FAKE NEWS DETECTION

MINOR PROJECT REPORT

A thesis submitted in partial fulfillment of the requirements for the award of the degree of

B.Tech

in
INFORMATION TECHNOLOGY

BY

VICKY KUMAR PRASAD
(2K16/IT/125)
UMESH SARASWAT
(2K16/IT/120)

Under the supervision of

Mrs. Priyanka Meel

DEPARTMENT OF INFORMATION TECHNOLOGY
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering) Bawana Road, New Delhi-110042

DECEMBER, 2019

DEPARTMENT OF INFORMATION TECHNOLOGY
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, New Delhi-110042

CANDIDATE'S DECLARATION

We, Vicky Kumar Prasad(2K16/IT/125), Umesh Saraswat (2K16/IT/120) students of B. Tech (Information Technology), hereby declare that project Dissertation titled "DETECTING ONLINE FAKE NEWS BY CONSIDERING STANCE OF HEADLINE+BODY TEXT+IMAGE ON ONLINE NEWS ARTICLES" which is submitted to the Department of Information Technology, Delhi Technological University, New Delhi in partial fulfillment of the requirement for the award of the degree of Bachelor of Technology, is original and not copied from any source without proper citation. This work has not previously formed the basis for the award of any Degree, Diploma Associateship, Fellowship or other similar title or recognition.

Place: New Delhi

Date:

VICKY KUMAR
PRASAD
2K16/IT/125

UMESH
SARASWAT
2K16/IT/120

DEPARTMENT OF INFORMATION TECHNOLOGY

DELHI TECHNOLOGICAL UNIVERSITY (Formerly Delhi College of Engineering)
Bawana Road, New Delhi-110042

CERTIFICATE

I hereby certify that the Project Dissertation titled "DETECTING ONLINE FAKE NEWS BY CONSIDERING STANCE OF HEADLINE+BODY TEXT+IMAGE ON ONLINE NEWS ARTICLES" which is submitted by Vicky Kumar Prasad (2K16/IT/125), Umesh Saraswat (2K16/IT/120) students of B. Tech (Information Technology), Delhi Technological University, New Delhi in partial fulfillment of the requirement for the award of the degree of Bachelor of Technology, is a record of the project work carried out by the students under my supervision. To the best of my knowledge this work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere.

Place: New Delhi

Date:

Mrs. Priyanka Meel

(SUPERVISOR)

ASSISTANT PROFESSOR

ACKNOWLEDGEMENT

With this project towards its completion, we would like to extend our sincere gratitude towards Mrs. Priyanka Meel who has acted as a direct guide during the entire course. We are sincerely grateful for her invaluable insight, the time and effort she dedicated to this work and, especially, for her contagious enthusiasm and optimism.

We are also grateful to the Department of Information Technology at Delhi Technological University for presenting us with this research opportunity which was essential in enhancing learning and promote research culture among ourselves.

We would like to thank our families for their unconditional support and faith in our endeavors. Their prayers and wishes have played a major role in the successful completion of this project. We would also like to thank our friends for the love, support and faith in us.

Above all, I would like to thank God without whom none of this would have been possible.

DECLARATION

This is to declare that this report has been written by us. No part of the report is plagiarized from other resources. All information included from other sources has been duly acknowledged. We are aware that if any part of the report is found to be plagiarized, we shall take full responsibility for it.

Date :

VICKY KUMAR PRASAD
2K16/IT/125

UMESH SARASWAT
2K16/IT/120

CONTENTS

ABSTRACT

In recent years, due to the booming development of online social networks, fake news for various commercial and political purposes has been appearing in large numbers and widespread in the online world. With deceptive words, online social network users can get infected by these online fake news easily, which has brought about tremendous effects on the offline society already. An important goal in improving the trustworthiness of information in online social networks is to identify the fake news timely. This paper aims at investigating the principles, methodologies and algorithms for detecting fake news articles, creators and subjects from online social networks and evaluating the corresponding performance.

This paper addresses the challenges introduced by the unknown characteristics of fake news and diverse connections among news articles, creators and subjects. This paper introduces a novel automatic fake news credibility inference model, namely FAKEDETECTOR. Based on a set of explicit and latent features extracted from the textual information, FAKEDETECTOR builds a deep diffusive network model to learn the representations of news articles, creators and subjects simultaneously. Extensive experiments have been done on a real-world fake news dataset to compare FAKEDETECTOR with several state-of-the-art models, and the experimental results have demonstrated the effectiveness of the proposed model.

Index Terms—Fake News Detection; Diffusive Network; Text Mining; Data Mining

LIST OF FIGURES

1.1 INTRODUCTION

Fake news dissemination is very common in social networks. Due to the extensive social connections among users, fake news on certain topics, e.g., politics, celebrities and product promotions, can propagate and lead to a large number of nodes reporting the same (incorrect) observations rapidly in online social networks.
A type of yellow journalism, fake news encapsulates pieces of news that may be hoaxes and is generally spread through social media and other online media. This is often done to further or impose certain ideas and is often achieved with political agendas. Such news items may contain
false and/or exaggerated claims, and may end up being viralized by algorithms, and users may end
up in a filter bubble.and end points are extracted.

