OTT V/S THEATRES: ENSEMBLE MODEL OF OPINION MINING ON SOCIAL MEDIA

HIRAN CHRISTA BEL BOMMELLA

18951A0552

OTT V/S THEATRES: ENSEMBLE MODEL OF OPINION MINING ON SOCIAL MEDIA

A Project Report

submitted in partial fulfillment of the

requirements for the award of the degree of

Bachelor of Technology

in

Computer Science and Engineering

by

HIRAN CHRISTA BEL BOMMELLA

18951A0552

Department of Computer Science and Engineering

INSTITUTE OF AERONAUTICAL ENGINEERING

(Autonomous)

Dundigal, Hyderabad-500043, Telangana

May,2022

CERTIFICATE

This is to certify that the project report entitled **OTT v/s THEATRE: Ensemble Model of Opinion Mining on Social Media** submitted by **Ms. Hiran Christa Bel Bommella** to the Institute of Aeronautical Engineering, Hyderabad in partial fulfilment of the requirements for the award of the Degree Bachelor of Technology in **Computer Science and Engineering**, is a Bonafide record of work carried out by her under my guidance and supervision. The contents of this report, in full or in parts, have not been submitted to some other Institute for the respect of any Degree.

Supervisor	**Head of the Department**
Dr. J. Sirisha Devi	Dr. B. J. D. Kalyani
Associate Professor, CSE	Associate Professor, CSE

Date**:**

APPROVAL SHEET

This project report entitled **OTT v/s THEATRE: Ensemble Model of Opinion Mining on Social Media** by **Hiran Christa Bel Bommella** is approved for the award of the Degree Bachelor of Technology in **Computer Science and Engineering**.

Examiners **Supervisor**

Principal

Date:

Place: Hyderabad

DECLARATION

I certify that

a. The work contained in this report is original and has been done by me under the guidance of my supervisor(s).
b. The work has not been submitted to any other Institute for any degree or diploma.
c. I have followed the guidelines provided by the Institute in preparing the report.
d. I have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.
e. Whenever I have used materials (data, theoretical analysis, figures, and text) from other sources, I have given due credit to them by citing them in the text of the report and giving their details in the references. Further, I have taken permission from the copyright owners of the sources, whenever necessary.

Place: Hyderabad **Signature of the student**

Date: **Roll No:**

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task would be in complete without the mention of the people who made it possible and whose constant guidance and encouragement crown all the efforts with success. I thank our college management and respected Sri M. Rajashekar Reddy, Chairman, IARE, Dundigal for providing me the necessary infrastructure to carry out the project work.

I express my sincere thanks to Dr. L.V. Narasimha Prasad, Professor and Principle who has been a reliable source of information for my work, and Dr. B. J. D. Kalyani, Professor and Head, Department of CSE, for extending her support to carry on this project work.

I am especially thankful to our supervisor Dr. J. Sirisha Devi, Associate Professor, Department of CSE, for her internal support and professionalism who helped me in shaping the project into successful one.

I take this opportunity to express my thanks to everyone directly helped me in bringing this effort to present form.

ABSTRACT

Opinion Mining has gained interest lately due to the uptick in social media. People began expressing public and private outlooks on varied subjects and sharing their perspectives on online forums. In this report, we concentrate on the implementation of opinion classification with a long short-term memory (LSTM) network with a convolutional neural network (CNN) deep learning model. Moreover, we present an ensemble model (LSTM-CNN) for opinion mining. This paper reviews the public mindset towards the OTT platforms and Movie Theatre with an LSTM-CNN ensemble model of Opinion Mining. Thus, Results categorize users' viewpoints through comments into, "*Ott biased"* and "*Movie Theatre biased"* illustrated in a pie chart.

Keywords: Opinion mining, Ensemble model, Ott, Theatres, long short-term memory, convolutional neural networks.

