

k 1 fake news

Word embedding is the most popular representation of document vocabulary. It is capable of capturing context of a word in a document, semantic and syntactic similarity, relation with other words, etc.

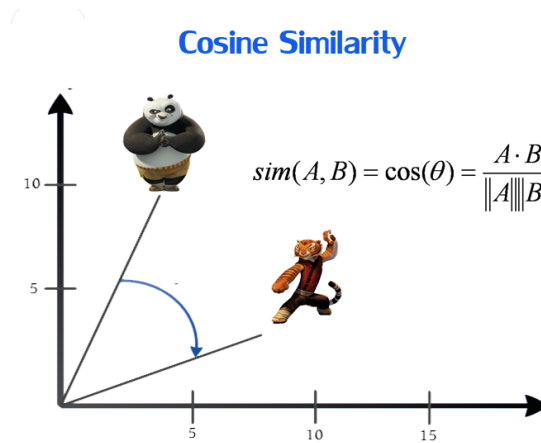
Cosine similarity is the cosine of the angle between two normalized vectors. The idea is that the length of the vectors in question should not influence the similarity measure.

Tokenizing Text Representing each word by a number

mapping of original word to number is preserved in word_index property of tokenizer. Tokenized applies basic processing like changing it

lower case, explicitly setting that as False

labeling the fake data
labeling the Real data



with zeros and
with ones and

concatenating both of them

```
frames = [Fake_df, True_df]
Data_df = pd.concat(frames)
Data_df
```

	text	subject	CLASS
0	donald trump just couldn t wish all americans ...	News	0
1	house intelligence committee chairman devin nu...	News	0
2	on friday it was revealed that former milwauke...	News	0
3	on christmas day donald trump announced that h...	News	0
4	pope francis used his annual christmas day mes...	News	0
...
21412	brussels reuters nato allies on tuesday welcom...	worldnews	1
21413	london reuters lexisnexis a provider of legal ...	worldnews	1
21414	minsk reuters in the shadow of disused soviete...	worldnews	1
21415	moscow reuters vatican secretary of state card...	worldnews	1
21416	jakarta reuters indonesia will buy 11 sukhoi f...	worldnews	1

44898 rows × 3 columns

```
def cleanDF(df):
    df.drop(['date', 'title'], axis=1, inplace=True)
    df.text = df.text.str.lower()
    df.text = df.text.str.replace(r'http[\w:/\.\.]+\>', '<URL>') # remove urls
    df.text = df.text.str.replace(r'^\.\w\s', '') #remove everything but characters and punctuation
    df.text = df.text.str.replace(r'\.\.\+', '.') #replace multiple periods with a single one
    df.text = df.text.str.replace(r'\.', ' ') #replace multiple periods with a single one
    df.text = df.text.str.replace(r'\s\s+', ' ') #replace multiple white space with a single one
    df.text = df.text.str.strip()
    print(df.shape)
    df.head()
    return df
```

This function from the preprocessing step as we remove all URLs , characters punctuation To clean the text .

```

y = Data_df["CLASS"].values
#Converting X to format acceptable by gensim, removing annd punctuation stopwords in the process
X = []
stop_words = set(nltk.corpus.stopwords.words("english"))
tokenizer = nltk.tokenize.RegexpTokenizer(r'\w+')
for par in Data_df["text"].values:
    tmp = []
    sentences = nltk.sent_tokenize(par)
    for sent in sentences:
        sent = sent.lower()
        tokens = tokenizer.tokenize(sent)
        filtered_words = [w.strip() for w in tokens if w not in stop_words and len(w) > 1]
        tmp.extend(filtered_words)
    X.append(tmp)

del Data_df

```

This step for removing the stop words set , Stop Words: A stop word is a commonly used word (such as "the", "a", "an", "in") that a search engine has been programmed to ignore, both when indexing entries for searching and when retrieving them as the result of a search query.

