Chapt3. Methodology

3.1 approach

This report is designed to investigate the effectiveness of different deep learning models for predicting malicious comments by modelling the same dataset and comparing the accuracy of different models. This section describes the basic principle and algorithm of the core RNN-based deep learning models used in this paper to analyze user-posted comments in the Wikipedia talk page. The two models are used to determine whether a comment contains malicious words and to classify comments that contain malicious words into different categories.

1. prepare the input data

Wikipedia, a multilingual encyclopedic collaborative project based on wiki technology, whose headquarter based in the United States, is an online encyclopedia written in several languages[1]. There is an online forum page called talk page (discussion page) in Wikipedia where editors can post comments about articles or other Wikipedia pages in talk pages (also known as discussion pages).

Kaggle, a platform for developers and data scientists to run machine learning competitions, host databases, and write and share code, has crawled comments of approximately 130M in size from the Wikipedia talk page and has collated these comments into a dataset in csv format for different kinds of model training[2]. In this paper, a set of classified comments of 59.8M was used as the input training set and a set of unclassified comments of approximately 70M was used as the test set to evaluate the performance of different deep learning models.

2. pre-process the data

The entire dataset contains ids of users who posted the comment, the content of the comments and different types of labels of the malicious comments in the form of one-hot codes. For contents of the comments analyzed as textual data in this project, dataset may contain duplicate or irrelevant observations, structural errors, or unwanted outliers. Consequently, some pre-processing of the raw data is applied.

All letters were firstly converted into lower case for better feature extraction. Then, in order to visualize the cleaned data，the number of words per comment are displayed as a frequency chart. Next, useless words in the content that do not have an impact on the overall experiment are removed to improve the quality of the text. These words, which usually have no real meaning, such as pronouns, prepositions, or conjunctions, are collectively referred to as stop words[3]. Natural Language Toolkit (NTLK), the most used word segmentation package developed at the University of Pennsylvania, was called to remove these common stop words from the comments. The word frequency chart and word cloud of all the malicious and non-malicious comments of the entire training set are displayed separately below.

图形用户界面, 应用程序, Teams

描述已自动生成

Figure Word Frequency

文本

描述已自动生成

Figure WordCloud for Toxic Comments

文本

描述已自动生成

Figure WordCloud for non-Toxic Comments

After the overview of the data and initial processing, most of the comments in the entire dataset are concentrated in less than 500 characters, so data pre-processing operations are applied to retain the important information in the comments for better word separation operations: The comment text was converted into processed text input by limiting its length and size. From the below histogram showing the word length after the pre-processing progress, it is evident that most of the sentences falling in range 1-200 lengths, so the max length of 200 words was kept when padding the sentence. As the last step of preprocessing, each comment was vectorized from text into array of integer with the help of tokenizer, and any comment above the max length was trimmed, and zeros were padded in all the sentence below 200.

图形用户界面, 应用程序

描述已自动生成

Figure Word Length

3. build the deep learning model

In this project, recurrent neural networks (RNN) are used as the main model. The use of RNN networks dated to the 1980s and has evolved into one of the main network models in the field of deep learning[4]. One thing that distinguishes RNN from other neural network is that RNN can be applied to scenarios where the input of this network can be a sequence of sequence type of data. In other words, the output of an RNN network is somewhat related to its previous output. Due to its recurrent feedback connections, RNN model can establish relationships between the inputs of the preceding and following sequences, allowing the output values of the RNN at each moment to be influenced by the input values of multiple previous moments, yielding more accurate results[5]. Different RNN models were used in this project to compare their performance. The structure of the RNN is illustrated in Figure 5.