CONTENTS

Title Page	**I**
Certificate by the Supervisor	II
Approval Sheet	III
Declaration	IV
Acknowledgement	V
Abstract	VI
Contents	VII
List of Figures	IX
List of Tables	X
Abbreviations	XI
Chapter 1 Introduction	1
1.1 Background and Basics	1
1.2 Problem Statement	3
1.3 Objective	3
1.4 Organization of Project Report	4
Chapter 2 Literature Review	5
Chapter 3 Methodology	9
3.1 Project Planning & Management	9
3.1.1 Detail System Requirement Specifications	9
3.1.1.1 System Overview	9
3.1.1.2 Functional Requirements	9
3.1.1.3 Non-Functional Requirements	10

3.1.1.4 Deployment Environment 11

3.1.2 Project Process Modelling 12

3.1.2.1 Ensemble Model 12

3.2 Analysis and Design 16

3.3 UML Diagrams 24

3.4 Implementation and Coding 29

3.4.1 Operational Details 29

3.4.2 Code Listing 30

Chapter 4 Results and Discussions 45

Chapter 5 Conclusion and Future Scope of Study 49

References 50

LIST OF FIGURES

Figure	Title	Page
3.1	Ensemble Bagging	13
3.2	Ensemble Boosting	14
3.3	Ensemble Stacking	15
3.4	Architecture of CNN	17
3.5	Architecture of LSTM	19
3.6	Flow Chart	21
3.7	data flow diagram for ensemble model	22
3.8	Work-flow Diagram	23
3.9	Use Case Diagram	24
3.10	Sequence Diagram	25
3.11	Activity Diagram	26
3.12	Class Diagram	27
3.13	Interface Diagram	27
3.14	Entity Relationship Diagram (ERD)	28
4.1	Executing the main file	47
4.2	Results	48

LIST OF TABLES

Table	Title	Page
4.1	Raw Dataset	45
4.2	Snapshot of preprocessed data after removing null values	46

LIST OF ABBREVIATIONS

LSTM	**Long-Term Short-Term Memory**
CNN	convolutional neural network
OTT	Over the top
NB	Naïve Bayes
ME	Maximum Entropy
SVM	Support Vector Machine
SRS	System Requirement Specification

CHAPTER 1

INTRODUCTION

1.1 Background and Basics

Sentiment Analysis or Opinion Mining is the text analysis technique that can analyse the text/tweets on social media. With this natural language processing (NLP) technique, we can determine if the data is positive, negative, or neutral in the state. One of the applications of these techniques involves the breakdown of social media messages automatically, based on the sentiments and feelings expressed. It plays a crucial role in commerce and research fields as it applies to any domain. So, this analysis is performed on the textual data to help monitor the businesses with their products and brands through customer feedback. Sentiment Analysis aids in improving decision-making, customer satisfaction, etc.

Sorting public statements or surveys manually is a perplexing task. A large amount of data is mined to obtain the necessary analytics. Hence, sentiment mining assists in analyzing unstructured data with greater efficiency without costing a lot of money. Based on your requirement, you can represent it according to your classifications to suit your conditions. Opinion Mining permits enterprises to comprehend their customer’s intents by mining their feedback, replies, and social media discussions.

Many generic errors are triggered while operating on a single model while forecasting. To overcome this pitfall ensemble classifiers are introduced. This report seeks to convey advanced sentiment-primary techniques and tools to get analytics from the social forums.

Corresponding to a single model, ensemble learning also permits better accuracy General idea is to understand a collection of experts (classifiers) and is permitted to choose. Ensemble approaches hold more elevated predictive accuracy, corresponding to the models. Ensemble techniques are useful when there is unstructured knowledge in the dataset; Since various models converge to deal with this type of data. So, you will utilize ensemble learning ways once you need to boost the performance of machine learning models. as an instance to decrease the mean absolute error for regression models or extend the preciseness of classification models. The whole also results in a very stable model.

Online reviews are a generic way for users to share their thought on the services/products. With so much data, humans cannot read every review. With the changing lifestyles of people and the increasing use of smartphones with affordable internet services, OTT platforms are becoming increasingly widespread since the pandemic. The combat between cinema and Ott has both pros and cons. Streaming apps permit you to have entertainment in your comfort zone while watching a movie in a theatre has another level of joy to capture the cinematic experience. So, to find the people's perspectives regarding them, we need to analyse their comments with the LSTM-CNN ensemble model.

1.2 Problem Statement

The analysis of sentiments on data from social networks, namely Twitter or Facebook, is a research area with growing demands today. Even though significant work was accomplished, numerous challenges remain, including applying techniques developed for various data and certain data fields, decreasing time intervals, and enhancing model accuracy. In recent years, deep learning models have become widespread in the field of opinion mining, where their significant possibilities are showing.

Several studies have focused solely on assembling a single model from a single (or a few) datasets in a relevant domain, for example, medical research, marketing tactics, and financial forecasts. When applied to specific fields, a single machine learning model is well-grounded So, each deep learning technique has its pros and cons. The approach of combining two (or more) methods is introduced as a means of incorporating the advantages of both and thus fills some shortcomings of individual methods.

