Tiện ích phát hiện tin giả bằng trí tuệ nhân tạo AI

Lĩnh vực dự thi: Phần mềm hệ thống, Rô bốt và máy thông minh

Học sinh thực hiện:

Nguyễn Vũ Khánh Uy, 12/6 THPT PHAN CHÂU TRINH

Nguyễn Văn Thành Đạt, 12/6 THPT PHAN CHÂU TRINH

SỞ GIÁO DỤC VÀ ĐÀO TẠO THÀNH PHỐ ĐÀ NẴNG
Trường THPT Phan Châu Trinh

CUỘC THI KHOA HỌC KĨ THUẬT

CẤP TRƯỜNG NĂM HỌC 2020 – 2021

Dự án kĩ thuật:

Fake News Detection

Phát hiện tin giả bằng trí tuệ nhân tạo AI

***Lĩnh vực dự thi*: Phần mềm hệ thống, Rô bốt và máy thông minh**

***Tác giả* :**

1. Nguyễn Vũ Khánh Uy, Lớp 12/6

2. Nguyễn Văn Thành Đạt, Lớp 12/6

***Giáo viên hướng dẫn:* Nguyễn Mậu Thắng**

Đà Nẵng, ngày 30 tháng 10 năm 2020

MỤC LỤC

Mở đầu

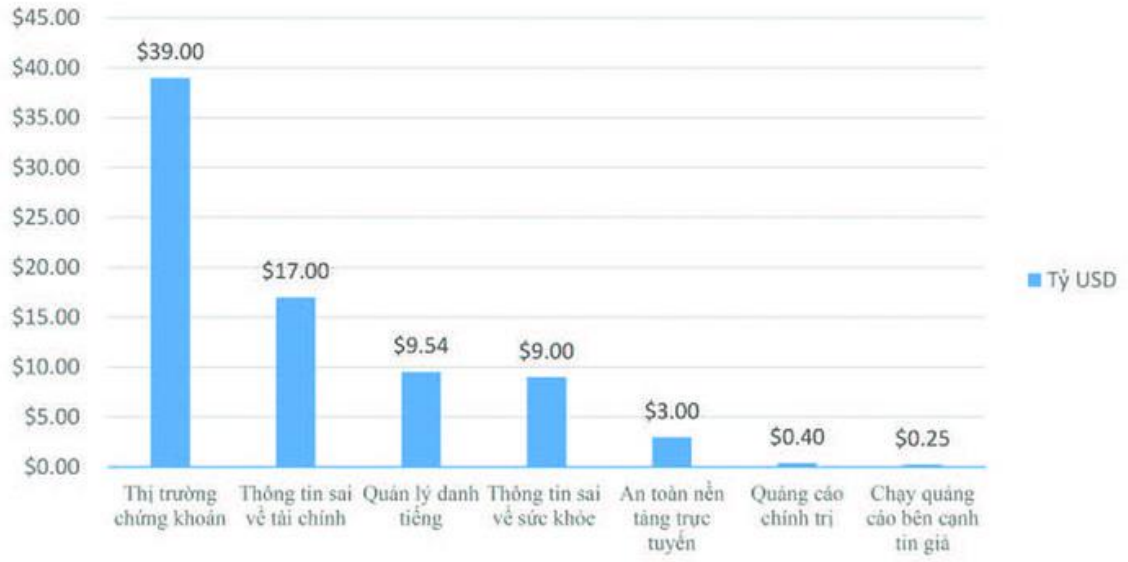
1 Câu hỏi nghiên cứu:

Việc phát triển công nghệ giúp mọi người có khả năng tiếp nhận thông tin nhanh và thuận lợi hơn nhưng cùng với đó là vô số rủi ro và tin giả là một vấn đề nghiêm trọng trong thời kì hiện nay khi có nhiều người tiếp cận thông tin nhưng không có khả năng chọn lọc, loại bỏ các thông tin xấu độc và các thành phần xấu(lừa đảo, phản động..) vẫn đang lộng hành. Và theo nghiên cứu [1] thì năm 2020 ở các nước phát triển tin giả sẽ còn nhiều hơn tin chính thống và **việc phát hiện tin giả bằng việc xem xét từ ngữ được sử dụng trong bài viết là khả thi**. VD: như bài viết sau, ta hoàn toàn có thể thấy được những chỗ từ ngữ quá đáng, lỗi font chữ, lặp từ..:

Thảo Ngọc (Danlambao) - Trong đợt hạn mặn đầu năm nay xảy ra ở một số tỉnh Tây Nguyên, nơi vốn đa số người dân là đồng bào thiểu số vốn đã nghèo, nay lại nghèo thêm.

Vì hạn hán kéo dài nên người dân không gieo cấy được. Một số khác tuy có tận dụng những nguồn nước ít □ỏi để gieo cấy, nhưng lại bị héo rụi do hạn và mặn kéo dài. Khiến cuộc sống của đồng bào nơi đây vốn đã khó khăn, nay lại khó khăn chồng chất.

Hay trong một báo cáo nghiên cứu của công ty An ninh mạng CHEQ và đại học Baltimore của Mỹ đã chứng minh rằng, tin tức giả mạo trực tuyến hiện gây thiệt hại cho nền kinh tế toàn cầu 78 tỷ USD hàng năm và chi phí mà các doanh nghiệp và chính phủ phải trả để ngăn chặn tin giả đang ngày càng gia tăng. Báo cáo, phân tích chi phí kinh tế trực tiếp từ tin tức giả ước tính đã góp phần làm mất giá trị thị trường chứng khoán khoảng 39 tỷ USD mỗi năm.



