Importing Libraries

```
'''from google.colab import drive
drive.mount('/content/drive')'''
```

```
"from google.colab import drive\ndrive.mount('/content/drive')"
```

```
#Importing important libraries


import numpy as np, pandas as pd
!pip install pad_sequences
!pip install talos

import re
import spacy
from spacy.lang.en import English
from spacy.lang.en.stop_words import STOP_WORDS
from nltk.tokenize import word_tokenize
import nltk
nltk.download('wordnet')
from nltk.stem import WordNetLemmatizer

import string
from string import ascii_lowercase

from tqdm import tqdm_notebook
import itertools
import io

import matplotlib.pyplot as plt
%matplotlib inline

from functools import reduce
from tensorflow import keras
from keras.preprocessing.text import Tokenizer
from keras_preprocessing.sequence import pad_sequences
from keras.layers import Dense, Input, LSTM, Embedding, Dropout,
Activation
from keras.layers import Bidirectional, GlobalMaxPool1D
from keras.models import Model
from keras.models import Sequential
from keras.layers import Conv1D, MaxPooling1D
from keras.layers import BatchNormalization
from keras import initializers, regularizers, constraints, optimizers,
layers
```

```python
import talos
```

```
Defaulting to user installation because normal site-packages is not
writeable
Requirement already satisfied: pad_sequences in c:\users\ojas\appdata\
roaming\python\python39\site-packages (0.6.1)
Defaulting to user installation because normal site-packages is not
writeable
Requirement already satisfied: talos in c:\users\ojas\appdata\roaming\
python\python39\site-packages (1.3)
Requirement already satisfied: tqdm in c:\programdata\anaconda3\lib\
site-packages (from talos) (4.64.1)
Requirement already satisfied: numpy in c:\programdata\anaconda3\lib\
site-packages (from talos) (1.21.5)
Requirement already satisfied: wrangle in c:\users\ojas\appdata\
roaming\python\python39\site-packages (from talos) (0.7.2)
Requirement already satisfied: tensorflow>=2.0.0 in c:\programdata\
anaconda3\lib\site-packages (from talos) (2.9.1)
Requirement already satisfied: pandas in c:\programdata\anaconda3\lib\
site-packages (from talos) (1.4.4)
Requirement already satisfied: requests in c:\programdata\anaconda3\
lib\site-packages (from talos) (2.28.1)
Requirement already satisfied: kerasplotlib in c:\users\ojas\appdata\
roaming\python\python39\site-packages (from talos) (1.0)
Requirement already satisfied: statsmodels>=0.11.0 in c:\programdata\
anaconda3\lib\site-packages (from talos) (0.13.2)
Requirement already satisfied: sklearn in c:\users\ojas\appdata\
roaming\python\python39\site-packages (from talos) (0.0.post2)
Requirement already satisfied: chances in c:\users\ojas\appdata\
roaming\python\python39\site-packages (from talos) (0.1.9)
Requirement already satisfied: astetik in c:\users\ojas\appdata\
roaming\python\python39\site-packages (from talos) (1.13)
Requirement already satisfied: scipy>=1.3 in c:\programdata\anaconda3\
lib\site-packages (from statsmodels>=0.11.0->talos) (1.7.3)
Requirement already satisfied: patsy>=0.5.2 in c:\programdata\
anaconda3\lib\site-packages (from statsmodels>=0.11.0->talos) (0.5.2)
Requirement already satisfied: packaging>=21.3 in c:\programdata\
anaconda3\lib\site-packages (from statsmodels>=0.11.0->talos) (21.3)
Requirement already satisfied: pytz>=2020.1 in c:\users\ojas\appdata\
roaming\python\python39\site-packages (from pandas->talos) (2022.7)
Requirement already satisfied: python-dateutil>=2.8.1 in c:\
programdata\anaconda3\lib\site-packages (from pandas->talos) (2.8.2)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in
c:\users\ojas\appdata\roaming\python\python39\site-packages (from
tensorflow>=2.0.0->talos) (0.28.0)
Requirement already satisfied: astunparse>=1.6.0 in c:\programdata\
anaconda3\lib\site-packages (from tensorflow>=2.0.0->talos) (1.6.3)
Requirement already satisfied: keras-preprocessing>=1.1.1 in c:\
programdata\anaconda3\lib\site-packages (from tensorflow>=2.0.0-
>talos) (1.1.2)
Requirement already satisfied: termcolor>=1.1.0 in c:\programdata\
anaconda3\lib\site-packages (from tensorflow>=2.0.0->talos) (2.1.0)
Requirement already satisfied: google-pasta>=0.1.1 in c:\programdata\
anaconda3\lib\site-packages (from tensorflow>=2.0.0->talos) (0.2.0)
Requirement already satisfied: absl-py>=1.0.0 in c:\programdata\
anaconda3\lib\site-packages (from tensorflow>=2.0.0->talos) (1.3.0)
Requirement already satisfied: gast<=0.4.0,>=0.2.1 in c:\users\ojas\
appdata\roaming\python\python39\site-packages (from tensorflow>=2.0.0-
>talos) (0.4.0)
Requirement already satisfied: six>=1.12.0 in c:\programdata\
anaconda3\lib\site-packages (from tensorflow>=2.0.0->talos) (1.16.0)
Requirement already satisfied: protobuf<3.20,>=3.9.2 in c:\users\ojas\
appdata\roaming\python\python39\site-packages (from tensorflow>=2.0.0-
>talos) (3.19.6)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in c:\programdata\
anaconda3\lib\site-packages (from tensorflow>=2.0.0->talos) (1.42.0)
Requirement already satisfied: wrapt>=1.11.0 in c:\programdata\
anaconda3\lib\site-packages (from tensorflow>=2.0.0->talos) (1.14.1)
Requirement already satisfied: tensorboard<2.10,>=2.9 in c:\
programdata\anaconda3\lib\site-packages (from tensorflow>=2.0.0-
>talos) (2.9.0)
Requirement already satisfied: tensorflow-estimator<2.10.0,>=2.9.0rc0
in c:\programdata\anaconda3\lib\site-packages (from tensorflow>=2.0.0-
>talos) (2.9.0)
Requirement already satisfied: typing-extensions>=3.6.6 in c:\
programdata\anaconda3\lib\site-packages (from tensorflow>=2.0.0-
>talos) (4.3.0)
Requirement already satisfied: libclang>=13.0.0 in c:\users\ojas\
appdata\roaming\python\python39\site-packages (from tensorflow>=2.0.0-
>talos) (14.0.6)
Requirement already satisfied: keras<2.10.0,>=2.9.0rc0 in c:\
programdata\anaconda3\lib\site-packages (from tensorflow>=2.0.0-
>talos) (2.9.0)
Requirement already satisfied: flatbuffers<2,>=1.12 in c:\users\ojas\
appdata\roaming\python\python39\site-packages (from tensorflow>=2.0.0-
>talos) (1.12)
Requirement already satisfied: setuptools in c:\programdata\anaconda3\
lib\site-packages (from tensorflow>=2.0.0->talos) (63.4.1)
Requirement already satisfied: h5py>=2.9.0 in c:\programdata\
anaconda3\lib\site-packages (from tensorflow>=2.0.0->talos) (3.7.0)
Requirement already satisfied: opt-einsum>=2.3.2 in c:\programdata\
anaconda3\lib\site-packages (from tensorflow>=2.0.0->talos) (3.3.0)
Requirement already satisfied: IPython in c:\programdata\anaconda3\
lib\site-packages (from astetik->talos) (7.31.1)
Requirement already satisfied: seaborn in c:\programdata\anaconda3\
lib\site-packages (from astetik->talos) (0.11.2)
Requirement already satisfied: geonamescache in c:\users\ojas\appdata\
roaming\python\python39\site-packages (from astetik->talos) (1.5.0)
Requirement already satisfied: matplotlib in c:\programdata\anaconda3\
lib\site-packages (from kerasplotlib->talos) (3.5.2)
Requirement already satisfied: idna<4,>=2.5 in c:\programdata\
anaconda3\lib\site-packages (from requests->talos) (3.3)
Requirement already satisfied: charset-normalizer<3,>=2 in c:\
programdata\anaconda3\lib\site-packages (from requests->talos) (2.0.4)
Requirement already satisfied: certifi>=2017.4.17 in c:\programdata\
anaconda3\lib\site-packages (from requests->talos) (2022.9.24)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\
programdata\anaconda3\lib\site-packages (from requests->talos)
(1.26.11)
Requirement already satisfied: colorama in c:\programdata\anaconda3\
lib\site-packages (from tqdm->talos) (0.4.5)
Requirement already satisfied: wheel<1.0,>=0.23.0 in c:\programdata\
anaconda3\lib\site-packages (from astunparse>=1.6.0-
>tensorflow>=2.0.0->talos) (0.37.1)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in c:\
programdata\anaconda3\lib\site-packages (from packaging>=21.3-
>statsmodels>=0.11.0->talos) (3.0.9)
Requirement already satisfied: google-auth<3,>=1.6.3 in c:\
programdata\anaconda3\lib\site-packages (from tensorboard<2.10,>=2.9-
>tensorflow>=2.0.0->talos) (2.6.0)
Requirement already satisfied: markdown>=2.6.8 in c:\programdata\
anaconda3\lib\site-packages (from tensorboard<2.10,>=2.9-
>tensorflow>=2.0.0->talos) (3.3.4)
Requirement already satisfied: tensorboard-data-server<0.7.0,>=0.6.0
in c:\programdata\anaconda3\lib\site-packages (from
tensorboard<2.10,>=2.9->tensorflow>=2.0.0->talos) (0.6.0)
Requirement already satisfied: google-auth-oauthlib<0.5,>=0.4.1 in c:\
programdata\anaconda3\lib\site-packages (from tensorboard<2.10,>=2.9-
>tensorflow>=2.0.0->talos) (0.4.4)
Requirement already satisfied: tensorboard-plugin-wit>=1.6.0 in c:\
programdata\anaconda3\lib\site-packages (from tensorboard<2.10,>=2.9-
>tensorflow>=2.0.0->talos) (1.8.1)
Requirement already satisfied: werkzeug>=1.0.1 in c:\programdata\
anaconda3\lib\site-packages (from tensorboard<2.10,>=2.9-
>tensorflow>=2.0.0->talos) (2.0.3)
Requirement already satisfied: pygments in c:\programdata\anaconda3\
lib\site-packages (from IPython->astetik->talos) (2.11.2)
Requirement already satisfied: jedi>=0.16 in c:\programdata\anaconda3\
lib\site-packages (from IPython->astetik->talos) (0.18.1)
Requirement already satisfied: backcall in c:\programdata\anaconda3\
lib\site-packages (from IPython->astetik->talos) (0.2.0)
Requirement already satisfied: traitlets>=4.2 in c:\programdata\
anaconda3\lib\site-packages (from IPython->astetik->talos) (5.1.1)
Requirement already satisfied: matplotlib-inline in c:\programdata\
anaconda3\lib\site-packages (from IPython->astetik->talos) (0.1.6)
Requirement already satisfied: prompt-toolkit!=3.0.0,!
=3.0.1,<3.1.0,>=2.0.0 in c:\programdata\anaconda3\lib\site-packages
(from IPython->astetik->talos) (3.0.20)
Requirement already satisfied: pickleshare in c:\programdata\
anaconda3\lib\site-packages (from IPython->astetik->talos) (0.7.5)
Requirement already satisfied: decorator in c:\programdata\anaconda3\
lib\site-packages (from IPython->astetik->talos) (5.1.1)
Requirement already satisfied: cycler>=0.10 in c:\programdata\
anaconda3\lib\site-packages (from matplotlib->kerasplotlib->talos)
(0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\programdata\
anaconda3\lib\site-packages (from matplotlib->kerasplotlib->talos)
(4.25.0)
Requirement already satisfied: pillow>=6.2.0 in c:\programdata\
anaconda3\lib\site-packages (from matplotlib->kerasplotlib->talos)
(9.2.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\programdata\
anaconda3\lib\site-packages (from matplotlib->kerasplotlib->talos)
(1.4.2)
Requirement already satisfied: rsa<5,>=3.1.4 in c:\programdata\
anaconda3\lib\site-packages (from google-auth<3,>=1.6.3-
>tensorboard<2.10,>=2.9->tensorflow>=2.0.0->talos) (4.7.2)
Requirement already satisfied: cachetools<6.0,>=2.0.0 in c:\
programdata\anaconda3\lib\site-packages (from google-auth<3,>=1.6.3-
>tensorboard<2.10,>=2.9->tensorflow>=2.0.0->talos) (4.2.2)
Requirement already satisfied: pyasn1-modules>=0.2.1 in c:\
programdata\anaconda3\lib\site-packages (from google-auth<3,>=1.6.3-
>tensorboard<2.10,>=2.9->tensorflow>=2.0.0->talos) (0.2.8)
Requirement already satisfied: requests-oauthlib>=0.7.0 in c:\
programdata\anaconda3\lib\site-packages (from google-auth-
oauthlib<0.5,>=0.4.1->tensorboard<2.10,>=2.9->tensorflow>=2.0.0-
>talos) (1.3.0)
Requirement already satisfied: parso<0.9.0,>=0.8.0 in c:\programdata\
anaconda3\lib\site-packages (from jedi>=0.16->IPython->astetik->talos)
(0.8.3)
Requirement already satisfied: wcwidth in c:\programdata\anaconda3\
lib\site-packages (from prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0-
>IPython->astetik->talos) (0.2.5)
Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in c:\programdata\
anaconda3\lib\site-packages (from pyasn1-modules>=0.2.1->google-
auth<3,>=1.6.3->tensorboard<2.10,>=2.9->tensorflow>=2.0.0->talos)
(0.4.8)
Requirement already satisfied: oauthlib>=3.0.0 in c:\programdata\
anaconda3\lib\site-packages (from requests-oauthlib>=0.7.0->google-
auth-oauthlib<0.5,>=0.4.1->tensorboard<2.10,>=2.9->tensorflow>=2.0.0-
>talos) (3.2.1)

[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\Ojas\AppData\Roaming\nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
```

