**Text Classification:**

**Data**1. we have total of 20 types of documents(Text files) and total 18828 documents(text files).

2. You can download data from this link (https://drive.google.com/open?id=1rxD15nyeIPIAZ-J2VYPrDRZI66-T BWvM), in that you will get documents.rar folder. If you unzip that, you will get total of 18828 documn ets. document name is defined as 'ClassLabel\_DocumentNumberInThatLabel'. so from document name, you can extract the label for that document. 4. Now our problem is to classify all the documents into any one of the class. 5. Below we provided count plot of all the labels in our data.

In [0]:

<IPython.core.display.Javascript object>

*### count plot of all the class labels.* 1

**Assignment:**

**sample document**

Subject: A word of advice From: jcopelan@nyx.cs.du.edu (The One and Only)

In article < 65882@mimsy.umd.edu > mangoe@cs.umd.edu (Charley Wingate) writes: > >I've said 100 times that there is no "alternative" that should think you >might have caught on by now. And there is no "alternative", but the point >is, "rationality" isn't an alternative either. The problems of metaphysical >and religious knowledge are unsolvable-- or I should say, humans cannot >solve them.

How does that saying go: Those who say it can't be done shouldn't interrupt those who are doing it.

Jim -- Have you washed your brain today?

**Preprocessing:**

useful links: http://www.pyregex.com/ (http://www.pyregex.com/)

**1.** Find all emails in the document and then get the text after the "@". and then split those texts by

'.' . after that remove the words whose length is less than or equal to 2 and also remove'com' word. a nd then combine those words by space. In one doc, if we have 2 or more mails, get all. **Eg:[test@dm1.d.com, test2@dm2.dm3.com]-->[dm1.d.com, dm3.dm4.com]-->[dm1,d,com,dm2,dm3,com]-->[dm1,dm2, dm3]-->"dm1 dm2 dm3"** append all those into one list/array. ( This will give length of 18828 sentences i.e one list for each

of the document). Some sample output was shown below.

> In the above sample document there are emails [jcopelan@nyx.cs.du.edu, 65882@mimsy.umd.edu, mangoe@c s.umd.edu]

preprocessing: [jcopelan@nyx.cs.du.edu, 65882@mimsy.umd.edu, mangoe@cs.umd.edu] ==> [nyx cs du edu mimsy umd edu cs um d edu] ==> [nyx edu mimsy umd edu umd edu]

**2.** Replace all the emails by space in the original text.

In [0]:

Out[28]: In [0]:

**3.** Get subject of the text. i.e. get the total lines where "Subject:" occur. and the remove the word wh ich are before the ":". remove the newlines, tabs, punctuations, any special chars. **Eg: if we have sentance like "Subject: Re: Gospel Dating @ \r\r\n" --> You have to get "Gospel Dating"** 1*# we have collected all emails and preprocessed them, this is sample output* 2preprocessed\_email

array(['juliet caltech edu',

'coding bchs edu newsgate sps mot austlcm sps mot austlcm sps mot com dna bchs edu', 'batman bmd trw', ..., 'rbdc wsnc org dscomsa desy zeus desy', 'rbdc wsnc org morrow stanford edu pangea Stanford EDU', 'rbdc wsnc org apollo apollo'], dtype=object)

1

len(preprocessed\_email)

Out[29]: 18828

Save all this data into another list/array.

**4.** After you store it in the list, Replace those sentances in original text by space.

**5.** Delete all the sentances where sentence starts with **"Write to:"** or **"From:"**. > In the above sample document check the 2nd line, we should remove that

**6.** Delete all the tags like "< anyword >" > In the above sample document check the 4nd line, we should remove that "< 65882@mimsy.umd.edu >"

**7.** Delete all the data which are present in the brackets. In many text data, we observed that, they mai ntained the explanation of sentence or transolation of sentence to another language in brackets so remo ve all those. **Eg: "AAIC-The course that gets you HIRED(AAIC - Der Kurs, der Sie anstellt)" --> "AAIC-The course that**

**gets you HIRED"**

> In the above sample document check the 4nd line, we should remove that "(Charley Wingate)"

**8.** Remove all the newlines('\n'), tabs('\t'), "-", "\".

**9.** Remove all the words which ends with **":"**. **Eg: "Anyword:"** > In the above sample document check the 4nd line, we should remove that "writes:"

**10.** Decontractions, replace words like below to full words. please check the donors choose preprocessin g for this **Eg: can't -> can not, 's -> is, i've -> i have, i'm -> i am, you're -> you are, i'll --> i will**

