Department of Informatics
King’s College London
United Kingdom

7CCSMPRJ MSc Project

AUTOMATIC DETECTION AND BLOCK OF CYBERBULLYING THROUGH BROWSER EXTENSIONS

Name: WANG JIAYING
Student Number: 20006542
Degree Programme: Master of Science

Supervisor’s Name: Islam, Tasmina

This dissertation is submitted for the degree of MSc in Cyber Security

ACKNOWLEDGEMENT

The work of this thesis was completed under the careful guidance of my tutor Islam, Tasmina. Dr. Islam, Tasmina have given me great help in the analysis of the topic of the thesis, the control of the research direction, and the sorting out of the specific research content. The teacher's rigorous attitude towards academic papers, the meticulous perseverance towards engineering projects, and the amiable and gracious teaching to students have profoundly affected me, and it will influence a lot for my academic career and career path in the future.

In addition, I am very grateful to my classmates for recommending learning methods and related books and materials in terms of theoretical knowledge, carefully guiding in familiarization with the operating environment, putting forward constructive suggestions in experimental design, and patiently helping me to infer the wrong steps when specific programming problems occur. Besides, I also want to thank the school for its experimental support. Finally, I want to thank my parents for encouraging me in difficult times and supporting me unconditionally. They are my most reliable and solid backing.

ABSTRACT

A new type of violence is growing rapidly with the popularity of the Internet and online social media such as Facebook, Instagram, Twitter, etc. Cyberbullies use Internet tools for initiating violent behavior against target victims, especially young people, such as intimidation, threats, insult, abuse, and slander. Their strong subjective and hostile comments at the same time instigated emotions from net-conscious people, who did not grasp the truth about the incident, broadened the ranks of criminals, and finally battered victims spiritually. This project uses stacking of RNN and CNN model trained with the data set from Kaggle in advance to detect malicious comments on the web version of the social network platform. Besides, purifying the malicious comments classified as "positive", i.e., replacing them into harmless kaomoji through the browser plug-in, Cyberbullying Blocker. The RNN-CNN model achieves a better classification result after multiple sets of comparative experiments whose train AUC is up to 0.99 and the best test accuracy is about 0.93.

Keywords: cyberbullying, automatic detection, NLP, RNN, TextCNN, browser plug-in

NOMENCLATURE

NLP	Natural Language Processing
AUC	Area under the curve
CNN	Convolutional Neural Networks
RNN	Recurrent neural network
LSTM	Long short-term memory
GRU	Gated Recurrent Unit

CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 INTRODUCTION

With the popularity of the Internet and online social media, a new type of violence is rapidly growing. Cyberbullies use the Internet as a tool to initiate violent behaviors such as intimidation, threats, insults, abuse, and slander against targeted victims, particularly young people. Unscrupulously express malicious remarks in cyberspace that cannot be fully covered by the law. At the same time, their strong subjective and hostile comments instigated the emotions of netizens who did not understand the truth of the incident, expanded the amounts of perpetrators, and finally caused a spiritually devastating blow to the victims.

To combat this new type of cyber violence, it is often necessary to combine the philosophical ideas of counterattack commonly used in cyberspace security, which means deter, detect, deny, delay, and defend the criminals i.e. 5D strategies(Morgan Landry, 2016). Besides, advanced technologies such as machine learning, deep learning, and natural language processing(NLP) algorithms are used as technical means to assist. This is because a small amount of cyberbullying of the victim can be prevented by manually setting a blacklist, but a large number of malicious language attacks like a kind of DDoS must be prevented by some automatic detection algorithms and powerful computing power.

1.1 Aims, Objectives, and Specifications

Cyberbullying uses the power of malicious language to cause great harm to the victim's psychology. Besides, as the Internet spreads more widely and rapidly, more people might be incited to join cyberbullying, causing secondary harm to the victims. So what this project wants to do is to cut off the victim's reception of malicious speech, like a firewall. After all, if it is impossible to block others' "free speech", at least, there should be a chance to be selective blind. After some investigation, it shows that there are many algorithms of automatic detection based on machine learning, also, enterprise-oriented digital content detection services. However, there are few user-friendly and adaptable functional plug-ins applied to multiple online social media. So this project will do some work to use advanced technology to improve the user experience of victims and potential victims on online social platforms.

The problem that this project aims to solve is to use the deep neural network model trained with a data set in advance to detect malicious comments on the web version of the social network platform, and to purify the malicious content that is classified as "positive", that is, to replace it into harmless kaomoji, similar to "o(*￣▽￣*)ブ". From the general user's point of view, this feature will appear as a browser extension that can automatically identify and block bad reviews.

This dissertation will first discuss the enlightenment of automatic detection of cyberbullying models obtained from the literature review, and then describe in detail the theoretical knowledge and measurement standards required by this dissertation. After that, the design and implementation of the entire system will be described in detail, and finally, the performance of the model in the given measurement system will be analyzed and summarized.

1.2 Background and Literature Survey of Cyberbullying

What should be clear before the entire project is the definition of cyberbullying: “Cyberbullying is bullying that takes place using electronic technology and communication tools. It includes the mean text messages or emails, rumors sent by email or posted on social networking sites, and embarrassing pictures, videos, websites, or fake profiles”(U.S. Department of Health and Human Services, 2020).

Although anti-cyberbullying developed for many years, there are still some challenges with cyberbullying research. For example, APIs limitations and over-emphasis of a single platform (e.g., Twitter). Furthermore, the choice of keywords or hashtags will specify the boundaries of data collection which might cause relevant data to be missed and result in overrepresentation of one class. Therefore, this project focuses on breaking the barriers between platforms and chooses to identify and block cyberbullying texts on the web version of major online social platforms based on the browser. At the same time, it pays attention to user experience, encapsulates the complex machine learning model used to automatically identify bad reviews in the back end, and only provides users with easy-to-operate interfaces, so that users can directly use the function of blocking bad reviews.

2 LITERATURE REVIEW

2.1 Automatic Detection of Cyberbullying

Cyberbullying, as a bullying strategy that has increased with the emergence of the Internet, has had a rising impact on the physical and emotional health of young people. Over the years, numerous research has focused on how bad information might be identified for the protection of victims in the internet environment. In recent years, the growth of artificial intelligence has dramatically enhanced the effectiveness of detecting cyberbullying. The combination of diverse natural language processing techniques, neural network models, and data sets all has a significant influence on recognition accuracy.

2.1.1 Pros and Cons of Existing Automatic Detection Algorithms

In cyberspace, the most common and dominant way to implement bullying is to convey offensive information through text. The first step to prevent victims from being stung by these sword-like text messages is to distinguish them from all normal texts. From the initial K-Nearest Neighbor(KNN), Conditional Random Field(CRF), Decision Tree, Random Forest, Support Vector Machine(SVM), and to the deep learning algorithm flourished due to the breakthrough of the computing power limit of the chip, the academic community has been continuously exploring text classification. The following is a comparison of the advantages and disadvantages of the existing classic automatic text detection models.

Algorithms	**pros**	**cons**
K-Nearest Neighbor	Mature theory; simple implementation; easy to understand; no training required (Peterson, 2009).	It is difficult to find the optimal value of K and a meaningful distance function; the amount of calculation is large; the prediction accuracy of the rare category in the imbalanced data set is low(Christopher, 2021).
Conditional Random Field	Feature design is flexible and can contain information of any context; overcome the shortcomings of label deviation by calculating the conditional probability of the global optimal output node (Sutton, 2012).	The training cost is high, and the calculation complexity is high; it is difficult to retrain the model when new data enters during online learning (Wang, Xu, Li & Ge, 2015).

Decision Tree	Simple and easy to understand; can process large amounts of data in a short time and the results converge quickly (Quinlan, 1986).	Easy to overfit; weak anti-interference ability to the data set; does not support online learning (Safavian & Landgrebe, 1991).
Random Forest	There is no need to preprocess the input data; when it is fed into an unbalanced data set, it can effectively balance errors (Breiman, 2001).	In the face of some specific noises, it is prone to overfitting; it is very slow when used for prediction after training; it is difficult to intuitively explain the internal operation of the model(Liaw, A., & Wiener, M, 2002).
Support Vector Machine	Can solve the problem of dividing the boundary of nonlinear decision-making; has a strong ability to resist over-fitting (Cortes & Vapnik, 1995).	It consumes a lot of memory; training on large data sets is difficult to implement (van Gestel et al., 2004).
Deep learning	High classification accuracy; strong parallel processing ability; because it is easy to transfer and retrain the trained model, it can handle online learning problems (Hao, Zhang & Ma, 2016).	A large number of training data sets are required; the computational cost of training the model is high; the discovery of an effective structure requires adjustment of parameters and a great number of experiments (Poomka, Kerdprasop & Kerdprasop, 2021).

Table 1. Analysis and comparison of classical automatic detection algorithms

With the continuous improvement of computing power, there are many adjustable aspects such as optimizable structure and training parameters of deep learning algorithms, and there is more room for development. Deep learning has greater potential in dealing with the explosive growth of text data in cyberspace over time, so this project chooses to choose better models in the deep learning algorithm for training and fusion, so as to maximize the realization of the function of automatically detecting cyberbullying text.

2.1.2 Classification Results of Existing Models and Comparison

The first and most critical step to identify and prevent cyberbullying is classification. From the perspective of the models trained on the Kaggle data set, they can be divided into two main types, one is traditional classifiers such as LR and GBM models, and the other is neural network models.

Based on research, C4.5 decision tree learner and an instance-based learner could identify 78.5%(true positives) using the data from the website Formspring. me (Reynolds, K., Kontostathis, A., & Edwards, L., 2012); compared with Logistic Regression (LR), Light Gradient Boosting Machine (LGBM), Stochastic Gradient Descent (SGD), Random Forest (RF), AdaBoost (ADB), Naive Bayes (NB), and Support Vector Machine (SVM) machine learning classifiers on a global Twitter dataset with 37,373 unique tweets, LR is superior(average accuracy 90.57%, F1 score 0.928), SGD had the best precision (0.968), and SVM reached the best recall (1.00) (Muneer, A., & Fati, S. M., 2020).

Although the training results of classical text classifiers based on small data sets are excellent, there will be issues like overfitting and lower resilience when faced with application scenarios including a significant quantity of continuously updated data, such as online social platforms. The usage of neural network models frequently employed in computer vision for natural language processing has been a prominent trend in recent years. When fresh data comes in, deep learning can retrain the model more simply and quickly, similar to online learning. It has parallel processing capabilities and can handle several jobs at the same time. The requirement for feature engineering is less in neural networks, and the design of extracting feature portions is more flexible, and they are better at mapping complicated inputs to outputs (Kowsari, 2019).

It is known that CNN(Convolutional Neural Networks) and RNN(Recurrent Neural Network) are both famous neural network models on classification(Kim, Y., 2014). According to the results of the classification competition on the Kaggle data set, CNN can better distinguish local semantic features, but it may destroy word order, while RNN can better preserve word order features; pseudo labeling is very effective when training the model, that is, when the model's true positive rate is high, the results of the test set predicted by the model will be added to the training set to continue training, thereby further improving TP(Kowsari, K., Brown, D. E., 2017). Overall, CNN can capture local semantic features, but it may destroy word order features by mistake. While RNN can retain word order features. So the better choice is to merge these two models (Girshick, R., 2015).

Model	**FastText crawl 300d 2M**	**Glove.twitter.27B**
TextCNN	0.9837	No data
KmaxTextCNN	0.9849	0.9832
RCNN	0.9847	0.9835
RNN	0.9858	0.9843

Table 2. The classification results of classical NN models scored by AUC

After emphatically investigating the performance of four NN (neural network) models under the same data set, the results are sorted into the following table according to the existing classification results. The data in table 2 is scored by AUC, which will be described in detail in the Methodology part. The closer the AUC value is to 1, the better the model classification performance. It can be seen from table 2 that KmaxTextCNN and RNN have the best effect among them, so this project considers combining the two models for training.