Importing Data

```
train=pd.read_csv('train.csv')
```

```
train.head()
```

```
                 id                                       comment_text
toxic  \
0  0000997932d777bf  Explanation\nWhy the edits made under my usern...
0
1  000103f0d9cfb60f  D'aww! He matches this background colour I'm s...
0
2  000113f07ec002fd  Hey man, I'm really not trying to edit war. It...
0
3  0001b41b1c6bb37e  "\nMore\nI can't make any real suggestions on ...
0
4  0001d958c54c6e35  You, sir, are my hero. Any chance you remember...
0

   severe_toxic  obscene  threat  insult  identity_hate
0             0        0       0       0              0
1             0        0       0       0              0
2             0        0       0       0              0
3             0        0       0       0              0
4             0        0       0       0              0
```

```
test=pd.read_csv('test.csv',skiprows=[18522])
```

```
test.head()
```

```
                 id                                       comment_text
0  00001cee341fdb12  Yo bitch Ja Rule is more succesful then you'll...
1  0000247867823ef7  == From RfC == \n\n The title is fine as it is...
2  00013b17ad220c46  " \n\n == Sources == \n\n * Zawe Ashton on Lap...
3  00017563c3f7919a  :If you have a look back at the source, the in...
4  00017695ad8997eb          I don't anonymously edit articles at all.
```

Data Exploration

Checking for missing values

```
train.isnull().any()
```

```
id               False
comment_text     False
toxic            False
severe_toxic     False
obscene          False
threat           False
insult           False
identity_hate    False
dtype: bool
```

```
test.isnull().any()
```

```
id                False
comment_text      False
dtype: bool
```

```
labels = ['toxic', 'severe_toxic', 'obscene', 'threat', 'insult',
'identity_hate']
y = train[labels].values
```