Mặt khác trong những năm gần đây, trí tuệ nhân tạo AI đã đạt được nhiều thành tựu to lớn, giải quyết được nhiều vấn đề nan giải mà con người rất tốn kém mới làm được. **Vậy nếu nghiên cứu sử dụng AI(Deep Learning) để phát hiện tin giả thì sẽ hạn chế thiệt hại đáng kể cho nền kinh tế và tạo ra một môi trường mạng an toàn, lành mạnh hơn.**

2 Đối tượng, công cụ nghiên cứu:

Đối tượng nghiên cứu:

- Mô hình Recurrent neural network(RNN), Bidirectional recurrent neural network(BRNN), lớp Word Embedding.. để tạo mô hình trí tuệ nhân tạo(model).
- Thư viện tensorflow và Keras để dễ dàng tạo ra các model và huấn luyện(train), thư viện flask để làm server chạy trí tuệ nhân tạo, thư viện newspaper để lấy thông tin bài viết cần xử lí.
- Nền tảng tiện ích (Extension).
- Bóc tách dữ liệu từ các file json và tiền xử lí về dạng phù hợp để huấn luyện.

Công cụ nghiên cứu:

- Google Colab hay Colaboratory: là một sản phẩm từ Google Research, nó cho phép chạy các dòng code python thông qua trình duyệt, đặc biệt phù hợp với Data analysis, machine learning và giáo dục. Colab không cần yêu cầu cài đặt hay cấu hình máy tính, mọi thứ có thể chạy thông qua trình duyệt, bạn có thể sử dụng tài nguyên máy tính từ CPU tốc độ cao và cả GPUs và cả TPUs đều được cung cấp miễn phí.
- Anaconda: là nền tảng (platform) mã nguồn mở về Khoa học dữ liệu (Data Science) trên Python thông dụng nhất hiện nay. Với hơn 6 triệu người dùng, Anaconda là cách nhanh nhất và dễ nhất để học Khoa học dữ liệu với Python

hoặc R trên Windows, Linux và Mac OS X. Đây là công cụ để nhóm dùng để tạo môi trường ảo cài các thư viện cần thiết và chạy server AI.

- Sublime Text 3: dùng để code server và extension vì có nhiều package hỗ trợ cho quá trình viết code dễ dàng hơn.
- Github: dùng để lưu mã nguồn và tải bộ dự liệu VFND sẽ đề cập ở dưới.

3 Tính mới của đề tài:

Việc phát hiện tin giả bằng trí tuệ nhân tạo (Deep Learning) đã được nhiều công ty công nghệ lớn và các nhóm, tổ chức trên thế giới nghiên cứu và thực hiện. Như Google có bert model, towardsdatascience, Stanford.. **Nhưng sản phẩm của của họ không thể được dùng để dự đoán tin giả với ngôn ngữ Tiếng Việt** vì trong một model như vậy luôn phải có một lớp word embedding (đó đơn giản là một lớp giúp chuyển các từ trong ngôn ngữ tữ nhiên thành các vector để phục vụ cho việc dự đoán tin giả). Và hiện tại **chưa có tổ chức nào nêu trên công bố phát triển công cụ phát hiện tin giả trên Tiếng Việt**. Và ở Việt Nam có một số tổ chức có làm được một số model với Embedding Tiếng Việt về các vấn đề khác có kết quả cao như: PhoBERT vào các tác vụ như sentiment analysis và đạt được kết quả cao như phân loại cảm xúc bình luận - Khôi Nguyễn.. nhưng các vấn đề các sản phẩm trên làm khi là quá khác biệt so với phát hiện tin giả nên không thể dùng kĩ thuật transfer learning để dùng lại Word Embedding của họ được, có thể xem thêm tại [7]. Và cũng chưa có tổ chức nào công bố ý tưởng tích hợp công nghệ này lên nền tảng tiện ích(Extension). Vì vậy sản phẩm của nhóm em có thể nói là **sản phẩm phát hiện tin giả đầu tiên bằng Tiếng Việt và tích hợp trên Extension.**

4 Phương án thiết kế và thu thập dữ liệu:

Đầu tiên khi bắt tay vào làm sản phẩm nhóm nhận ra là hầu hết các tin giả đều được đọc từ trình duyệt, hầu hết tin giả trên facebook cũng dẫn nguồn đến bài báo khác nên việc chọn một phương án thiết kế sản phẩm cho thuận tiện với người dùng khi họ đang đọc bài báo trên trình duyệt là rất cần thiết. Từ đó nhóm đã quyết định chọn làm sản phẩm là một **extension** với ưu điểm gọn nhẹ tích hợp trên trình duyệt giúp người dùng có thể sử dụng xử lí thông tin ngay trên trình duyệt mà không cần chuyển sang một phần mền thứ ba.

Thời gian hạn chế và việc thu thập dữ liệu về tin giả khá khó khăn nhưng nhóm đã tìm ra bộ dữ liệu VIETNAMESE FAKE NEWS DATASET – VFND [2], đó là bộ dataset về các tin tức giả bằng ngôn ngữ tiếng Việt được tập hợp trong khoảng thời gian từ 2017 đến 2019, các tin tức được đưa vào đây được phân loại thật giả dựa trên một số nguồn tin, tham chiếu chéo đến các nguồn tin được dẫn hoặc được phân loại bởi cộng đồng. Dữ liệu được lưu dưới dạng các file json và nhóm đã dùng ngôn ngữ lập trình python để bóc tách dữ liệu từ đó chuyển vào để huấn luyện model. Đến nay

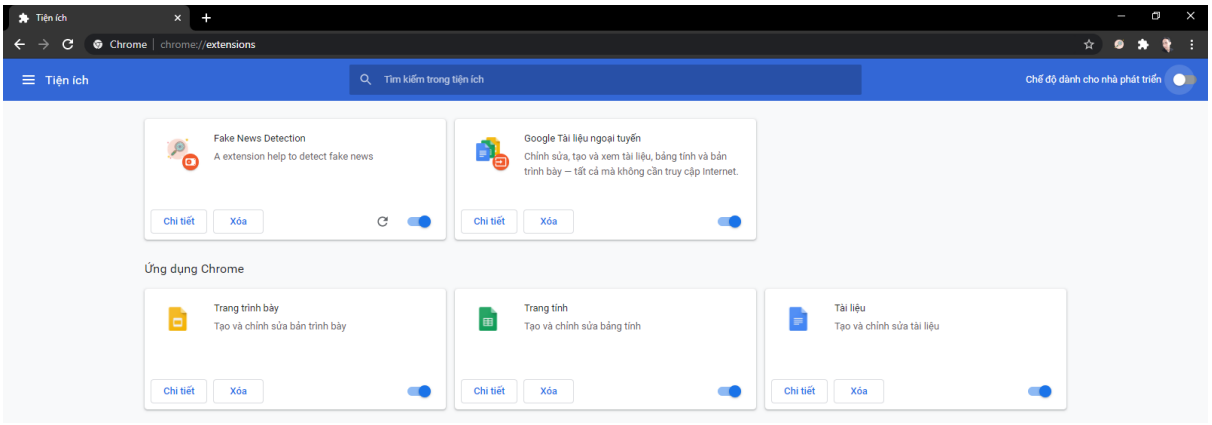
nhóm vẫn chưa có thời gian để bổ sung thêm dữ liệu khác ngoài bộ VFND nhưng trong tương lai thì sẽ cố kiếm thêm để tối ưu model.