**There is no order to do point 6 to 10. but you have to get final output correctly**

**11.** Do chunking on the text you have after above preprocessing. Text chunking, also referred to as shal low parsing, is a task that follows Part-Of-Speech Tagging and that adds more structure to the sentenc e. So it combines the some phrases, named entities into single word. So after that combine all those ph rases/named entities by separating **" "**. And remove the phrases/named entities if that is a "Person". Yo

ases/ a ed e t t es by sepa at g **\_** . d e o e t e p ases/ a ed e t t es t at s a e so . o u can use **nltk.ne\_chunk** to get these. Below we have given one example. please go through it.

useful links: https://www.nltk.org/book/ch07.html (https://www.nltk.org/book/ch07.html) https://stackoverflow.com/a/31837224/4084039 (https://stackoverflow.com/a/31837224/4084039) http://www.nltk.org/howto/tree.html (http://www.nltk.org/howto/tree.html) https://stackoverflow.com/a/44294377/4084039 (https://stackoverflow.com/a/44294377/4084039)

In [0]:

1*#i am living in the New York* 2print("i am living in the New York -->", list(chunks)) 3print(" ") 4print("-"**\***50) 5print(" ") 6*#My name is Srikanth Varma* 7

print("My name is Srikanth Varma -->", list(chunks1))

i am living in the New York --> [('i', 'NN'), ('am', 'VBP'), ('living', 'VBG'), ('in', 'IN'), ('the', 'DT'), T ree('GPE', [('New', 'NNP'), ('York', 'NNP')])]

--------------------------------------------------

My name is Srikanth Varma --> [('My', 'PRP$'), ('name', 'NN'), ('is', 'VBZ'), Tree('PERSON', [('Srikanth', 'NN P'), ('Varma', 'NNP')])]

We did chunking for above two lines and then We got one list where each word is mapped to a POS(parts o f speech) and also if you see "New York" and "Srikanth Varma", they got combined and represented as a t ree and "New York" was referred as "GPE" and "Srikanth Varma" was referred as "PERSON". so now you have to Combine the "New York" with **"\_"** i.e "New\_York". and remove the "Srikanth Varma" from the above sente nce because it is a person.

**13.** Replace all the digits with space i.e delete all the digits. > In the above sample document, the 6th line have digit 100, so we have to remove that.

**14.** After doing above points, we observed there might be few word's like **"\_word\_" (i.e starting and en ding with the \_), "\_word" (i.e starting with the \_), "word\_" (i.e ending with the \_)** remove the **\_** from

these type of words.

**15.** We also observed some words like **"OneLetter\_word"- eg: d\_berlin, "TwoLetters\_word" - eg: dr\_berli n** , in these words we remove the "OneLetter\_" (d\_berlin ==> berlin) and "TwoLetters\_" (de\_berlin ==> be rlin). i.e remove the words which are length less than or equal to 2 after spliiting those words by

"\_".

**16.** Convert all the words into lower case and lowe case and remove the words which are greater than or

equal to 15 or less than or equal to 2.

**17.** replace all the words except "A-Za-z\_" with space.

**18.** Now You got Preprocessed Text, email, subject. create a dataframe with those. below are the columns of the df.

In [0]:

1data.columns

Index(['text', 'class', 'preprocessed\_text', 'preprocessed\_subject',

'preprocessed\_emails'], dtype='object')

In [0]:

1

data.iloc[400]

text From: arc1@ukc.ac.uk (Tony Curtis)\r\r\r\nSubj... class alt.atheism preprocessed\_text said re is article if followed the quoting rig... preprocessed\_subject christian morality is preprocessed\_emails ukc mac macalstr edu Name: 567, dtype: object

**To get above mentioned data frame --> Try to Write Total Preprocessing steps in One Function Named Preprocess as below.**

In [0]:

1**def** preprocess(Input\_Text): 2"""Do all the Preprocessing as shown above and 3return a tuple contain preprocess\_email,preprocess\_subject,preprocess\_text for that Text\_data""" 4

**return** (list\_of\_preproessed\_emails,subject,text)

**Code checking:**

After Writing preprocess function. call that functoin with the input text of 'alt.atheism\_49960' doc and print the output of the preprocess function

This will help us to evaluate faster, based on the output we can suggest you if there are any changes.

**After writing Preprocess function, call the function for each of the document(18828 docs) and then create a dataframe as mentioned above.**

**Training The models to Classify:**

1. Combine "preprocessed\_text", "preprocessed\_subject", "preprocessed\_emails" into one column. use that column to model.