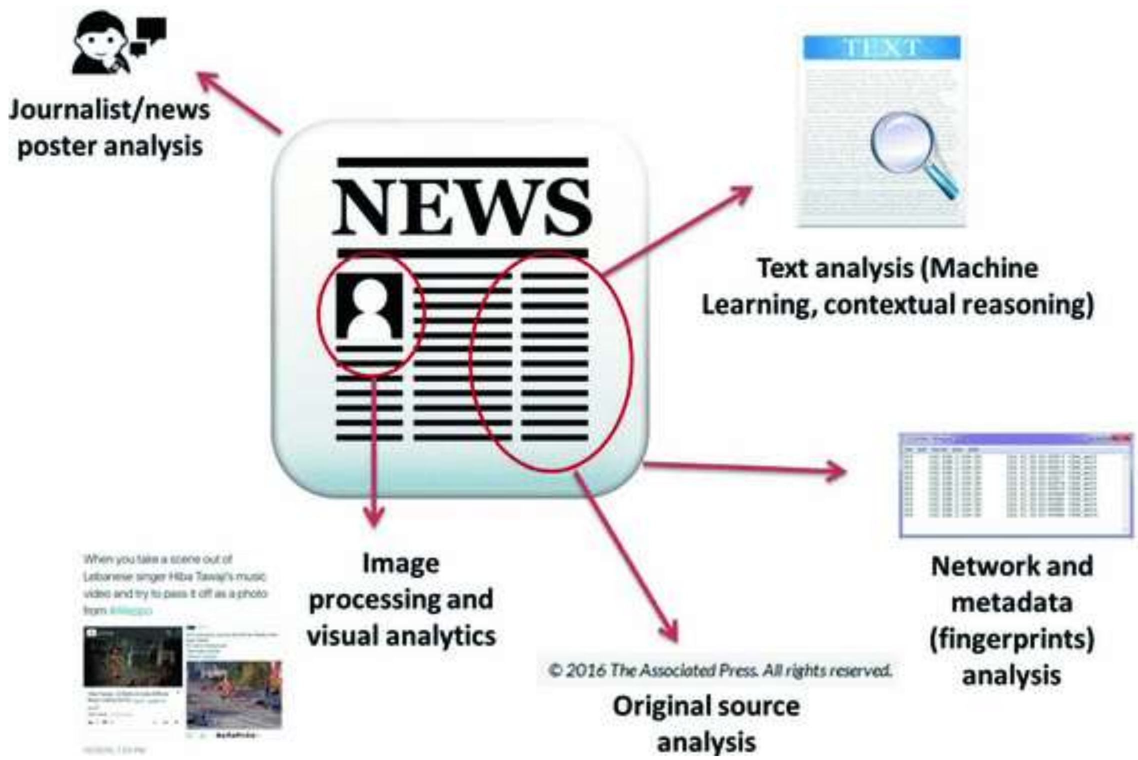
Fake news denotes a type of yellow press which intentionally presents misinformation or hoaxes spreading through both traditional print news media and recent online social media. Fake news has been existing for a long time, since the "Great moon hoax" published in 1835. In recent years, due to the booming developments of online social networks, fake news for various commercial and political purposes has been appearing in large numbers and widespread in the online world. With deceptive words, online social network users can get infected by these online fake news easily, which has brought about tremendous effects on the offline society already. During the 2016 US president election, various kinds of fake news about the candidates widely spread in the online social networks, which may have a significant effect on the election results.

According to a post-election statistical report, online social networks account for more than 41.8% of the fake news data traffic in the election, which is much greater than the data traffic shares of both traditional TV/radio/print medium and online search engines respectively. An important goal in improving the trustworthiness of information in online social networks is to identify the fake news timely, which will be the main tasks studied in this paper. Fake news has significant differences compared with traditional suspicious information, like spams, in various aspects: (1) impact on society: spams usually exist in personal emails or specific review websites and merely have a local impact on a small number of audiences, while the impact fake news in online social networks can be tremendous due to the massive user numbers globally, which is further boosted by the extensive information sharing and propagation among these users; (2) audiences' initiative: instead of receiving spam emails passively, users in online social networks may seek for, receive and share news information actively with no sense about its correctness; and (3) identification difficulty: via comparisons with abundant regular messages (in emails or review websites), spams are usually easier to be distinguished; meanwhile, identifying fake news with erroneous information is incredibly challenging, since it requires both tedious evidence-collecting and careful factchecking due to the lack of other comparative news articles available.