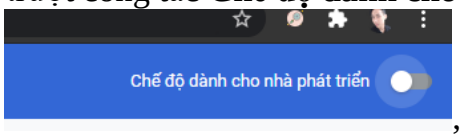
5. Hướng dẫn cài đặt và sử dụng sản phẩm:

Hướng dẫn cài đặt

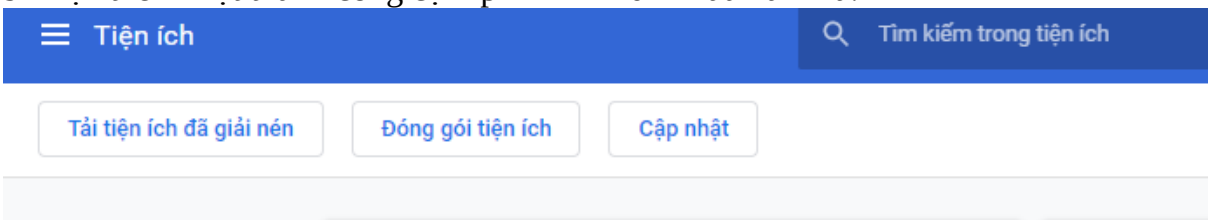
Hiện tại vì mục đích dự thi KHKT nên nhóm chưa upload sản phẩm lên google extension store nhưng trong tương lai nếu được tải lên rồi thì mọi người có thể lên đó tải về chỉ với một cú click chuột. Còn bây giờ nhóm sẽ hướng dẫn cài thủ công chỉ với vài bước đơn giản:

- B1: Mở trình duyệt lên và vào phần cài đặt extension, trong chrome bọn có thể truy cập nhanh bằng đường dẫn: chrome://extensions/

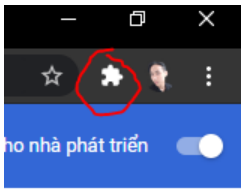


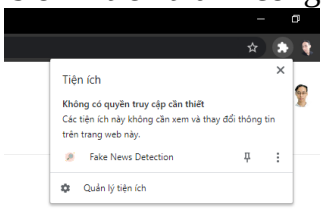
- B2: Khi ra giao diện như trên trượt công tắc **Chế độ dành cho nhà phát triển** ở góc trên bên phải màn hình 

, khi thực hiện thì sẽ hiện thêm một thanh công cụ ở phần dưới dải màu xanh là:



- B3: Chọn vào nút **Tải tiện ích đã giải nén** ở trên thanh công cụ vừa hiện ra ở trên và chọn tới folder chứa extension. Như vậy là đã cơ bản cài đặt xong nhưng muốn cho tiện sử dụng thì vui lòng đọc thêm B4.

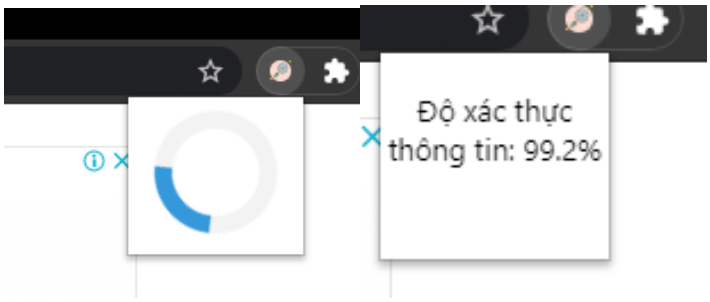
- B4: Chọn vào biểu tượng chung của các extension ở trên thanh công cụ trình duyệt: 

nó sẽ hiện ra như sau 

và chọn vào biểu tượng  để ghim extension ra ngoài cho dễ sử dụng. Sau khi làm xong thì tiện ích của chúng ta đã hiện ra ngoài như sau:  .

Hướng dẫn sử dụng

Khi truy cập vào một bài viết bất kì người dùng có thể yêu cầu dự đoán bằng cách click chuột vào biểu tượng của extension như trên, và hệ thống sẽ hiện thị khung đang xử lí sau khoảng 2 – 3 giây thì sẽ hiển thị độ xác thực thông tin được dự đoán:



Ngoài ra có thể xem thêm video hướng dẫn sử dụng nếu chưa hiểu tại: https://drive.google.com/file/d/1jRarsHhwwc4acrtYXQc7FnWV5N9kX6O5/view?usp=sharing