2.2 Cyberbullying Content-blocker Browser Extension

Although there is currently a lack of research on browser plug-ins that block Internet bullying content in the academic area, there are many browser plug-ins that block advertising links and malicious content. This project modifies the function based on the advertising content blocker so that the original function used to block URLs existing in the elements of webpage Html files could be used to block malicious comments. Most ad blockers have their filtering lists and matching rules. The initialized ad blockers mainly maintain their own blacklist library, and then the number of ad links in the blacklist can be expanded by manually adding ad URLs by the user(C. E. Wills & D. C. Uzunoglu, 2016). This project will also imitate the operating mechanism of the advertising filter, and first put some obvious insulting, highly offensive words into the blacklist to block. Then use the trained machine learning model to judge the obscure comments that may be cyberbullying.

3 Background Theories and Methodology

3.1 Background Theories of NN models

3.1.1 TextCNN Model

Experiments on the visual brain of cats aided in the development of convolutional neural networks (Kim & Jeong, 2019). It stimulates neurons using tiny matrices with weights (ie convolution kernels) to extract all implicit semantic information in the original data, then transfers the data in the concrete receptive field to the abstract feature surface using the backpropagation method. Consider the effects on the weights of connected neurons. During this process, the model will increasingly focus on specific features, then make matching judgments based on the likelihood of the presence of different types of characteristics, and produce a list of the percentages of similarity to each category.

In 2014, based on the widely used CNN in the field of computer vision, Yoon Kim proposed a text classification model TextCNN, which created an interesting connection between natural language processing and computer vision(Kim, Y., 2014). Compared with the traditional CNN network used to process images, TextCNN has simplified the number and operation of the convolutional layer and pooling layer and has made some changes in the input layer.

3.1.1.1 Convolutional neural network hierarchy

i. Input layer

The main task of the data input layer is to preprocess the original image or text data. The first step is to label all the words in the data set, build a vocabulary, which means digitize the natural language, and using the word as the smallest indivisible granularity. For example, divide the sentence "The moonlight is so beautiful tonight" into the following figure and labels each word.

word	**Vector**				
The	0	0	0	0	1
moonlight	0	0	0	1	0
is	0	0	1	0	0
so	0	1	0	0	0
beautiful	1	0	0	0	0
tonight	0	0	0	1	1

Table 3. Example of word vector matrix

ii. Convolutional layer

The convolutional layer is the biggest advantage of the CNN model and the innovation that is most different from other algorithms. It can be said that it is the core of the entire model. It is actually a combination of multiple convolution kernels with different weights and the output feature surface of the image or text sample after the convolution operation.

Each convolution kernel is equivalent to a filter used to filter a certain feature, which is a matrix of weights, which can be known from the discrete sequence convolution formula (1):

$$output[n] = x[n] * w[n] = \sum_{i=1}^{k} x(i) \cdot w(n-i) \qquad (1)$$

Among them, w[n] is the weight matrix of the convolution kernel serialization, x[n] is the input serialized word vector matrix, and output[n] is the output value after the convolution of x[n] and w[n], That is, the value of the corresponding position (the region in the word vector matrix consistent with the size of the convolution kernel) is multiplied by the convolution kernel after being flipped and then summed.

It should be noted that the focus of different convolution kernels is different, for example, some convolution kernels pay attention to the position of the word "tonight" in the input sentence; some n-dimensional convolution kernels focus on the relationship between specific words and n-words before and after them when they appear.

Besides, it should be mentioned that CNN has some differences in natural language processing and computer vision. On the one hand, unlike the thickness of the convolution kernel is three when processing pictures, that is, RGB channels, the convolution kernel can be set to multi-channel when processing text to prevent problems such as over-fitting, but the most widely used is single-channel. Another point is that unlike when convolving with the pixel matrix, the convolution kernel slides from left to right and from top to bottom in the form of a square to traverse the entire image. The convolution operation on the word vector matrix will only slide from top to bottom and meanwhile, the width of the convolution kernel is consistent with the maximum width of the word vector.

In addition to the convolution kernel, another important term in the convolution layer is the feature map, which is actually the output result of the convolution kernel and the word vector matrix after the convolution operation. Its number is equal to the number of convolution kernels, and its size is related to the size of the word vector matrix sample (M×N×H, M is the number of rows of the matrix, N is the number of columns of the matrix, H is the number of channels of the matrix, generally H=1 in text classification), the size of the convolution kernel (m×N×H, m is the number of rows of the kernel matrix) and the vertical sliding step (j), see formula (2) as follows:

$$\begin{cases} FeatureMap_l = \frac{M-m}{j} + 1 \\ FeatureMap_w = 1 \end{cases} \qquad (2)$$

In formula 2, $FeatureMap_l$ is the length of the feature surface, that is, the number of rows, and $FeatureMap_w$ is the width of the feature surface, that is, the number of columns. The feature map is actually a numerical matrix and each element in the matrix is equivalent to a neuron. The input neuron is the result of convolution output[n] and the output of the neuron is output[n] based on the input. The setting bn is shown in formula (3):

$$Output[n] = output[n] + b_n \quad (3)$$

The effect of adding the offset is to enable Output[n] to cover the entire n-Output plane instead of being trapped by the origin so that the output function can correctly divide the data.

Following the convolutional layer is usually the ReLU excitation layer, which is responsible for the nonlinear transformation of the data results output by the convolutional layer. In fact, the excitation layer can also use other functions, but after many researchers' experiments, it is found that the ReLU function works best, so most of the existing models choose to use the ReLU function to activate neurons. The reason for introducing the activation function is that the previous series of operations are based on linearity, but the linear model cannot solve all the problems, so by using the activation function to non-linearly map the input of the neuron to the output end, the CNN model is no longer simple calculations, but also has the ability to understand compound features. The characteristic of the ReLU function is to set all inputs less than or equal to 0 to 0, and inputs greater than 0 remain unchanged as shown in formula (4):

$$f(x) = \begin{cases} 0, & x \leq 0 \\ x, & x > 0 \end{cases} \quad (4)$$

The advantage is that most of the values in the matrix will be set to 0 after ReLU, which means that the matrix is sparsed so as to speed up the training of the CNN model, and at the same time reduce the computational complexity. Since the derivative of f(x) is always equal to 1 when x>0, it will not cause the explosion or disappearance of the gradient (when the CNN model uses the result to update the weights layer by layer, it will be caused by the chain derivative: The gradient explosion occurs when the derivative of the activation function>1, and the gradient disappears when the derivative of the activation function<1).

iii. Pooling layer

The pooling layer is connected to the convolutional layer to compress and purify the data by taking the maximum or average value in the area while ensuring that the number of extracted features remains unchanged, reducing overfitting. Similar to the processing method of the convolutional layer, the pooling layer also has a pooling kernel, but it is more like a window. The numerical matrix of the characteristic surface selected by this window will follow certain rules, such as finding the maximum value of the data in the box or the data in the box The average value is used to filter out a value representing the data characteristics of this box. Then slide this window to shrink the entire eigenface value matrix.

TextCNN simplifies the operation of the pooling layer, using the maximum value in the feature map to represent the most important feature information in the special feature map so that the dimension of each feature map will be reduced to 1, which means it becomes a specific value. Therefore, the method of hiding important information such as expanding the sentence is ineffective and achieving a high degree of extraction of the information in the sentence.

iv. Fully-connected layer

The fully connected layer uses the matrix-vector multiplication as formula (5) to combine the large complex features scattered after the local combination into complete features in the global scope:

$$\begin{cases} Y=XW \\ y_{ij}=\sum_{k=1}^{s} x_{ik}y_{kj}=x_{i1}y_{1j}+x_{i2}y_{2j}+\ldots+x_{is}y_{sj} \end{cases} \quad (5)$$

Explanation of formula 5: where i=1,2,...m; j=1,2,...n, Y=(y_{ij}) is an m×n output matrix, X=(x_{ij}) is an m×s input matrix, which is also the output matrix after convolution and pooling. W=(w_{ij}) is an s×n weight matrix.

After the weight matrix is trained, the features associated with a certain category will be changed to higher weights, and then after weighted summation, if a feature is found to exist in this word vector matrix sample, then this feature will be Activate, that is, the corresponding output value of this feature will be very large. Matching these features with the possible labels according to the probability size can complete the task of classification.

3.1.1.2 Backpropagation algorithm

The backpropagation algorithm helps the model to know how to set the value of the filter of the convolution kernel in order to filter out the features associated with its category, and how does the fully connected layer know how to assign high weights so that some features are seen. The method for the model to adjust its filter value (ie weight) is a back-propagation training process, which is divided into three steps: forward propagation, backward propagation, and weight update. Before the CNN model starts to operate, randomly set weights to filter features. At this time, it enters the forward transfer process, uses a random convolution kernel to filter the image features and then transfers it to the entire network model, and finally gets an incorrect classification result. Then the error function (also called loss function) is shown in formula (6):

$$Error=\frac{1}{2n}\sum_{x}\|t(x)-r^{L}(x)\|^{2} \quad (6)$$

In formula 6, n is the number of all training samples in a training batch, t(x) is the expected output of the x-th input sample, r(x) is the actual output after the activation function, and L is the number of layers in this network. Then, using the chain derivation rule, on each neuron (that is, the convolution kernel), use the error function to obtain the partial derivation of the weight and bias as shown in formula (7) to find the direction of adjustment:

$$\begin{cases} \frac{\partial Error}{\partial b_j^l} = \delta_j^l \\ \frac{\partial Error}{\partial w_{jk}^l} = r_k^{l-1}\delta_j^l \end{cases} \quad (7)$$

In formula 7, l and j are the l-th layer and j-th neuron in the network model; k is the neuron in the l-1th layer connected to j, which is used to distinguish the weights between different neurons; δ is the partial derivative of the loss function to Output (that is, the element before the input activation function). Then, the formula for adjusting the weight w and the bias value b using the gradient descent method is shown in (8):

$$\begin{cases} w_{jk(new)}^l = w_{jk(old)}^l - \eta \times \frac{\partial Error}{\partial w_{jk(old)}^l} \\ b_{jk(new)}^l = b_{jk(old)}^l - \eta \times \frac{\partial Error}{\partial b_{j(old)}^l} \end{cases} \quad (8)$$

In formula 8, η is the learning rate, which mainly determines the speed of the gradient descent. If the learning rate is too small, the convergence will be too slow. While if the learning rate is too large, it may not find the most suitable weights and oscillate around it.

3.1.2 RNN Model

When the human brain reads and understands an article, if a new word appears, it often guesses the meaning of the word through the knowledge accumulated in the past and the context-related information it has mastered. Different from CNN's non-relevant feature extraction of text, the Recurrent Neural Networks(RNN) increases the connection between keywords and context by adding a cyclic module to the network structure, so that the entire neural network has a similar memory function to information(Mandic & Chambers, 2001).

3.1.2.1 Chain structure of the recurrent neural network

RNN and TextCNN have the same input layer for processing text, mapping words and numbers in the vocabulary, and a fully connected layer for summarizing feature results for probabilistic judgment, which will not be repeated here. The most distinctive structure in RNN is the loop operation that can be expanded into a chain structure as shown in the figure below.