1.2 PROBLEM STATEMENT

The main objective of project is to carry out fake news detection of online available news material (images) by building a model that can classify images into Fake, Somewhat Fake, Somewhat Real and Real categories with high accuracy.
A CNN based approached is used to detect any fakeness in the news images by finding any hidden distortion while a LSTM based approach is used to find a latent pattern in the news text body so that the text can be classified as the Fake or Real news instance. Finally both of the approaches can be combined to get a highly accurate news instance detector.

Figure 1



2. DATASETS

There are two datasets used till now in this project:-

2.1. Image dataset : https://minyoungg.github.io/selfconsistency/in_wild/in_wild.tar.gz

This dataset contains 201 images of different news collection most of them (around 153) being fake.

This dataset has been divided into two separate folder named train set and test set which are then further subdivided into two given classes of label “Fake images” and “Real Images”.

2.1 Text Dataset : https://drive.google.com/open?id=0B3e3qZpPtccsMFo5bk9Ib3VCc2c

This dataset contains 54 columns of different fake news collection taken from kaggle site in the all_data.csv format. Though it contains 25,824 instances of news but most of them have missing data in different features, so the data is preprocessed manually in order to reatain only those instance which have all values of necessary feature, by which the number of useful instances turn out to be 16,508 out of which ~7200 instances are having “Real” label and ~9200 instances have “Fake” news label.

The important features which can be used for the classification of the fake news are:

(I) **Title** : This column of csv contains the title related to each news instance in small or large sentences.

(ii) **Text** : This column contains the body text related to the news instance. It may or may not be related to the context of title of news. Usually the length of text in “Real” news is greater than the “Fake” news one. “Fake” news contains many frequent words like, “Email”, “Tweet”, “Print”, “Source”, “Halloween” etc. and are mostly exaggerations while the “Real” News contains only large sentence out of which most of them are grammatically correct.

(iii) **Type** : This column contains label to each news instance either “Fake” or “Real”.

(iv) **Main_img** : The image related to the news instance given in the form of URL.

(v) **Comments** : The no. of comments in each news post.

(vi) **Likes** : The no. of like in each news post.

(vii) **Site_url** : The url of website from which each news instance is taken from.

(viii) **Replies** : The no. of replies related to each news post.

(ix) **Shares** : The no. of shares users have done to each news post.

(x) **Spam Scores** : This columns an integral number corresponds to the "spamming level" of each news post.

(xi) **Sentimental scores** : There are 8 different columns which contains weighted numbers for each of the 8 sentiments namely anger, disgust, fear, joy, anticipate, sadness, surprise and trust.

2.3 Dataset Preprocessing

Both of the dataset is splited into two sets one for training the classifier model and the other for validation purposes in the ratio 80:20.

1. In Image dataset both parts contain 4 categories of sub folder of images for completely fake, somewhat fake, somewhat real and completely real. The images are manually assigned to each folder.

2. In Text dataset all_data.csv a subset is taken which contains only news text and news type and named text_data.csv in order to retrieve the latent hidden sequential pattern in the text body so that the LSTM method can do the pattern classification on the dataset by its own.

3. CONVULATION NEURAL NETWORK

Artificial Intelligence has been witnessing a monumental growth in bridging the gap between the capabilities of humans and machines. Researchers and enthusiasts alike, work on numerous aspects of the field to make amazing things happen. One of many such areas is the domain of Computer Vision.

The agenda for this field is to enable machines to view the world as humans do, perceive it in a similar manner and even use the knowledge for a multitude of tasks such as Image & Video recognition, Image Analysis & Classification, Media Recreation, Recommendation Systems, Natural Language Processing, etc. The advancements in Computer Vision with Deep Learning has been constructed and perfected with time, primarily over one particular algorithm — a **Convolutional Neural Network**.

3.1 Introduction

A **Convolutional Neural Network (ConvNet/CNN)** is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics.

The architecture of a ConvNet is analogous to that of the connectivity pattern of Neurons in the Human Brain and was inspired by the organization of the Visual Cortex. Individual neurons respond to stimuli only in a restricted region of the visual field known as the Receptive Field. A collection of such fields overlap to cover the entire visual area.

Why ConvNets over Feed-Forward Neural Nets?

An image is nothing but a matrix of pixel values, right? So why not just flatten the image (e.g. 3x3 image matrix into a 9x1 vector) and feed it to a Multi-Level Perceptron for classification purposes? Uh.. not really.

In cases of extremely basic binary images, the method might show an average precision score while performing prediction of classes but would have little to no accuracy when it comes to complex images having pixel dependencies throughout.

A ConvNet is able to **successfully capture the Spatial and Temporal dependencies** in an image through the application of relevant filters. The architecture performs a better fitting to the image dataset due to the reduction in the number of parameters involved and reusability of weights. In other words, the network can be trained to understand the sophistication of the image better.

Input Image

In the figure, we have an RGB image which has been separated by its three color planes — Red, Green, and Blue. There are a number of such color spaces in which images exist — Grayscale, RGB, HSV, CMYK, etc.

