INTERNSHIP: PROJECT REPORT

Internship Project Title	Automate Emotion Analysis of Textual comments and Feedback.
Project Title	To develop deep learning algorithms with an aim to detect different types of emotion contained in a collection of English sentences or a large paragraph and accurately predict the overall emotion of the paragraph.
Name of the Company	Tata Consultancy Service
Name of the Industry Mentor	Mr. Debashis Roy
Name of the Institute	B. K. Birla College, Kalyan.

Start Date	End Date	Total Effort (hrs.)	Project Environment	Tools used
05/12/2021	14/12/2021	10	Google Colab, Chrome, Windows 10	Python 3

Project Synopsis:

Textual Emotion Analysis (TEA) is the task of extracting and analyzing user emotional states in texts. TEA not only acts as a standalone tool for information extraction but also plays an important role for various Natural Language Processing (NLP) applications, including e-commerce, public opinion analysis, big search, information prediction, personalized recommendation, healthcare, and online teaching. Usually, emotions are expressed as joy, sadness, anger, surprise, hate, fear, etc. Recognizing this type of emotion from a text written by a person plays an important role in applications such as chatbots, customer support forum, customer reviews etc.

Solution Approach:

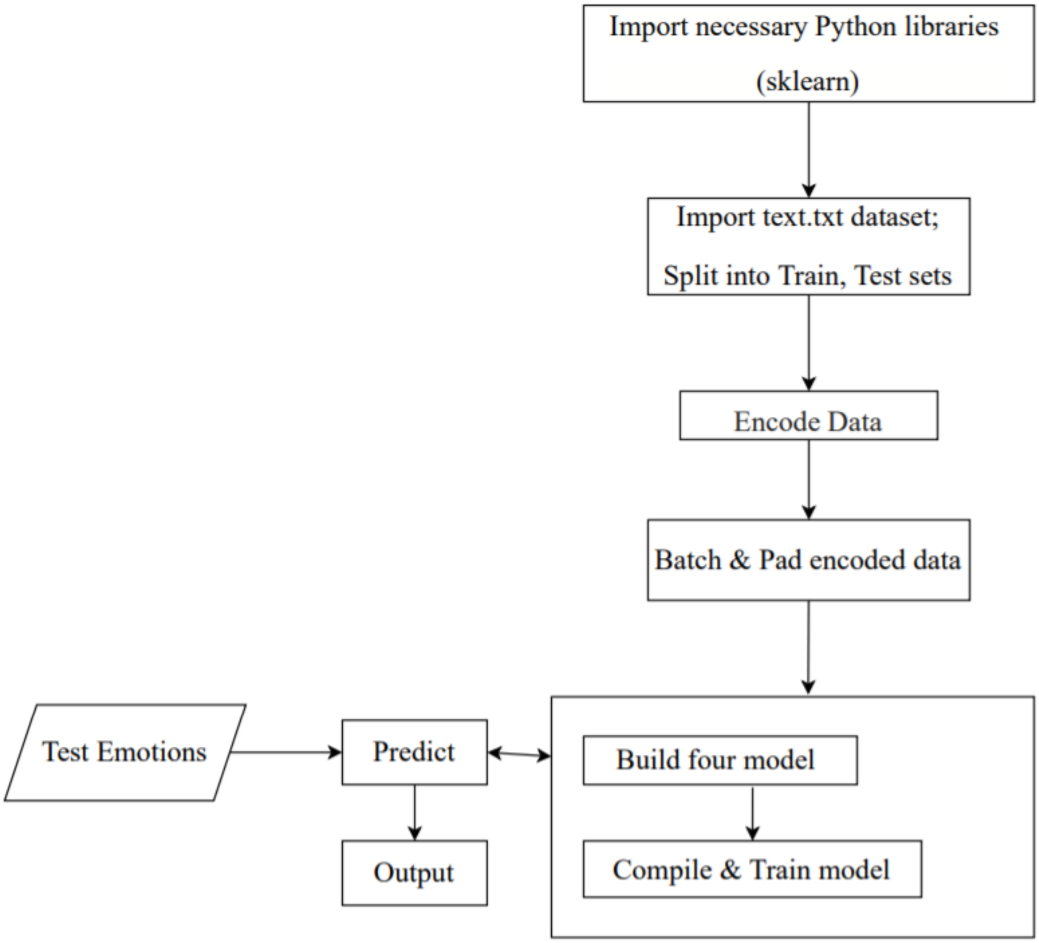
Deep Learning methods improved on statistical learning results in many fields. At present, a neural network-based NLP framework has achieved new levels of quality and become the dominating technology for NLP tasks, such as sentiment analysis, machine translation, and question answering systems. There are numerous emotions in textual conversations. As people use text messaging applications (such as Wechat, Facebook) and conversation agents (such as Amazon Alexa) to communicate more frequently than ever, context emotion detection in text is becoming more important to emotion analysis. If we can effectively detect the emotion in a conversation, it has great commercial value (e.g., online customer service of an e-commerce platform).