#Data Pre-processing

Text Normalization

- Data normalization is the systematic process of grouping similar values into one common value, bringing greater context
- This includes -
- Removing Characters in between Text
- Removing Repeated Characters
- Converting data to lower-case
- Removing Numbers from the data
- Remove Punctuation
- Remove Whitespaces
- Removing spaces in between words
- Removing "\n"
- Remove Non-english characters

```
RE_PATTERNS = {
    ' american ':
        [
            'amerikan'
        ],

    ' adolf ':
        [
            'adolf'
        ],


    ' hitler ':
        [
            'hitler'
        ],

    ' fuck':
        [
            '(f)(u|[^a-z0-9 ])(c|[^a-z0-9 ])(k|[^a-z0-9 ])([^ ])*',
            '(f)([^a-z]*)(u)([^a-z]*)(c)([^a-z]*)(k)',
            ' f[!@#\$%\^\&\*]*u[!@#\$%\^&\*]*k', 'f u u c',
            '(f)(c|[^a-z ])(u|[^a-z ])(k)', r'f\*',
            'feck ', ' fux ', 'f\*\*', 'f**k','fu*k',
            'f\-ing', 'f\.u\.', 'f###', ' fu ', 'f@ck', 'f u c k', 'f
uck', 'f ck'
        ],

    ' ass ':
        [
            '[^a-z]ass ', '[^a-z]azz ', 'arrse', ' arse ', '@\$\$',
            '[^a-z]anus', ' a\*s\*s', '[^a-z]ass[^a-z ]',
            'a[@#\$%\^&\*][@#\$%\^&\*]', '[^a-z]anal ', 'a s s','a55',
'@$$'
        ],

    ' ass hole ':
        [
            ' a[s|z]*wipe', 'a[s|z]*[w]*h[o|0]+[l]*e', '@\$\$hole',
'a**hole'
        ],

    ' bitch ':
        [
            'b[w]*i[t]*ch', 'b!tch',
            'bi\+ch', 'b!\+ch', '(b)([^a-z]*)(i)([^a-z]*)(t)([^a-z]*)
(c)([^a-z]*)(h)',
            'biatch', 'bi\*\*h', 'bytch', 'b i t c h', 'b!tch',
'bi+ch', 'l3itch'
        ],

    ' bastard ':
        [
            'ba[s|z]+t[e|a]+rd'
        ],

    ' trans gender':
        [
            'transgender'
        ],

    ' gay ':
        [
            'gay'
        ],

    ' cock ':
        [
            '[^a-z]cock', 'c0ck', '[^a-z]cok ', 'c0k', '[^a-
z]cok[^aeiou]', ' cawk',
            '(c)([^a-z ])(o)([^a-z ]*)(c)([^a-z ]*)(k)', 'c o c k'
        ],
    ' dick ':
        [
            ' dick[^aeiou]', 'deek', 'd i c k', 'dik'
        ],

    ' suck ':
        [
            'sucker', '(s)([^a-z ]*)(u)([^a-z ]*)(c)([^a-z ]*)(k)',
'sucks', '5uck', 's u c k'
        ],

    ' cunt ':
        [
            'cunt', 'c u n t'
        ],

    ' bull shit ':
        [
            'bullsh\*t', 'bull\$hit'
        ],

    ' homo sex ual':
        [
            'homosexual'
        ],

    ' jerk ':
        [
            'jerk'
        ],

    ' idiot ':
        [
            'i[d]+io[t]+', '(i)([^a-z ]*)(d)([^a-z ]*)(i)([^a-z ]*)(o)
([^a-z ]*)(t)', 'idiots'

'i d i o t'
        ],

    ' dumb ':
        [
            '(d)([^a-z ]*)(u)([^a-z ]*)(m)([^a-z ]*)(b)'
        ],

    ' shit ':
        [
            'shitty', '(s)([^a-z ]*)(h)([^a-z ]*)(i)([^a-z ]*)(t)',
'shite', '\$hit', 's h i t', '$h1t'
        ],
    ' shit hole ':
        [
            'shythole'
        ],

    ' retard ':
        [
            'returd', 'retad', 'retard', 'wiktard', 'wikitud'
        ],

    ' rape ':
        [
            ' raped'
        ],

    ' dumb ass':
        [
            'dumbass', 'dubass'
        ],

    ' ass head':
        [
            'butthead'
        ],

    ' sex ':
        [
            'sexy', 's3x', 'sexuality'
        ],


    ' nigger ':
        [
            'nigger', 'ni[g]+a', ' nigr ', 'negrito', 'niguh', 'n3gr',
'n i g g e r'
        ],

    ' shut the fuck up':
        [
            'stfu', 'st*u'
        ],

    ' pussy ':
        [
            'pussy[^c]', 'pusy', 'pussi[^l]', 'pusses', 'p*ssy'
        ],

    ' faggot ':
        [
            'faggot', ' fa[g]+[s]*[^a-z ]', 'fagot', 'f a g g o t',
'faggit',
            '(f)([^a-z ]*)(a)([^a-z ]*)([g]+)([^a-z ]*)(o)([^a-z ]*)
(t)', 'fau[g]+ot', 'fae[g]+ot',
        ],

    ' mother fucker':
        [
            ' motha ', ' motha f', ' mother f', 'motherucker',
        ],

    ' whore ':
        [
            'wh\*\*\*', 'w h o r e'
        ],
    ' fucking ':
        [
            'f*$%-ing'
        ],
}

# Function to clean data.
def clean_text(text,remove_repeat_text=True,
remove_patterns_text=True, is_lower=True):

  if is_lower:
    text=text.lower()

  if remove_patterns_text:
    for target, patterns in RE_PATTERNS.items():
      for pat in patterns:
        text=str(text).replace(pat, target)

  if remove_repeat_text:
    text = re.sub(r'(.)\1{2,}', r'\1', text)

  text = str(text).replace("\n", " ")
  text = re.sub(r'[^\w\s]',' ',text)
  text = re.sub('[0-9]',"",text)
  text = re.sub(" +", " ", text)
  text = re.sub("([^\x00-\x7F])+"," ",text)
  return text
```

Cleaning Training Data

```
train['comment_text']=train['comment_text'].apply(lambda x:
clean_text(x))
train['comment_text'][1]
```

```
'd aww he matches this background colour i m seemingly stuck with
thanks talk january utc '

train

                      id
comment_text  \
0       0000997932d777bf  explanation why the edits made under my
userna...
1       000103f0d9cfb60f  d aww he matches this background colour i m
se...
2       000113f07ec002fd  hey man i m really not trying to edit war it
s...
3       0001b41b1c6bb37e   more i can t make any real suggestions on
imp...
4       0001d958c54c6e35  you sir are my hero any chance you remember
wh...
...                  ...
...
159566  ffe987279560d7ff   and for the second time of asking when your
v...
159567  ffea4adeee384e90  you should be ashamed of yourself that is a
ho...
159568  ffee36eab5c267c9  spitzer umm theres no actual article for
prost...
159569  fff125370e4aaaf3  and it looks like it was actually you who
put ...
159570  fff46fc426af1f9a   and i really don t think you understand i
cam...

        toxic  severe_toxic  obscene  threat  insult  identity_hate
0           0             0        0       0       0              0
1           0             0        0       0       0              0
2           0             0        0       0       0              0
3           0             0        0       0       0              0
4           0             0        0       0       0              0
...       ...           ...      ...     ...     ...            ...
159566      0             0        0       0       0              0
159567      0             0        0       0       0              0
159568      0             0        0       0       0              0
159569      0             0        0       0       0              0
159570      0             0        0       0       0              0

[159571 rows x 8 columns]

test

                     id
comment_text
0      00001cee341fdb12  Yo bitch Ja Rule is more succesful then
you'll...
```

```
1      0000247867823ef7  == From RfC == \n\n The title is fine as it 
is...
2      00013b17ad220c46  " \n\n == Sources == \n\n * Zawe Ashton on 
Lap...
3      00017563c3f7919a  :If you have a look back at the source, the 
in...
4      00017695ad8997eb          I don't anonymously edit articles at 
all.
...                   ...                                         
...
18516  1f1d024a23558d69  " \n\n HEY YOU KNOW WHAT? I THINK IT""S TIME 
T...
18517  1f1d588bfa2c48df  " \n\n == Userboxes == \n\n Hi, If you want 
to...
18518  1f1e7a517fe588ac              the flu shot, raise an eyebrow or 
two
18519  1f1e856cb41f254b  " \n\n == Open proxy page == \n\n Watch out! 
V...
18520  1f1eef220e12dbfd  (But of course there should be a redirect 
from...

[18521 rows x 2 columns]
```

```
#test
test['comment_text']
```

```
0        Yo bitch Ja Rule is more succesful then you'll...
1        == From RfC == \n\n The title is fine as it is...
2        " \n\n == Sources == \n\n * Zawe Ashton on Lap...
3        :If you have a look back at the source, the in...
4                I don't anonymously edit articles at all.
                               ...                        
18516    " \n\n HEY YOU KNOW WHAT? I THINK IT""S TIME T...
18517    " \n\n == Userboxes == \n\n Hi, If you want to...
18518                the flu shot, raise an eyebrow or two
18519    " \n\n == Open proxy page == \n\n Watch out! V...
18520    (But of course there should be a redirect from...
Name: comment_text, Length: 18521, dtype: object
```

Cleaning Test Data

```
test['comment_text']=test['comment_text'].apply(lambda x: 
clean_text(x))
test['comment_text'][1048]
```

```
'this is a university ip address just fyi '
```