You can imagine how computationally intensive things would get once the images reach dimensions, say 8K (7680×4320). . This is important when we are to design an architecture which is not only good at learning features but also is scalable to massive datasets.

The objective of the Convolution Operation is to **extract the high-level features** such as edges, from the input image. ConvNets need not be limited to only one Convolutional Layer. Conventionally, the first ConvLayer is responsible for capturing the Low-Level features such as edges, color, gradient orientation, etc. With added layers, the architecture adapts to the High-

Level features as well, giving us a network which has the wholesome understanding of images in the dataset, similar to how we would.

There are two types of results to the operation — one in which the convolved feature is reduced in dimensionality as compared to the input, and the other in which the dimensionality is either increased or remains the same. This is done by applying **Valid Padding** in case of the former, or **Same Padding** in the case of the latter.

3.2. Fake Image Detection (by CNN):

1. Model :

A CNN model is built with 2 convoluted layers which takes the training images as input and produces a flattened one dimensional feature vector which is then allowed to fully connect through an ANN of 1 hidden layer.

2. Model compilation:
The whole model is then compiled with the stochastic gradient descent algorithm and cross entropy loss function.

3. Improvement in performance :
The training is improved by using k-fold cross validation with suitable value of k so that the variance can be lower down. The bias can be decreased by hyperparameter tuning using GridSearch() method from keras library.

4. Prediction :
The model is tested for the given test set to predict the categories of images.

5. Result:
The accuracy and the standard variation is used as the performance measure for the model.
The k-fold cross validation function finds the average accuracy for all of the iteration in each epoch
and hence give the final mean accuracy and standard variance.

There are various architectures of CNNs available which have been key in building algorithms which power and shall power AI as a whole in the foreseeable future. Some of them have been listed below:

1.LeNet

2.AlexNet

3.VGGNet

4.GoogLeNet

5.ResNet

6.ZFNet

4. RECURRENT NEURAL NETWORK

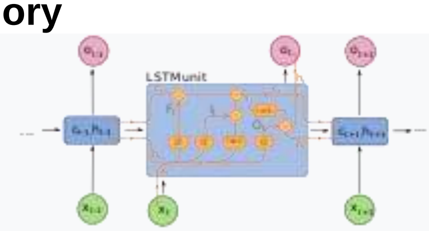
3.1Introduction

A **recurrent neural network** (**RNN**) is a class of ANN where connections between nodes form a directed graph along a temporal sequence. This allows it to exhibit temporal dynamic behavior. Unlike feedforward neural networks, RNNs can use their internal state (memory) to process sequences of inputs. This makes them applicable to tasks such as unsegmented, connected handwriting recognition[1] or speech recognition.

The term "recurrent neural network" is used indiscriminately to refer to two broad classes of networks with a similar general structure, where one is finite impulse and the other is infinite impulse. Both classes of networks exhibit temporal dynamic behavior.[4] A finite impulse recurrent network is a directed acyclic graph that can be unrolled and replaced with a strictly feedforward neural network, while an infinite impulse recurrent network is a directed cyclic graph that can not be unrolled.

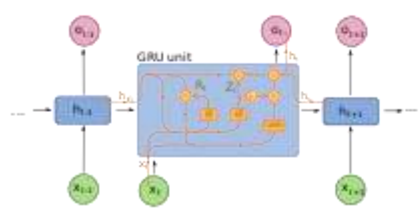
Variants of RNN

Long short-term memory



Long short-term memory (LSTM) is a deep learning system that avoids the vanishing gradient problem. LSTM is normally augmented by recurrent gates called "forget" gates LSTM prevents backpropagated errors from vanishing or exploding. Instead, errors can flow backwards through unlimited numbers of virtual layers unfolded in space