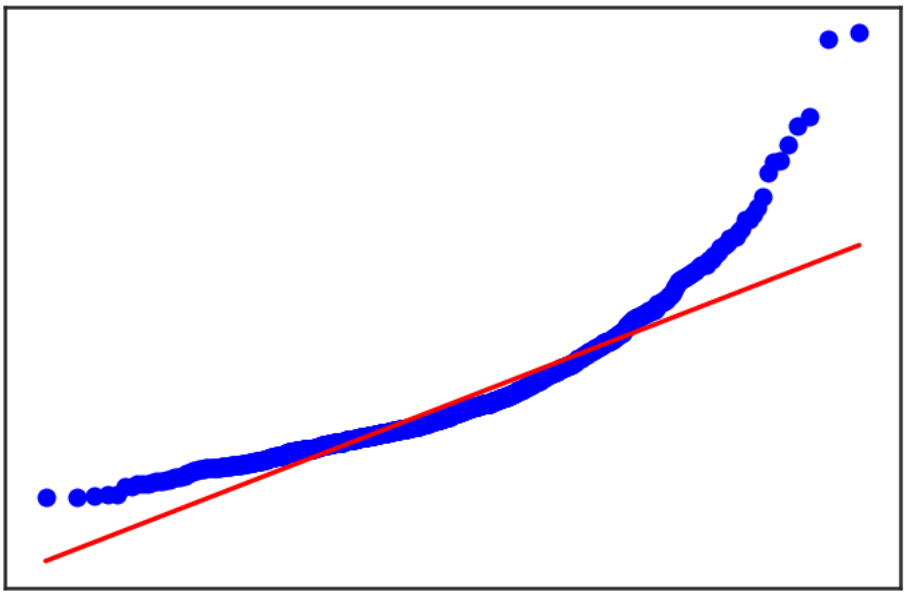
Nội dung

Phần I: Cơ sở lý thuyết:

Tiện ích(Extension): Một tiện ích mở rộng trình duyệt hay tiện ích mở rộng là một phần mềm nhỏ dùng để tùy biến một trình duyệt web. Trình duyệt thường có nhiều loại tiện ích khác nhau với chức năng đa dạng, bao gồm chỉnh sử giao diện người dùng, chặn quảng cáo, và quản lý cookie..

Trước khi giới thiệu về deep learning(học sâu) ta cần giới thiệu một chút về machine learning(học máy). Có thể đọc kĩ hơn ở [8] còn nhóm xin giải thích một cách đơn giản dễ hiểu nhất có thể:

Machine learning: Khi các nhà khoa học nghiên cứu về các bài toán VD như bài toán dự đoán giá tiền của ngôi nhà dựa vào diện tích của nó(tất nhiên là giá trị ngôi nhà còn phụ thuộc vào vị trí của ngôi nhà nhưng để cho đơn giản thì ví dụ chỉ như vậy):



Người ta nhận ra rằng giá nhà(đường xanh dương) gần như là một đường thẳng tuyến tính và việc dùng thuật toán để xác định một đồ thị hàm số(đường đỏ) giá nhà theo diện tích để tiện cho việc sự dụng lại với độ sai sót không đáng kể là khả thi. Đó là tư tưởng cơ bản của machine learning, cố gắng tạo ra một đồ thị khớp với dữ liệu mẫu hết sức có thể(đương nhiên VD trên chỉ là dạng đơn giản của machine learning).

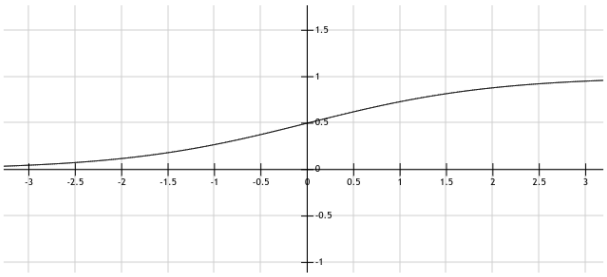
Vậy thì deep learning(học sâu) là gì? Nó là một dạng của machine learning, các nhà khoa học nhận ra là việc chỉ cố vẽ một đồ thị hàm số từ bộ dữ liệu hoàn chỉnh như vậy với một số bài toán là không khả thi. Vì vậy họ mới nghĩ ra cách để biểu diễn đơn giản giữ liệu lại thành nhiều lớp khác nhau rồi vẽ các đồ thị hàm số giữa các lớp và hợp lại ở một lớp cuối. Đó là ý tưởng cơ bản của Deep learning. Và ở dưới phần Cách thức hoạt động và kiến trúc model sẽ giới thiệu về một mô hình của deep learning đó là RNN. Mô hình này không dự đoán ngay từ dữ liệu ban đầu mà đọc từng từ một và ý nghĩ của từ đó sẽ được lấy từ các dự đoán trước rồi mới suy ra ý nghĩa của từ hiện tại và quy nạp đến hết bài viết.

Activation function(hàm kích hoạt phi tuyến)[9]: như ở trên, các hàm dùng thuật toán để vẽ ra các đồ thị và thuật toán đó cơ bản là cố gắng tối ưu các trọng số(weight) từ các hàm activation. Nhóm sẽ giới thiệu sơ về công thức hai hàm nhóm sử dụng còn sẽ không giải thích cách tối ưu trọng số hay các chứng minh toán học:

- Thứ nhất là hàm Sigmoid: nhận đầu vào là một số thực và chuyển thành một giá trị trong khoảng (0;1)

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

- Phân tích

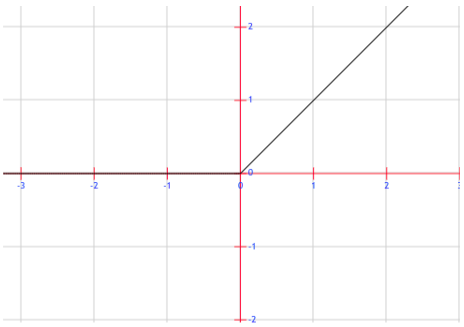


Đồ thị hàm Sigmoid - nguồn: Wikipedia

- Thứ hai là hàm relu: ReLU đơn giản lọc các giá trị < 0

$$f(x) = max(0, x)$$

- Phân tích



Đồ thị hàm ReLU

Phần II: Cách thức hoạt động và kiến trúc model (kèm mã nguồn)

Cách thức hoạt động:

Khi người dùng bấm vào extension thì qua thư viện **Tabs** lấy được đường dẫn của bài viết hiện tại đang đọc và dùng thư viện **XMLHttpRequest** để gửi đường dẫn đó đến server đang chạy model để đưa ra dự đoán:

Lấy url, hiện thị khung loading, gọi hàm gửi dự đoán đến server

```
const server_address = '127.0.0.1';
const port = '5000';

function showResult(prediction) {
  // prediction = -1: đang xử lí
  if (prediction < 0) {
    document.getElementById("result").style.visibility = "hidden";
    document.getElementById("loader").style.visibility = "visible";
  } else {
    document.getElementById("result").style.visibility = "visible";
    document.getElementById("loader").style.visibility = "hidden";
    document.getElementById("prediction").innerHTML = prediction + "%";
  }
}

chrome.tabs.query({active: true, currentWindow: true}, tabs => {
  let url = tabs[0].url;
  getPrediction(url);
  // console.log(url);
  // chrome.tabs.sendMessage(tabs[0].id, {url_target: url}, function(response) {
  //   // console.log(response.result);
  // });
});
```