Lemmatization

- Lemmatization is the processs of grouping together the different inflected from of a word so they can be analyzed as a single item. Lemmatization helps to reduce word into stem words such as 'studies' to study.

```
comments_train=train['comment_text']
comments_test=test['comment_text']

comments_train=list(comments_train)
comments_test=list(comments_test)

print(comments_train[:10])
print(comments_test[:10])
```

```
['explanation why the edits made under my username hardcore metallica
fan were reverted they weren t vandalisms just closure on some gas
after i voted at new york dolls fac and please don t remove the
template from the talk page since i m retired now ', 'd aww he matches
this background colour i m seemingly stuck with thanks talk january
utc ', 'hey man i m really not trying to edit war it s just that this
guy is constantly removing relevant information and talking to me
through edits instead of my talk page he seems to care more about the
formatting than the actual info ', ' more i can t make any real
suggestions on improvement i wondered if the section statistics should
be later on or a subsection of types of accidents i think the
references may need tidying so that they are all in the exact same
format ie date format etc i can do that later on if no one else does
first if you have any preferences for formatting style on references
or want to do it yourself please let me know there appears to be a
backlog on articles for review so i guess there may be a delay until a
reviewer turns up it s listed in the relevant form eg wikipedia
good_article_nominations transport ', 'you sir are my hero any chance
you remember what page that s on ', ' congratulations from me as well
use the tools well   talk ', 'cock suck before you piss around on my
work', 'your vandalism to the matt shirvington article has been
reverted please don t do it again or you will be banned ', 'sorry if
the word nonsense was offensive to you anyway i m not intending to
write anything in the article wow they would jump on me for vandalism
i m merely requesting that it be more encyclopedic so one can use it
for school as a reference i have been to the selective breeding page
but it s almost a stub it points to animal breeding which is a short
messy article that gives you no info there must be someone around with
expertise in eugenics ', 'alignment on this subject and which are
contrary to those of dulithgow']
['yo bitch ja rule is more succesful then you ll ever be whats up with
you and hating you sad mofuckas i should bitch slap ur pethedic white
faces and get you to kiss my ass you guys sicken me ja rule is about
pride in da music man dont diss that shit on him and nothin is wrong
bein like tupac he was a brother too fuckin white boys get things
right next time ', ' from rfc the title is fine as it is imo ', '
sources zawe ashton on lapland ', ' if you have a look back at the
source the information i updated was the correct form i can only guess
the source hadn t updated i shall update the information once again
but thank you for your message ', 'i don t anonymously edit articles
at all ', 'thank you for understanding i think very highly of you and
would not revert without discussion ', 'please do not add nonsense to
wikipedia such edits are considered vandalism and quickly undone if
you would like to experiment please use the sandbox instead thank you
', ' dear god this site is horrible ', ' only a fool can believe in
such numbers the correct number lies between to ponder the numbers
carefully this error will persist for a long time as it continues to
reproduce the latest reproduction i know is from encyclop dia
britannica almanac wich states magnittude fair enough victims today to
is not a lot so i guess people just come out with a number that
impresses enough i don t know but i know this it s just a shameless
lucky number that they throw in the air gc ', ' double redirects when
fixing double redirects don t just blank the outer one you need edit
it to point it to the final target unless you think it s inappropriate
in which case it needs to be nominated at wp rfd']
```

```python
wordnet_lemmatizer = WordNetLemmatizer()

def lemma(text, lemmatization=True):
  output=""
  if lemmatization:
    text=text.split(" ")
    for word in text:
       word1 = wordnet_lemmatizer.lemmatize(word, pos = "n")
       word2 = wordnet_lemmatizer.lemmatize(word1, pos = "v")
       word3 = wordnet_lemmatizer.lemmatize(word2, pos = "a")
       word4 = wordnet_lemmatizer.lemmatize(word3, pos = "r")
       output=output + " " + word4
  else:
    output=text

  return str(output.strip())
```

Lemmatizing Training Data

```python
nltk.download('averaged_perceptron_tagger')
```

```
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     C:\Users\Ojas\AppData\Roaming\nltk_data...
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!

True
```

```python
nltk.download('omw-1.4')
```

```
[nltk_data] Downloading package omw-1.4 to
[nltk_data]     C:\Users\Ojas\AppData\Roaming\nltk_data...
[nltk_data]   Package omw-1.4 is already up-to-date!

True
```

```python
lemmatized_train_data = []
```

```
for line in tqdm_notebook(comments_train, total=159571):
    lemmatized_train_data.append(lemma(line))
```

{"model_id":"c0f38d8e898945e5bb83c1a94d3c13d1","version_major":2,"version_minor":0}

```
lemmatized_train_data[152458]
```

Lemmatizing Test Data

```
lemmatized_test_data = []

for line in tqdm_notebook(comments_test, total=len(comments_test)):
    lemmatized_test_data.append(lemma(line))
```

Stopwords Removal

- Removing stopwords ensures that more focus is on those word that define the meaning of the text.
- To remove stopwords from data "Spacy" library which remove stopwords from any textual data.

```
stopword_list=STOP_WORDS
```

Adding Single and Dual to STOP_WORDS

- adding custom words to the list of stopwords.

```
def iter_all_strings():
    for size in itertools.count(1):
        for s in itertools.product(ascii_lowercase, repeat=size):
            yield "".join(s)

dual_alpha_list=[]
for s in iter_all_strings():
    dual_alpha_list.append(s)
    if s == 'zz':
        break

# Removing stopwords from the text

dual_alpha_list.remove('i')
dual_alpha_list.remove('a')
dual_alpha_list.remove('am')
dual_alpha_list.remove('an')
dual_alpha_list.remove('as')
dual_alpha_list.remove('at')
dual_alpha_list.remove('be')
dual_alpha_list.remove('by')
dual_alpha_list.remove('do')
dual_alpha_list.remove('go')
dual_alpha_list.remove('he')
dual_alpha_list.remove('hi')
dual_alpha_list.remove('if')
dual_alpha_list.remove('is')
dual_alpha_list.remove('in')
dual_alpha_list.remove('me')
dual_alpha_list.remove('my')
dual_alpha_list.remove('no')
dual_alpha_list.remove('of')
dual_alpha_list.remove('on')
dual_alpha_list.remove('or')
dual_alpha_list.remove('ok')
dual_alpha_list.remove('so')
dual_alpha_list.remove('to')
dual_alpha_list.remove('up')
dual_alpha_list.remove('us')
dual_alpha_list.remove('we')

for letter in dual_alpha_list:
    stopword_list.add(letter)
print("Done!!")
```

Checking for other words that we may need in STOP_WORDS

```
def search_stopwords(data, search_stop=True):
  output=""
  if search_stop:
    data=data.split(" ")
    for word in data:
      if not word in stopword_list:
        output=output+" "+word
  else:
    output=data

  return str(output.strip())

potential_stopwords = []

for line in tqdm_notebook(lemmatized_train_data, total=159571):
    potential_stopwords.append(search_stopwords(line))

len(potential_stopwords)
```

Combining all the sentences in the list into a single string

```
def string_combine_a(stopword):
  final_a=""
  for item in range(39893):
    final_a=final_a+" "+stopword[item]
  return final_a

def string_combine_b(stopword):
  final_b=""
  for item in range(39893,79785):
    final_b=final_b+" "+stopword[item]
  return final_b

def string_combine_c(stopword):
  final_c=""
  for item in range(79785,119678):
    final_c=final_c+" "+stopword[item]
  return final_c

def string_combine_d(stopword):
  final_d=""
  for item in range(119678,159571):
    final_d=final_d+" "+stopword[item]
  return final_d

total_string_potential_a=string_combine_a(potential_stopwords)
total_string_potential_b=string_combine_b(potential_stopwords)
total_string_potential_c=string_combine_c(potential_stopwords)
total_string_potential_d=string_combine_d(potential_stopwords)
```

Counting the number of words in each of the 4 strings

```
def word_count(str):
    counts = dict()
    words = str.split()

    for word in words:
        if word in counts:
            counts[word] += 1
        else:
            counts[word] = 1

    return counts

total_string_potential_a_dict=word_count(total_string_potential_a)
total_string_potential_b_dict=word_count(total_string_potential_b)
total_string_potential_c_dict=word_count(total_string_potential_c)
total_string_potential_d_dict=word_count(total_string_potential_d)
```

Converting Dictionaries to Dataframe

```
total_string_potential_a_df =
pd.DataFrame(list(total_string_potential_a_dict.items()),columns =
['Word','Count'])
total_string_potential_b_df =
pd.DataFrame(list(total_string_potential_b_dict.items()),columns =
['Word','Count'])
total_string_potential_c_df =
pd.DataFrame(list(total_string_potential_c_dict.items()),columns =
['Word','Count'])
```

```
total_string_potential_d_df =
pd.DataFrame(list(total_string_potential_d_dict.items()),columns =
['Word','Count'])
```

Getting Dataframe output in descending order

```
top50_potential_stopwords_a=total_string_potential_a_df.sort_values(by
=['Count'],ascending=False).head(50)
top50_potential_stopwords_b=total_string_potential_b_df.sort_values(by
=['Count'],ascending=False).head(50)
top50_potential_stopwords_c=total_string_potential_c_df.sort_values(by
=['Count'],ascending=False).head(50)
top50_potential_stopwords_d=total_string_potential_d_df.sort_values(by
=['Count'],ascending=False).head(50)
```