Gated recurrent unit



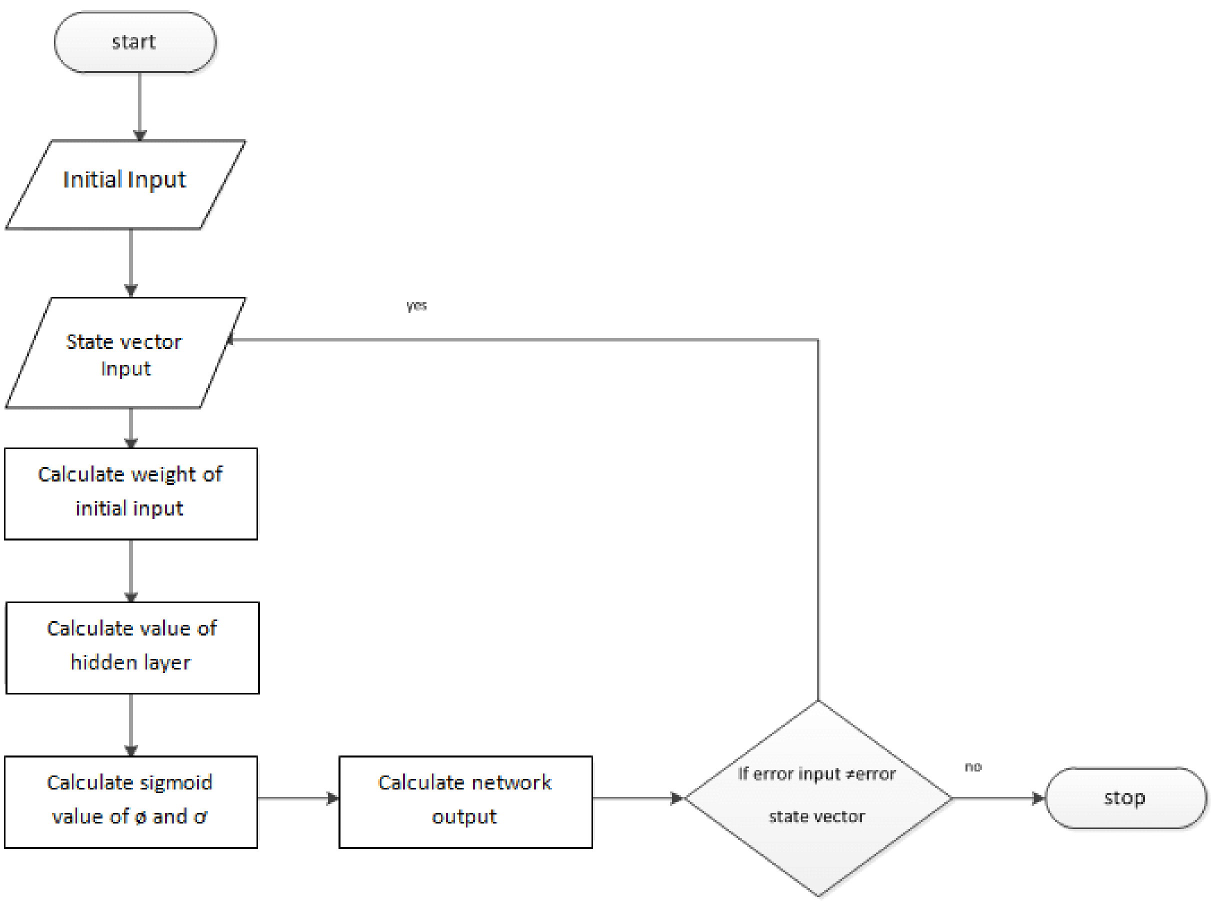
Gated recurrent units (GRUs) are a gating mechanism in recurrent neural networks introduced in 2014. They are used in the full form and several simplified variants. Their performance on polyphonic music modeling and speech signal modeling was found to be similar to that of long short-term memory.[45] They have fewer parameters than LSTM, as they lack an output gate.

Bi-directional

Bi-directional RNNs use a finite sequence to predict or label each element of the sequence based on the element's past and future contexts. This is done by concatenating the outputs of two RNNs, one processing the sequence from left to right, the other one from right to left. The combined outputs are the predictions of the teacher-given target signals. This technique proved to be especially useful when combined with LSTM RNNs

3.2 Flow Chart

figure 2. flow chart



FULLY RECURRENT

Basic RNNs are a network of neuron-like nodes organized into successive "layers." Each node in a given layer is connected with a directed (one-way) connection to every other node in the next successive layer.[*citation needed*] Each node (neuron) has a time-varying real-valued activation. Each connection (synapse) has a modifiable real-valued weight. Nodes are either input nodes (receiving data from outside the network), output nodes (yielding results), or hidden nodes (that modify the data *en route* from input to output).

For supervised learning in discrete time settings, sequences of real-valued input vectors arrive at the input nodes, one vector at a time. At any given time step, each non-input unit computes its current activation (result) as a nonlinear function of the weighted sum of the activations of all units that connect to it. Supervisor-given target activations can be supplied for some output units at certain time steps. For example, if the input sequence is a speech signal corresponding to a spoken digit, the final target output at the end of the sequence may be a label classifying the digit.

In reinforcement learning settings, no teacher provides target signals. Instead a fitness function or reward function is occasionally used to evaluate the RNN's performance, which influences its input stream through output units connected to actuators that affect the environment. This might be used to play a game in which progress is measured with the number of points won.

Each sequence produces an error as the sum of the deviations of all target signals from the corresponding activations computed by the network. For a training set of numerous sequences, the total error is the sum of the errors of all individual sequences.

