STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LXV**, Number 2, 2020
DOI: 10.24193/subbi.2020.2.02

EMPLOYING LONG SHORT-TERM MEMORY NETWORKS IN TRIGGER DETECTION FOR EMETOPHOBIA

MARIA-MĂDĂLINA MIRCEA

Abstract. Research focused on mental health-related issues is vital to the modern person's life. Specific phobias are part of the anxiety disorder umbrella and they are distressing afflictions. *Emetophobia* is the rarely known, yet fairly common and highly disruptive specific phobia of vomiting. Unlike other phobias, emetophobia is triggered not only by the object of the specific fear, but also by verbal and written mentions of said object. This paper proposes and compares ten neural network-based architectures that discern between triggering and non-triggering groups of written words. An interface is created, where the best models can be used in emetophobia-related applications. This interface is then integrated into an application that can be used by emetophobes to censor online content such that the exposure to triggers is controlled, patient-centered, and patient-paced.

1. Introduction

For the longest time, mental health has been a largely overlooked aspect of people's lives. This fact is worrisome, especially given the fact that globally, over 10% of the population suffers from at least one mental health disorder [9]. Ranging from mild anxiety to substance abuse and severe depression, mental health problems affect daily activities and even become life-threatening if left untreated.

The specific phobia, a very common affliction, is part of the most widely-spread set of mental health conditions, namely, anxiety disorders. *Emetophobia*, the specific phobia of vomiting and sickness, is fairly familiar to the average person, but scarcely by name. It is, however, a life-altering, highly disruptive phobia, with around 2% of male and 7% of female sufferers [10].

Received by the editors: 24 August 2020.

2010 *Mathematics Subject Classification.* 68T05, 68T50.

1998 *CR Categories and Descriptors.* I.2.6 [**Artificial Intelligence**]: Learning – *Induction*; I.2.7 [**Artificial Intelligence**]: Natural Language Processing – *Text analysis*.

Key words and phrases. machine learning, text classification, long short-term memory, trigger detection, natural language processing, neural networks.

Emetophobia is highly disruptive, mostly because the main trigger comes from inside the body, which people cannot "escape". The act of being physically ill is usually the main trigger for emetophobes, but it is definitely not the only one. What is uncommon about this affliction is the fact that triggers also include verbal or written communications, images, audio recordings or videos that suggest, either implicitly or explicitly, the act of being ill.

The purpose of the current paper is to demonstrate the use of modern Machine Learning (ML) techniques in written trigger identification for emetophobia, going into detail about the technical aspects, from the architectures we used to the plugin we developed. We propose and compare multiple neural-network architectures, from simple *artificial neural networks* (ANNs) to Bidirectional Long Short-Term Memory Networks (BiLSTMs). We then choose the best performing ensemble of models and create an application that is useful for the therapy of people with emetophobia.

Emetophobia is rarely studied, especially in relation to Computer Science or Artificial Intelligence. The task we focus on is, to the best of our knowledge, the first of its kind. Text classification, however, is not new and BiLSTM nerworks are state-of-the-art in this regard. This motivates our chosen approach. We developed our own models using Tensorflow and Keras, and tested 10 architectures. We achieved outstanding results on our dataset, which were also tested "in the wild" within our own API and trigger-censoring application.

The rest of the paper is organized as follows. Section 2 provides more details about the issue at hand. Section 3 introduces the methodology we propose for emetophobia trigger identification and details the process of collecting the dataset, constructing and comparing the models, and implementing the API and the final application. Section 4 provides an analysis of our results. Section 5 presents what we intend to do in the future to provide more help for people suffering from this condition.

2. Background

This section provides details about emetophobia and people suffering from it. Then, we describe the ML techniques we used and the reason they were the best choice for the task. Finally, we provide an analysis of related work.

2.1. Emetophobia.

Emetophobia is not usually regarded as serious or worthy of extensive research. The main reason for this is that "No one likes being sick". It is a very common perception that emetophobia is just that: fear of being sick. However, it is much more than that. People suffering from emetophobia construct their whole lives around their avoidance and safety-seeking behaviours. Be it founded or completely unfounded, the fear is constant and consistent throughout their day and the anxiety is scarcely reduced in intensity.

Avoidance behaviours include, but are not limited to, staying inside, not eating at restaurants, rarely attending events, avoiding places where people usually drink, etc. Safety behaviours include carrying plastic bags in case they feel sick, taking many pills, sanitizing their body and surroundings too often, burning their food to make sure it is cooked, etc.