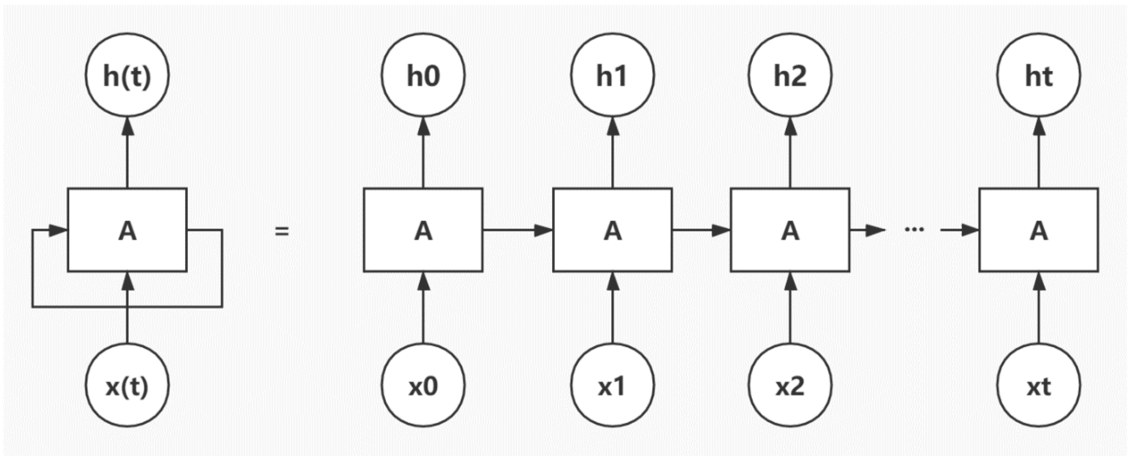


Figure 1. Example of chain structure of the recurrent neural network

In Figure 1 above, module A of the neural network is inputted with a certain input x(t) and outputs a value h(t). At the same time, the current h(t) is passed to the next step through the loop. In the whole cycle, it can be seen that the same neural network is assigned multiple times, and each neural network module passes the processed message to the next one.

3.1.2.2 Back-propagation through time

Similar to the backward propagation algorithm used in the weight update of the CNN model, when training the RNN model, since the processed input is a time series, the weight update also uses a time-based backward propagation algorithm called Back-propagation through time(BPTT).

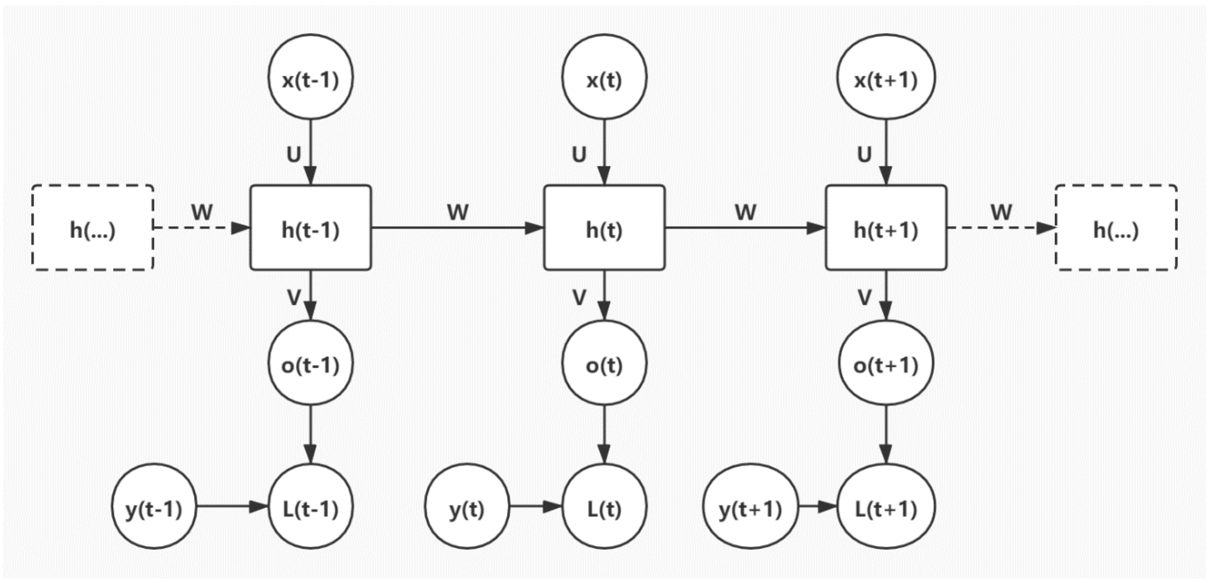


Figure 2. The detailed chain structure of the RNN model

In Figure 2, x(t) is the input at time t, h(t) is the function of the hidden layer unit at time t, o is the output of the function h at time t, L is the loss function which is equal to the Error in formula 6, and y(t) is t The label of the training set at all times. U, V, and W are the weight of input x(t) at time t, the weight of h(t-1) at time t-1, and the weight of h(t) at time t.

$$\begin{cases} h(t) = \phi(U \cdot \mathrm{x(t)} + \mathrm{W} \cdot \mathrm{h(t-1)} + \mathrm{b}) \\ o(t) = V \cdot h(t) + c \\ y'(t) = \sigma(o(t)) \end{cases} \tag{9}$$

In formula 9, ϕ is usually the activation function tanh or function ReLU, σ is usually the activation function softmax, and b and c are the offsets. When the weights of the RNN model(U, V, W) are updated during the training process, the time-based changes are made based on formulas 6, 7, and 8, as shown in formula 10.

$$\begin{cases} L = \sum_{t=1}^{n} L(t) \\ \frac{\partial L}{\partial V} = \sum_{t=1}^{n} \frac{\partial L(t)}{\partial o(t)} \cdot \frac{\partial o(t)}{\partial V} \\ \frac{\partial L(t)}{\partial W} = \sum_{k=0}^{t} \frac{\partial L(t)}{\partial o(t)} \cdot \frac{\partial o(t)}{\partial h(t)} \cdot \left(\prod_{j=k+1}^{t} \frac{\partial h(j)}{\partial h(j-1)}\right) \cdot \frac{\partial h(k)}{\partial W} \\ \frac{\partial L(t)}{\partial U} = \sum_{k=0}^{t} \frac{\partial L(t)}{\partial o(t)} \cdot \frac{\partial o(t)}{\partial h(t)} \cdot \left(\prod_{j=k+1}^{t} \frac{\partial h(j)}{\partial h(j-1)}\right) \cdot \frac{\partial h(k)}{\partial U} \end{cases} \quad (10)$$

3.1.3 Main problems and solutions of NN models

3.1.3.1 Gradient diffusion

The gradient diffusion problem is that when the backpropagation algorithm transfers the gradient backward, as the propagation depth increases, that is, the number of layers increases, the amplitude of the gradient decreases sharply, which leads to the problem that the weights of shallow neurons are updated very slowly to be unable to learn effectively.

The main solution to this problem is structural reforming such as residual network or batch normalization processing. Since the distribution of data will lead to the different inverse of gradient descent, the normalization process is to normalize the corresponding activation through batch gradient descent every time the gradient falls randomly, so that the mean value of the result is 0, and the variance is 1. So as to maintain the stability of the network. For the problem that RNN can only maintain short-term memory due to the disappearance of gradients, long short-term memory(LSTM) networks have emerged as an improvement method. It realizes the preservation and copying of long-term memory by adding a complex gate control mechanism, thereby increasing the possibility of correct prediction of missing words in the text.

3.1.3.2 Degenerate

Degradation problem refers to that by increasing the number of network layers to increase the network accuracy to a certain level, it will reach saturation, and the learning effect will be far inferior to the network with a moderate number of layers. In other words, in each convolutional layer or fully connected layer, only a few neurons that meet the constraints will change their weight value matrix according to different inputs and feedback results, and most hidden neurons are due to the redundant constraints of the deep network (Lower degrees of freedom) make multiple feasible weight values smaller and smaller until the weight matrix drops to 0, so different inputs do not represent different outputs. After the degradation problem occurs, the result will be reflected in the increase of training and testing errors.

The main solution is the deep residual network, that is, adding some control gates in the network model so that the input can choose to take shortcuts and jump several redundant convolutional pooling layers, reducing the number of activation functions that reduce the number of nonlinear transformations The irreversible loss of information can solve the degradation problem.

3.1.3.3 Overfitting

The over-fitting phenomenon occurs when there is noise in the training data, or too many parameters in the network model cause the computational complexity of the network to increase, or the training data is insufficient. When any of the above conditions are met, the

NN models will have good classification performance when facing the trained data set, but when facing the untrained test set, the classification performance will deteriorate, which is overfitting. This is related to the fact that the focus of the model learning is biased towards noise features or the common features of too few samples, rather than general regular trends.

The main solutions to this problem are Dropout and network regularization. Dropout is to temporarily discard neurons in the fully connected layer from the network according to a certain probability during the forward and backward propagation process and to ensure the same number of input and output neurons. This action can avoid the model's dependence on certain samples. Network regularization is to restrict the size of the weight so that it can not fit random noise disorderly.

3.2 Background Theories of Technical Tools

3.2.1 Browser Extension

This project chooses to use the Chrome extension as an approach to achieve the user-friendly cyberbullying firewall because Chrome has a higher market share, more users, easier development, and a wider range of application scenarios.

As for the software packages, Tampermonkey will be used, because it is suitable and convenient to manage scripts. It is a free browser extension and the most popular user script manager, so in this project, it was decided to make changes to some extensions similar to ad-blocking in Tampermonkey to implement my functions("Tampermonkey", 2021).

3.2.2 Programming Language

For the front end, the programming languages will involve Javascript and CSS, while for the back end(also for the API) which is the part of building the model and training it, python would be used. It is because the Python language has the characteristics of orienting objects and having powerful function libraries. This paper mainly uses the 'NumPy library' that supports a large number of dimensional arrays and matrix operations (such as the function 'np.mean' to calculate the average value, and the standard deviation calculation function 'np.std', etc.), the 'pandas library' for basic data serialization and deserialization, and the 'random library' for generating pseudo-random numbers.

Besides, Python also has many mature machine learning-related function libraries and frames like TensorFlow, PyTorch, etc. This project mainly uses the multi-layer structure Tensorflow framework, which is characterized by containing a large number of function libraries required by deep learning algorithms and can support GPU and TPU high-performance numerical calculations. The TensorFlow functions (the following tf is the abbreviation of TensorFlow) mainly used in this paper are:

i. The 'tf.nn' library provides functions related to the calculation of neural networks. This paper mainly uses the '.conv2d' function that can be used for convolution operations; the '.max_pool' maximum pooling function that can be used for pooling operations; the '.dropout' deactivation function that can be used to randomly lose neurons; the '.relu' activation function that is used to make linear input have nonlinear output characteristics; the '.softmax' function that can be used to turn the classification result into a probability output.

ii. Keras, an open-source artificial neural network library mainly developed by Francois Chollet, contains many APIs for linking neural network models and encapsulates many mature deep learning algorithms. This project mainly uses: the 'keras.preprocessing.text.Tokenizer' class used to preprocess text sequences and generate text vocabulary vectors; the 'keras.preprocessing.sequence.pad_sequences' function used to enhance the data set and perform zero paddings; 'Model.layer' used to load objects in the neural network hierarchy; interface 'keras.models.Model' used to map the input and output of the model; 'model.complie' used to train the model and 'model.fit' used for the model's transfer learning; as well as the 'model.evaluate' used for the overall evaluation of the model classification performance and 'model.predict' functions used to predict the results of specific data in the test data set.

3.3 Metrics

Similar to an intrusion prevention system, a cyberbully can be considered an intruder to a good and friendly network atmosphere, and a bad comment can be considered an intrusion that causes substantial harm to the victim. Therefore, the method used by the intrusion detection system to determine the real intrusion behavior can be used for reference by the automatic detection network bullying model. In the process of model training, there are two key factors that determine the accuracy of the final prediction results and other properties of the model itself. One is the actual label of each comment in the data set, and the other is the predicted label by the model.