Gửi dự đoán tới server và khi có kết quả gọi hàm để ẩn khung loading và hiển thị kết quả

```
function getPrediction(url) {
  var xhttp = new XMLHttpRequest();
  xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
      res = (parseFloat(this.responseText) * 100).toFixed(1);
      // console.log(res);
      showResult(res);
    }
  };
  xhttp.open("GET", ("http://" + server_address + ":" + port + "/cn?url=" + url), true);
  xhttp.send();
}

window.onload = function() {
  showResult(-1);
}
```

Server dùng thư viện **flask** để bắt các request sau đó dùng thư viện **newspaper** để lấy dữ liệu các bài viết và gọi các **hàm tiền xử lý** data để đưa về dạng phù hợp cho việc dự đoán, cuối cùng là gọi các **hàm dự đoán** từ model đã khởi chạy sẵn:

```
74  def RorF(domain, title, content, model):
75    if type(domain) == list:
76      target = create_predict_data(domain, title, content)
77    else:
78      temp = domain + ' ' + title + ' ' + content
79      target = pre_progress(encoder(temp), ml)
80    res = model.predict(target)
81    return res
82
83  @app.route('/cn')
84  def predict():
85
86      url_target = request.args.get('url')
87
88      article = Article(url_target, language='vi')
89      article.download()
90      article.parse()
91      # print(article.title)
92
93      domain = [getDomain(url_target)]
94      title = [article.title]
95      content = [article.text]
96
97      return str((RorF(domain, title, content, model))[0][0])
98      # return str(RorF(domain, title, content, model))
99
100 if __name__ == '__main__':
101     CORS(app)
102     app.run(debug=True, port=5000)
```

Các hàm tiền xử lý dữ liệu sử dụng **numpy**, khởi chạy và dự đoán model dùng **tensorflow**:

Khởi chạy model cho dự đoán

```python
from flask import Flask, request
from newspaper import Article
import tensorflow as tf
import numpy as np
from flask_cors import CORS

ml = 2500
model_path = 'model/V2/'

# load vocab
dict_vocab = []
try:
    with open(model_path + 'vocab.txt', 'r') as inf:
        for line in inf:
            dict_vocab.append(eval(line))
    vocab = dict_vocab[0]
    print(len(vocab))
except Exception:
    pass

# load model
f = open(model_path + "model.json", "r")
# print(f.read())
model = tf.keras.models.model_from_json(f.read())

# load weight
model.load_weights(model_path + "model.h5")

```

Tiền xử lý

```python
# preprocess
def getDomain(url):
    p1 = url.find('//') + 2
    p2 = url.find('/', p1)
    return url[p1:p2]

app = Flask(__name__)

def encoder(data):
    res = []
    for i in data:
        k = []
        for j in i.split():
            try:
                k.append(vocab[j])
            except Exception:
                k.append(0);
        res.append(k)
    return res

def pre_progress(data, max_len):
    # res = np.empty(223, dtype=list)
    # j = 0
    res = []
    for i in data:
        zeros = [0] * (max_len - len(i))
        res.append(zeros + i)
    # res[j] = zeros + i
    # j += 1
    return np.array(res)

# def RorF(domain, title, content, model):
#   temp = domain + ' ' + title + ' ' + content
#   target = pre_progress(encoder(temp), ml)
#   res = model.predict(target)
#   return res

def create_predict_data(ddata, tdata, cdata):
  res = []
  for i in range(len(ddata)):
    temp = ddata[i] + ' ' + tdata[i] + ' ' + cdata[i]
    res.append(temp)
  # print(res)
  return pre_progress(encoder(res), ml)

```

Sau khi model đưa ra dự đoán sẽ được đọc lại từ extension và hiển thị kết quả. Đó là cách thức hoạt động cơ bản của thệ thống và có thể được biểu diễn đơn giản qua sơ đồ khối sau:

Kiến trúc model: Gồm 5 lớp

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, None, 64)          8073856
_________________________________________________________________
bidirectional (Bidirectional (None, 128)               66048
_________________________________________________________________
dense (Dense)                (None, 64)                8256
_________________________________________________________________
dropout (Dropout)            (None, 64)                0
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 65
=================================================================
Total params: 8,148,225
Trainable params: 8,148,225
Non-trainable params: 0
_________________________________________________________________
```

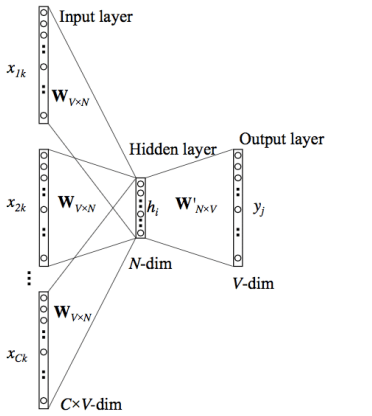
Lớp đầu là lớp Word Embedding [3]:

Lớp này giúp chuyển đổi từ ngữ trong ngôn ngữ tự nhiên sang word embedding. Word embedding được coi là cách tốt nhất để thể hiện các từ trong văn bản. Kỹ thuật này cũng gán mỗi từ với một vector, nhưng ưu việt hơn kỹ thuật vector ngẫu nhiên vì các vector này được tính toán để biểu diễn quan hệ tương đồng giữa các từ. Để tạo word embedding thì có hai phương pháp phổ biến là: **Skip Gram** và Continuous Bag of Word (**CBOW**). Ở đây nhóm em làm theo phương pháp thứ hai vì có ưu điểm là train nhanh và biểu hiện tốt trên bộ dữ hiệu lớn nhưng khi thu thập dữ liệu thì nhóm nhận ra là khá mất thời gian nên chưa tìm được nhiều nên kết quả vẫn chưa tối ưu lắm

và trong tương lai có thể nhóm sẽ nghiên cứu thử dùng phương pháp thứ nhất. Vì vậy ở đây sẽ nói luôn về cả hai phương pháp:

Mô hình CBOW: Phương pháp này lấy ngữ cảnh của mỗi từ làm đầu vào và cố gắng dự đoán từ tương ứng với ngữ cảnh. Hãy xem xét ví dụ: Hôm nay tôi đi học. Chúng ta sẽ cố gắng dự đoán từ mục tiêu (đi) bằng cách sử dụng duy nhất một từ ngữ cảnh đầu vào (học). Cụ thể hơn, chúng tôi sử dụng mã hóa one-hot của từ đầu vào và đo lỗi đầu ra của mạng nơ ron đối với mã hóa one-hot của từ mục tiêu (đi). Ngoài ra, chúng ta có thể xây dựng các kiến trúc dự đoán một từ bằng nhiều từ xung quanh. Trong quá trình dự đoán từ mục tiêu, mô hình sẽ học được biểu diễn vectơ của từ mục tiêu.

Mô hình Skip Gram:



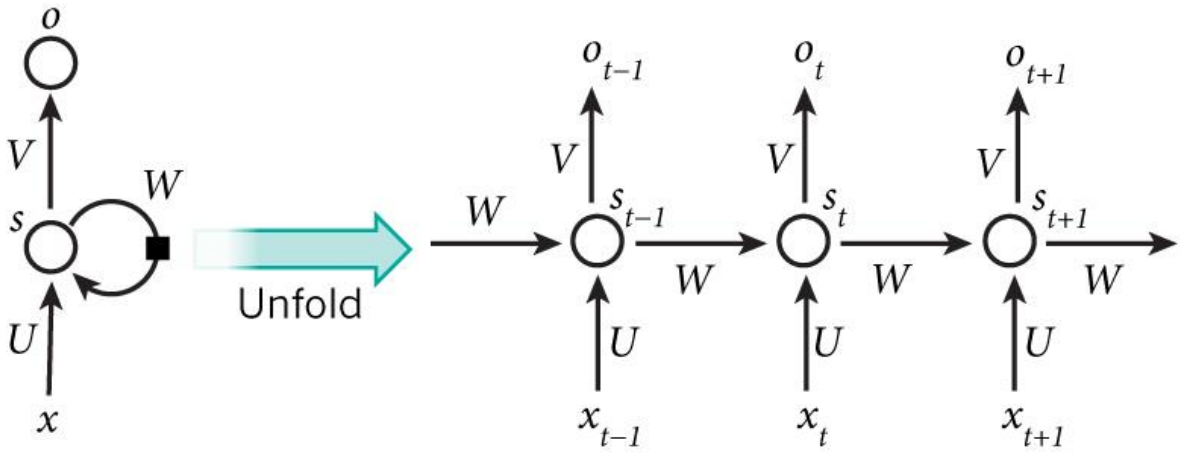
Mô hình Skip Gram còn được coi là phiên bản đảo ngược của mô hình CBOW. Cho trước một vị trí ngữ cảnh, mô hình cần đưa ra được phân bố xác suất của mỗi từ ở vị trí đó. Trong cả hai trường hợp, mạng sử dụng lan truyền ngược để học ra biểu diễn vector của từ.

Lớp thứ hai là lớp Bidirectional RNN [4]:

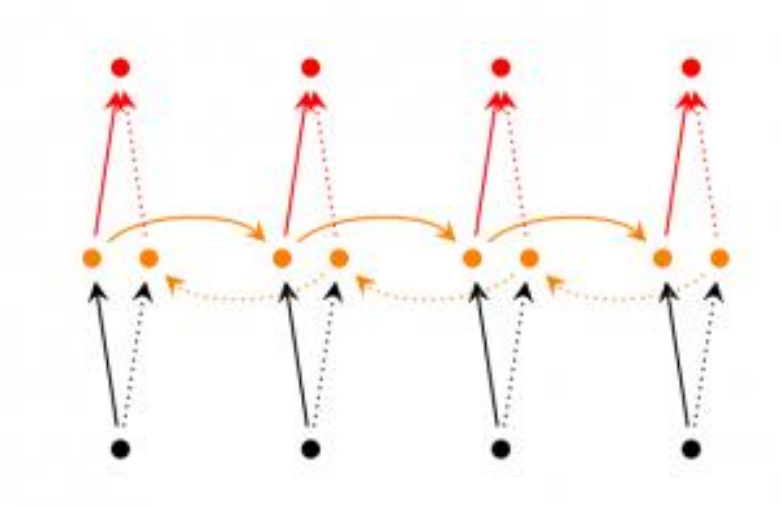
Để giải thích về lớp này em xin giải thích trước về một lớp RNN[5] trước vì BRNN là một lớp cải tiến của RNN. Thì ý tưởng cơ bản của việc sử dụng RNN là do trong ngôn ngữ ý nghĩa của một câu từ có thể phụ thuộc vào ngữ cảnh mà nó đang được đặt vào. VD trong một bài viết:

Ta có thể thấy sự việc Mr Bean qua đời là không chính xác vì các từ ngữ ở trước đã phủ định nó. Để xử lí vấn đề đó RNN sử dụng chuỗi các thông tin. RNN được gọi là hồi quy (Recurrent) bởi lẽ chúng thực hiện cùng một tác vụ cho tất cả các phần tử của một chuỗi với đầu ra phụ thuộc vào cả các phép tính trước đó. Nói cách khác, RNN có khả năng nhớ các thông tin được tính toán trước đó. Trên lý thuyết, RNN có thể sử dụng được thông tin của một văn bản rất dài, tuy nhiên thực tế thì nó chỉ có thể nhớ

được một vài bước trước đó (ta cùng bàn cụ thể vấn đề này sau) mà thôi. Về cơ bản một mạng RNN có dạng như sau:



Nhưng RNN có một hạn chế là ý nghĩa của một từ không chỉ phụ thuộc vào các từ đứng trước nó mà còn là các từ đứng sau đó. VD khi ta muốn dùng RNN để đánh giá bình luận của người dùng về một bộ phim là tốt hay tệ, bình luận của người dùng đó như sau: "Phim hay thế, biết vậy khỏi xem" ta có thể thấy đó là một câu nói mỉa mai và ý nghĩ của từ "hay" phụ thuộc vào các từ ngữ đứng sau nó mà RNN chỉ có thể nhớ những từ ngũ đứng trước. Để giải quyết vấn đề đó thì người ta đã phát minh ra BRNN, đó cơ bản là cài đặt 2 mô hình RNN chạy ngược hướng nhau với mô hình chạy theo hướng đọc(thường là trái sang phải nhưng tùy theo ngôn ngữ) là chính và mô hình kia để hiệu chỉnh kết quả của mô hình chính. Mô hình BRNN:



Các lớp tiếp theo không có gì quá nổi bật nên sẽ giải thích đơn giản:

Lớp thứ ba và thứ năm là lớp Dense:

Dense layer hay Fully-connected layer là một lớp cổ điển trong mạng nơ ron nhân tạo. Mỗi nơ ron nhận đầu vào từ tất cả nơ ron lớp trước đó.

Có thể hiểu cơ bản là như đã nói ở phần **Cơ sở lý thuyết** thì đó là các lớp chứa các điểm nút trên đồ thị dựa vào đó các phần đồ thị giữa hai điểm nút kề nhau có thể được các hàm activation uốn đồ thị theo dữ liệu mẫu.

Lớp thứ tư là lớp Dropout:

Dropout là việc chúng ta sẽ bỏ qua một vài unit trong suốt quá trình train trong mô hình, những unit bị bỏ qua được lựa chọn ngẫu nhiên. Ở đây, chúng ta hiểu "bỏ qua - ignoring" là unit đó sẽ không tham gia và đóng góp vào quá trình huấn luyện (lan truyền tiến và lan truyền ngược).

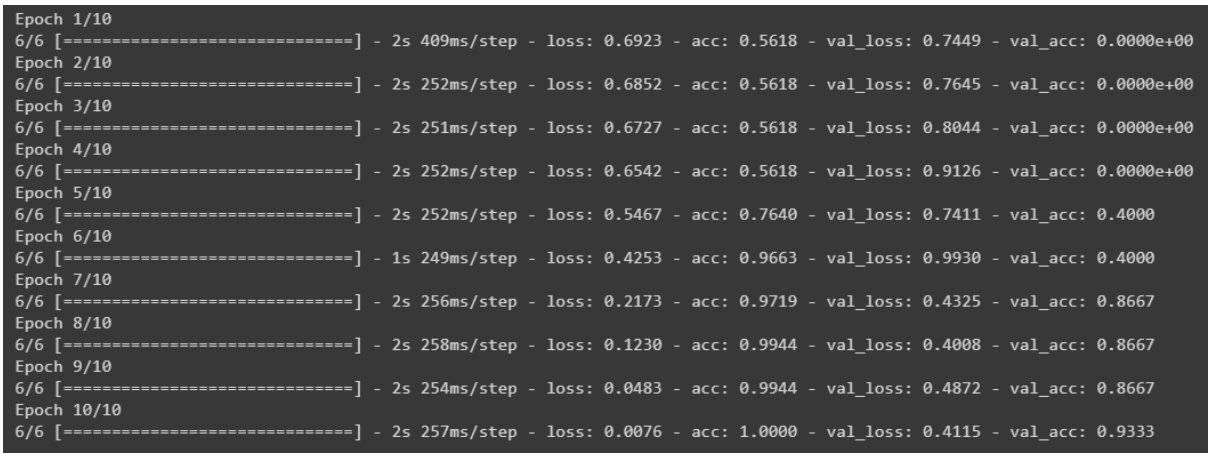
Dropout được sử dụng để chống over-fitting[6]. Nó cơ bản là vấn đề do mô hình cố gắng uốn đồ thị quá khớp bộ dữ liệu do train không phù hợp dẫn đến thể hiện rất tệ trong thực tế.

Kết luận

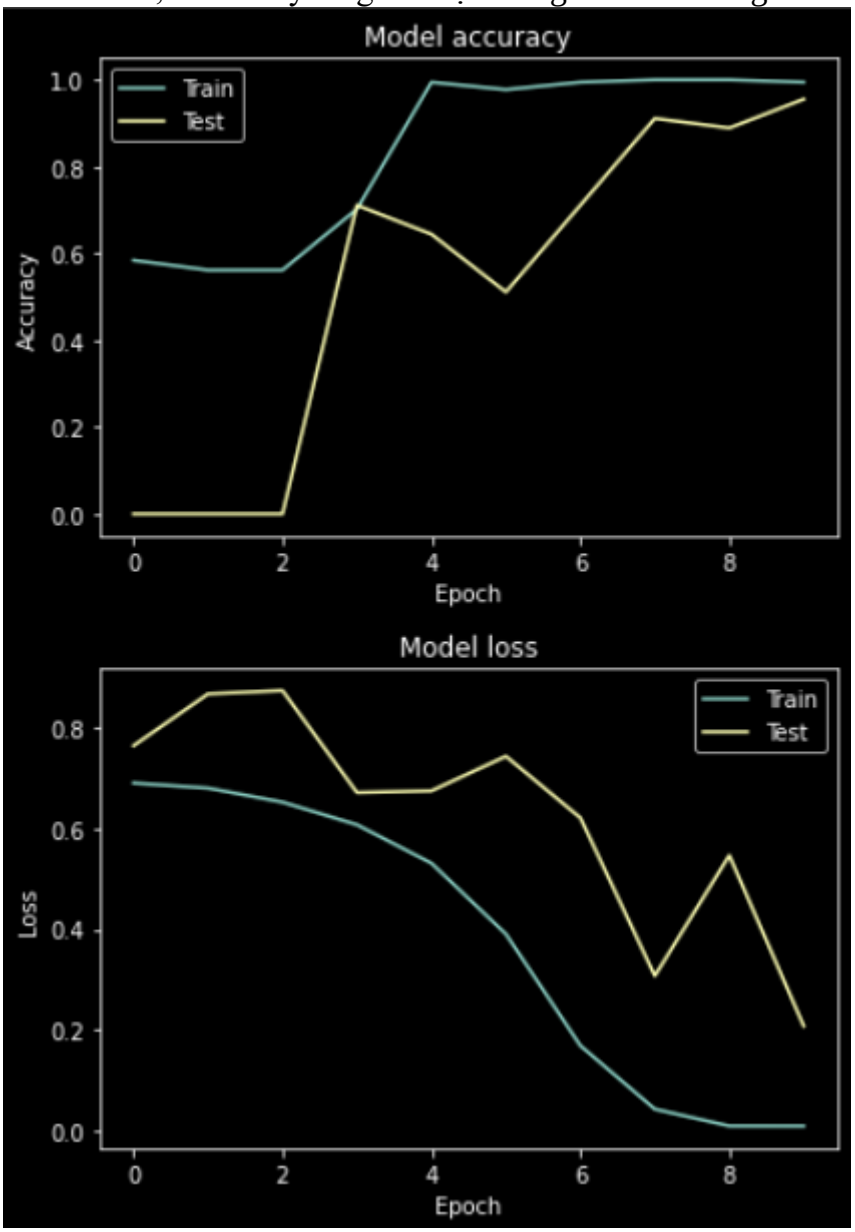
1 Kết quả:

Thiết kế chế tạo được sản phẩm đáp ứng mục tiêu nghiên cứu đặt ra, cụ thể:

- Cho kết quả khá tốt trên bộ dữ liệu kiểm thử(độ chính xác val_acc thời điểm cao nhất đạt 93.33%):



Và không hề có dấu hiệu over-fitting(có thể thấy trong biểu đồ được chiết xuất từ mô hình, Accuracy tăng liên tục trong khi loss vẫn giảm tốt):



- Tích hợp thành công lên nền tảng extension và giao tiếp tốt với server AI.

2 Tính ưu việt của đề tài

- Kích thước nhỏ gọn, dễ cài đặt, không đòi hỏi cấu hình cao, đưa ra kết quả nhanh chóng(thời gian mỗi lần dự đoán trả về kết quả chỉ 2 – 3 giây), hoàn toàn không có hiện tượng giật lag, chiếm ít bộ nhớ(vì hoàn toàn xử lí trên server), tính bảo mật cao(vì cái được gửi đi chỉ là đường link của bài viết).