1.3 Objective

The purpose of this analysis is first, to study the ensemble model and secondly, to examine the data according to the polarity of, "*individuals who favor OTT"* and "*individuals who favor Movie Theatre*" and deliver the outcome utilizing an ensemble LSTM-CNN model of Opinion Mining.

1.4 Organization of Project Report

The whole report pursuits on finding the outlooks on the Theatres and Ott platforms based on the user's data.

The first chapter begins with an introduction to opinion mining and its requirements. In this section, the background and basics are stated along with its problem statement and objectives.

The second chapter talks about relevant literature applied in the research studies.

The third chapter presents the methodology. This chapter deals with project planning and management.

The fourth chapter confers the results.

The fifth chapter concludes the report along with the future scope. Thus, we conclude the document in the provided sequence of the chapters.

CHAPTER 2

LITERATURE REVIEW

2.1 Sentiment analysis based on a social media customised dictionary

Authors: Milene Dias Almeida and Vinicius Mothé Maia and Roberto Tommasetti and Rodrigo de Oliveira Leite

The authors studied on developing a customized dictionary based on perceptual mapping, assembling the sentiment indicator according to a business enterprise or an organization. It works on Naïve Bayes classifier. Their work mainly focuses on the business enterprises where these markers have a facility to create their own customized dictionary according to their requirements.

2.2 Twitter sentiment analysis

Authors: Sarlan, A., Nadam, C., & Basri, S.

Most studies are conducted on the Twitter datasets as Twitter is the best online forum to collect users' sentiments. Opinion mining is a helpful tool for analyzing the user's tweets to get the required insights. They worked on the twitter sentiments and researched on the microblog for a product based on the customer opinion. Their work is related to the ecommerce websites to get the insights for a product based on the user review on the product.

2.3 Tweet Classification and Sentiment Analysis on Metaverse Related Messages

Authors: Özgür Ağrali and Ömer Aydın

Özgür Ağrali and Ömer Aydın examined the Twitter sentiments about the user's opinion on the Metaverse when Mark Zuckerberg revealed to revise its name to Meta. An open-source, Vader (Valence Aware Dictionary and Sentiment Reasoner) was used to analyse the twitter sentiments.

2.4 Improving Sentiment Analysis for Social Media Applications Using an Ensemble Deep Learning Language Model

Author: Alsayat A

Alsayat researched the hybrid ensemble model using long-term short-term memory (LSTM) network over the Twitter datasets. This paper introduces custom ensemble learning with deep learning, and data polarization is done on the CrowdFlower platform. The experiment is performed on the three distinct types of datasets through various hidden layers and neurons as model parameters.

2.5 An ensemble approach to stabilize the features for multi-domain sentiment analysis using supervised machine learning.

Authors: Ghosh M and Sanyal G

Ghosh and Sanyal used feature selection methods namely Gini Index, IG, and Chi-Square are evaluated on three datasets (Electronic reviews, kitchen reviews, and IMDb movie reviews) individually as well as an integrated approach. At last trained the data with LR (logical regression), SMO, RF, and MNB classifiers with this feature vector model.

2.6 Mining Using Ensemble Classification Models

Authors: Matthew Whitehead and Larry Yaeger

Matthew Whitehead and Larry Yaeger worked on the problem of classifying the written text of public opinions. A comparative study was conducted on the different models and the most accurate model was opted to work on the further results. They worked on the three different classifiers namely, SVM, NB, and ME. White et al. researched on the SVM. In the paper, they proposed four different ensemble techniques such as bagging, random subspace, and boosting. Through random subspace best execution was accomplished. This paper reported the accuracy of ensemble models that gives higher efficiencies.

2.7 Applications and challenges for sentiment analysis

Authors: Vohra M.S. and Teraiya J

Vohra and teraiya worked on the sentiments of customers' reviews about the services and products. Their work was focused on the applications and challenges that are faced during the process of sentiment analysis. A survey was conducted on the online forums.

2.8 Real-Time Sentiment Analysis of Twitter Streaming data for Stock Prediction

Authors: Sushree Das, Ranjan Kumar Behera, Mukeshkumar, Santanu Kumar Rath

Tweet feel, an app used to examine the real-time tweets. It is an exceptional Application. Everyone is aware of stocks and are ready to buy them but before buying the stocks, people need some guidance. So, with the help of the sentiment analysis, a business enterprise stock's prices can be predicted. Twitter is an ultimate source of vast data and with this data a real time streaming can be conducted for the stock predictions with RNN.