Before elaborating on the criteria for evaluating the performance of models for machine learning, the confusion matrix should be introduced as a basic methodology. In the diagram below, 1 means positive and 0 means negative, and if the predicted classification results match the actual, it will get true(11 or 00), otherwise, it will get false(10 or 01). Besides, the positive or negative of the cross point is based on the predicted result which means if the predicted result is 1, then the cross point will be positive no matter what the actual result is, vice versa.

During model training		**Actual**	
		1	0
Predict	1	True Positive(TP)	False Positive(FP)
	0	False Negative(FN)	True Negative(TN)

Table 4. Confusion matrix

Furthermore, there are four standards related to the content shown in Table 4 that can be used as a measure of the quality of the model, the accuracy rate, the recall rate, the precision rate, and the F1 score, which will be described in detail below. In addition, by using ROC (Receiver Operating Characteristic Curve), it is possible to know how the relationship between the recall rate and the accuracy rate will change when the threshold value recognized as a positive example in the model changes. At the same time, the use of cross-validation can measure whether the model has problems such as selection bias or overfitting when facing a new data set that has not been trained.

3.3.1 Accuracy

The definition of accuracy is the sum of true positives cases and true negative cases divided by the sum of all samples, which means that all correctly classified samples account for the probability of the total samples in the data set.

$$Accuracy = \frac{True\ Positives + True\ Negatives}{True\ Positives + False\ Positives + True\ Negatives + False\ Negatives} \quad (11)$$

However, a model may have a high accuracy rate but cannot be a comprehensive measure of the true classification performance of the model, especially on unbalanced classification problems. For example, the bad reviews on the Internet are regarded as negative cases, and the rest of the normal reviews are regarded as positive cases. In real life, such negative cases are very lethal but account for a very small part of all reviews. When the negative cases are much smaller than the positive cases, the number of harmless comments successfully identified far exceeds the number of harmful comments successfully identified. However, the numerator of the accuracy rate is the sum of the two, so even if the result of the accuracy rate is very close to 1. , Nor can it prove that the model has a very effective ability to distinguish bad reviews. Therefore, we should focus more on the recall rate and accuracy mentioned below to maximize the model's ability to find bad reviews in the data set.

3.3.2 Recall rate and Precision rate

The definition of recall rate is the true positives divided by the sum of the true positives and the false negatives, that is, the probability that a sample that is actually a positive case is predicted to be correct. It can be used to measure the model's ability to find a small number of positive data samples in the unbalanced data set.

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives} \quad (12)$$

The definition of precision rate is the true positives divided by the sum of the true positives and the false positives, which means the probability that the sample predicted to be a positive case is actually a positive case. It can be used to measure the correct rate of a small number of positive data samples found by the model in an unbalanced data set

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives} \quad (13)$$

The relationship between recall rate and the precision rate is inversely proportional, that is, when true positives and true negatives remain unchanged, the recall rate will decrease as the precision rate increases, and vice versa.

3.3.3 F1 score

Under certain conditions, the model may be anticipated to have a high recall or high accuracy, even if this means sacrificing another indication to some amount. If the goal is to find all the bad reviews in a webpage, then it can be tolerated that the precision of the model is not high, that is, some normal comments are misjudged as bad reviews, but after feeding the webpage comments into the model several times, all of the bad reviews can be recognized. If the aim is to minimize the number of regular comments that are misinterpreted as negative evaluations, the precision must be improved, and the recall rate could be lower. The F1 score is derived by combining precision and recall and reconciling the average value of the two to find the best combination.

$$F_1 = 2 \times \frac{Recall \times Precision}{Recall + Precision} \quad (14)$$

The reason for using harmonic mean instead of simple arithmetic average in the F1 score calculation is that harmonic mean is more sensitive to extreme situations. For example, a classifier A with a precision rate of 1.0 but a recall rate of 0, if the purely arithmetic average is used, the result will be 0.5. While the arithmetic average result of classifier B with a precision rate of 0.5 and a recall rate of 0.5 is also 0.5, so it is impossible to distinguish between this two. However, for this extreme case, using the F1 score calculation, the result of classifier A will be 0, and the result of classifier B will be 0.5 so that the extreme classifier A could be clearly distinguished.

3.3.4 ROC

In addition to the above indicators, there is a more intuitive curve used to show the performance of the classification model, which is ROC (Receiver Operating Characteristic curve). When the classification model classifies a sample, there will be a threshold. If the score of the sample exceeds this threshold, it will be classified as a positive case, and if the score is below this threshold, it will be classified as a negative case. The ROC curve shows how the relationship between recall and accuracy will change as this threshold changes.

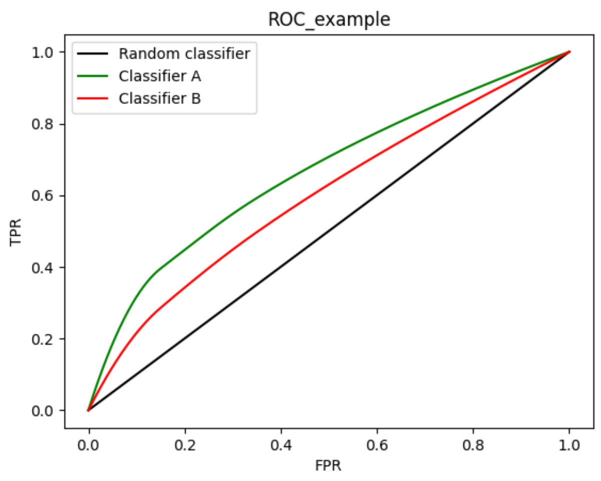


Figure 3. ROC curves of three different classifiers

The Y-axis of the ROC curve is the Recall rate (TPR), and the X-axis is the False Positive rate (FPR). FPR means the probability that a negative case is misjudged as a positive case.

$$False\ Positive\ rate = \frac{False\ Positives}{False\ Positives + True\ Negatives} \quad (15)$$

Assuming that the score of the sample is between 0 and 1 when the threshold is 1, no sample will be identified as a positive case, that is, there are no true positives and false positives. The ROC curve is located at the coordinate axis origin at the bottom left of figure 3. When the threshold decreases from 1 to 0, more samples will be identified as positive cases, that is, both true positives and false positives will increase. Finally, when the threshold is 0, all samples will be recognized as positive, and thus, the ROC curve is located at the upper right of figure 3.

3.3.5 AUC

In order to quantify the ROC curve of the model, a value can be obtained by calculating the area under the ROC curve, which is also called AUC (Area under the curve). The value of AUC is between 0 and 1, and the closer it is to 1, the better the classification performance of this model. As shown in Figure 3, the AUC value of Classifier A is greater than B which means that A has better classification performance.

4 Design and Implementation

As mentioned in the introduction part, this project aims to build an automatic text firewall on a variety of online social platforms like Youtube, Twitter, Facebook, Ins, etc. but that needs to be connected to these platforms one-to-one, so the workload is too large and inconvenient for users. Thus, deploying this firewall on the browser, as a browser extension could be a better choice. When users read comments on social media web pages through their browsers, any hate, aggression, insults, and toxicity speech to anyone would be replaced by some warm and cute kaomoji, so that the victim could be protected unconsciously, because they will only know that there are bad comments, but they are not sure whether it is directed at themselves or others. This will not only protect the victims but also reduce the probability of users being instigated by malicious speech and becoming cyberbullying participants.

This project plans to divide tasks into two parts, one is to choose several existing automatic detection algorithms with good performance, and stacking or blending them to get better results. The second part is to complete the development of a browser extension that replaces the detected cyberbullying text with another delightful kaomoji dataset randomly(or users can customize the style of replacement pictures). To connect these two parts, there should be a communication model which could also be called application program interface (API) to transfer the Html data from online social media and input it into a trained neural network model. Then, get the output of classification results from this trained model and pass it to the browser extension. So that the browser extension could complete replacing operation according to the classification results.

4.1 Data Acquisition

4.1.1 Pre-trained Embedding Data Set Analysis

This project uses *FastText crawl 300d* data set for pre-trained embeddings, that is, the behavior of pre-mapping common English words into vocabulary vectors at the word level to facilitate subsequent formal training. This English word vector dataset is released by Facebook in 2018(https://www.kaggle.com/yekenot/fasttext-crawl-300d-2m), and 300d means that the dimension of the word vector is 300 dimensions. The format of the data set is .vec, the size is 4.21GB, and it contains 2 million words (shown in the first row), sorted in descending order of frequency of use. Starting from the second line, each line is in the form of a word followed by a vector, and each value in the vector is separated by a space.

4.1.2 Data Set Analysis

This project chooses *Kaggle*(https://www.kaggle.com/surekharamireddy/malignant-comment-classification) as training as well as the test set for the machine learning model. The data format is CSV and it was collected from 2004 to 2015. It contains 159 thousand samples of the training set and nearly 153 thousand samples of the test set. And the fields of it are "id, comments, Malignant, highly malignant, rude, Threat, Abuse and Loathe", while all of these fields will be blocked in this project, so they would only get the label 1 which is equal to yes and other benign words will be labeled zero which is equal to no.

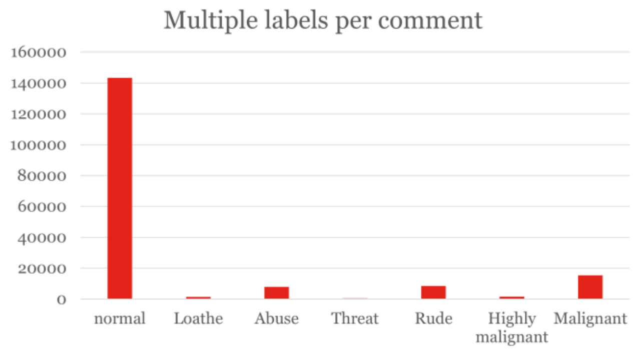


Figure 4.a. Multiple labels per comment in the Kaggle malignant-comment-classification dataset

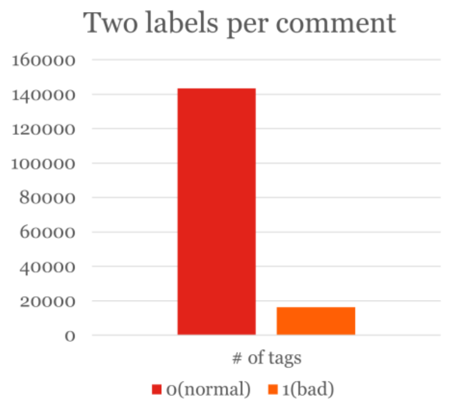


Figure 4.b. Two labels per comment after processing

4.1.3 Data Preprocessing

This project performed the following preprocessing operations on the data set of the input model:

i. Remove special characters and punctuation like '$,./{}#%@' etc.
ii. Expanding the abbreviation, like changing can't to cannot, what's to what is
iii. Replace emoji with words such as :), :p to smile, and :(to sad
iv. Use regular expression processing, replace words like fuckkkk / fucccck with fuck
v. Corrected some wrong words with TextBlob
vi. Remove extra spaces between letters in a word, such as 'ha te' to 'hate'

4.2 Software Design

The functional requirement of the software design of this project is to realize the automatic recognition and shielding of cyberbullying comments. In order to realize this function, the whole system can be divided into three modules.

The first is the RNN-CNN model, which is also the most important part of the function of automatically detecting negative review texts. It will be deployed on the backend server after training to complete the classification of each comment on the web page, namely Malicious comments were identified. The second module is a browser plug-in, which is used to replace the original malicious comments with harmless kaomoji after receiving the classification results of the comments from the back-end model. At the same time, the plug-in deployed on the front-end browser will also complete real-time data collection and provide convenience for users to open or close this function. The last module is a middleware script that connects the first two modules, which can also be called an API. It is used to process the input and output information interaction between the front and back ends. Its meaning is to separate the read and write operations and increase the security of the entire system. Besides, it also provides scalability for the addition of new functions in the future.