- Tính tái sử dụng là cao vì có thể sử dụng cài đặt trên các nền tảng khác vì đa số phần xử lí là trên server.
- Hoạt động tốt trên các bài viết từ năm 2017 đến 2019.

- Có tính mới, tính sáng tạo cao.

3 Hạn chế

- Do thời gian hạn chế nên chưa tìm được thêm dữ liệu và trong bộ dữ liệu cũng có một số bài viết đặt sai nhãn dẫn đến sự khó khăn khi huấn luyện mô hình. Bên cạnh đó định dạng dữ liệu cũng khá tệ dẫn đến các bước tiền sử lý khó khăn.
- Cũng vì do vấn đề thời gian nên chưa tích hợp thử các mô hình tạo embedding mới như đã đề cập ở trên và chưa thử tích hợp lớp Pooling(Đây là một lớp khá tốt trong việc bắt những từ hiểm vì trong các bài viết tin giả đặc biệt là các bài có nội dung phản động thì thường sử dụng những từ ngữ đặc biệc).
- Do bộ dữ liệu nhỏ(chưa đến 250 bài viết) chỉ có các bài viết từ 2017 đến 2019 và kèm với sự phát triển của tin giả.

=> Việc dự đoán các tin giả trong nửa cuối năm 2019 đến 2020 là không ổn định lắm.

4 Hướng phát triển của đề tài

- Bổ sung thêm các tin giả trong thời gian gần đây, lọc và đánh nhãn lại các bài viết. Thực sự các nguồn tin giả của Việt Nam có rất nhiều, nhất là trong các trang phản động như: https://danlambaovn.blogspot.com/ nên nếu được đi tiếp thì việc bổ sung thêm là khá dễ dàng.
- Cải thiện thuật toán, cấu trúc model. Cụ thể là thử tích hợp lớp Pooling và mô hình tạo word embedding mới. Nghiên cứu áp dụng thêm bert model.
- Vì bản chất hệ thống chỉ dự đoán một tin là thật hay giả dựa trên từ ngữ của bài viết nên việc dự đoán tin giả và huấn luyện thêm các sự kiện nội bật trong thời gian đó là việc phải đi song hành nhau thì mới có đạt được hiệu quả cao. Nên có thể sẽ tạo ra một hệ thống cho phép người dùng có thể đóng góp thêm cho sản phẩm bằng việc đánh giá một tin là thật hay giả bằng kiến thức của mình và sẽ được duyệt để đưa vào huấn luyện như mô hình làm việc của hệ thống Google Dịch chứ không phải chính tác giả phải đi tìm dữ liệu như hiện tại.
- Hỗ trợ thêm phát hiện tin giả trên các nên tảng mạng xã hội như facebook..

Các nguồn tham khảo thêm:

[1]: Titcomb, J., Carson, J.: www.telegraph.co.uk. Fake news: What exactly is it – and how can you spot it?

[2]: https://github.com/thanhhocse96/vfnd-vietnamese-fake-news-datasets

[3]: https://trituenhantao.io/kien-thuc/word-embedding-la-gi-tai-sao-no-quan-trong/

[4]: https://en.wikipedia.org/wiki/Bidirectional_recurrent_neural_networks

[5]: https://dominhhai.github.io/vi/2017/10/what-is-rnn/

[6]: https://en.wikipedia.org/wiki/Overfitting

[7]: https://www.quora.com/In-recurrent-neural-networks-like-LSTMs-is-it-possible-to-do-transfer-learning-Has-there-been-any-research-in-this-area

[8]: https://viblo.asia/p/gioi-thieu-ve-deep-learning-deep-learning-hoat-dong-nhu-the-nao-3Q75w2nDlWb

[9]: https://aicurious.io/posts/2019-09-23-cac-ham-kich-hoat-activation-function-trong-neural-networks/