2.9 Ensemble of feature sets and classification algorithms for sentiment classification

Authors: Rui Xia, Chengqing Zong, and Shoushan Li

An ensemble framework combining different classifiers and subsets of algorithms was designed to make comparative studies. Firstly, feature sets were designed based on the speech and the text based. Secondly, with the well-known classifiers namely, NB, SVM and ME are applied. They are considered as base classifiers and based on the ensemble strategies; they are experimented on the five varied datasets.

2.10 Twitter Sentiment Analysis using lstm-cnn models

Author: Pedro M. Sosa

Research was conducted on the LSTM and CNN networks to test their accuracies by using the pretrained embedded words. These models were trained on the twitter datasets. Four models were designed to study their efficiencies. So, each model is trained on the data and forecast the accuracy.

CHAPTER 3

METHODOLOGY

3.1 Project Planning and Management

3.1.1 Detail System Requirement Specification (SRS)

In this section, system requirements are articulated. SRS is considered as the ground for effort analyses and task schedules.

3.1.1.1 System Overview

Opinion Analysis is a text-based analysis that is carried out on the users' outlooks from YouTube comments and replies. The data is collected using the YouTube API and trained on the ensemble model on visual studio code IDE. The model forecasts the results on a pie chart.

3.1.1.2 Functional Requirements

Cleansing the data plays a vital role in opinion analysis. The raw data is unsupervised data with noise and inconsistency. Datasets must be stored in proper file format. We stored the dataset in a .csv file.

The following are used to design the Ensemble Model:

2 Long short-term memory (LSTM)

3 Convolution Neural Networks (CNN)

The working of these networks demonstrated in the coming sections.

The ensemble model built using LSTM and CNN is trained on the data sets collected from the YouTube forum. Additional measures are taken to optimize them to enhance their precision. They enclose data cleaning and data preprocessing.

The models were assessed according to their precisions and it was marked that the LSTM-CNN is the most precise out of other models with 75.2% efficiency. It has opted for the prognosis of the datasets on Ott and theatres.

Inputs: raw user dataset

Outputs: result, accuracy

3.1.1.3 Non-Functional Requirements

Performance Requirements

- The system needs to be reliable.
- Check every module before executing the code.
- Hardware and Software interfaces are crucial while performing a machine learning project.
- Accuracy differs if there are missing data.
- The model with greater efficiency should be chosen. As LSTM-CNN has the highest value of 75.2%, it was chosen.

Safety and Security Requirements:

- This project uses the data of users only for the forecasts on Ott and theatres.
- Data will be safe and secure.

3.1.1.4 Deployment Environment

Hardware Requirements

Minimum hardware requirements are:

Ram	:	16 GB (8GB is good but do not perform efficiently)
Processor	:	core intel i5 or more
GPU	:	1-2 NVidia cards (to set up a virtual environment)

Software Requirements:

Operating System	:	Windows 10
IDE	:	Visual Studio Code /Anaconda2 /Spyder
Python Modules	:	TensorFlow, NLTK package, Ipython, Batchgen, Matplotlib

Other Requirements:

APIs & Services	:	YouTube API (to extract YouTube comments)
Completeness	:	All external libraries including their respective license should be documented.

3.1.2 Project Process Modeling

3.1.2.1 Ensemble Model

Ensemble learning is a strategy where different models join to envision the insights.

To design the model at least two varied algorithms are mandated despite the way that there is intently an endless number of ways through which it tends to be done however at most just three procedures are applied progressively they are undeniable due to their effortlessness of execution and accomplishments on a wide extent of envision prototype issues.

Typical procedures applied in Ensemble Modeling:

there are three procedures employed in ensemble learning; they are additionally called “meta-algorithms”

1. Stacking Boosting
2. Bagging
3.

Each algorithm compromises of two stages:

1. Creating an allocation of straightforward ML models on subsets of the actual data.
2. Integrating the distribution into the “cluster” model.

Bootstrap Aggregating:

Bootstrap Aggregation or bagging, is a strategy used to lower the variance of the training data of the forecasting model by producing the hybrids with recurrences from the actual datasets to get multi-sets. So, it is more useful in adjusting the model for anticipated results by reducing the conflicts rather than expanding the training sets. The subsets are randomly created.