As shown in figure 5 below, the information interaction between modules and the tasks that modules need to do can be divided into two processes, the input process, and output process, on the timeline.

i. Input process

a. What the browser extension has to do is:

① Monitor the webpage to keep the plug-in always running.

② Locate the position of each comment on the Html page of the social media platform on the browser by marking the id.

③ Traverse the Html component, extract the text from it and pass it to the middleware script.

b. What the API does:

④ Pass the text and text id given by the plug-in to the RNN-CNN model.

⑤ Pre-train the model with data sets such as crawl-300d-2M, glove.840B.300d, etc. in advance.

⑥ Use the Kaggle data set to train the model so that the model has the ability to judge whether a sentence is bad or normal.

c. What the RNN-CNN model does:

⑦ Make a judgment on the text passed by the script, and give a result of 0 or 1.

ii. output process

a. What the RNN-CNN model does:

⑧ Pass the judgment result 0 or 1 of each sentence in the text to the script.

b. What the script does:

⑨ Modify the text string according to the result 0 or 1, and modify the string with the result of 1 (that is, with bad reviews) to emoji similar to ‘o(*￣▽￣*)ブ’.

c. What the plug-in should do:

⑩ Replace the text on the original Html with the new modified text submitted by the API (to be located according to the id).

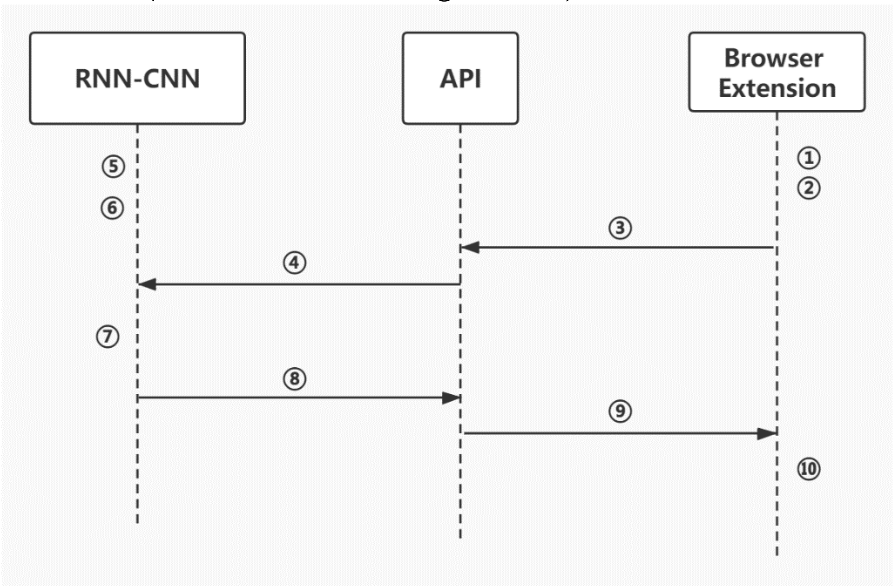


Figure 5. Flow chart of module information interaction and tasks

4.2.1 RNN-CNN Classification Model Design

According to the background theories in section 3.1, the network hierarchical structure design diagram of RNN-CNN is shown below in Figure 6.

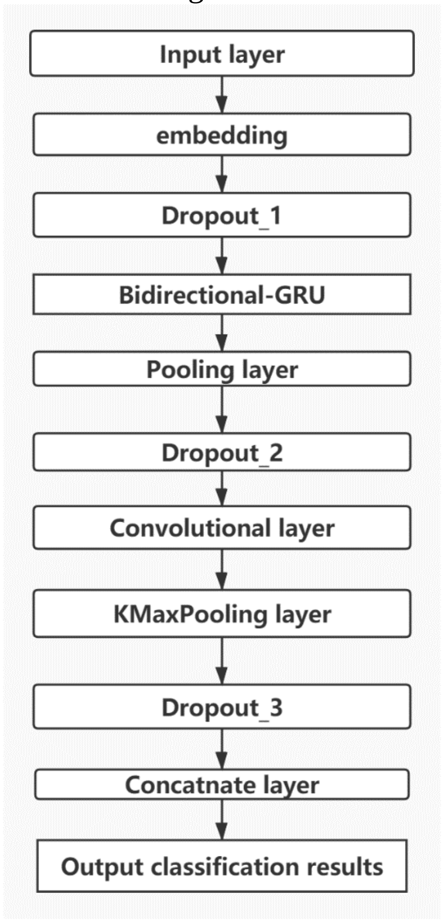


Figure 6. Network hierarchical structure design diagram of RNN-CNN model

The design of the network structure combines RNN and CNN models. After data pre-processing, there would be a layer containing Bidirectional-GRU (bi-GRU) cells. They can broadly capture and save the information in the pre-text and post-text of the keyword which is also the reason for choosing the bidirectional RNN. After the RNN, the data enters the convolution layer, and the feature information is extracted through the convolution kernel and the Kmax-pooling layer is used to purify the largest value in the maximum feature information vector value. Finally, the feature information is integrated through the fully concatenate layer, and the softmax function is used to convert it into the possible probability of the predicted category, that is, the classification result. At the same time, the three Dropout layers interspersed in the structure are to prevent overfitting, so that the model can have similar classification performance on the test set and training set.

4.2.2 Cyberbullying Blocker Google Plug-in Design

The browser plug-in Cyberbullying_Blocker has three main functional modules:

i. Monitor the Document Object Model (DOM) tree and call API.
By monitoring the DOM tree of the browser's Html webpage, when the structure changes and the URL of the webpage being browsed are included in Facebook,

Twitter, Ins, Cyberbullying_Blocker will call middleware API and pass the URL as input to it. Then, it will locate those elements that contain comments and capture their content. The specific work content completed by the API will be described in detail in section 4.2.3.

ii. Receive classification results and replace the malicious comment with kaomoji
After receiving the comment classification result of the RNN-CNN model feedback through the API script, for the comment whose predicted label is bad, that is, the classification result is 1, use the getElementById() function to locate the element with the specified id and replace it with kaomoji.

4.2.3 API Design

The middleware API that connects the front-end plug-in and the back-end classification model has three main tasks that need to be completed:

i. When the API is called, it traverses and captures all the comment text on the webpage.
First, it imports the requests_html and BeautifulSoup libraries in the python language and uses the ‘request.get’ function to crawl text information from the input specific URL webpage. Then, according to the incoming target page number address, it will grab the content of the web page, return the response object, use the BeautifulSoup library function to parse the code, and lock the page number to specify the label content. In the process, record the position of the element in the DOM tree where the text information is located according to the tag path.

ii. Call well-trained RNN-CNN model and pass the comments obtained.
After obtaining the comment text, this API will call the pre-saved trained RNN-CNN model, and pass in the comment samples that need to be judged, and save the returned results.

iii. Upload the classification results of comments to the plug-in, Cyberbullying_Blocker.
Lastly, it will use the ‘request.post’ function in the request_html library to submit a post to the Html page, including the previously saved tag path of the comment element and the predicted classification result of this comment.

4.3 Implementation Procedures

The implementation process is mainly divided into three steps, namely training the model, optimizing the model, and combining the trained model with the fully functional browser plug-in.

4.3.1 Model Training

First, importing the processed and purified Kaggle train.csv data set (which means the cleaned train.csv has no problems such as messy characters, spelling errors, and non-English words mixed, etc.) into the model, and then use the public pre-trained word vector table 'FastText crawl 300d' data set as embedding layer into the model. Then use the training set and test set to train and test the model crossly according to the flowchart shown in Figure 8, and use Tensorboard to record the data and draw the chart. In the process of training the model, manually adjusting the learning rate, batch size, and the number of epochs according to the learning situation.

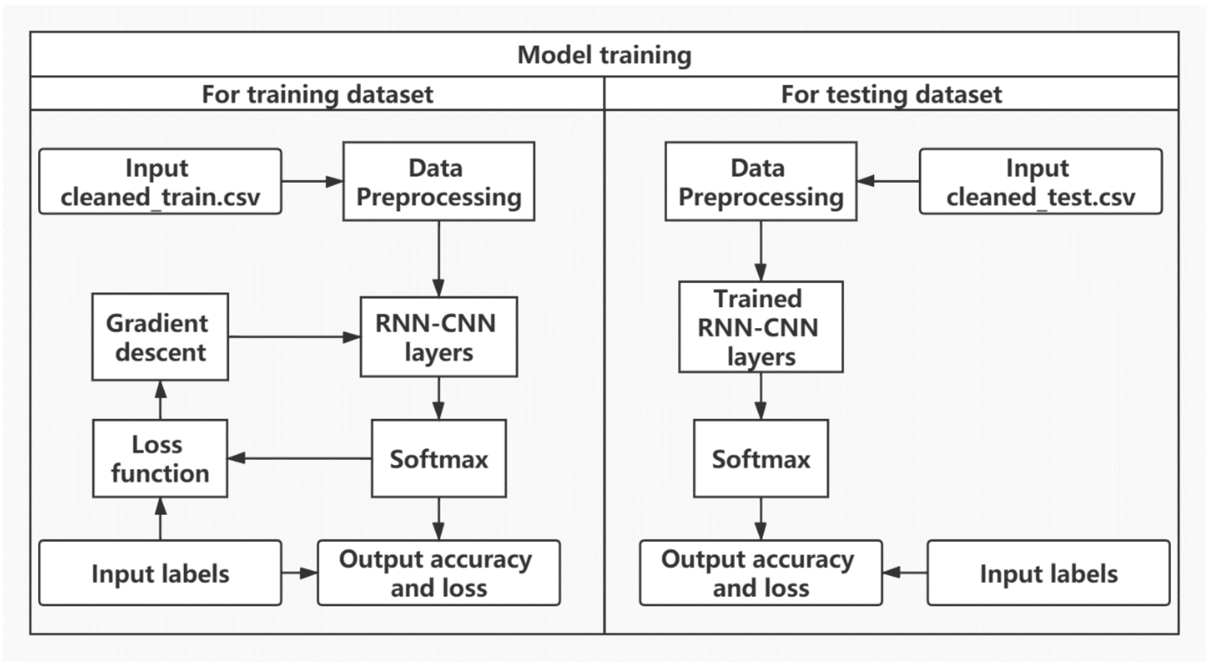


Figure 7. Model training flowchart

In figure 7, the flowchart on the left is for the training set samples, and the flowchart on the right is for the test set samples. The difference between the two is that every time the model is trained with a batch of samples according to the process on the left, the weight matrixes of the bi-GRU layer, convolutional layer, pooling layer, and the fully connected layer will be adjusted by the loss function. While the test set is directly adjusted through the model, and the classification accuracy and prediction loss are obtained by calculation. Each generation of training (epoch) is to first learn all batches of training set samples and then start batch testing, and the two modules work alternately until all the set epochs are completed.

4.3.2 Combine Trained Model with Cyberbullying Blocker Google Plug-in

The ideal mature plug-in deployment is to host the plug-in program to the script manager Tampermonkey, and so that users can choose to turn it on or off every time they open the browser and use it. When Cyberbullying_Blocker is enabled, by monitoring the DOM tree, when the user browses to social platforms such as Facebook, Instagram, Twitter, etc., the comments appearing on the web page will be crawled by the API script and the text data will be passed to the trained RNN- CNN model. Just after the model evaluates the received comments, the outcome will be sent to Cyberbullying_Blocker via the API script. Following that, this plug-in will make no changes to the normal comments while sanitizing the elements that are judged to be malicious comments, i.e. replacing them with kaomoji.

5 RESULTS, ANALYSIS AND EVALUATION

5.1 Performance of Trained RNN-CNN Model

5.1.1 Accuracy and AUC

According to the structural design of the model in Chapter 4, five comparative experiments were carried out by adjusting the epoch number and batch size respectively. The experimental results are shown in the following table.