图表, 散点图

描述已自动生成

Figure 5 Structure of RNN

Long Short-Term Memory (LSTM) [6] is one of the RNN variants created to solve the RNN gradient vanishing problem. LSTM has a linear unit called constant error carousels (CECs) and is controlled by three gates that are used to store the input into the model from real time information entered the model[7]. The input gate controls whether information from the current moment is allowed to be added to the CEC, the output gate controls whether information from the previous moment's CEC will be output to affect the output of the next moment's node, and the forget gate controls whether information from the current moment's CEC will be formatted. Figure 6 illustrates an architecture of a single LSTM unit, where c represents the entire memory cell and a represents the output of this single unit.

图示

描述已自动生成

Figure 6 Architecture of LSTM Unit

The calculation of an individual cell of the LSTM will be shown as Eq. (1) ~ Eq. (6):

Different weights wi, wf, w0, ui, uf, u0 and biases bi, bf and b0 in each gate are computed with the input values xt at time t and the cell state vector vt-1 at time t-1 to form the input values zi, zf and z0 for each gate. These input values are computed into values between 0 and 1 by means of a specific activation function f(zi), f(zf) and f(z0) for mimicking the extent of openness of gate.

In the identical way, the input of the LSTM cell z enters the cell through a specific activation function g(z).

As the value stored in the memory cell, the update of c is displayed in Eq. (5).

The updated c is then calculated with the activation function h(c), and multiplied with f(z0) to yield the final output a of the LSTM unit at time t.

Another model used in this report is called Gated Recurrent Unit neural network (GRU). GRU [8] is a neural network that improves on the LSTM by not only retaining the gate feature, but also simplifying it by reducing the three gates in the model to two gates: the update gate and the reset gate[5]. The update gate is used to control how much of the information in the CEC from the previous moment will be output and thus affect the output of the node at the next moment, while the reset gate is used to control how much information can be added to the CEC. Figure 3 shows a schematic diagram of the GRU structure.

图示

描述已自动生成

Figure 7 schematic diagram of GRU

The calculation of an individual cell of the GRU will be shown as Eq. (7) ~ Eq. (10):

Value s(uu) and s(ur) of Update Gate and Reset Gate are formed by having input value xt at time t and the cell state ht-1 at moment t-1 firstly calculated with weights wu and wr and the bias bu and br of teach gate, and then calculated with the corresponding activation function s.

Then subsequent judgement is made using the update gate to determine the degree to which the new input value can be stored in the candidate where ⊙ indicates an all-element multiplication.

Reset gate is used to do the determination to choose whether to store the candidate as the new value in the memory cell.

3.1.4 Testing

Two generic model-like tests, pre-train and post-train tests, are written[8].

### Pre-train Test

Some tests can be used to test the data without adjusting the parameters.

* Check if the type of labels of the training and testing sets are the same one-hot code
* Check if the data of the training and testing sets are both containing id and comment\_text
* Check if the output of the LSTM and GRU model matches all types in the label
* Check that the range of the LSTM and GRU model output matches the range of the label of 0 to 1
* Adding assertions to the model to control the operation of the model

### Post-train Test

* Invariance Tests: Manually make changes (e.g.: change “a” to “A”) to the data entered the LSTM and GRU model to see if there is an impact on the model's predictions while ensuring that the output is not affected
* Directional Expectation Test: Manually make changes (e.g.: change “hate” to “like”) to the data entered the LSTM and GRU model to see if there is an impact on the model's predictions if it interferes with the model output
* Data Unit Test: Classify possible erroneous results in the model

Once these processes had been done, the evaluation and testing of the model can be used as a basis for modifying and refining the model.

3.2 Technology 变表格

The experimental environment used in this paper: M1 ProM1 Pro integrates several different components, including CPU, GPU, unified memory architecture (RAM), neural engine, etc.

The 8-core M1 Pro is equipped with a 14-core GPU and a 16-core neural engine for machine learning.

Python 3.8, TensorFlow 2.7.0 with Jupyter Notebook are used to implement the methods.

3.3 Project Version Management

Versions of the project are stored in the GitHub:

Progress of the project can be seen in the sharing folder with the URL <https://github.com/Ivvvvvvvy/OBU_Project>