The two fundamental components in bagging are bootstrap and aggregation as it is the abbreviation of "Bootstrap AGGregatING."

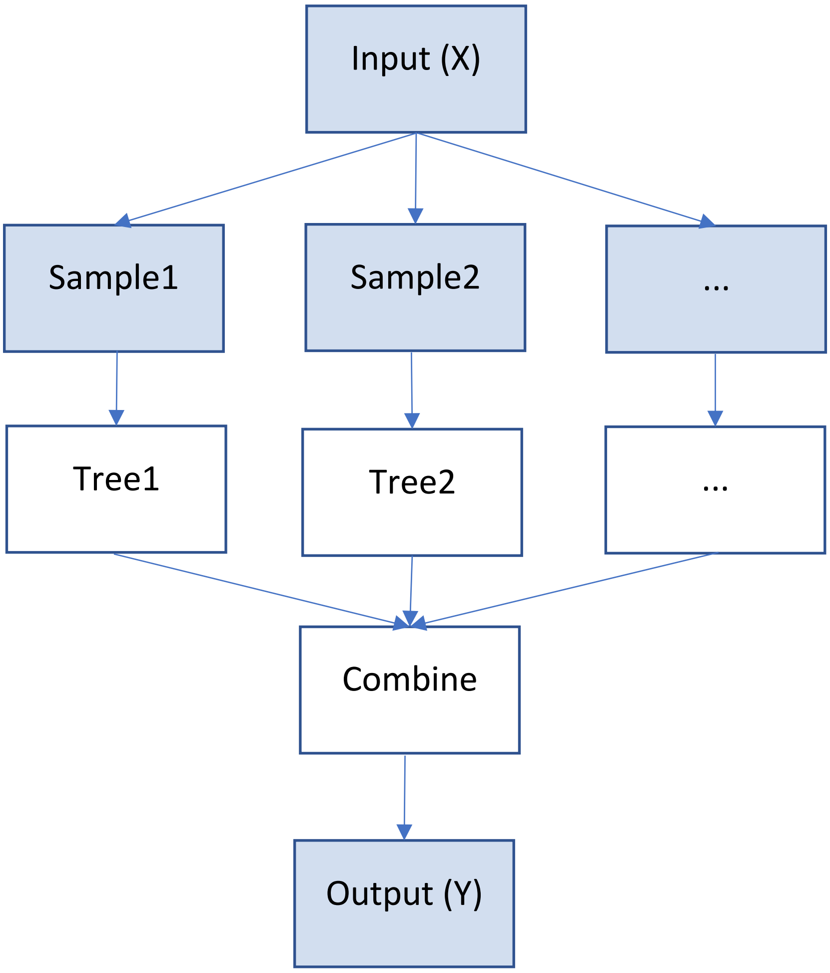

Fig (3.2) Ensemble Bagging

Boosting:

Boosting is a method with two phases, Firstly, the subsets of the initial data create a sequence of mediocre-performing prototypes and afterward "boosts" their execution by merging them jointly based on the bulk voting. Secondly, the subset production is not arbitrary and relies on the previous fits.

Hence, each new subset holds the components that are prone to be misconceptions by last fits. The primary concept of it is to rectify the forecast mistakes.

With the help of weak classifier predictions, a strong classifier is generated by uniting them using an averaging or poll.