	Epoch	Train_Acc	Test_Acc	Train_Loss	Test_Loss	Train_AUC
①	55	0.7725	0.7012	0.7963	0.8136	0.6581
②	100	0.9991	0.9294	0.01586	0.2845	0.9883
③	164	0.9947	0.8514	0.2428	0.4739	0.9866
④	240	0.9980	0.8417	0.2368	0.4872	0.9876
⑤	164	0.9994	0.9324	6.4543e-3	0.2939	0.9884

Table 5. Training and test results of the RNN-CNN model on the Kaggle dataset

The constant parameter in experiment ① to ⑤ is the comment classification category class_num is 7 (because a bad_comment class is added, if any of the first 6 items is 1, the seventh item is 1), and the weight_decay is 0.0003 (This parameter is a coefficient used in front of the regular term in the loss function. The regular term reflects the complexity of the model. Weight_decay can adjust the influence of the model complexity on the loss function and its ultimate purpose is to prevent overfitting.) Inactivation probability dropout_rate is 0.5 (because inactivation has only two results: discard or not to discard, so the probability of half-divided discard is set to 50%), the momentum in the optimizer momentum_rate is 0.9 (the increase in momentum can speed up the iterative process and its value range is 0 to 1). The variable parameters in the experiment and the results of the five experiments are shown in the following figures.

Experiment ①: the total_epoch is 55, the training set batch_size is 100, the learning rate is 0.1, and the test set batch_size is 100.

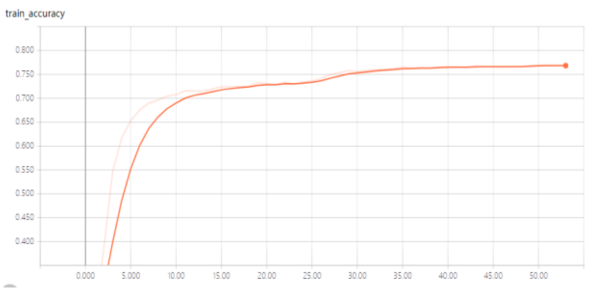


Figure 8.a. The classification accuracy of experiment ① on the training set

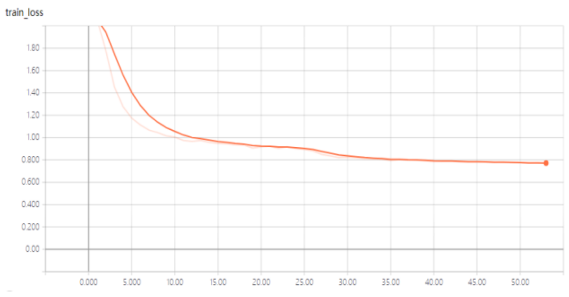


Figure 8.b. The loss of experiment ① on the training set

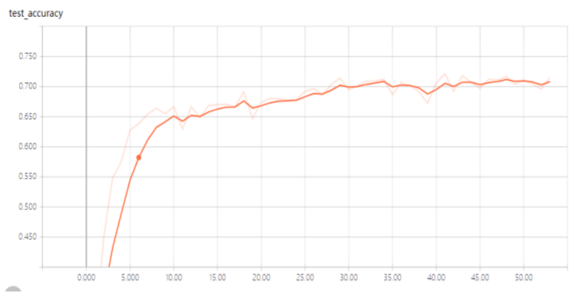


Figure 8.c. The classification accuracy of experiment ① on the test set

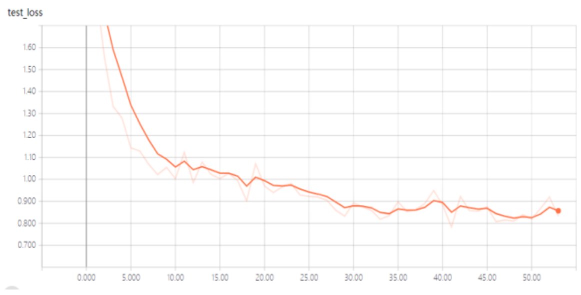


Figure 8.d. The loss of experiment ① on the test set

Experiment ②: total_epoch is 100, batch_size of the training set is 256. When the number of iterations is less than 81, the learning rate is 0.1, and when the number of iterations is greater than 81 and less than 100, the learning rate is 0.01, and the test set batch_size is 1000.

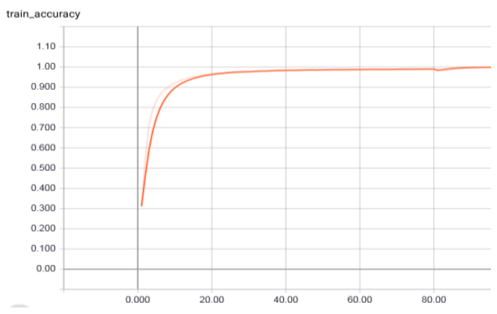


Figure 9.a. The classification accuracy of experiment ② on the training set

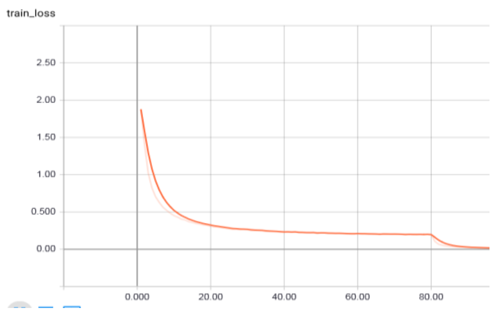


Figure 9.b. The loss of experiment ② on the training set

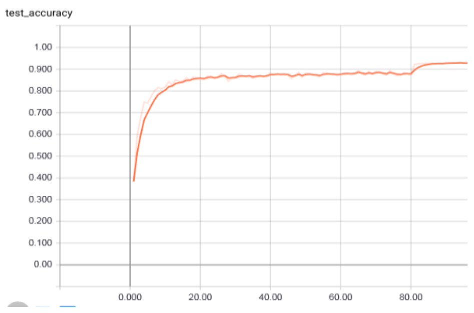


Figure 9.c. The classification accuracy of experiment ② on the test set

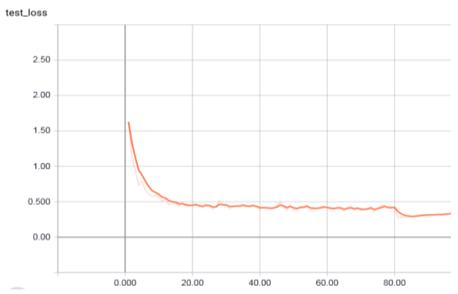


Figure 9.d. The loss of experiment ② on the test set

Experiment ③: total_epoch is 164, batch_size of the training set is 100. When the number of iterations is less than 81, the learning rate is 0.1. When the number of iterations is less than 121 and greater than 81, the learning rate is 0.01. When the number of iterations is less than 164 and greater than 121, the learning rate is 0.001. The batch_size of the test set is 100.

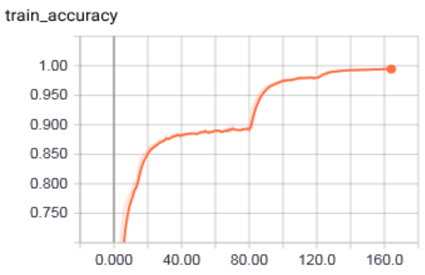


Figure 10.a. The classification accuracy of experiment ③ on the training set

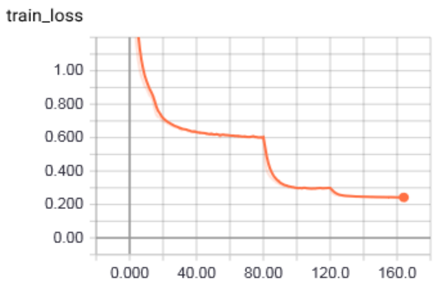


Figure 10.b. The loss of experiment ③ on the training set

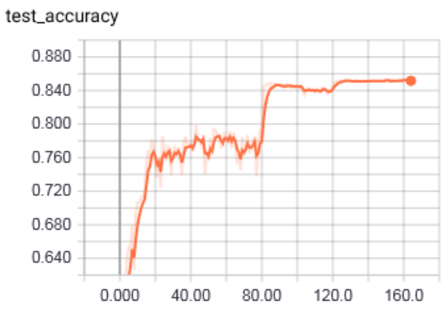


Figure 10.c. The classification accuracy of experiment ③ on the test set

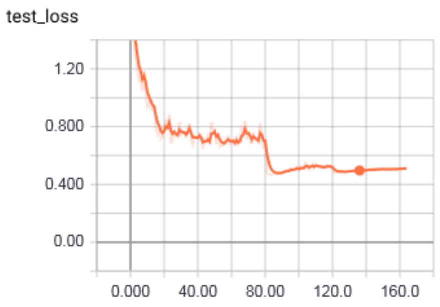


Figure 10.d. The loss of experiment ③ on the test set

Experiment ④: total_epoch is 240, batch_size of the training set is 100. When the number of iterations is less than 136, the learning rate is 0.1. When the number of iterations is less than 185 and greater than 136, the learning rate is 0.01. When the number of iterations is less than 240 and greater than 136, the learning rate is 0.001. The batch_size of the test set is 100.

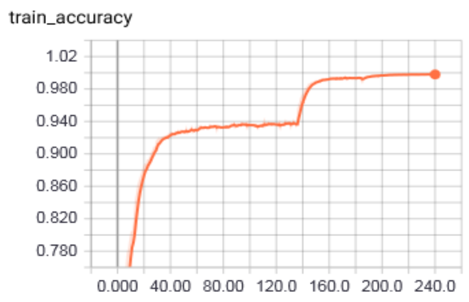


Figure 11.a. The classification accuracy of experiment ④ on the training set

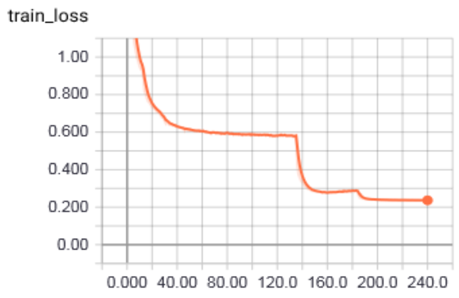


Figure 11.b. The loss of experiment ④ on the training set

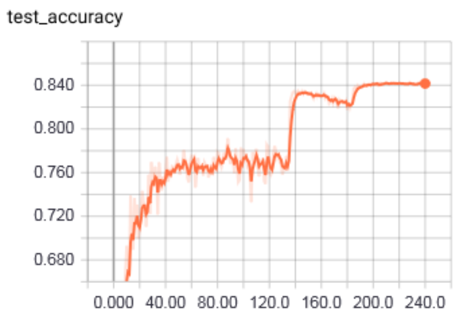


Figure 11.c. The classification accuracy of experiment ④ on the test set

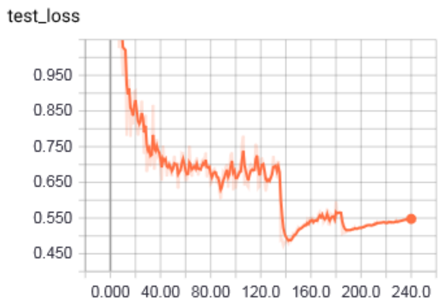


Figure 11.d. The loss of experiment ④ on the test set

Experiment ⑤: total_epoch is 164, batch_size of the training set is 250. When the number of iterations is less than 81, the learning rate is 0.1. When the number of iterations is less than 105 and greater than 81, the learning rate is 0.01. When the number of iterations is less than 164 and greater than 105, the learning rate is 0.001. The batch_size of the test set is 1000.