4.3 LSTM

Recurrent networks can in principle use their feedback connections to store representations of recent input events in form of activations (\short-term memory", as opposed to \long-term memory" embodied by slowly changing weights). This is potentially signicant for many applications, including speech processing, non-Markovian control, and music composition (e.g., Mozer 1992). The most widely used algorithms for learning what to put in short-term memory, however, take too much time or do not work well at all, especially when minimal time lags between inputs and corresponding teacher signals are long. Although theoretically fascinating, existing methods do not provide clear practical advantages over, say, backprop in feedforward nets with limited time windows. This paper will review an analysis of the problem and suggest a remedy. The problem. With conventional \Back-Propagation Through Time" (BPTT, e.g., Williams and Zipser 1992, Werbos 1988) or \Real-Time Recurrent Learning" (RTRL, e.g., Robinson and Fallside 1987), error signals \owing backwards in time" tend to either (1) blow up or (2) vanish: the temporal evolution of the backpropagated error exponentially depends on the size of the weights (Hochreiter 1991). Case (1) may lead to oscillating weights, while in case (2) learning to bridge long time lags takes a prohibitive amount of time, or does not work at all (see section 3). The remedy. This paper presents \Long Short-Term Memory" (LSTM), a novel recurrent network architecture in conjunction with an appropriate gradient-based learning algorithm. LSTM is designed to overcome these error back-ow problems. It can learn to bridge time intervals in excess of 1000 steps even in case of noisy, incompressible input sequences, without loss of short time lag capabilities. This is achieved by an ecient, gradient-based algorithm for an architecture 1 enforcing constant (thus neither exploding nor vanishing) error ow through internal states of special units (provided the gradient computation is truncated at certain architecture-specic points | this does not a ect long-term error ow though).

4.4 Different Gates of LSTM

Forget gate

First, we have the forget gate. This gate decides what information should be thrown away or kept. Information from the previous hidden state and information from the current input is passed through the sigmoid function. Values come out between 0 and 1. The closer to 0 means to forget, and the closer to 1 means to keep.

Input Gate

To update the cell state, we have the input gate. First, we pass the previous hidden state and current input into a sigmoid function. That decides which values will be updated by transforming the values to be between 0 and 1. 0 means not important, and 1 means important. You also pass the hidden state and current input into the tanh function to squish values between -1 and 1 to help regulate the network. Then you multiply the tanh output with the sigmoid output. The sigmoid output will decide which information is important to keep from the tanh output.

Cell State

Now we should have enough information to calculate the cell state. First, the cell state gets pointwise multiplied by the forget vector. This has a possibility of dropping values in the cell state if it gets multiplied by values near 0. Then we take the output from the input gate and do a pointwise addition which updates the cell state to new values that the neural network finds relevant. That gives us our new cell state.

Output Gate

Last we have the output gate. The output gate decides what the next hidden state should be. Remember that the hidden state contains information on previous inputs. The hidden state is also used for predictions. First, we pass the previous hidden state and the current input into a sigmoid function. Then we pass the newly modified cell state to the tanh function. We multiply the tanh output with the sigmoid output to decide what information the hidden state should carry. The output is the hidden state. The new cell state and the new hidden is then carried over to the next time step.

4.5 Application of LSTM

LSTM networks have been used successfully in the following tasks
1.Language modeling (The tensorflow tutorial on PTB is a good place to start Recurrent Neural Networks) character and word level LSTM's are used

2.Machine Translation also known as sequence to sequence learning (https://arxiv.org/pdf/1409.3215.pdf)

3.Image captioning (with and without attention, https://arxiv.org/pdf/1411.4555v…)

4.Hand writing generation (http://arxiv.org/pdf/1308.0850v5…)

5.Image generation using attention models - my favorite (https://arxiv.org/pdf/1502.04623…)

6.Question answering (http://www.aclweb.org/anthology/…)

7.Video to text (https://arxiv.org/pdf/1505.00487...)

4.6 Advantages of LSTM

1.Neural Networks is a very powerful technique and is used for image recognition and many other applications. One of the limitation is that, there is no memory associated with the model. Which is a problem for sequential data, like text or time series.

2.RNN addresses that issue by including a feedback look which serves as a kind of memory. So the past inputs to the model leave a footprint. LSTM extends that idea and by creating both a short-term and a long-term memory component.

3.Hence, LSTM is great tool for anything that has a sequence. Since the meaning of a word depends on the ones that preceded it. This paved the way for NLP and narrative analysis to leverage Neural Networks.

4.LSTM can be used for text generation. You can train the model on the text of a writer, say, and the model will be able to generate new sentences that mimics the style and interests of the writer.

5.Sequence-to-Sequence LSTM models are the state of the technique for translations. They also have a wide array of applications like time series forecasting.

5. OUTPUT AND VISUALISATION OF RESULTS

Often when I talk to organizations that are looking to implement data science into their processes, they often ask the question, “How do I get the most accurate model?”. And I asked further, “What business challenge are you trying to solve using the model?” and I will get the puzzling look because the question that I posed does not really answer their question. I will then need to explain why I asked the question before we start exploring if Accuracy is the be-all and end-all model metric that we shall choose our “best” model from.

5.1 Confusion Matrix

Firstly, let us look at the following confusion matrix. What is the accuracy for the model?