The exact situation emetophobes fear varies. A 2011 Dutch study [10] arrived to the conclusion that a large majority of people fear vomiting themselves, but another large percentage fear vomiting around others or seeing others vomit. Around one-fifth of the participants fear all 3 of these events [10]. Another interesting characteristic of many people suffering from this phobia is that they, ironically, experience nausea when they are anxious or scared, which makes for a gruesome vicious cycle.

Avoidance and safety-seeking behaviours seem to be the primary factor that makes one's fear stagnate or increase with time [10]. Since censoring triggers can be considered avoidance, we feel it is necessary to explain the primary purpose of our work. The application we developed should be used in order for emetophobes to be exposed to triggers gradually, in a controlled environment, without fear that their free time spent online will trigger their phobia and disrupt their regular day. This application is not meant to replace therapy or encourage avoidance of any kind. On top of that, the API has additional uses, which we will discuss in Section 5.

2.2. **Machine Learning Techniques.** *Artificial Neural Networks* (ANNs) started to be hypothesized and, later on, implemented, in the 1940s. They are considered by many to be the basis of Artificial Intelligence (AI). ANNs loosely take after anatomical neurons and synapses, in an attempt to eventually reach human-level perception and understanding. Even though AI is still far from this goal, ANNs are useful in many Computer Science tasks and applications, from stock market predictions to image recognition and cybersecurity.

As an improvement to simple ANNs, *Recurrent Neural Networks* (RNNs) were introduced to allow the algorithm to analyse sequences of information of arbitrary length. RNNs, however, come with a high risk of exploding or vanishing gradient, which is where Long Short-Term Memory Networks (LSTMs) step in. LSTMs choose when to remember and when to forget information they parsed. They, however, only analyse information from the past. Bidirectional LSTMs are an enhancement of LSTMs, where information is parsed forwards, as well as backwards, in order to provide a better understanding of the input.

Machine Learning techniques have been growing in efficiency, as well as complexity, since their very beginning. At this moment in time, BiLSTMs are considered to be state-of-the-art in sequence analysis tasks, such as text classification.

2.3. **Related work.** Since emetophobia is scarcely studied in relation to Computer Science and Machine Learning, the next best comparison would be hate speech detection, followed closely by sentiment analysis. With the freedom of speech offered by the internet and, more specifically, social media, the issue of hate speech has become more and more prevalent and worrisome. Automated detection of hate speech is a need humanity could not have predicted a number of years ago. This task has been tackled by many researchers lately, with relevant work starting to emerge in 2015 or even earlier.

Davidson et al. [3] used 3 output classes (Clean, Offensive, and Hate) instead of the previously-popular 2 (Clean vs Hate) to differentiate between hate speech and offensive language. This is because many offensive terms are used online daily in a non-pejorative manner and the distinction is crucial. The dataset was made up of 25 thousand examples of tweets. The tweets were labeled by hand, by at least 3 workers, the final label being decided in a "most votes" fashion. The ML models compared in this article were Logistic Regression, Naive Bayes, Support Vector Machines, etc. [3].

Badjatiya et al. [1] compared different combinations of neural networks and embeddings to find the most appropriate one for tweet classification. The 3 classes used for the task are sexist, racist, and neither sexist nor racist. The dataset contained 16k instances of annotated tweets. Training was performed using 10-fold cross-validation. One of the best combinations proved to be LSTM with Random Embeddings, improved with Gradient Boosted Decision Trees. CNNs and LSTMs also performed admirably when combined with GloVe embeddings, and even better when also using GBDT [1].

Recently, Do et al. [5] employed a Bidirectional LSTM network to identify hate speech in Vietnamese social media text. The dataset used contained over 25 thousand Twitter comments, about 20,000 being used for training and 5,000 for testing. Each item was assigned a label from 1 to 3, 1 meaning CLEAN, 2 meaning OFFENSIVE, and 3 meaning HATE [5]. The model was compared with Support Vector Machines, Logistic Regression, and Gated Recurrent Units, this comparison clearly showcasing that the BiLSTM was the best choice for the task.

Other recent articles used SVMs [11], Bag-of-words [4], and other similar techniques, which are not necessarily comparable to our proposed approach.

3. Methodology

This section introduces the methodology we propose for emetophobia trigger identification. As with any similar endeavour, the first step was to extract a specific dataset of words and phrases, then to develop ML models capable of discerning between triggering and non-triggering phrases, followed by a thorough evaluation of the models. The final steps consisted of developing an API and a useful application where this API is integrated.

