a promising approach to automatically detect cyberbullying in text messages. A great obstacle to train a deep neural network for this task is the lack of suitable datasets. There is a need to produce more datasets that contain verbal exchanges so the algorithms can be properly trained and produce accurate results. Besides the manual search for the optimal hyperparemeters it is also possible to automate this task. Automated Hyperparameter Tuning has the potential to find nearly optimal configurations in the parameter space. Training process and model architecture are regarded as a black box, which is executed with a specific hyperparameter configuration and outputs the resulting model accuracy. In order to minimize the number of training runs, stochastic methods (Gaussian processes / Bayesian optimization) are typically used to simultaneously explore the parameter space and iteratively approach an optimum. [33, p. 275 ff.] This is something that could be applied to our model to improve the accuracy even more.

[2]https://channels.readthedocs.io/en/latest/

References

[1] B. Liu, *Sentiment Analysis and Opinion Mining*. Morgan & Claypool Publishers, 2012.

[2] M. Anderson, "A majority of teens have experienced some form of cyberbullying," https://www.pewinternet.org/2018/09/27/a-majority-of-teens-have-experienced-some-form-of-cyberbullying/, 2018, accessed: 2019-07-25.

[3] F. Kock. (2017) 1,4 millionen schülerinnen und schüler von cybermobbing betroffen. [Online]. Available: https://www.sueddeutsche.de/panorama/jugendliche-im-internet-13-prozent-der-schueler-sehen-sich-als-opfer-von-cybermobbing-1.3507917

[4] D. L. Hoff and S. N. Mitchell, "Cyberbullying: causes, effects, and remedies," *Journal of Educational Administration*, vol. 47, pp. 652–665, 2009.

[5] L. Deng and Y. Liu, *Deep Learning in Natural Language Processing*. Springer Singapore, 2018.

[6] A. Yousef, W. Medhat, and H. Mohamed, "Sentiment analysis algorithms and applications: A survey," *Ain Shams Engineering Journal*, vol. 5, 05 2014.

[7] K. Van Royen, K. Poels, W. Daelemans, and H. Vandebosch, "Automatic monitoring of cyberbullying on social networking sites: From technological feasibility to desirability," *Telematics and Informatics*, vol. 32, 01 2014.

[8] L. Zhang, S. Wang, and B. Liu, "Deep learning for sentiment analysis : A survey," *CoRR*, vol. abs/1801.07883, 2018. [Online]. Available: http://arxiv.org/abs/1801.07883

[9] M. Dadvar and K. Eckert, "Cyberbullying detection in social networks using deep learning based models; a reproducibility study," *ArXiv*, vol. abs/1812.08046, 2018.

[10] X. Zhang, J. Tong, N. Vishwamitra, E. Whittaker, J. P. Mazer, R. Kowalski, H. Hu, F. Luo, J. Macbeth, and E. Dillon, "Cyberbullying detection with a pronunciation based convolutional neural network," in *2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)*, Dec 2016, pp. 740–745.

[11] R. I. Rafiq, H. Hosseinmardi, R. Han, Q. Lv, S. Mishra, and S. A. Mattson, "Careful what you share in six seconds: Detecting cyberbullying instances in vine," in *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015*, ser. ASONAM '15. New York, NY, USA: ACM, 2015, pp. 617–622. [Online]. Available: http://doi.acm.org/10.1145/2808797.2809381

[12] S. Agrawal and A. Awekar, *Deep Learning for Detecting Cyberbullying Across Multiple Social Media Platforms*. Springer, Cham, 01 2018, pp. 141–153.

[13] M. Abdullah Al-Ajlan and M. Ykhlef, "Deep learning algorithm for cyberbullying detection," *International Journal of Advanced Computer Science and Applications*, vol. 9, 01 2018.

[14] ——, "Optimized twitter cyberbullying detection based on deep learning," pp. 1–5, 04 2018.

[15] A. Géron, *Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*, 2nd ed. O'Reilly Media, Inc., 2019.

[16] T. Ganegedara, *Natural Language Processing with TensorFlow: Teach language to machines using Python's deep learning library*, 1st ed. Packt Publishing, 2018.