Looking for common terms in all top 50 dataframes

```
common_potential_stopwords=list(reduce(set.intersection,map(set,
[top50_potential_stopwords_a.Word,top50_potential_stopwords_b.Word,top
50_potential_stopwords_c.Word,top50_potential_stopwords_d.Word])))

print(common_potential_stopwords)
```

Retaining certain words and removing others from the above list

```
potential_stopwords=['editor', 'reference', 'thank', 'work','find',
'good', 'know', 'like', 'look', 'thing', 'want', 'time', 'list',
'section','wikipedia', 'doe', 'add','new', 'try', 'think',
'write','use', 'user', 'way', 'page']
```

Adding above retrived words into the stopwords list

```
for word in potential_stopwords:
    stopword_list.add(word)
print("Done!!")
```

Removing Stopwords from Training Data

```
def remove_stopwords(text, remove_stop=True):
  output = ""
  if remove_stop:
    text=text.split(" ")
    for word in text:
      if word not in stopword_list:
        output=output + " " + word
  else :
    output=text

  return str(output.strip())

processed_train_data = []
```

```
for line in tqdm_notebook(lemmatized_train_data, total=159571):
    processed_train_data.append(remove_stopwords(line))
```

```
processed_train_data[152458]
```

Removing Stopwords from Test Data

```
processed_test_data = []

for line in tqdm_notebook(lemmatized_test_data, total=153164):
    processed_test_data.append(remove_stopwords(line))
```

Model Building

```
max_features=100000
maxpadlen = 200
val_split = 0.2
embedding_dim_fasttext = 300
```

Tokenization

```
tokenizer = Tokenizer(num_words=max_features)
tokenizer.fit_on_texts(list(processed_train_data))
list_tokenized_train = 
tokenizer.texts_to_sequences(processed_train_data)
list_tokenized_test = 
tokenizer.texts_to_sequences(processed_test_data)

word_index=tokenizer.word_index
print("Words in Vocabulary: ",len(word_index))
```

Padding

- Variable-length sentences are converted into variable-length sequence vectors and we cannot pass vectors of inconsistent lengths to our deep-learning model.

```
X_t=pad_sequences(list_tokenized_train, maxlen=maxpadlen, padding = 
'post')
X_te=pad_sequences(list_tokenized_test, maxlen=maxpadlen, padding = 
'post')

print('Tokenized sentences: \n', X_t[10])
print('One hot label: \n', y[10])
```

```
Tokenized sentences: 
 [   116    575     12 33222   1131      1    193     12 33222    349     91
12
   577     12    116    368    575      2   1084    116    339   5390    116
120
    12    387    265    368    575     12      2   1343    116      1      1
12
   387      3     32    116    575      1      1    193    116    173     47
84
   577   116   575    12     3   488   106    11  1088   406  1071
12
  2396   496    37   116    12   193   407   368    10   254   193
242
   154   109    19    21    30   173    77    21     4   256     1
4561
     5    12 33222  1131     1   193    12 33222   349    91   170
387
   308    69   577    35    46    77   425  1547    35   170     9
577
  1273    77  1636   135    11   135  4698   135    95    46   561
1251
    17    77  1337   118   135  1577    77  1132     1     1     5
507
    77   436    33   170    69    35   172  2213   450    33   317
1524
    71   173   237   154   116    33    33    12    77    33   116
12
    77    33    77    33     1     1   193   170    47    84   577
5
    33   170   193    81    11   860  3130    12    10   254    33
242
   154   109    19    12    77   124    68   436   116    12    10
357
    21    30   173    77    21     4   256     0]
One hot label:
 [0 0 0 0 0 0]
```

```
indices = np.arange(X_t.shape[0])
np.random.shuffle(indices)

X_t = X_t[indices]
labels = y[indices]
```

Splitting data into Training and Validation Set

```
num_validation_samples = int(val_split*X_t.shape[0])
x_train = X_t[: -num_validation_samples]
y_train = labels[: -num_validation_samples]
x_val = X_t[-num_validation_samples: ]
y_val = labels[-num_validation_samples: ]

print('Number of entries in each category:')
print('training: ', y_train.sum(axis=0))
print('validation: ', y_val.sum(axis=0))
```

```
Number of entries in each category:
training:  [12244  1260  6789   374  6342  1125]
validation:  [3050  335 1660  104 1535  280]
```

Importing Fast Text

```
embeddings_index_fasttext = {}
f = open('/content/drive/MyDrive/Profanity_dataset/test_labels.csv',
encoding='utf8')
for line in f:
    values = line.split()
    word = values[0]
    embeddings_index_fasttext[word] = np.asarray(values[1:],
dtype='float32')

f.close()

---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call
last)
~\AppData\Local\Temp\ipykernel_18036\3811139651.py in <module>
      1 embeddings_index_fasttext = {}
----> 2 f =
open('/content/drive/MyDrive/Profanity_dataset/test_labels.csv',
encoding='utf8')
      3 for line in f:
      4     values = line.split()
      5     word = values[0]

FileNotFoundError: [Errno 2] No such file or directory:
'/content/drive/MyDrive/Profanity_dataset/test_labels.csv'

embedding_matrix_fasttext = np.random.random((len(word_index) + 1,
embedding_dim_fasttext))
for word, i in word_index.items():
    embedding_vector = embeddings_index_fasttext.get(word)
    if embedding_vector is not None:
        embedding_matrix_fasttext[i] = embedding_vector
print(" Completed!")
```

Creating Model

Talos Grid Search for LSTM Model

```
def toxic_classifier(x_train,y_train,x_val,y_val,params):

  inp=Input(shape=(maxpadlen, ),dtype='int32')

  embedding_layer = Embedding(len(word_index) + 1,
                              embedding_dim_fasttext,
                              weights = [embedding_matrix_fasttext],
                              input_length = maxpadlen,
                              trainable=False,
                              name = 'embeddings')
  embedded_sequences = embedding_layer(inp)
  x = LSTM(params['output_count_lstm'],
return_sequences=True,name='lstm_layer')(embedded_sequences)

  x = GlobalMaxPool1D()(x)

  x = Dropout(params['dropout'])(x)

  x = Dense(params['output_count_dense'],
activation=params['activation'], kernel_initializer='he_uniform')(x)

  x = Dropout(params['dropout'])(x)

  preds = Dense(6, activation=params['last_activation'],
kernel_initializer='glorot_uniform')(x)

  model = Model(inputs=inp, outputs=preds)

  model.compile(loss=params['loss'], optimizer=params['optimizer'],
metrics=['accuracy'])

  model_info=model.fit(x_train,y_train, epochs=params['epochs'],
batch_size=params['batch_size'],  validation_data=(x_val, y_val))

  return model_info, model

p={
    'output_count_lstm': [40,50,60],
    'output_count_dense': [30,40,50],
    'batch_size': [32],
    'epochs':[2],
    'optimizer':['adam'],
    'activation':['relu'],
    'last_activation': ['sigmoid'],
    'dropout':[0.1,0.2],
    'loss': ['binary_crossentropy']
}

scan_results = talos.Scan(x=x_train,
               y=y_train,
               x_val=x_val,
               y_val=y_val,
               model=toxic_classifier,
               params=p,
               experiment_name='tcc',
               print_params=True)

model_id = scan_results.data['val_accuracy'].astype('float').argmax()
model_id

analyze_object = talos.Analyze(scan_results)
```

```
analyze_object.best_params('val_accuracy', ['accuracy', 'loss',
'val_loss'])

analyze_object.plot_line('val_accuracy')

analyze_object.plot_line('accuracy')
```