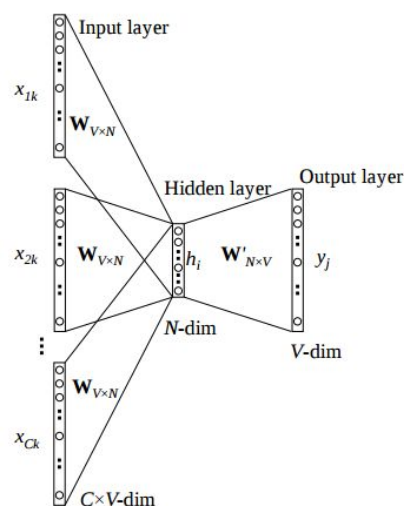
Weights step

W : Input-hidden layer matrix

W' : Hidden-output layer matrix

Semantics encoder from n-hot vector: Word-list to semantics

Semantics decoder which outputs a probability vector: Semantics to probability distribution over words.



Network part : LSTM

These Vectors will be passed to LSTM/GRU instead of words. 1D-CNN can further be used to extract features from the vectors.

Keras has implementation called "Embedding Layer" which would create word embeddings(vectors). Since we did that with gensim's word2vec, we will load these vectors into embedding layer and make the layer non-trainable.

```
max_nb_words: maximum number of words in corpus
```

```
embedding_dim: the size of the embedding dimension
```

The *Embedding* has a vocabulary a . We will choose a small embedding space of 8 dimensions.

The model is a simple binary classification model. Importantly, the output from the *Embedding* layer We flatten this to a one 32-element vector to pass on to the *Dense* output layer.

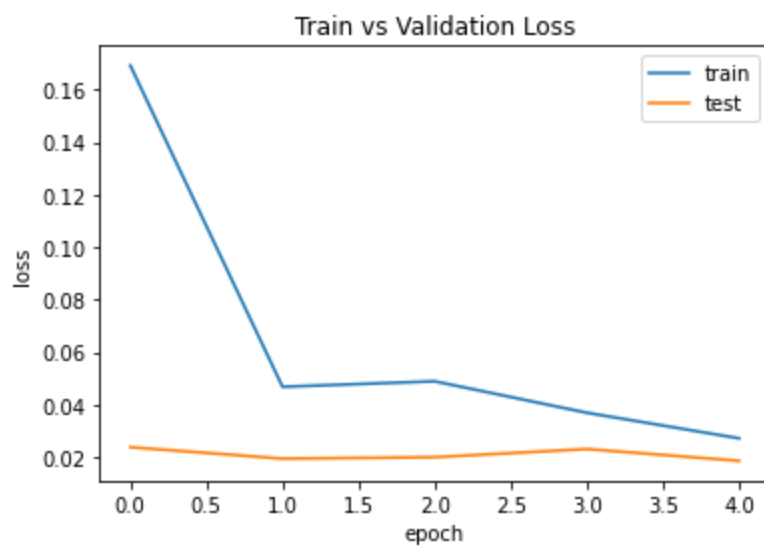
```
from keras.layers import Dense, Embedding, LSTM, SpatialDropout1D
MAX_NB_WORDS = 10000
EMBEDDING_DIM = 3000
```

```
from keras.regularizers import l2
from keras import regularizers
```