2. Now Split the data into Train and test. use 25% for test also do a stratify split.

3. Analyze your text data and pad the sequnce if required. Sequnce length is not restricted, you can us e anything of your choice. you need to give the reasoning

4. Do Tokenizer i.e convert text into numbers. please be careful while doing it. if you are using tf.ke ras "Tokenizer" API, it removes the **"\_"**, but we need that.

5. code the model's ( Model-1, Model-2 ) as discussed below and try to optimize that models.

6. For every model use predefined Glove vectors. **Don't train any word vectors while Training the model.**

7. Use "categorical\_crossentropy" as Loss.

8. Use **Accuracy and Micro Avgeraged F1 score** as your as Key metrics to evaluate your model.

9. Use Tensorboard to plot the loss and Metrics based on the epoches.

10. Please save your best model weights in to **'best\_model\_L.h5' ( L = 1 or 2 )**.

11. You are free to choose any Activation function, learning rate, optimizer. But have to use the same

architecture which we are giving below.

12. You can add some layer to our architecture but you **deletion** of layer is not acceptable.

13. Try to use **Early Stopping** technique or any of the callback techniques that you did in the previous

assignments.

14. For Every model save your model to image ( Plot the model) with shapes and upload those imgages to

Classroom. You can use "plot\_model" please refer this (https://www.tensorflow.org/api\_docs/python/tf/k eras/utils/plot\_model) if you don't know how to plot the model with shapes.

**Model-1**

Input\_Text: InputLayer

EmbeddingLayer\_ToGet\_Embedding\_Matrix: Embedding

ConvLayer\_with\_filter\_size\_N1xMl: Conv2D

ConvLayer\_with\_filter\_size\_N2XM1: Conv2D

concatenate\_above\_two\_conv\_layers\_1: Concatenate

SpatialDropout2DLayer1: SpatialDropout2D

MaxPoolingLayerl: MaxPooling2D

ConvLayer\_with\_filter\_size\_N3xM2: Conv2D

ConvLayer\_with\_filter\_size\_N4XM2: Conv2D

concatenate\_above\_two\_conv\_layers\_2: Concatenate

SpatialDropout2DLayer2: SpatialDropout2D

MaxPoolingLayer2: MaxPooling2D

ConvLayer\_with\_filter\_size\_N4xM3: Conv2D

ConvLayer\_with\_filter\_size\_N5xM3: Conv2D

concatenate\_above\_two\_conv\_layers\_3: Concatenate

flatten the data: Flatten

Dropoutl: Dropout

**Densel: Dense**

ref: https://i.imgur.com/tps37NM.png

1. **Input\_Text** --> Input the total text data. Then Create a Matrix with Embedding layer as shown Below. Here d = 5, but in our case we will get d = dimension of Word vectors we are using. So if i have maximu m of 350 words in a sentence and 300 dim word vector for each word, we will get 350\*300 dimensional mat rix as output after embedding layer

Ref: https://i.imgur.com/kiVQuk1.png

2. We have given the conv layers ( Conv2D ) with filter sizes N\_i \* M\_i --> Here M\_i must be size of Co lumn i.e for the first Conv layer it will be n\_i \* 300 size filters if you are using 300 dim word vecto r. so for Every layer please use M\_i = Column dimension of previous layer.

3. concatenate layer is to concatenate all the filters.

y

4. You can use any pool size and stride for maxpooling layer, we will recommed you to pool size greater than 3.

5. Don't use more than 16 filters in one Conv layer becuase it will increase the no of params. ( Only r ecommendation if you have less computing power )

6. You can use any layers from the Flatten Layer.

**Model-2**

Input\_Text: InputLayer

EmbeddingLayer\_ToGet\_Embedding\_Matrix: Embeddling

ConviD\_with\_filter\_size\_M: ConviD

ConviD\_with\_filter\_size\_N: ConviD

ConviD\_with\_filter\_size\_O: ConviD

concatenatel\_above\_three\_conv\_layers: Concatenate

MaxPoolLayerl: MaxPooling2D

ConvlD\_with\_filter\_size\_i: Convid

ConvlD\_with\_filter\_size\_j: ConviD |

ConviD\_with\_filter\_size\_k: ConviD

concatenate2\_above\_Three\_conv\_layers: Concatenate

MaxPoolLayer2: MaxPoolinglD

Convit\_with\_filter\_size\_P: ConviD

ref: 'https://i.imgur.com/fv1GvFJ.png'

1. all are Conv1D layers with any number of filter and filter sizes, there is no restriction on this.

2. concatenate layer is to concatenate all the filters/channels.

3. You can use any pool size and stride for maxpooling layer.

4. Don't use more than 16 filters in one Conv layer becuase it will increase the no of params. ( Only r ecommendation if you have less computing power )

5. You can use any number of layers after the Flatten Layer.