Talos Grid Search for LSTM-CNN Model

```
def toxic_classifier(x_train,y_train,x_val,y_val,params):

  inp=Input(shape=(maxpadlen, ),dtype='int32')

  embedding_layer = Embedding(len(word_index) + 1,
                             embedding_dim_fasttext,
                             weights = [embedding_matrix_fasttext],
                             input_length = maxpadlen,
                             trainable=False,
                             name = 'embeddings')
  embedded_sequences = embedding_layer(inp)

  x = LSTM(params['output_count_lstm'],
return_sequences=True,name='lstm_layer')(embedded_sequences)

  x = Conv1D(filters=params['filters'],
kernel_size=params['kernel_size'], padding='same', activation='relu',
kernel_initializer='he_uniform')(x)

  x = MaxPooling1D(params['pool_size'])(x)

  x = GlobalMaxPool1D()(x)

  x = BatchNormalization()(x)

  x = Dense(params['output_1_count_dense'],
activation=params['activation'], kernel_initializer='he_uniform')(x)

  x = Dropout(params['dropout'])(x)

  x = Dense(params['output_2_count_dense'],
activation=params['activation'], kernel_initializer='he_uniform')(x)

  x = Dropout(params['dropout'])(x)

  preds = Dense(6, activation=params['last_activation'],
kernel_initializer='glorot_uniform')(x)

  model = Model(inputs=inp, outputs=preds)

  model.compile(loss=params['loss'], optimizer=params['optimizer'],
metrics=['accuracy'])
```

```
  model_info=model.fit(x_train,y_train, epochs=params['epochs'],
batch_size=params['batch_size'],  validation_data=(x_val, y_val))

  return model_info, model

p={
    'output_count_lstm': [50,60],
    'output_1_count_dense': [40,50],
    'output_2_count_dense': [30,40],
    'filters' : [64],
    'kernel_size' : [3],
    'batch_size': [32],
    'pool_size': [3],
    'epochs':[2],
    'optimizer':['adam'],
    'activation':['relu'],
    'last_activation': ['sigmoid'],
    'dropout':[0.1,0.2],
    'loss': ['binary_crossentropy']
}

scan_results = talos.Scan(x=x_train,
               y=y_train,
               x_val=x_val,
               y_val=y_val,
               model=toxic_classifier,
               params=p,
               experiment_name='tcc',
               print_params=True)

model_id = scan_results.data['val_accuracy'].astype('float').argmax()
model_id

scan_results.data[8:9]

analyze_object = talos.Analyze(scan_results)

analyze_object.best_params('val_accuracy', ['accuracy', 'loss',
'val_loss'])

analyze_object.plot_line('val_accuracy')

analyze_object.plot_line('accuracy')
```

Training Model with Best Parameters

LSTM

```
inp=Input(shape=(maxpadlen, ),dtype='int32')
```

```
embedding_layer = Embedding(len(word_index) + 1,
                            embedding_dim_fasttext,
                            weights = [embedding_matrix_fasttext],
                            input_length = maxpadlen,
                            trainable=False,
                            name = 'embeddings')
embedded_sequences = embedding_layer(inp)

x = LSTM(40, return_sequences=True,name='lstm_layer')
(embedded_sequences)
x = GlobalMaxPool1D()(x)
x = Dropout(0.1)(x)
x = Dense(30, activation="relu", kernel_initializer='he_uniform')(x)
x = Dropout(0.1)(x)
preds = Dense(6, activation="sigmoid",
kernel_initializer='glorot_uniform')(x)

model_1 = Model(inputs=inp, outputs=preds)
model_1.compile(loss='binary_crossentropy',
                  optimizer='adam',
                  metrics=['accuracy'])

model_1.summary()

model_info_1=model_1.fit(x_train,y_train, epochs=2, batch_size=32,
validation_data=(x_val, y_val))
```

LSTM-CNN

```
inp=Input(shape=(maxpadlen, ),dtype='int32')

embedding_layer = Embedding(len(word_index) + 1,
                            embedding_dim_fasttext,
                            weights = [embedding_matrix_fasttext],
                            input_length = maxpadlen,
                            trainable=False,
                            name = 'embeddings')
embedded_sequences = embedding_layer(inp)

x = LSTM(50, return_sequences=True,name='lstm_layer')
(embedded_sequences)
x = Conv1D(filters=64, kernel_size=3, padding='same',
activation='relu', kernel_initializer='he_uniform')(x)
x = MaxPooling1D(3)(x)
x = GlobalMaxPool1D()(x)
x = BatchNormalization()(x)
x = Dense(40, activation="relu", kernel_initializer='he_uniform')(x)
x = Dropout(0.2)(x)
x = Dense(30, activation="relu", kernel_initializer='he_uniform')(x)
x = Dropout(0.2)(x)
preds = Dense(6, activation="sigmoid",
kernel_initializer='glorot_uniform')(x)
```

```
model_2 = Model(inputs=inp, outputs=preds)
model_2.compile(loss='binary_crossentropy',
                  optimizer='adam',
                  metrics=['accuracy'])
```

```
model_2.summary()
```

```
model_info_2=model_2.fit(x_train,y_train, epochs=2, batch_size=32,
validation_data=(x_val, y_val))
```

Plotting Graphs

LSTM

```
loss = model_info_1.history['loss']
val_loss = model_info_1.history['val_loss']

epochs = range(1, len(loss)+1)

plt.plot(epochs, loss, label='Training loss')
plt.plot(epochs, val_loss, label='Validation loss')
plt.title('Training and Validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show();
```

```
accuracy = model_info_1.history['accuracy']
val_accuracy = model_info_1.history['val_accuracy']

plt.plot(epochs, accuracy, label='Training accuracy')
plt.plot(epochs, val_accuracy, label='Validation accuracy')
plt.title('Training and validation accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epochs')
plt.legend()
plt.show();
```

LSTM-CNN

```
loss = model_info_2.history['loss']
val_loss = model_info_2.history['val_loss']

epochs = range(1, len(loss)+1)

plt.plot(epochs, loss, label='Training loss')
plt.plot(epochs, val_loss, label='Validation loss')
plt.title('Training and Validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show();
```

```
accuracy = model_info_2.history['accuracy']
val_accuracy = model_info_2.history['val_accuracy']

plt.plot(epochs, accuracy, label='Training accuracy')
plt.plot(epochs, val_accuracy, label='Validation accuracy')
plt.title('Training and validation accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epochs')
plt.legend()
plt.show();
```

Saving the Model

```
trick=
pd.read_csv('/content/drive/MyDrive/Profanity_dataset/test.csv',skipro
ws=[18522])
trick.head()

#/content/drive/MyDrive/Profanity_dataset/test.csv
model_1.save(filepath="/content/drive/MyDrive/Profanity_dataset/Model1
save.h5")

model_2.save(filepath="File_Path")
```

Loading Saved Model

```
loaded_model_1 =
keras.models.load_model(filepath="C:/Users/Ojas/Documents/Toxic
comment final/website/Model2save.h5")

loaded_model_2 = keras.models.load_model(filepath="File_Path")

loaded_model_1.summary()

Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 input_1 (InputLayer)        [(None, 200)]             0         

 embeddings (Embedding)      (None, 200, 300)          44675400  

 lstm_layer (LSTM)           (None, 200, 40)           54560     

 global_max_pooling1d (Globa  (None, 40)               0         
 lMaxPooling1D)                                                  

 dropout (Dropout)           (None, 40)                0         

 dense (Dense)               (None, 30)                1230      
 dropout_1 (Dropout)         (None, 30)                0

 dense_1 (Dense)             (None, 6)                 186

=================================================================
Total params: 44,731,376
Trainable params: 55,976
Non-trainable params: 44,675,400
_________________________________________________________________
```

```
loaded_model_1.optimizer
```

```
loaded_model_1.get_weights()
```

Generating the Output

LSTM

```
test_values_1 = loaded_model_1.predict([X_te], batch_size=1,
verbose=1)

sample_submission = pd.read_csv('File_Path')
test_values_1=pd.DataFrame(test_values_1,columns=['toxic',
'severe_toxic', 'obscene', 'threat', 'insult', 'identity_hate'])
submission = pd.DataFrame(sample_submission["id"])
combined_submission=pd.concat([submission,test_values_1],axis=1)
combined_submission.to_csv('File_Path', index=False)
```

LSTM-CNN

```
test_values_2 = loaded_model_2.predict([X_te], batch_size=1,
verbose=1)

sample_submission = pd.read_csv('File_Path')
test_values_2=pd.DataFrame(test_values_2,columns=['toxic',
'severe_toxic', 'obscene', 'threat', 'insult', 'identity_hate'])
submission = pd.DataFrame(sample_submission["id"])
combined_submission=pd.concat([submission,test_values_2],axis=1)
combined_submission.to_csv('File_Path', index=False)
```