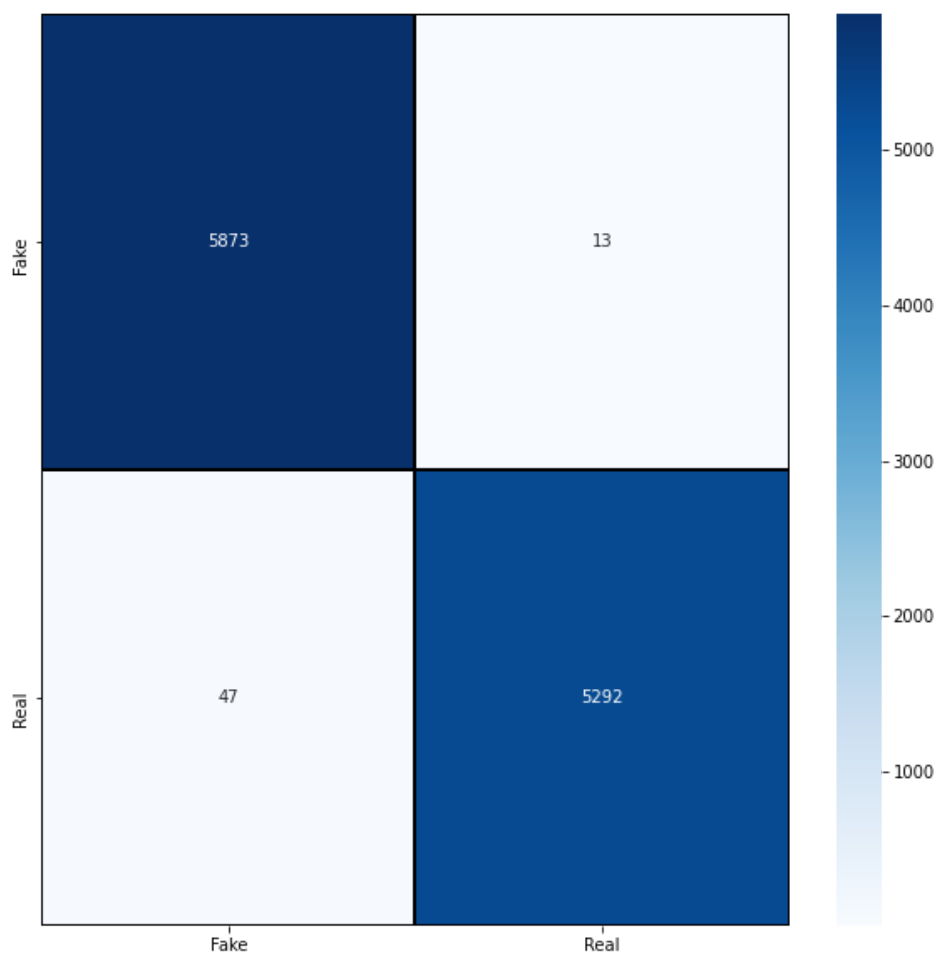
```
model = Sequential()
model.add(Embedding(MAX_NB_WORDS, EMBEDDING_DIM, input_length=X.shape[1]))
#Defining Neural Network
#Non-trainable embedding Layer
#LSTM
model.add(LSTM(units=128, return_sequences = True, recurrent_dropout = 0.25, dropout = 0.5))
model.add(LSTM(units=64, recurrent_dropout = 0.2, dropout = 0.5))
model.add(Dense(units = 32, activation = 'relu'))
model.add(Dense(1, activation='sigmoid'))
model.layers[0].trainable = False
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
history_lstm = model.fit(X_train, y_train, epochs=5, batch_size=128, workers=10, validation_data=(X_test, y_test))
```

Results

```
33673/33673 [=====] - 361s 11ms/step
Accuracy of the model on Training Data is - 99.37041401863098
11225/11225 [=====] - 120s 11ms/step
Accuracy of the model on Testing Data is - 99.46547746658325
```



Confusion matrix



Digit recognizer .

<https://towardsdatascience.com/a-guide-to-an-efficient-way-to-build-neural-network-architectures-part-ii-hyper-parameter-42efca01e5d7>

Improved Handwritten Digit Recognition Using Convolutional Neural Networks (CNN)

In the ensemble method, it is necessary that the correlation between multiple learner is low (or the independence of multiple learner is high). In order to obtain various classifiers with high independence, it is necessary to use a significantly different algorithm. With 3 ensemble