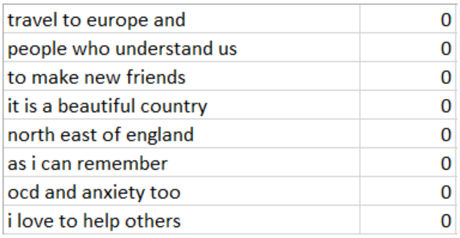
3.1. **Data set.** The dataset was chosen manually by searching the internet, specifically social media websites like Facebook and Instagram, and extracting words and phrases. Social Media was chosen as source for the data because nowadays, most people have a habit of spending a long time on the internet, hence this is where the probability of an encounter with a trigger rises drastically. The main reason for manually creating a new dataset is that research in emetophobia in relation to Computer Science is scarce, so there is no dataset that we were able to find that can be used in the task at hand.

We first selected 600 sentences, which were split into 300 triggers and 300 non-triggers. Triggers included sentences like “I feel sick” or “my child just threw up”. Non-triggers included everything from “Happy birthday” to “I am traveling to Europe”. The first version of the dataset contained phrases of up to 17 words. This version could have worked well with other approaches to the same task, but not to the one we chose. For example, if the models were directly trained to take as input longer texts of varying lengths and highlight the triggering parts of each text, then this dataset would have been useful. However, this is not the approach we chose.

Our models were trained to determine if a short piece of text is triggering or not, using an output value from 0 (meaning not triggering) to 1 (meaning triggering). The model has no information about the text as a whole. It does not receive a whole webpage, whole paragraphs or even whole sentences. The model only receives a small segment of the text and determines if that segment is triggering, then it receives the next segment of the text, and so on. The segment of the text the model receives is controlled from the outside by the algorithm in the API, which uses a sliding window approach, as detailed in Section 4.1.

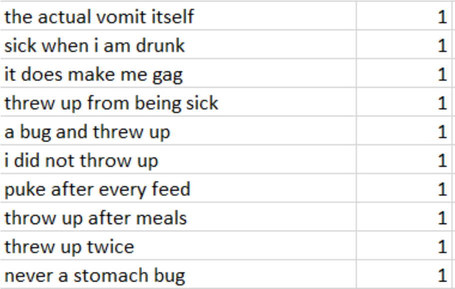
Since the model receives short segments of the text and determines if they are triggering or not triggering, we found it fitting to modify the dataset so that the training is performed on appropriate sentences. If the model received a sentence of 17 words which was marked as triggering, it would be impossible for it to determine exactly which part of the sentence makes it triggering. It is unlikely that all of the 17 words are, in themselves, triggering. However, for shorter sentences, say, up to 5 words, the model can assume that all of the words form a triggering phrase and, thus, it can label the sentence as a whole as triggering. This is why we decided to manually parse our dataset and extract the precise 5-word segment of each entry that determines if it is triggering. For example, the sentence ”I traveled to Europe last month and the turbulence made me sick, but I felt fine later” became ”the turbulence made me sick”. The final dataset contains 600 sentences of up to 5 words.

The pre-processing of the dataset consisted of making all letters into lowercase, removing punctuation, numbers, emojis, special characters, repeating white space, etc. The next step parsed the dataset and replaced each word with its lemma using the *morphy* function [6] provided by Princeton's WordNet. The tokenization was performed using the *TwitterTokenizer* provided by NLTK [2]. Finally, the triggering phrases were marked with a value of 1, while the non-triggering phrases were marked with 0. Two fragments from the dataset can be seen in Figures 1 and 2. The dataset was made publicly available on Github [7].



travel to europe and	0
people who understand us	0
to make new friends	0
it is a beautiful country	0
north east of england	0
as i can remember	0
ocd and anxiety too	0
i love to help others	0

FIGURE 1. Non-triggering phrases in the dataset



the actual vomit itself	1
sick when i am drunk	1
it does make me gag	1
threw up from being sick	1
a bug and threw up	1
i did not throw up	1
puke after every feed	1
throw up after meals	1
threw up twice	1
never a stomach bug	1

FIGURE 2. Triggering phrases in the dataset

3.2. **Proposed ML models.** Since trigger identification requires analysis of word sequences, LSTMs are the most suitable ML technique for the task. Most of the model architectures used regression. However, to achieve as complete a picture as possible, we also considered simple ANN and BiLSTM architectures, as well as classification models for each of these approaches. Some models use the NLTK stopwords dictionary during the tokenization process, while others do not. Other differences between models consisted of different dropout values, a different number of epochs, different learning rates, etc. Something that all of the models had in common was the first layer (excluding the input layer), which was an Embedding layer. In all instances, this layer used Stanford's GloVe Twitter pre-trained word vectors [8] with 100 dimensions.