		Predicted/Classified	
		Negative	Positive
Actual	Negative	998	0
	Positive	1	1

Figure 3. confusion matrix

Very easily, you will notice that the accuracy for this model is very very high, at 99.9%!! Wow! You have hit the jackpot and holy grail (*scream and run around the room, pumping the fist in the air several times*)!

But….(well you know this is coming right?) what if I mentioned that the positive over here is actually someone who is sick and carrying a virus that can spread very quickly? Or the positive here represent a fraud case? Or the positive here represents terrorist that the model says its a non-terrorist? Well you get the idea. The costs of having a mis-classified actual positive (or false negative) is very high here in these three circumstances that I posed.

OK, so now you realized that accuracy is not the be-all and end-all model metric to use when selecting the best model…now what?

5.2 Precision and Recall

Let me introduce two new metrics (if you have not heard about it and if you do, perhaps just humor me a bit and continue reading? :D)

So if you look at Wikipedia, you will see that the the formula for calculating Precision and Recall is as follows:

$$\text{Precision} = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

$$\text{Recall} = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

Let me put it here for further explanation.

Let me put in the confusion matrix and its parts here.

Precision

Great! Now let us look at Precision first.

What do you notice for the denominator? The denominator is actually the Total Predicted Positive! So the formula becomes

Immediately, you can see that Precision talks about how precise/accurate your model is out of those predicted positive, how many of them are actual positive.

Precision is a good measure to determine, when the costs of False Positive is high. For instance, email spam detection. In email spam detection, a false positive means that an email that is non-spam (actual negative) has been identified as spam (predicted spam). The email user might lose important emails if the precision is not high for the spam detection model.

		Predicted	
		Negative	**Positive**
Actual	**Negative**	True Negative	False Positive
	Positive	False Negative	True Positive

Recall

So let us apply the same logic for Recall. Recall how Recall is calculated.

There you go! So Recall actually calculates how many of the Actual Positives our model capture through labeling it as Positive (True Positive). Applying the same understanding, we know that

Recall shall be the model metric we use to select our best model when there is a high cost associated with False Negative.

For instance, in fraud detection or sick patient detection. If a fraudulent transaction (Actual Positive) is predicted as non-fraudulent (Predicted Negative), the consequence can be very bad for the bank.

Similarly, in sick patient detection. If a sick patient (Actual Positive) goes through the test and predicted as not sick (Predicted Negative). The cost associated with False Negative will be extremely high if the sickness is contagious.

F1 Score

Now if you read a lot of other literature on Precision and Recall, you cannot avoid the other measure, F1 which is a function of Precision and Recall. Looking at Wikipedia, the formula is as follows:

F1 Score is needed when you want to seek a balance between Precision and Recall. Right…so what is the difference between F1 Score and Accuracy then? We have previously seen that accuracy can be largely contributed by a large number of True Negatives which in most business circumstances, we do not focus on much whereas False Negative and False Positive usually has business costs (tangible & intangible) thus

I hope the explanation will help those starting out on Data Science and working on Classification problems, that Accuracy will not always be the metric to select the best model from.

I wish all readers a FUN Data Science learning journey and if you have liked this blog posts, do give a clap or two. Please visit my other blog posts and LinkedIn profile.

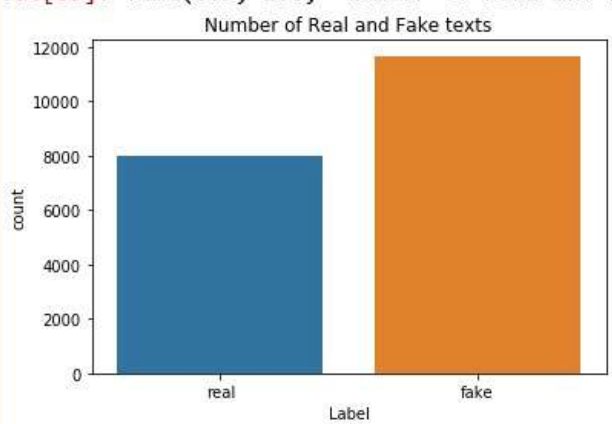
CNN on image dataset:

Validation Accuracy: 81.25%

```
IPython console
Console 1/A    Console 3/A
val_loss: 0.0108 - val_accuracy: 0.8750
Epoch 9/15
153/153 [==============================] - 28s 183ms/step - loss: 0.1106 - accuracy: 0.9567 -
val_loss: 4.1981e-04 - val_accuracy: 0.8542
Epoch 10/15
153/153 [==============================] - 27s 176ms/step - loss: 0.0455 - accuracy: 0.9818 -
val_loss: 3.6087 - val_accuracy: 0.8750
Epoch 11/15
153/153 [==============================] - 27s 179ms/step - loss: 0.0250 - accuracy: 0.9917 -
val_loss: 0.3322 - val_accuracy: 0.8542
Epoch 12/15
153/153 [==============================] - 27s 178ms/step - loss: 0.0804 - accuracy: 0.9800 -
val_loss: 2.0896e-04 - val_accuracy: 0.8542
Epoch 13/15
153/153 [==============================] - 28s 181ms/step - loss: 0.0316 - accuracy: 0.9917 -
val_loss: 2.6913 - val_accuracy: 0.8958
Epoch 14/15
153/153 [==============================] - 27s 178ms/step - loss: 0.0659 - accuracy: 0.9783 -
val_loss: 1.4217 - val_accuracy: 0.8750
Epoch 15/15
153/153 [==============================] - 29s 191ms/step - loss: 0.0364 - accuracy: 0.9833 -
val_loss: 4.4354 - val_accuracy: 0.8125
Out[12]: <keras.callbacks.callbacks.History at 0x20dc02bb108>
```