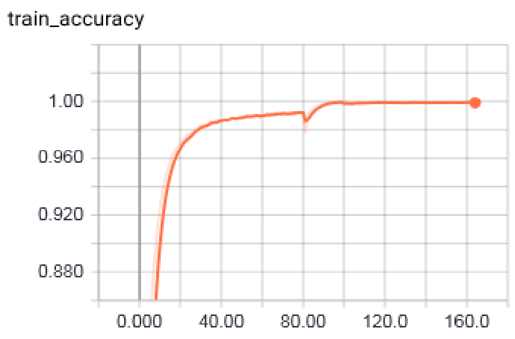


Figure 12.a. The classification accuracy of experiment ⑤ on the training set

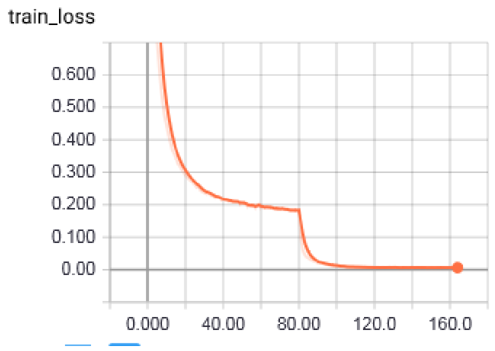


Figure 12.b. The loss of experiment ⑤ on the training set

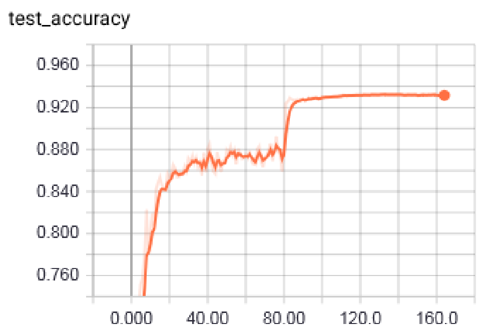


Figure 12.c. The classification accuracy of experiment ⑤ on the test set

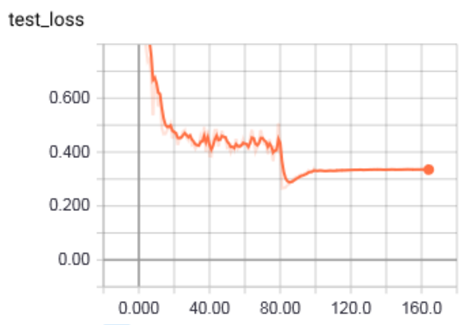


Figure 12.d. The loss of experiment ⑤ on the test set

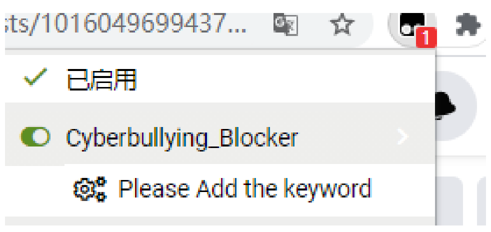
5.1.2 Analysis and Evaluation

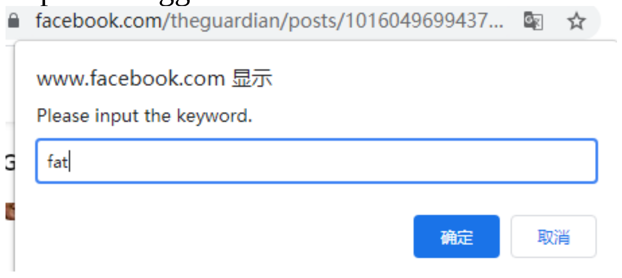
Among the data shown in chapter 5.1.1, comparing ① and ③, the increase in the number of epochs during training will increase the classification accuracy. Comparing ③ and ④, it can be seen that increasing the number of epochs to a certain extent will cause over-fitting, that is, the training situation will get better but the test situation will get worse. Compared with ③ and ⑤, it is found that the increase of batch_size will increase the classification accuracy, but the batch_size cannot be too large due to hardware conditions.

Within a reasonable range, the larger the number of batch image samples, the more accurate the direction of gradient descent and the smaller the oscillation. However, if the number of batch image samples is too large, it may meet the optimal locally and deviate from the actual category from a global perspective. The small number of batch image samples makes the training process more random, so it is difficult to achieve convergence, and only in rare cases, the effect may be better. If the hardware can support or the data set is relatively small, all data samples can be trained at once. The advantage is that the gradient descent direction determined by the full data set can better and faster characterize the entire sample population so that it can drop in the direction of the extremum more accurately.

5.2 Performance of Trained Model that Receives the Webpage Data Transmitted by The Browser Extension in Real-Time

The plug-in in the testing phase of this project is based on one of the online social media, Facebook. The comparison of the browser plug-in when it is turned on and when it is not is shown in Figure 13. However, since real-time comments are not labeled as samples, they cannot be used to evaluate the quality of the model test results. The following section 5.2.1 will show the results of randomly selecting 10 specific tagged comments to test the model.





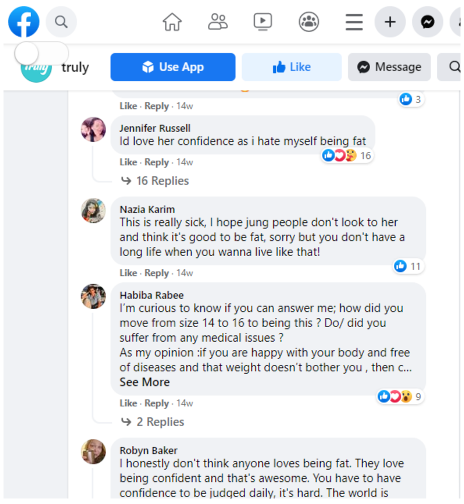
www.facebook.com 显示

Please input the keyword.

fat

Figure 13.a. The pop-up window shows that the Cyberbullying_Blocker plug-in is turned on when the user browses the page.

Figure 13.b. An example of entering a keyword to be blocked



Like · Reply · 14w

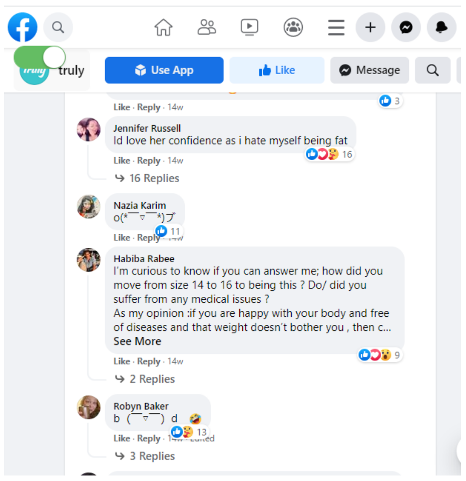
Jennifer Russell
Id love her confidence as i hate myself being fat
Like · Reply · 14w
16 Replies

Nazia Karim
This is really sick, I hope jung people don't look to her and think it's good to be fat, sorry but you don't have a long life when you wanna live like that!
Like · Reply · 14w

Habiba Rabee
I'm curious to know if you can answer me; how did you move from size 14 to 16 to being this ? Do/ did you suffer from any medical issues ?
As my opinion :if you are happy with your body and free of diseases and that weight doesn't bother you , then c...
See More
Like · Reply · 14w
2 Replies

Robyn Baker
I honestly don't think anyone loves being fat. They love being confident and that's awesome. You have to have confidence to be judged daily, it's hard. The world is

Figure 13.c. The webpage when the Cyberbullying_Blocker plug-in is closed



Like · Reply · 14w

Jennifer Russell
Id love her confidence as i hate myself being fat
Like · Reply · 14w
16 Replies

Nazia Karim
o(*￣▿￣*)ブ
Like · Reply

Habiba Rabee
I'm curious to know if you can answer me; how did you move from size 14 to 16 to being this ? Do/ did you suffer from any medical issues ?
As my opinion :if you are happy with your body and free of diseases and that weight doesn't bother you , then c...
See More
Like · Reply · 14w
2 Replies

Robyn Baker
b (￣▿￣) d
Like · Reply
3 Replies

Figure 13.d. The webpage when the Cyberbullying_Blocker plug-in is opened

5.2.1 Accuracy and F1 score

Since the comments crawled from the social media platform in real-time are not labeled, in order to avoid the influence of subjective factors on the comment classification, 10 samples are randomly selected from the test data set as the test samples. At the same time, because the test set does not participate in the weight adjustment in the trained model, that is, the model has no memory of the test set. Thus, it can be ensured that the random samples from the test set and the comments collected in real-time from the online social platforms are equally unfamiliar to the model. Therefore, random samples from the test set can test the classification performance of the model on behalf of the comments collected in real-time.

	Comments	**Actual label**	**Predict label**
1	'what the hell justin'	0	0
2	'= = racist girls = = how can this article possibly not mention the famous incident regarding the two racist girls whose video became viral and got them expelled from the school unconstitutionally ?'	0	1
3	': : : so far you have offered zero information just silly assumptions like a make - believe dispute over a court case you have not studied'	0	0
4	': yes she does although it says king and not monarch i have just cut it as it does seem faintly misleading'	0	0
5	'the true truth is that only my balls have official status in the principality of monaco	1	0

	stupid warm musky regards his most serene highness prince scruffy musky coinpurse iv’		
6	‘random deletion = = deleted xanax bars fuck you up mah nigga fuck wit em they good from the links section’	1	1
7	‘: redirected to arnold jtoynbee#civilizations’	0	1
8	‘hey shithead stop vandilizing articles’	1	1
9	‘i think the i is lowercase wikipedia capitalised the i most likely’	0	0
10	‘* i will be there’	0	0

Table 6. The actual labels and predicted results of 10 randomly selected samples

According to the actual labels and prediction results of the randomly selected samples, as well as the metrics mentioned in section 3.3, the results of accuracy and F1 score can be calculated as shown in table 7.

Metrics	Percentage(%)
Accuracy	70%
Recall rate	66.7%
Precision rate	50%
F1 score	57.2%

Table 7. The accuracy and F1 score of test samples

5.2.2 Analysis and Evaluation

Combining the accuracy of the model in section 5.1.1 on the test set in the five comparative experiments and the test results of the predictive classification on the randomly sampled samples, it can be seen that after the overall structure and parameters of the trained model are saved, and then tested with test samples, the result will be different from the test during training, that is, the model transfer will have a certain degree of accuracy loss. Although the classification accuracy of the model on the training set is very high (close to 1), but there is still a gap of accuracy when facing the specific sampling samples from the test set. This is probably because even if the dropout layer is used to avoid overfitting, this problem still exists.

At the same time, the gap between F1-score and accuracy shows that even if a model's accuracy is up to a tolerable level, it is difficult to find bad comments that account for a small proportion of the total. This result demonstrates the importance of introducing more than one measurement standard of accuracy in this article. The significance of extending the test result with a recall rate higher than the precision rate to real life is that even if the model sometimes misjudges the normal comment as a bad comment, while there is only a small possibility that the bad comment will not be identified. In view of the fact that one fierce criticism may cause trauma to the victim, even a model that seems a little overprotective has great value.

6 Legal, Social, Ethical, and Professional Issues

At present, all software, browser extension, and scripts involved in this project are in the testing phase, and all data sets used are public data sets. Besides, the comment texts collected by the browser plug-in from the social platform web pages are also public and can be browsed without any account login. And this paper does not involve any interviews with the public or the collection of private information. It is obvious that this project can not replace the law to exercise the power to punish bullies, but only uses science and technology to implement some protective measures on the side of the victim. At the same time, when processing the data set, all people including the perpetrators of bullying are anonymized without discrimination, that is, there are only renumbered ids without names.

The original intention and significance of the design of this project are also to protect and help the victims of cyberbullying and at the same time create a warm, peaceful, friendly, and non-hostile public-friendly environment for the majority of netizens.