The first architecture we trained and tested was a LSTM model with one layer of 100 units and a dropout value of 25%. This layer was then flattened and connected to a fully connected layer of 100 units, a fully connected layer of 25 units, and, finally, the output layer with 1 unit. This model used the stopwords dictionary, 100 training epochs and the default TensorFlow value for the learning rate. The architecture is illustrated in Figure 3. This model is henceforth denoted LSTM_1. For the second model, denoted LSTM_2, the first architecture was modified by adding one more LSTM layer with 100 units and a dropout value of 25%.

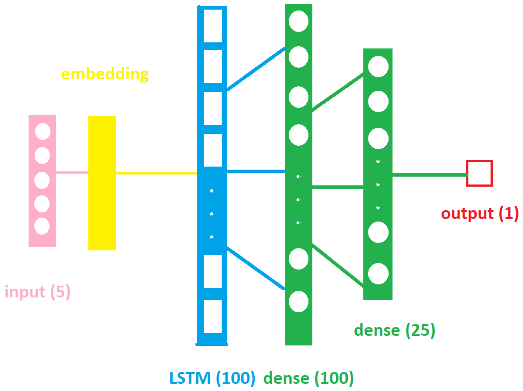
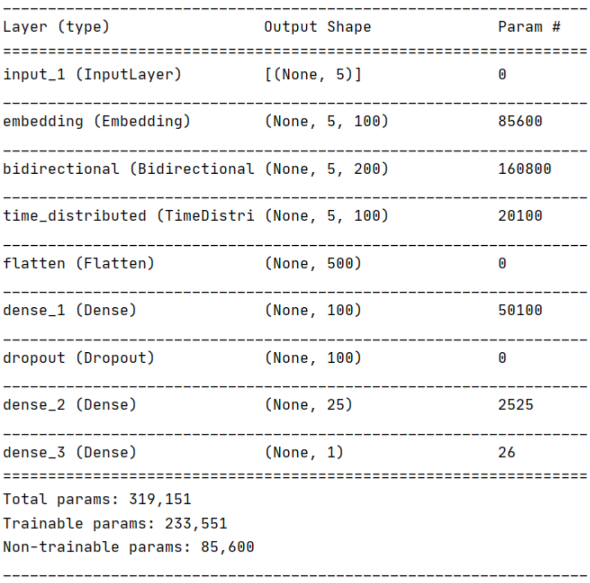


FIGURE 3. The architecture of the LSTM_1 model

The following 3 proposed models are all BiLSTM models. They use the stopwords dictionary modified by removing the word "up". This modification was performed because phrases like "throw up" were deemed important for this study. The first BiLSTM model (denoted BiLSTM_1) consisted of the input layer, the embedding layer, one BiLSTM layer with 200 units, which was then flattened and fed into a fully connected layer with 100 neurons, and then into the output layer. This model was trained for 100 epochs with the default learning rate value. The second BiLSTM model (denoted BiLSTM_2) had different fully connected layers, (one with 100 neurons, one dropout layer with a value of 25%, one fully connected layer with 25 neurons). The architecture, as given by the *summary()* method provided by Keras, is displayed in Figure 4. This model was trained for 200 epochs with a learning rate of 0.0001.