Text dataset:

```
Out[80]: Text(0.5, 1.0, 'Number of Real and Fake texts')
```



Validation Accuracy on Text: 92.38%

```
IPython console
Console 1/A    Console 3/A
Epoch 14/20
12599/12599 [==============================] - 55s 4ms/step - loss: 0.1002 - accuracy: 0.9644 -
val_loss: 0.1923 - val_accuracy: 0.9356
Epoch 15/20
12599/12599 [==============================] - 55s 4ms/step - loss: 0.0947 - accuracy: 0.9662 -
val_loss: 0.2047 - val_accuracy: 0.9375
Epoch 16/20
12599/12599 [==============================] - 55s 4ms/step - loss: 0.0892 - accuracy: 0.9680 -
val_loss: 0.2801 - val_accuracy: 0.9324
Epoch 17/20
12599/12599 [==============================] - 55s 4ms/step - loss: 0.1171 - accuracy: 0.9566 -
val_loss: 0.2511 - val_accuracy: 0.9063
Epoch 18/20
12599/12599 [==============================] - 56s 4ms/step - loss: 0.1181 - accuracy: 0.9548 -
val_loss: 0.2183 - val_accuracy: 0.9352
Epoch 19/20
12599/12599 [==============================] - 55s 4ms/step - loss: 0.0798 - accuracy: 0.9726 -
val_loss: 0.2217 - val_accuracy: 0.9305
Epoch 20/20
12599/12599 [==============================] - 55s 4ms/step - loss: 0.0828 - accuracy: 0.9715 -
val_loss: 0.2346 - val_accuracy: 0.9238
Out[90]: <keras.callbacks.callbacks.History at 0x1be24d1c748>
```

Validation Accuracy on Title: 83.56%

```
IPython console
Console 1/A    Console 3/A
val_accuracy: 0.8232
Epoch 46/50
12599/12599 [==============================] - 56s 4ms/step - loss: 0.0324 - accuracy: 0.9864 - val_loss: 1.7742 -
val_accuracy: 0.8324
Epoch 47/50
12599/12599 [==============================] - 57s 4ms/step - loss: 0.0286 - accuracy: 0.9887 - val_loss: 1.7955 -
val_accuracy: 0.8270
Epoch 48/50
12599/12599 [==============================] - 56s 4ms/step - loss: 0.0308 - accuracy: 0.9877 - val_loss: 1.8353 -
val_accuracy: 0.8267
Epoch 49/50
12599/12599 [==============================] - 54s 4ms/step - loss: 0.0262 - accuracy: 0.9890 - val_loss: 1.9711 -
val_accuracy: 0.8263
Epoch 50/50
12599/12599 [==============================] - 58s 5ms/step - loss: 0.0301 - accuracy: 0.9887 - val_loss: 2.0110 -
val_accuracy: 0.8356
Out[91]: <keras.callbacks.callbacks.History at 0x1be24d4e648>
```


6. CONCLUSION

- We have concluded that for contrast enhancement of the image CLAHE(Adaptive Histogram Eqaulization) is best.
- Boundingbox and imcrops function can be used to automatically detect the region of interest of the hand dorsal vein pattern.
- End points and Bifurcate points can be used for authorization of a person as these points represents the identification points.

FUTURE SCOPE

- To Integrate the results from text, title and images to perform a single classification using the features of all 3.

- The concept of short term memory can be included in the fake image detection to improve the performance. As only using CNN approach will be not as good as combination of convolution and LSTM.

- 2. Different other parameters in the Text dataset can too be involved in order to increase the performance of the Text classifier.

- 3. The most important future can be that we can build the clasffier which combinely classify fake images and text so that our news detector can have more use cases.

7.REFERENCES

1. Yang Yang, Lei Zheng "TI-CNN: Convolutional Neural Networks for Fake News Detection"
2. Y-Lan Boureau, Jean Ponce, and Yann LeCun. A theoretical analysis of feature pooling in visual recognition. In Proceedings of the 27th international conference on machine learning (ICML-10),
3. Automatic Detection and Verification of Rumors on Twitter by Soroush Vosoughi
4.Keras Documentation.