Testing the Created Model

```
def toxicity_level(string):
    new_string = [string]
    new_string = tokenizer.texts_to_sequences(new_string)
    new_string = pad_sequences(new_string, maxlen=maxpadlen,
padding='post')

    prediction = loaded_model_1.predict(new_string) #(Change to
model_1 or model_2 depending on the preference of model type|| Model
```

```
1: LSTM, Model 2:LSTM-CNN)

    # print("Toxicity levels for '{}':".format(string))
    # print('Toxic:         {:.0%}'.format(prediction[0][0]))
    # print('Severe Toxic:  {:.0%}'.format(prediction[0][1]))
    # print('Obscene:       {:.0%}'.format(prediction[0][2]))
    # print('Threat:        {:.0%}'.format(prediction[0][3]))
    # print('Insult:        {:.0%}'.format(prediction[0][4]))
    # print('Identity Hate: {:.0%}'.format(prediction[0][5]))
    # print()
    return \
            {
                # "Toxic": str(result[0][0]),
                # "Very Toxic": str(result[0][1]),
                # "Obscene": str(result[0][2]),
                # "Threat": str(result[0][3]),
                # "Insult": str(result[0][4]),
                # "Hate": str(result[0][5]),
                # "Neutral": str(result[0][6])

        # "Toxicity levels for '{}':".format(string),
        "Toxic":         str(prediction[0][0]),
        "Severe Toxic":  str(prediction[0][1]),
        "Obscene":       str(prediction[0][2]),
        "Threat":        str(prediction[0][3]),
        "Insult":        str(prediction[0][4]),
        "Identity Hate": str(prediction[0][5])

 }


toxicity_level('go jump off a bridge jerk')

1/1 [==============================] - 4s 4s/step

{'Toxic': '0.72555375',
 'Severe Toxic': '0.03741559',
 'Obscene': '0.4122267',
 'Threat': '0.018686421',
 'Insult': '0.38985068',
 'Identity Hate': '0.042486805'}

toxicity_level('i will kill you')

1/1 [==============================] - 0s 66ms/step

{'Toxic': '0.6452304',
 'Severe Toxic': '0.043562613',
 'Obscene': '0.39242083',
 'Threat': '0.023309777',
 'Insult': '0.32728735',
 'Identity Hate': '0.04148046'}
```

```
toxicity_level('have a nice day')
```

```
1/1 [==============================] - 0s 74ms/step

{'Toxic': '0.055214457',
 'Severe Toxic': '0.0010245952',
 'Obscene': '0.016113892',
 'Threat': '0.0014553948',
 'Insult': '0.018033411',
 'Identity Hate': '0.0029318666'}
```

```
toxicity_level('fuck ofF!!')
```

```
1/1 [==============================] - 0s 55ms/step

{'Toxic': '0.9718479',
 'Severe Toxic': '0.25001442',
 'Obscene': '0.91191465',
 'Threat': '0.057230383',
 'Insult': '0.72776455',
 'Identity Hate': '0.18474112'}
```

```
toxicity_level('Hello, How are you?')
```

```
1/1 [==============================] - 0s 63ms/step

{'Toxic': '0.10698262',
 'Severe Toxic': '0.0019811825',
 'Obscene': '0.032743525',
 'Threat': '0.0024176212',
 'Insult': '0.03736955',
 'Identity Hate': '0.0044639404'}
```

```
toxicity_level('get the fuck away from me @sshole!!')
```

```
1/1 [==============================] - 0s 63ms/step

{'Toxic': '0.97336113',
 'Severe Toxic': '0.25326455',
 'Obscene': '0.9109043',
 'Threat': '0.06012877',
 'Insult': '0.7460838',
 'Identity Hate': '0.18878502'}
```

```
!pip install gradio
```

```
Defaulting to user installation because normal site-packages is not
writeable
Requirement already satisfied: gradio in c:\programdata\anaconda3\lib\
site-packages (3.16.2)
```

```
Requirement already satisfied: httpx in c:\programdata\anaconda3\lib\
site-packages (from gradio) (0.23.3)
Requirement already satisfied: aiofiles in c:\programdata\anaconda3\
lib\site-packages (from gradio) (22.1.0)
Requirement already satisfied: websockets>=10.0 in c:\users\ojas\
appdata\roaming\python\python39\site-packages (from gradio) (10.4)
Requirement already satisfied: altair>=4.2.0 in c:\programdata\
anaconda3\lib\site-packages (from gradio) (4.2.0)
Requirement already satisfied: python-multipart in c:\programdata\
anaconda3\lib\site-packages (from gradio) (0.0.5)
Requirement already satisfied: pyyaml in c:\programdata\anaconda3\lib\
site-packages (from gradio) (6.0)
Requirement already satisfied: pandas in c:\programdata\anaconda3\lib\
site-packages (from gradio) (1.4.4)
Requirement already satisfied: uvicorn in c:\programdata\anaconda3\
lib\site-packages (from gradio) (0.20.0)
Requirement already satisfied: numpy in c:\programdata\anaconda3\lib\
site-packages (from gradio) (1.21.5)
Requirement already satisfied: pydub in c:\programdata\anaconda3\lib\
site-packages (from gradio) (0.25.1)
Requirement already satisfied: matplotlib in c:\programdata\anaconda3\
lib\site-packages (from gradio) (3.5.2)
Requirement already satisfied: orjson in c:\programdata\anaconda3\lib\
site-packages (from gradio) (3.8.5)
Requirement already satisfied: pycryptodome in c:\programdata\
anaconda3\lib\site-packages (from gradio) (3.16.0)
Requirement already satisfied: pydantic in c:\users\ojas\appdata\
roaming\python\python39\site-packages (from gradio) (1.10.4)
Requirement already satisfied: markdown-it-py[linkify,plugins] in c:\
programdata\anaconda3\lib\site-packages (from gradio) (2.1.0)
Requirement already satisfied: fsspec in c:\programdata\anaconda3\lib\
site-packages (from gradio) (2022.7.1)
Requirement already satisfied: fastapi in c:\programdata\anaconda3\
lib\site-packages (from gradio) (0.89.1)
Requirement already satisfied: ffmpy in c:\programdata\anaconda3\lib\
site-packages (from gradio) (0.3.0)
Requirement already satisfied: aiohttp in c:\programdata\anaconda3\
lib\site-packages (from gradio) (3.8.1)
Requirement already satisfied: markupsafe in c:\programdata\anaconda3\
lib\site-packages (from gradio) (2.0.1)
Requirement already satisfied: pillow in c:\programdata\anaconda3\lib\
site-packages (from gradio) (9.2.0)
Requirement already satisfied: requests in c:\programdata\anaconda3\
lib\site-packages (from gradio) (2.28.1)
Requirement already satisfied: jinja2 in c:\programdata\anaconda3\lib\
site-packages (from gradio) (2.11.3)
Requirement already satisfied: typing-extensions in c:\programdata\
anaconda3\lib\site-packages (from gradio) (4.3.0)
Requirement already satisfied: toolz in c:\programdata\anaconda3\lib\
site-packages (from altair>=4.2.0->gradio) (0.11.2)
```

```
Requirement already satisfied: entrypoints in c:\programdata\
anaconda3\lib\site-packages (from altair>=4.2.0->gradio) (0.4)
Requirement already satisfied: jsonschema>=3.0 in c:\programdata\
anaconda3\lib\site-packages (from altair>=4.2.0->gradio) (4.16.0)
Requirement already satisfied: python-dateutil>=2.8.1 in c:\
programdata\anaconda3\lib\site-packages (from pandas->gradio) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\users\ojas\appdata\
roaming\python\python39\site-packages (from pandas->gradio) (2022.7)
Requirement already satisfied: multidict<7.0,>=4.5 in c:\programdata\
anaconda3\lib\site-packages (from aiohttp->gradio) (6.0.2)
Requirement already satisfied: aiosignal>=1.1.2 in c:\programdata\
anaconda3\lib\site-packages (from aiohttp->gradio) (1.2.0)
Requirement already satisfied: charset-normalizer<3.0,>=2.0 in c:\
programdata\anaconda3\lib\site-packages (from aiohttp->gradio) (2.0.4)
Requirement already satisfied: yarl<2.0,>=1.0 in c:\programdata\
anaconda3\lib\site-packages (from aiohttp->gradio) (1.8.1)
Requirement already satisfied: async-timeout<5.0,>=4.0.0a3 in c:\
programdata\anaconda3\lib\site-packages (from aiohttp->gradio) (4.0.2)
Requirement already satisfied: attrs>=17.3.0 in c:\programdata\
anaconda3\lib\site-packages (from aiohttp->gradio) (21.4.0)
Requirement already satisfied: frozenlist>=1.1.1 in c:\programdata\
anaconda3\lib\site-packages (from aiohttp->gradio) (1.2.0)
Requirement already satisfied: starlette==0.22.0 in c:\programdata\
anaconda3\lib\site-packages (from fastapi->gradio) (0.22.0)
Requirement already satisfied: anyio<5,>=3.4.0 in c:\programdata\
anaconda3\lib\site-packages (from starlette==0.22.0->fastapi->gradio)
(3.5.0)
Requirement already satisfied: rfc3986[idna2008]<2,>=1.3 in c:\
programdata\anaconda3\lib\site-packages (from httpx->gradio) (1.5.0)
Requirement already satisfied: sniffio in c:\programdata\anaconda3\
lib\site-packages (from httpx->gradio) (1.2.0)
Requirement already satisfied: httpcore<0.17.0,>=0.15.0 in c:\
programdata\anaconda3\lib\site-packages (from httpx->gradio) (0.16.3)
Requirement already satisfied: certifi in c:\programdata\anaconda3\
lib\site-packages (from httpx->gradio) (2022.9.24)
Requirement already satisfied: mdurl~=0.1 in c:\programdata\anaconda3\
lib\site-packages (from markdown-it-py[linkify,plugins]->gradio)
(0.1.2)
Requirement already satisfied: linkify-it-py~=1.0 in c:\programdata\
anaconda3\lib\site-packages (from markdown-it-py[linkify,plugins]-
>gradio) (1.0.3)
Requirement already satisfied: mdit-py-plugins in c:\programdata\
anaconda3\lib\site-packages (from markdown-it-py[linkify,plugins]-
>gradio) (0.3.3)
Requirement already satisfied: pyparsing>=2.2.1 in c:\programdata\
anaconda3\lib\site-packages (from matplotlib->gradio) (3.0.9)
Requirement already satisfied: cycler>=0.10 in c:\programdata\
anaconda3\lib\site-packages (from matplotlib->gradio) (0.11.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\programdata\
anaconda3\lib\site-packages (from matplotlib->gradio) (1.4.2)
```

```
Requirement already satisfied: fonttools>=4.22.0 in c:\programdata\
anaconda3\lib\site-packages (from matplotlib->gradio) (4.25.0)
Requirement already satisfied: packaging>=20.0 in c:\programdata\
anaconda3\lib\site-packages (from matplotlib->gradio) (21.3)
Requirement already satisfied: six>=1.4.0 in c:\programdata\anaconda3\
lib\site-packages (from python-multipart->gradio) (1.16.0)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\
programdata\anaconda3\lib\site-packages (from requests->gradio)
(1.26.11)
Requirement already satisfied: idna<4,>=2.5 in c:\programdata\
anaconda3\lib\site-packages (from requests->gradio) (3.3)
Requirement already satisfied: click>=7.0 in c:\programdata\anaconda3\
lib\site-packages (from uvicorn->gradio) (8.0.4)
Requirement already satisfied: h11>=0.8 in c:\programdata\anaconda3\
lib\site-packages (from uvicorn->gradio) (0.14.0)
Requirement already satisfied: colorama in c:\programdata\anaconda3\
lib\site-packages (from click>=7.0->uvicorn->gradio) (0.4.5)
Requirement already satisfied: pyrsistent!=0.17.0,!=0.17.1,!
=0.17.2,>=0.14.0 in c:\programdata\anaconda3\lib\site-packages (from
jsonschema>=3.0->altair>=4.2.0->gradio) (0.18.0)
Requirement already satisfied: uc-micro-py in c:\programdata\
anaconda3\lib\site-packages (from linkify-it-py~=1.0->markdown-it-
py[linkify,plugins]->gradio) (1.0.1)
```

```python
import gradio as gr
comment = gr.inputs.Textbox(lines=20, placeholder="Enter your
comment!!")

title = "Toxic Comment Classifier"
description = "This application uses a Long Short-Term Memory (LSTM)
Recurrent Neural Network (RNN) " \
              "model to predict the inappropriateness of a comment"

gr.Interface(fn=toxicity_level,
             inputs=comment,
             outputs="label",
             title=title,
             description=description,
             server_name="0.0.0.0",
             server_port=8080).launch(share = True, debug=True)
```

```
Running on local URL:  http://127.0.0.1:7860

Could not create share link, please check your internet connection.

<IPython.core.display.HTML object>

1/1 [==============================] - 0s 77ms/step
```

```python
'''
def greet(name):
  return "Hello " + name + "!"
  '''

#iface = gr.Interface(fn=greet, inputs="text", outputs="text")
#iface.launch(share=True)

'''
import gradio as gr

def image_classifier(inp):
    return {'cat': 0.3, 'dog': 0.7}

demo = gr.Interface(fn=image_classifier, inputs="image",
outputs="label")
demo.launch(share=True)
iface.close()

'''
```