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 5)]               0
_________________________________________________________________
embedding (Embedding)        (None, 5, 100)            85600
_________________________________________________________________
bidirectional (Bidirectional (None, 5, 200)            160800
_________________________________________________________________
time_distributed (TimeDistri (None, 5, 100)            20100
_________________________________________________________________
flatten (Flatten)            (None, 500)               0
_________________________________________________________________
dense_1 (Dense)              (None, 100)               50100
_________________________________________________________________
dropout (Dropout)            (None, 100)               0
_________________________________________________________________
dense_2 (Dense)              (None, 25)                2525
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 26
=================================================================
Total params: 319,151
Trainable params: 233,551
Non-trainable params: 85,600
_________________________________________________________________
```

FIGURE 4. The Keras architecture of the BiLSTM_2 model

The final BiLSTM regression model (denoted BiLSTM_3) was also the largest and most computationally expensive one. The number of BiLSTM layers was doubled, and one more dropout layer was added between the last fully connected layer and the output neuron. This model was trained with the modified stop words dictionary, for 200 epochs, with a learning rate of 0.00005. Finally, two simple ANN models were constructed. The first model is illustrated in Figure 5. It was trained with the modified stop words dictionary, for 100 epochs. The second ANN model had the same structure, with additional dropout layers.

The classifications models we tested were variations of the architectures we already described. The main difference was at the output layer level, where two neurons were used instead of one. The output, which had so far been Boolean in nature, was one-hot-encoded to fit the new output layer structure.

After the best architecture was chosen, an ensemble of models was created. Each of the models in the ensemble was given a weight based on the metrics obtained in the testing phase and the final output was decided by computing a weighted average of all of the output values.

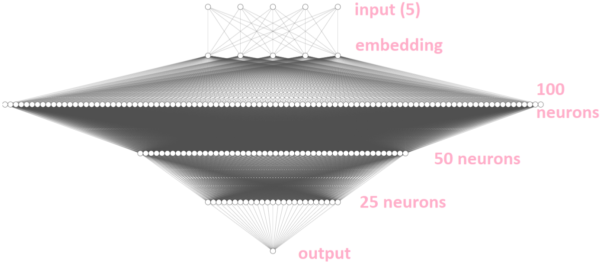
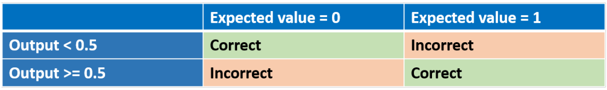


Figure 5. The architecture of the ANN_1 model

3.3. **Evaluation.** For the classification models, we used Binary Crossentropy as the loss function and accuracy for metrics. For the regression models, the main loss function used was *Mean Squared Error* (MSE). We implemented an accuracy measure for the regression models as well. The principle we used for this accuracy measure is illustrated in Figure 6. The same principle was applied in the final API.

We used 10-fold cross-validation, so that the result obtained was as objective as possible, with a lower probability of it becoming overfitted or biased towards parts of the dataset. One instance of each model was randomly given one fold for testing, and the other 9 folds for training. Thus, resulted were 10 different instances per architecture. When computing the best architecture, all of these models were taken into account. The 10 models were ordered ascendingly by loss. Then, the first 3 models (with the lowest loss) were assigned a weight of 0.1, the following 4, a weight of 0.2, and the final 3 were assigned a weight of 0.3. The weights given to the instances within an ensemble are displayed in Figure 7.

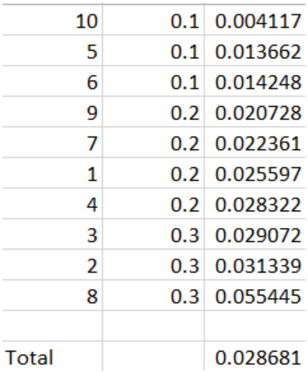


	Expected value = 0	Expected value = 1
Output < 0.5	Correct	Incorrect
Output >= 0.5	Incorrect	Correct

Figure 6. Accuracy measure for the regression models

4. Results and discussion

A comparison of the results obtained for all of the architectures can be seen in Figure 8. This table shows the model name, the model accuracy, and the model loss for all of the architectures. For the three BiLSTM regression architectures, more measures are displayed: Mean-Squared Error (denoted MSE), Precision (P), Recall (R), and F1 (the F1 score). These measures were obtained for the ensembles on the 600 entries in the whole dataset. All of



10	0.1	0.004117
5	0.1	0.013662
6	0.1	0.014248
9	0.2	0.020728
7	0.2	0.022361
1	0.2	0.025597
4	0.2	0.028322
3	0.3	0.029072
2	0.3	0.031339
8	0.3	0.055445
Total		0.028681

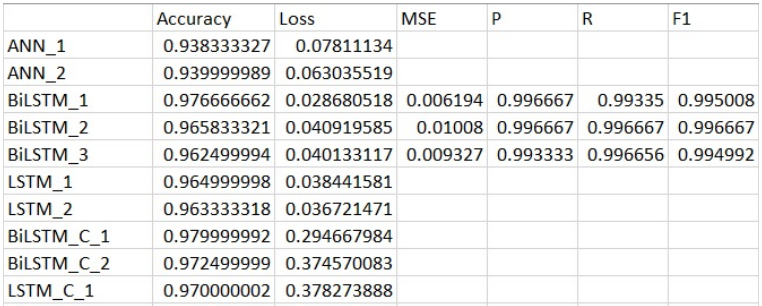