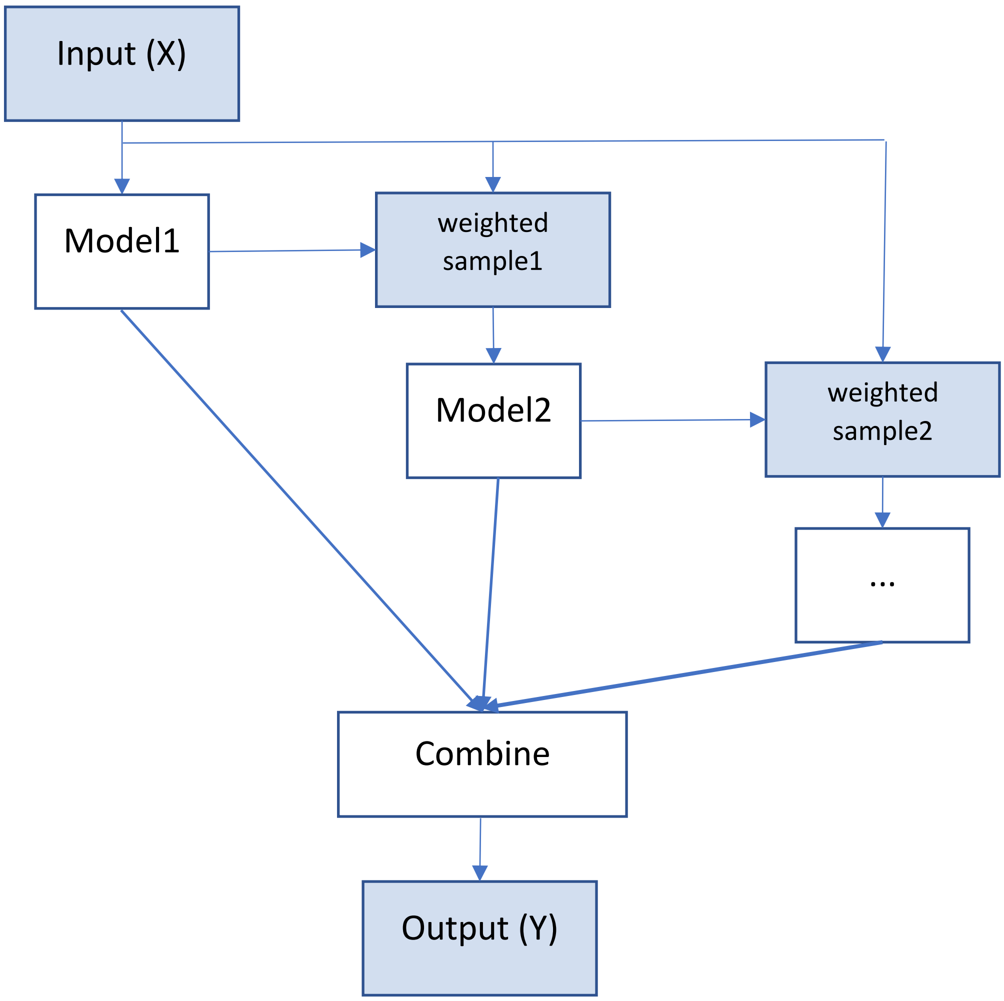

Fig (3.3) Ensemble Boosting

Stacking:

It is also known as Stacked Generalization. Stacking presents a meta-level. Individual learners are first-level or level-0 models and merging models for forecasts are level-1 models or meta-learner. With stacking, classifiers can be easily sorted into dependable and non-dependable.

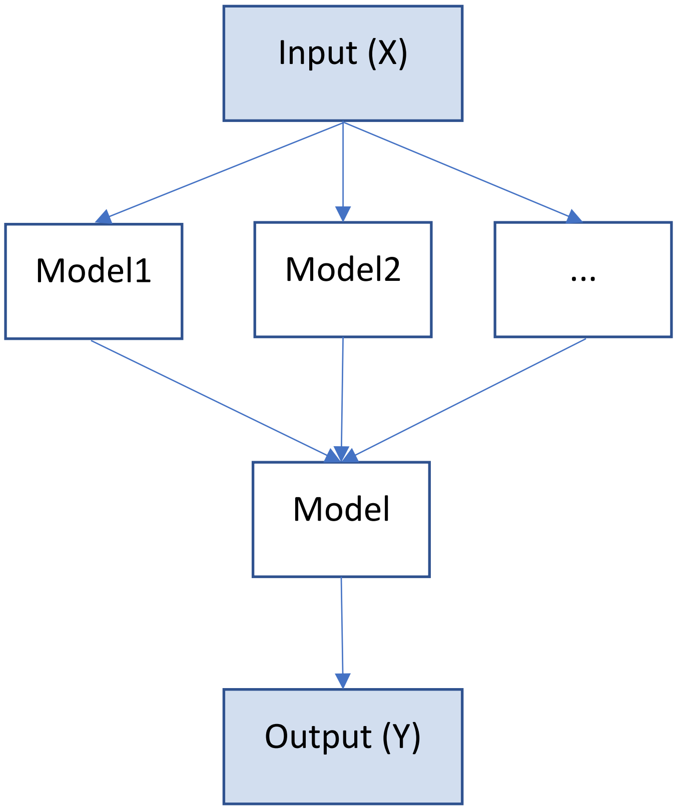

Fig (3.1) Ensemble Stacking

Advantages of Ensemble Learning:

- Rather than a single model, these have greater accuracy in forecasting.
- If dataset has structured and unstructured data; it is easier to get insights by using the ensemble learning.
- It is exceedingly rare to get a underfitted or overfitted model when ensembling is used.