7 CONCLUSION

7.1 Contributions

Under the premise of automatically preventing cyberbullying on online social media and protecting victims, this paper mainly contributes to the text classification based on the existing deep neural network model, RNN and CNN, on the Kaggle data set, that is, distinguishing the malicious comments (such as offensive, toxic, or insulting, etc.) and normal comments. Meanwhile, the hierarchical structure of the deep neural network and the backpropagation algorithm used for weight update are explained in detail. While integrating part of the network structure of RNN and CNN, the parameters in the model learning process are adjusted to get a better classification result. During the classification experiment process, the designed NN model was built, trained, and tested using basic functions under the TensorFlow framework in python language, and the classification test results were analyzed and compared.

After multiple sets of comparative experiments, and mastering the general rules of model training, the RNN-CNN model obtains a better classification result(Train_AUC from 0.6581 to 0.9876 and Test_Acc from 0.7012 to 0.9324). Besides, this project combined the well-trained model with the browser plug-in, Cyberbullying_Blocker to realize the user-friendly blocking function of malicious comments on the web page of online social platforms.

7.2 Issue and challenges

Word-level cyberbullying is easy to identify, sentence-level such as unwarranted charges and misunderstandings and ridicule are difficult to identify. The ambiguous comments that might be cyberbullying can only be reported by the victims and manually reviewed before they can be discovered. Technically, since most of the existing deep learning algorithm models are like black boxes, they can only adjust the actual judgment of the model on the text by adjusting the appearance parameters, such as the threshold value of the boundary between different categories. The choice of this threshold is difficult and complicated, just as the same sentence will be considered offensive by some people but not by others.

The problem of human flesh searches for victims and exposure of private information cannot be alleviated by shielding victims from bad reviews. It needs to cooperate with online social platforms and public security agencies.

7.3 Future plan

Due to time constraints and the shortage of King’s GPU cloud server equipment, the training of the model is not comprehensive enough. If conditions permit, in the future, there will be a plan to use multiple data sets to train models and blending more existing excellent models to achieve a more accurate recognition of cyberbullying comments.

However, blindly pursuing continuous improvement of the accuracy of automatic detection of cyberbullying cannot personally protect victims. This technology needs to be more implemented in actual application scenarios. When the victim's personal privacy is leaked by a human flesh search, he will suffer more severe physical bullying and substantial harm. Therefore, in order to protect personal privacy, the trained model (deep learning technology) can not only identify comments that humiliate or viciously attack victims on social platforms but also can be applied to text messages and emails that target the victims accurately.

8 REFERENCES

Breiman, L. (2001). Journal search results - Cite This For Me. Machine Learning, 45(1), 5-32. doi: 10.1023/a:1010933404324

C. E. Wills and D. C. Uzunoglu, "What Ad Blockers Are (and Are Not) Doing," 2016 Fourth IEEE Workshop on Hot Topics in Web Systems and Technologies (HotWeb), 2016, pp. 72-77, doi: 10.1109/HotWeb.2016.21.

Chen, Z., & Liao, I. (2020). Improved Fast R-CNN with Fusion of Optical and 3D Data for Robust Palm Tree Detection in High Resolution UAV Images. International Journal Of Machine Learning And Computing, 10(1), 122-127. doi: 10.18178/ijmlc.2020.10.1.908

Christopher, A. (2021, February 3). K-Nearest Neighbor. A Complete Explanation Of K-NN | By Antony Christopher | The Startup | Medium. Medium. https://medium.com/swlh/k-nearest-neighbor-ca2593d7a3c4.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine Learning, 20(3), 273-297. doi: 10.1007/bf00994018

Hao, X., Zhang, G., & Ma, S. (2016). Deep Learning. International Journal Of Semantic Computing, 10(03), 417-439. doi: 10.1142/s1793351x16500045

Kim, H., & Jeong, Y. (2019). Sentiment Classification Using Convolutional Neural Networks. Applied Sciences, 9(11), 2347. doi: 10.3390/app9112347

Kim, Y. . (2014). Convolutional neural networks for sentence classification. Eprint Arxiv.
Kowsari, Jafari Meimandi, Heidarysafa, Mendu, Barnes, & Brown. (2019). Text Classification Algorithms: A Survey. Information, 10(4), 150. doi:10.3390/info10040150

Kowsari, K. , Brown, D. E. , Heidarysafa, M. , Meimandi, K. J. , & Barnes, L. E. . (2017). Hdltex: hierarchical deep learning for text classification. IEEE International Conference on Machine Learning and Applications.

Landry, M. (2016, June 2). The Five D's Of Defense. Alamom Consulting, Inc. https://alamom.com/5defense/.

Liaw, A. , & Wiener, M. . (2002). Classification and regression by randomforest. R News, 23(23).

Malignant Comment Classification. (2021). Retrieved 11 June 2021, from https://www.kaggle.com/surekharamireddy/malignant-comment-classification

Mandic, D., & Chambers, J. (2001). Recurrent neural networks for prediction. New York: John Wiley.

Muneer, A. , & Fati, S. M. . (2020). A comparative analysis of machine learning techniques for cyberbullying detection on twitter. Future Internet, 12(11), 187.

Peterson, L. (2009). K-nearest neighbor. Scholarpedia, 4(2), 1883. doi: 10.4249/scholarpedia.1883

Poomka, P., Kerdprasop, N., & Kerdprasop, K. (2021). Machine Learning Versus Deep Learning Performances on the Sentiment Analysis of Product Reviews. International Journal Of Machine Learning And Computing, 11(2), 103-109. doi: 10.18178/ijmlc.2021.11.2.1021

Quinlan, J. (1986). Induction of decision trees. Machine Learning, 1(1), 81-106. doi: 10.1007/bf00116251

Reynolds, K. , Kontostathis, A. , & Edwards, L. . (2012). Using Machine Learning to Detect Cyberbullying. 2011 10th International Conference on Machine Learning and Applications and Workshops. IEEE.

Safavian, S., & Landgrebe, D. (1991). A survey of decision tree classifier methodology. IEEE Transactions On Systems, Man, And Cybernetics, 21(3), 660-674. doi: 10.1109/21.97458

Sutton, C. (2012). An Introduction to Conditional Random Fields. Foundations And Trends® In Machine Learning, 4(4), 267-373. doi: 10.1561/2200000013

Tampermonkey. (2021). Retrieved 1 September 2021, from https://addons.opera.com/en/extensions/details/tampermonkey-beta/

Van Gestel, T., Suykens, J., Baesens, B., Viaene, S., Vanthienen, J., & Dedene, G. et al. (2004). Benchmarking Least Squares Support Vector Machine Classifiers. Machine Learning, 54(1), 5-32. doi: 10.1023/b:mach.0000008082.80494.e0

Wang, X., Xu, B., Li, C., & Ge, W. (2015). Labeling Sequential Data Based on Word Representations and Conditional Random Fields. International Journal Of Machine Learning And Computing, 5(6), 439-444. doi: 10.18178/ijmlc.2015.5.6.548

9 APPENDICES

Both the RNN-CNN model and API script are python files(.py), and the browser plug-in is deployed on the browser of the host, so the code is saved as javascript format(.js). The pre-trained vocabulary vector and the clean word needed to purify the data set are in the features folder. The model saved in the experiment is in the checkpoints folder. The library functions needed in training are in the tools folder.

9.1 Appendix A: source code of RNN-CNN model

(The basic source code is referred from https://github.com/zake7749/DeepToxic#word-level)
The main network design is shown below:

```
    input_layer = Input(shape=(max_sequence_length,))
    embedding_layer = Embedding(nb_words,
                                embedding_dim,
                                weights=[embedding_matrix],
                                input_length=max_sequence_length,
                                trainable=False)(input_layer)#embedding

    Dropout_1 = SpatialDropout1D(0.2)(embedding_layer)#Dropout_1 layer
    rnn_1 = Bidirectional(CuDNNGRU(recurrent_units,
return_sequences=True))(Dropout_1)#Bi-GRU layer
    pooling = GlobalMaxPooling1D()(rnn_1)#pooling layer
    Dropout_2 = Dropout(0.5)(concatenated)#Dropout_2 layer
    conv_1 = Conv1D(filter1_nums, 2, kernel_initializer="normal", padding="valid",
activation="relu", strides=1)(Dropout_2)#Convolutional layer
    kmaxpooling = KMaxPooling(k=3)(conv_1)#KMaxpooling layer
    Dropout_3 = Dropout(0.5)(kmaxpooling) #Dropout_3 layer
    attn = AttentionWeightedAverage()(Dropout_3)
    average = GlobalAveragePooling1D()(Dropout_3)

    concatenated = concatenate([Dropout_3, attn, average], axis=1)# Concatnate layer
    x = Dense(120, activation="relu")(concatenated)
    output_layer = Dense(out_size, activation="sigmoid")(x)#Output classification results
```

```
model = Model(inputs=input_layer, outputs=output_layer)
```

9.2 Appendix B: source code of Cyberbullying_Blocker

(The basic source code is referred from https://github.com/ShoSatoJp/google_search_blocker)
The main function code is shown below:

```
        $(document.body).on('aThing', function () { //insert monitoring DOM tree
                hid(blacklist.host,blacklist.url,blacklist.word);
                $('[onmousedown]').attr('onmousedown','');

        $.each(jQuery('[href*=webcache]'),function(i,o){jQuery(o).attr('href',jQuery(o).attr(
'href').replace('p:','ps:'));});
        });
        hid(blacklist.host,blacklist.url,blacklist.word);
        $('#rso').delegate('a.block','click',function(e){
                $(this).parents('.g').addClass('hiden').hide();
                blacklist.host.push($(this).data('host'));
                GM_setValue('blacklist',JSON.stringify(blacklist));
                hid([$(this).data('host')],[],[]);
        });
        });

        function addKeyword() {
                var word=prompt("Please input the keyword." );
                if(word){
                        blacklist.word.push(word);
                        GM_setValue('blacklist',JSON.stringify(blacklist));
                        hid([],[],[word]);
                }
        }

  window.toggle = function(){
    var host = window.location.host;
    window.siteCfg[host] = !!!window.siteCfg[host];
    var hideThem = window.siteCfg[host];
    if( 'hwww.instagram.com' == host){
      window.killIns(hideThem);
    }else if('www.facebook.com' == host){
      window.killFacebook(hideThem);
    }else if('www.twitter.com' == host){
      window.killTwitter(hideThem);
    }
  };

  window.toggle();
```

9.3 Appendix B: source code of middleware API

(The basic source code is referred from https://blog.csdn.net/weixin43790560/article/details /86617630)

```
def get(sel_title):  # sel_title is the tag path of the comment's name
    r_title = r.html.find(sel_title)
    r_comment = r.html.find('id__u1i29bn7g1 > span')
    mylist = []

    print(r_title[0].text)  # output the title
    for result in r_code:
        mylist.append(result.text)
        mylist = " ".join(mylist)  #delete the [ ] and " "
        print(mylist)
        mylist = []  # Clear mylist to facilitate receiving the next result


if __name__ == "__main__":
    sel_title = 'head > title'  # the path of comments
    # sel_title = 'body > react-root'
    get(sel_title)
    CLASSES = ["malignant", "highly_malignant", "rude", "threat", "abuse", "loathe",
"bad_comment"]
    print("Predicting testing results...")
    pred_list = []
    for fold_id, model in enumerate(models):
        pred = model.predict(test_data, batch_size=1, verbose=1)
        pred_list.append(pred)
        np.save("Individual_Project/predict_path/", pred)

    pred = np.zeros(pred_list[0].shape)
    for fold_predict in pred_list:
        pred += fold_predict
    pred /= len(pred_list)

    test_ids = test_df["id"].values
    test_ids = test_ids.reshape((len(test_ids), 1))

    pred = pd.DataFrame(data=pred, columns=CLASSES)
    pred["id"] = test_ids
    pred = pred[["id"] + CLASSES]
    request.post(pred)
```