FIGURE 7. Final loss of an ensemble - The first column represents the order number of the instance; the second column represents the weight assigned to the instance; the third column represents the loss of the instance; The last line in the table, labeled "Total", shows the final loss of the ensemble, computed using a weighted sum of the model losses (i.e. $Total = \frac{\sum\limits_{m=1}^{10} loss(m)*weight(m)}{2}$, where m=model)

the models performed admirably, but the best performance was measured for the BiLSTM network with one layer (denoted BiLSTM_1). Even though the accuracy was comparable for most of the models, the loss for this architecture was lower than the other regression models and significantly lower than the classification models. After selecting this model as the winner, we constructed the ensemble, assigning weights to the instances following the rules described above.

Figure 9 illustrates a graph of the loss with respect to the training epoch for the 10 instances of the chosen architecture (namely, BiLSTM_1). Similar to the other architectures, this model learned quickly within the first 10 to 20 epochs, then slowed down until the final epoch. All of the 10 instances seem to have learned at the same rate, achieving a similar loss by the end of the training phase. We tried to slow down the learning process in hopes of achieving an even better result by decreasing the learning rate and increasing the number of epochs, but the performance did not exceed the best performance achieved previously.

4.1. **API and Application.**

We developed a server written in Flask (Python) that implements the approach described in the previous sections. The main endpoint of the API receives a POST request with the text to analyse. The



	Accuracy	Loss	MSE	P	R	F1
ANN_1	0.938333327	0.07811134				
ANN_2	0.939999989	0.063035519				
BiLSTM_1	0.976666662	0.028680518	0.006194	0.996667	0.99335	0.995008
BiLSTM_2	0.965833321	0.040919585	0.01008	0.996667	0.996667	0.996667
BiLSTM_3	0.962499994	0.040133117	0.009327	0.993333	0.996656	0.994992
LSTM_1	0.964999998	0.038441581				
LSTM_2	0.963333318	0.036721471				
BiLSTM_C_1	0.979999992	0.294667984				
BiLSTM_C_2	0.972499999	0.374570083				
LSTM_C_1	0.970000002	0.378273888				

Figure 8. Comparison of the metrics for the 10 architectures

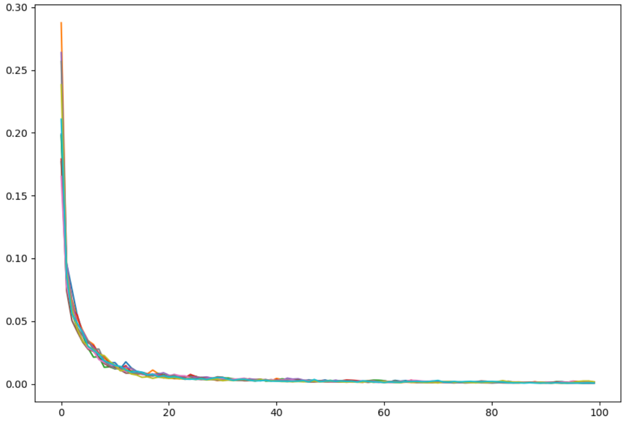


Figure 9. Loss graph for the winning model

response contains a text of the same length, with the triggering phrases censored using the asterisk character. The text is parsed using a “sliding-window” approach, detailed in what follows. The first 5 words are analysed. If they are triggering, they are replaced with asterisks and the algorithm goes on to the next 5 words, i.e. words 5-10 of the text. If they are non-triggering, the algorithm moves 3 words to the right, thus keeping the last two words from the previous input to ensure that these words were not part of a triggering phrase.

As an example, the text ”I traveled to Europe last month and the turbulence made me sick, but I felt fine later” will be parsed as follows. First, the words ”I traveled to Europe last” will be analysed and they are labeled as negative.

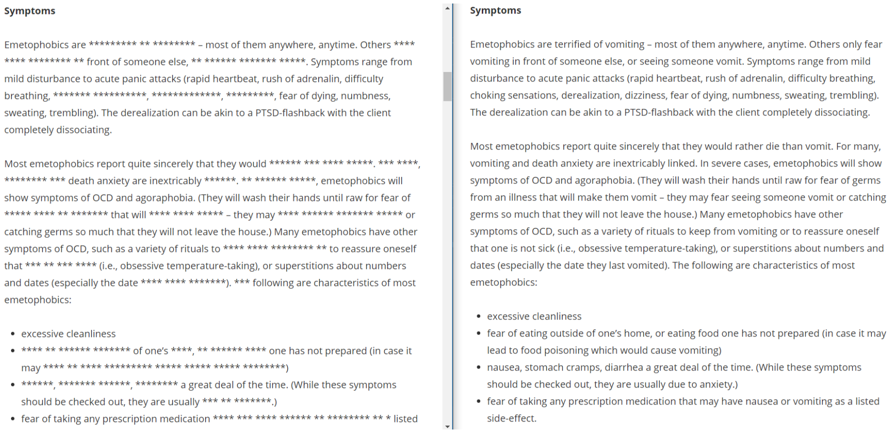
The algorithm appends the words ”I traveled to” to the response and moves on to analyse the phrase ”Europe last month and the”. It is not triggering, so the response becomes ”I traveled to Europe last month” and the algorithm goes on to analyse the phrase ”and the turbulence made me”. The input is not triggering, so the response becomes ”I traveled to Europe last month and the turbulence”. The algorithm now analyses the phrase ”made me sick, but I”, which is labeled as positive. The response thus becomes ”I traveled to Europe last month and the turbulence **** ** ****, *** *”. The algorithm now analyses the phrase ”felt fine later”, which is labeled as negative. The response ”I traveled to Europe last month and the turbulence **** ** ****, *** * felt fine later” is returned by the endpoint.

The API was utilized in a Chrome extension that censors written triggers encountered online. Figure 10 depicts a webpage with and without the extension. The algorithm is very strict with variations of the word “eat”. For our purposes, this could be considered extreme. The same applies for mentions of the word “anxiety”. As mentioned above, the word ”up” causes problems for the algorithm. These issues are all mild since they cause excessive censorship, which is better than no censorship at all.