3.2 Analysis and Design

We selected LSTM and CNN to design an ensemble model because of their unique features that are mentioned below:

Convolutional Neural Networks (CNNs):

CNN is precisely a kind of neural network; it has distinctive convolutions with respect to different networks. To commit picture classifying, CNN runs through every intersection, proportion, and vector of a pixel matrix. Functioning with these all attributes of a matrix completes CNN better feasible to data of matrix form.

Primarily CNNs are designed for image classification that can distinguish pictures of dogs vs. Pictures of cat. But here, we use these networks to catches negative phrases. For example, *“don’t like” / “don’t want”*.

1. I **do not like** chocolate.
2. I **do not** want to eat.

The channels in CNNs can assist with distinguishing important examples in text information - bigrams, trigrams, or n-grams (bordering succession of n words) contingent upon bit size. Since CNN's are interpretation invariant, they can distinguish these examples regardless of their situation in the sentence. The hierarchy of words is not that significant in-text characterization, so CNN can play out this errand successfully. Each channel/bit distinguishes a particular segment, for example, presuming the statement includes optimistic ('great', 'astonishing') or contrary ('awful', 'hate') terms on account of opinion mining.

Like the opinion analysis approach, most sentiment classification is set by the presence or non-attendance of a few key expressions present at any place in the statements. With CNN, it can be effectively fitted which is great at drawing nearby and position-invariant elements from the information. Henceforth we have picked CNNs for our project.

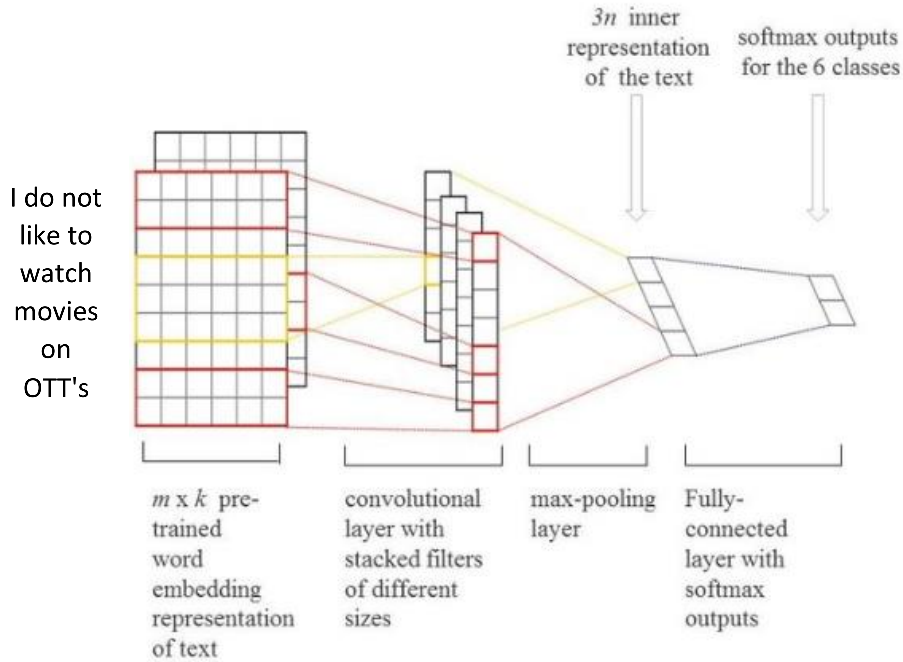

Fig (3.4) Architecture of CNN

The text data can be regarded as sequel data which is identical to the data in time series, that is a (1-D) one-dimensional layer. To function on TextCNN, a 1-D layer in CNN is employed. The possibility of the model is something similar, yet the aspect of convolution layers and data class varied. To accomplish with CNN on text classifying, we require a 1-D convolution and a word embedded layer.

Long-Short Term Memory (LSTM):

LSTMs are the networks that retain the data. It has a recollection from which the decisions are made. LSTM networks can pinpoint the changes in the statement,

> "I love watching movies in theatres but I ended up hating it."

With this illustration, it holds two intentions if we divide the sentence into two parts; By considering the part one, he used to love watching movies, typically other models capture this meaning which contradicts the presumptions but with LSTM, we can capture the true sentiments of the whole sentence. LSTM is an exceptional sort of RNN, which exhibits remarkable execution on a huge assortment of issues.