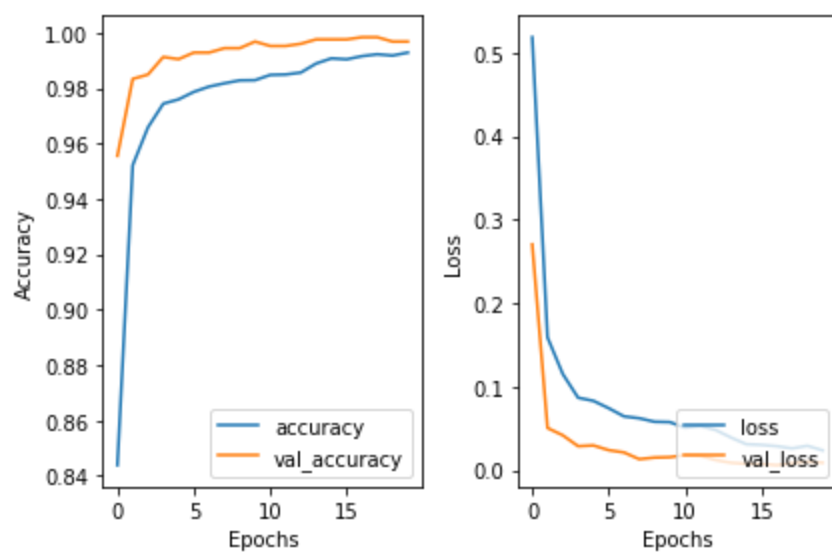
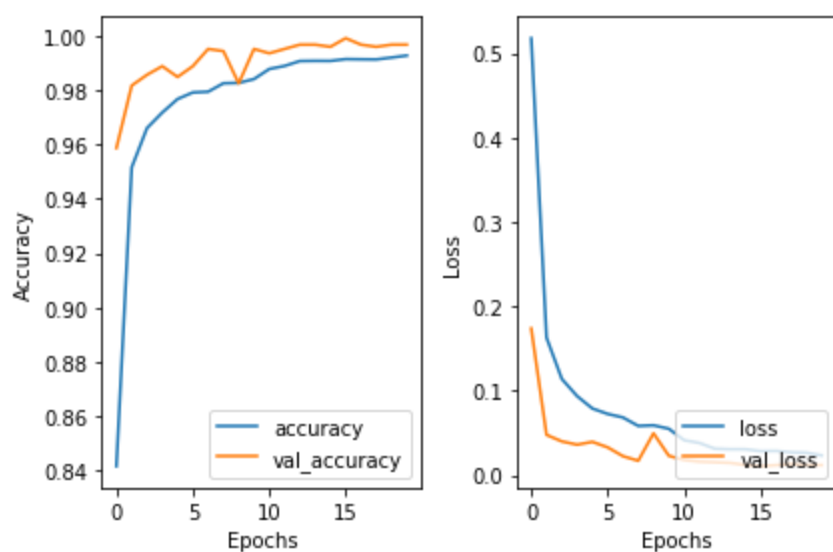
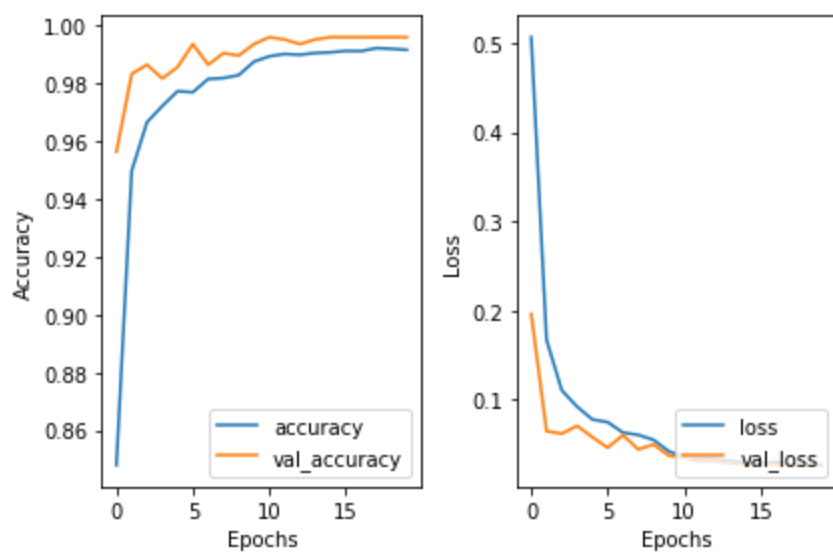
Results

1274/1274 [=====] - 47s 37ms/step - loss: 0.0084 - accuracy: 0.9976

Accuracy of the model on Training Data is - 99.76190328598022

40/40 [=====] - 1s 35ms/step - loss: 0.0089 - accuracy: 0.9968

Accuracy of the model on Testing Data is - 99.68253970146179



Task 3

1-If x is not a number or if base is given, then x must be a string, bytes, or bytearray instance representing an integer literal in radix base

The literal can be preceded by + or - (with no space in between) and surrounded by whitespace

A base-n literal consists of the digits 0 to n-1, with a to z (or A to Z) having values 10 to 35. The default base is 10.

The allowed values are 0 and 2-36. Base-2, -8, and -16 literals can be optionally prefixed with 0b/0B, 0o/0O, or 0x/0X, as with integer literals in code.


2-Use `timeit()` function to measure the running time.

```
3-info_list = [case.split(',') for case in info.split(' ')]
```

```
print(sorted(info_list))
```

Splitting & Sorting

4- using `collections.Counter()` to get count of each element in string



5-If the given string already ends with 'ing' then add 'ly' instead. If the string length of the given string is less than 3, leave it unchanged.

6-`import` datetime

The other Questions are solved