False positives are to be preferred in such a scenario, where the sufferer would rather see less of the non-triggering text of a webpage, rather than more of the triggering text. Furthermore, there are extreme cases where even the words ”eat” or ”anxiety” cause an increase in the stress levels of an individual, so such censorship, even when not expected or purposeful, could end up adding to the positive characteristics of the application. False negatives, on the other hand, would be a much graver issue. The problem of false negatives is one that, if present, should be brought to a minimum right away. The false negatives we encountered were only labeled as negative because of improper representation in the dataset.

4.2. **Comparison to related work.** Emetophobia is rarely focused on in research, especially with respect to Computer Science and Artificial Intelligence. This is the reason why we were unable to find papers on this topic that we could compare our results to. Our solution to this issue was to compare with similar works in slightly different domains. Hate speech detection is also an up and coming topic in research, since the time spent online by individuals is increasing and this leads to more and more exposure to hateful and rude comments. We deemed this task similar enough to emetophobia trigger detection to address comparable works in this section.

The pre-processing phase we employed in our approach is similar to the approaches we analysed. GloVe embeddings are also frequent in research that



Symptoms

Emetophobics are ********* ** ******** – most of them anywhere, anytime. Others **** **** ******** ** front of someone else, ** ****** ******* *****. Symptoms range from mild disturbance to acute panic attacks (rapid heartbeat, rush of adrenalin, difficulty breathing, ******* **********, *************, *********, fear of dying, numbness, sweating, trembling). The derealization can be akin to a PTSD-flashback with the client completely dissociating.

Most emetophobics report quite sincerely that they would ****** *** **** *****. *** ****, ******** *** death anxiety are inextricably ******. ** ****** *****, emetophobics will show symptoms of OCD and agoraphobia. (They will wash their hands until raw for fear of ***** **** ** ******* that will **** **** ***** – they may **** ****** ******* ***** or catching germs so much that they will not leave the house.) Many emetophobics have other symptoms of OCD, such as a variety of rituals to **** **** ******** ** to reassure oneself that *** ** *** **** (i.e., obsessive temperature-taking), or superstitions about numbers and dates (especially the date **** **** *******). *** following are characteristics of most emetophobics:

- excessive cleanliness
- **** ** ****** ******* of one’s ****, ** ****** **** one has not prepared (in case it may **** ** **** ********* ***** ***** ***** ********)
- ******, ******* ******, ******** a great deal of the time. (While these symptoms should be checked out, they are usually *** ** *******.)
- fear of taking any prescription medication **** *** **** ****** ** ******** ** * listed

Symptoms

Emetophobics are terrified of vomiting – most of them anywhere, anytime. Others only fear vomiting in front of someone else, or seeing someone vomit. Symptoms range from mild disturbance to acute panic attacks (rapid heartbeat, rush of adrenalin, difficulty breathing, choking sensations, derealization, dizziness, fear of dying, numbness, sweating, trembling). The derealization can be akin to a PTSD-flashback with the client completely dissociating.

Most emetophobics report quite sincerely that they would rather die than vomit. For many, vomiting and death anxiety are inextricably linked. In severe cases, emetophobics will show symptoms of OCD and agoraphobia. (They will wash their hands until raw for fear of germs from an illness that will make them vomit – they may fear seeing someone vomit or catching germs so much that they will not leave the house.) Many emetophobics have other symptoms of OCD, such as a variety of rituals to keep from vomiting or to reassure oneself that one is not sick (i.e., obsessive temperature-taking), or superstitions about numbers and dates (especially the date they last vomited). The following are characteristics of most emetophobics:

- excessive cleanliness
- fear of eating outside of one’s home, or eating food one has not prepared (in case it may lead to food poisoning which would cause vomiting)
- nausea, stomach cramps, diarrhea a great deal of the time. (While these symptoms should be checked out, they are usually due to anxiety.)