[17] "Introduction to lstms with tensorflow," https://www.oreilly.com/ideas/introduction-to-lstms-with-tensorflow, accessed: 2019-07-25.

[18] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[19] J. Chung, Çaglar Gülçehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *ArXiv*, vol. abs/1412.3555, 2014.

[20] M. Banko and E. Brill, "Scaling to very very large corpora for natural language disambiguation," in *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, ser. ACL '01. Stroudsburg, PA, USA: Association for Computational Linguistics, 2001, pp. 26–33.

[21] C. Van Hee, G. Jacobs, C. Emmery, B. Desmet, E. Lefever, B. Verhoeven, G. De Pauw, W. Daelemans, and V. Hoste, "Automatic detection of cyberbullying in social media text," *PLOS ONE*, 2018. [Online]. Available: https://journals.plos.org/plosone/article/file?id=10.1371/journal.pone.0203794&type=printable

[22] S. Salawu, Y. He, and J. Lumsden, "Approaches to automated detection of cyberbullying: A survey," *IEEE Transactions on Affective Computing*, pp. 1–1, 2018.

[23] R. Sprugnoli, S. Menini, S. Tonelli, F. Oncini, and E. Piras, "Creating a WhatsApp dataset to study pre-teen cyberbullying," in *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*. Brussels, Belgium: Association for Computational Linguistics, Oct. 2018, pp. 51–59. [Online]. Available: https://www.aclweb.org/anthology/W18-5107

[24] J. Bayzick, A. Kontostathis, and L. Edwards, "Detecting the presence of cyberbullying using computer software," 2011. [Online]. Available: http://webpages.ursinus.edu/akontostathis/BayzickHonors.pdf

[25] J.-M. Xu, K.-S. Jun, X. Zhu, and A. Bellmore, "Learning from bullying traces in social media," in *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, ser. NAACL HLT '12. Stroudsburg, PA, USA: Association for Computational Linguistics, 2012, pp. 656–666. [Online]. Available: http://dl.acm.org/citation.cfm?id=2382029.2382139

[26] E. Wulczyn, N. Thain, and L. Dixon, "Ex machina: Personal attacks seen at scale," *CoRR*, vol. abs/1610.08914, 2016. [Online]. Available: http://arxiv.org/abs/1610.08914

[27] Z. Waseem and D. Hovy, "Hateful symbols or hateful people? predictive features for hate speech detection on twitter," in *SRW@HLT-NAACL*, 2016.

[28] K. Reynolds, A. Kontostathis, and L. Edwards, "Using machine learning to detect cyberbullying," in *Proceedings of the 2011 10th International Conference on Machine Learning and Applications and Workshops - Volume 02*, ser. ICMLA '11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 241–244. [Online]. Available: http://dx.doi.org/10.1109/ICMLA.2011.152

[29] Y. Goldberg, *Neural Network Methods for Natural Language Processing*. Morgan Claypool Publishers, 2017.

[30] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543. [Online]. Available: http://www.aclweb.org/anthology/D14-1162

[31] M. Naili, A. Habacha, and H. Ben Ghezala, "Comparative study of word embedding methods in topic segmentation," *Procedia Computer Science*, vol. 112, pp. 340–349, 12 2017.

[32] "Understanding hyperparameters and its optimisation techniques," https://towardsdatascience.com/understanding-hyperparameters-and-its-optimisation-techniques-f0debba07568, accessed: 2019-07-25.

[33] U. Michelucci, *Applied Deep Learning: A Case-Based Approach to Understanding Deep Neural Networks*, 1st ed. Berkely, CA, USA: Apress, 2018.

[34] "Intro to optimization in deep learning: Gradient descent," https://blog.paperspace.com/intro-to-optimization-in-deep-learning-gradient-descent/, 2018, accessed: 2019-07-25.

[35] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *International Conference on Learning Representations*, 12 2014.

[36] "Dropout explained and implementation in tensorflow," http://laid.delanover.com/dropout-explained-and-implementation-in-tensorflow/, accessed: 2019-07-25.

[37] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jan. 2014. [Online]. Available: http://dl.acm.org/citation.cfm?id=2627435.2670313