Hyper parameter

'''Some of the popular optimization parameters are given below:

Learning Rate: The learning rate is the hyperparameter in optimization algorithms that controls how much the model needs to change in response to the estimated error for each time when the model's weights are updated. It is one of the crucial parameters while building a neural network, and also it determines the frequency of cross-checking with model parameters. Selecting the optimized learning rate is a challenging task because if the learning rate is very less, then it may slow down the training process. On the other hand, if the learning rate is too large, then it may not optimize the model properly. Note: Learning rate is a crucial hyperparameter for optimizing the model, so if there is a requirement of tuning only a single hyperparameter, it is suggested to tune the learning rate. Batch Size: To enhance the speed of the learning process, the training set is divided into different subsets, which are known as a batch. Number of Epochs: An epoch can be defined as the complete cycle for training the machine learning model. Epoch represents an iterative learning process. The number of epochs varies from model to model, and various models are created with more than one epoch. To determine the right number of epochs, a validation error is taken into account. The number of epochs is increased until there is a reduction in a validation error. If there is no improvement in reduction error for the consecutive epochs, then it indicates to stop increasing the number of epochs. Hyperparameter for Specific Models Hyperparameters that are involved in the structure of the model are known as hyperparameters for specific models. These are given below:

A number of Hidden Units: Hidden units are part of neural networks, which refer to the components comprising the layers of processors between input and output units in a neural network. It is important to specify the number of hidden units hyperparameter for the neural network. It should be between the size of the input layer and the size of the output layer. More specifically, the number of hidden units should be 2/3 of the size of the input layer, plus the size of the output layer.

For complex functions, it is necessary to specify the number of hidden units, but it should not overfit the model.

Number of Layers: A neural network is made up of vertically arranged components, which are called layers. There are mainly input layers, hidden layers, and output layers. A 3-layered neural network gives a better performance than a 2-layered network. For a Convolutional Neural network, a greater number of layers make a better model. Conclusion Hyperparameters are the parameters that are explicitly defined to control the learning process before applying a machine-learning algorithm to a dataset. These are used to specify the learning capacity and complexity of the model. Some of the hyperparameters are used for the optimization of the models, such as Batch size, learning rate, etc., and some are specific to the models, such as Number of Hidden layers, etc.'''

Word Embedding

https://towardsdatascience.com/deep-learning-for-nlp-word-embeddings-4f5c90bcdab5

'''Computers break everything down to numbers. Bits (zeros and ones) more specifically. What happens when a software inside a computer (like a Machine Learning algorithm for example) has to operate or process a word? Simple, this word needs to be given to the computer as the only thing it can understand: as numbers.

In NLP, the most simple way to do this is by creating a vocabulary with a huge amount of words (100.000 words let’s say), and assigning a number to each word in the vocabulary.

The first word in our vocabulary (‘apple’ maybe) will be number 0. The second word (‘banana’) will be number 1, and so on up to number 99.998, the previous to last word (‘king’) and 999.999 being assigned to the last word (‘queen’).

Then we represent every word as a vector of length 100.000, where every single item is a zero except one of them, corresponding to the index of the number that the word is associated with.

Vector representations of some of the examples from the previous paragraphs. This is called one-hot encoding for words.

The one-hot encoding have various different issues related with efficiency and context, that we will see in just a moment.

Word embeddings are just another form representing words through vectors, that successfully solve many of the issues derived from using a one-hot encoding by somehow abstracting the context or high-level meaning of each word.

The main takeaway here is that word embeddings are vectors that represent words, so that similar meaning words have similar vectors.'''

'''If we then plotted these word vectors in a 3 dimensional space, we would get a representation like the one shown in the following figure, where each axis represents one of the dimensions that we have, and the icons represent where the end of each word vector would be.

Representation of our one hot encoded word vectors in a 3 dimensional space. As we can see, the distance from any vector (position of the icons) to all the other ones is the same: two size 1 steps in different directions. This would be the same if we expanded the problem to 100.000 dimensions, taking more steps but maintaining the same distance between all the word vectors.

Ideally, we would want vectors for words that have similar meanings or represent similar items to be close together, and far away from those that have completely different meanings: we want apple to be close to banana but far away from king.

Also, one hot encodings are very inefficient. If you think about it, they are huge empty vectors with only one item having a value different than zero. They are very sparse, and can greatly slow down our calculations.

Word embeddings solve these problems by representing each word in the vocabulary by a fairly small (150, 300, 500 dimensional) fixed size vector, called an embedding, which is learned during the training.

These vectors are created in a manner so that words that appear in similar contexts or have similar meaning are close together, and they are not sparse vectors like the ones derived from one-hot embeddings.

Lastly, as we can see in the word embedding vectors, they usually have a smaller size (2 in our example, but most times they have 150, 200, 300, or 500 dimensions) and are not sparse, making calculations with them much more efficient than with one-hot vectors

: the algorithms learn similar word embedding for words that appear many times in similar contexts by guessing missing words in a huge corpus of text sentences.

An embedding matrix E (the matrix that translates a one hot embedding into a word embedding vector) is calculated by training something similar to a language model (a model that tries to predicts missing words in a sentence) using an Artificial Neural

Network to predict this missing word, in a similar manner to how the weights and biases of the network are calculated.

'''