- fear of taking any prescription medication that may have nausea or vomiting as a listed side-effect.

FIGURE 10. Comparison of a webpage with (left) and without (right) the extension enabled

utilizes social media text since these embeddings are trained on billions of tweets and have learned colloquial language well.

Perhaps the most obvious difference between the proposed approach and other comparable works is the dataset. Most similar articles had access to thousands of instances, while we only used 600. Some had workers manually label their data according to strict rules, thus ensuring accurate and consistent labels, while we relied on our own perception when extracting relevant phrases. Our dataset can and will be improved upon. Another important distinction is that most similar articles use classification models, while our proposed approach highlights regression models as better-performing for this specific task. Classification models were similar in accuracy, but with a significantly higher loss than regression models.

5. Conclusions and future work

This paper presented our approach to the issue of censoring online content that can be triggering for people with emetophobia (the extreme fear of vomiting). We collected a dataset, constructed different neural network architectures, and compared their performance on our dataset. The architecture with the best performance was turned into an ensemble. This ensemble was utilized in an API, which was then incorporated into a Chrome extension that censors triggering content on the internet.

Future improvements to the existing API will commence with expanding the dataset to include a larger number of instances. Tweaks need to be made to make sure that the model can discern between triggering and non-triggering uses of the word "up", "anxiety", "throw", etc. Additional endpoints of the API can be used in an upcoming project, a Virtual Reality therapy application currently in development. The amount of triggering phrases utilized by the user will decide if they are ready to pass to the next level or not.

Modern ML techniques have many uses in everyday life. While some uses are purely for entertainment or comfort, others can drastically improve the life of people suffering with certain afflictions. Since mental health is such an important aspect of one's life, applications that aid therapy or improve a patient's quality of life will remain a crucial part of Computer Science research for years to come.

References

[1] Badjatiya, P., Gupta, S., Gupta, M., and Varma, V. Deep learning for hate speech detection in tweets. In *Proceedings of ACM WWW'17 Companion* (Republic and Canton of Geneva, CHE, 2017), WWW '17 Companion, International World Wide Web Conferences Steering Committee, p. 759–760.

[2] Bird, S., Klein, E., and Loper, E. *Natural language processing with Python: analyzing text with the natural language toolkit.* " O'Reilly Media, Inc.", 2009.

[3] Davidson, T., Warmsley, D., Macy, M., and Weber, I. Automated hate speech detection and the problem of offensive language. *ICWSM* (2017), 512–515.

[4] Djuric, N., Zhou, J., Morris, R., Grbovic, M., Radosavljevic, V., and Bhamidipati, N. Hate speech detection with comment embeddings. In *Proceedings of the 24th International Conference on World Wide Web* (New York, NY, USA, 2015), WWW '15 Companion, Association for Computing Machinery, p. 29–30.

[5] Do, H. T.-T., Huynh, H. D., Van Nguyen, K., Nguyen, N. L.-T., and Nguyen, A. G.-T. Hate speech detection on vietnamese social media text using the bidirectional-lstm model. In *Proceedings of VLSP 2019* (2019).

[6] Fellbaum, C. *WordNet: An Electronic Lexical Database.* Bradford Books, 1998.

[7] Mircea, M.-M. Emetophobia dataset, Aug. 2020.

[8] Pennington, J., Socher, R., and Manning, C. D. Glove: Global vectors for word representation. In *In EMNLP* (2014).

[9] Ritchie, H., and Roser, M. Mental health. *Our World in Data* (2018). https://ourworldindata.org/mental-health.

[10] van Hout, W. J. P. J., and Bouman, T. K. Clinical features, prevalence and psychiatric complaints in subjects with fear of vomiting. *Clinical Psychology & Psychotherapy 19*, 6 (2012), 531–539.

[11] Warner, W., and Hirschberg, J. Detecting hate speech on the world wide web. In *Proceedings of the Second Workshop on Language in Social Media* (Montréal, Canada, June 2012), Association for Computational Linguistics, pp. 19–26.

Department of Computer Science, Faculty of Mathematics and Computer Science, Babeş-Bolyai University, 1 Kogălniceanu St., 400084 Cluj-Napoca, Romania

Email address: mmic2002@scs.ubbcluj.ro