Assumptions:

For detecting emotions from the text, it will perform a few steps that will start with preparing the data. Then the next step will be tokenization where the textual data will be converted into tokens to identify the emotional words. These emotional words will be the keyword to classify the emotions of a text. Next, It will frame this task in such a way that a text will be taken as an input and the emoji that represents the emotions in that text is generated as the output.

Project Diagrams:

Project Work Flow:



Outcome:

Trained four models and then choose the model that works best on the training and testing sets:

```
svc = SVC()
lsvc = LinearSVC(random_state=123)
rforest = RandomForestClassifier(random_state=123)
dtree = DecisionTreeClassifier()

clifs = [svc, lsvc, rforest, dtree]

# train and test them
print("| {:25} | {} | {} |".format("Classifier", "Training Accuracy", "Test Accuracy"))
print("| {} | {} | {} |".format("-"*25, "-"*17, "-"*13))
for clf in clifs:
    clf_name = clf.__class__.__name__
    train_acc, test_acc = train_test(clf, X_train, X_test, y_train, y_test)
    print("| {:25} | {:17.7f} | {:13.7f} |".format(clf_name, train_acc, test_acc))
```

The OUTPUT is:

```
| Classifier             | Training Accuracy | Test Accuracy |
| ------------------------ | ---------------- | ------------ |
| SVC                    |        0.8016377 |     0.4939840 |
/usr/local/lib/python3.7/dist-packages/sklearn/svm/_base.py:1208: ConvergenceWarning: Liblinear failed to converge, increase the number of iterations.
  ConvergenceWarning,
| LinearSVC              |        0.9901404 |     0.5467914 |
| RandomForestClassifier    |        0.9988302 |     0.5434492 |
| DecisionTreeClassifier    |        0.9988302 |     0.4532086 |
```

Detecting Emotion

Assigning an emoji to each label that is emotions in this problem, then I'll write 4 input sentences, then I'll use our trained model to take a look at the emotions of our input sentences:

```
l = ["joy", 'fear', "anger", "sadness", "disgust", "shame", "guilt"]
l.sort()
label_freq = {}
for label, _ in data:
   label_freq[label] = label_freq.get(label, 0) + 1

# print the labels and their counts in sorted order
for l in sorted(label_freq, key=label_freq.get, reverse=True):
   print("{:10}({})  {}".format(convert_label(l, emotions), l, label_freq[l]))
```

The OUTPUT is:

```
joy      (1. 0. 0. 0. 0. 0. 0.)  1084
anger    (0. 0. 1. 0. 0. 0. 0.)  1080
sadness   (0. 0. 0. 1. 0. 0. 0.)  1079
fear     (0. 1. 0. 0. 0. 0. 0.)  1078
disgust   (0. 0. 0. 0. 1. 0. 0.)  1057
guilt    (0. 0. 0. 0. 0. 0. 1.)  1057
shame    (0. 0. 0. 0. 0. 1. 0.)  1045
```

```
emoji_dict = {"joy":"😂", "fear":"😱", "anger":"😠", "sadness":"😥", "disgust":"😒", "shame":"😳", "
guilt":"😳"}
t1 = "This looks so impressive"
t2 = "I have a fear of dogs"
t3 = "My dog died yesterday"
t4 = "I don't love you anymore..!"

texts = [t1, t2, t3, t4]
for text in texts:
   features = create_features(text, nrange=(1, 4))
   features = vectorizer.transform(features)
   prediction = clf.predict(features)[0]
   print( text,emoji_dict[prediction])
```

The OUTPUT is:

```
This looks so impressive 😒
I have a fear of dogs 😱
My dog died yesterday 😢
I don't love you anymore..! 😂
```

Exceptions considered:

Accuracy and Loss Function values vary slightly on every compilation of the model. This may give a slight difference in the prediction value which may impact the neutral emotions.

Enhancement Scope:

Emotions detection is a growing research field. A lot of work has already been done on sentiment analysis, facial recognition and speech recognition. For now, I have implemented a supervised model to detect the emotion in a given sentence. Further My project aim to recognize emotions in a sentence and paragraph. Determining overall impact of the sentence can help in various fields especially in marketing. Consumer state of mind can be read by analyzing what they write. More features (emotions) can be added to detect separately like recognizing sarcasm. Further it aims to show the relationship between valence, arousal and Dominance.

Link to Code and executable file:

https://colab.research.google.com/drive/1L0wBF4uKfjGHEoLvqvftDV_KUAnC8kmM#scrollTo=xNrer2gIF7_V