LSTM v/s RNN

Consider, that you have the undertaking of altering specific data in a schedule. To do this, an RNN totally changes the current information by applying a capacity. While LSTM produces negligible alterations to the data by primary expansion or replication that move via cell states. This is the way LSTM overlooks and recalls things specifically, which constructs it a refinement over RNNs.

In any claim, they have two drawbacks detonating inclination and disappearing slope that make them excess.

Here; LSTM illustrates recollection units known as cell states to take cautiousness of this issue the planned cells might be regarded as differentiable memory.

Structure of Long-Short Term Memory (LSTM):

Long-Short Term Memory (LSTM) carries a network of chains that includes

1. Four neural networks
2. Extra memory chunks are known as cells.

Data is maintained by them and the memory rules are achieved by the given gates.

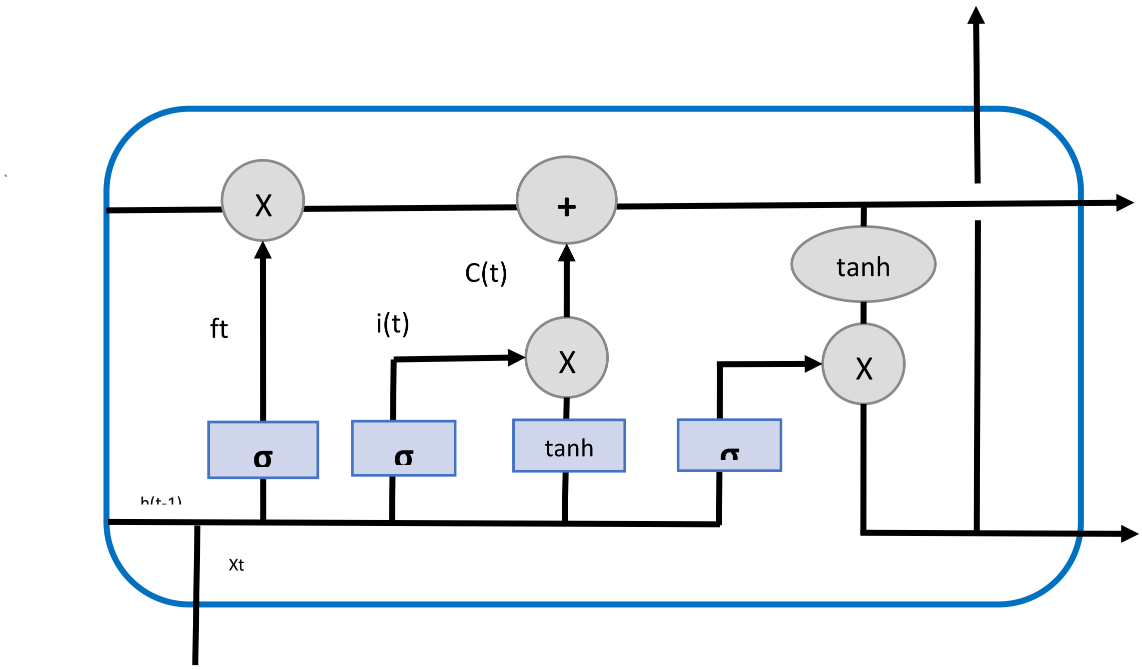

Fig (3.5) Architecture of LSTM

LSTM architecture is comprised of below gates:

1. Forget Gate (ft):

The dataset that is as of now not valuable in the cell state is taken out with the neglected entryway. Two bases of info x_t (info at the specific instance) and h_t-1 (past cell yield) are given to forget gate. Then, the obtained outcome is sent via an activation operation that delivers a result in a binary value that is either 0 or 1. If for a specific cell; if the result is 0, the snippet of data is disregarded and if the result is 1, the data is held for some time later.

2. Input Gate:

The input gate works as an input and valid data added to the cell state.

It comprises two sections;

1. Past State(ht)

2. Present Input(Xt)

The result is revised by the sigmoid rule when these two weights are handed to it.

Then, a vector is initiated employing the tan h rule. This vector delivers an outcome from -1 to +1, which includes all the feasible weights from h_t-1 and x_t. Later, the product of the sigmoid outcome and the tanh values are calculated to decide which one obtains the practical details.

3. Output Gate:

Past inputs are stored in ht and are used for forecasts. This gate handles these states

1. Current State
2. Past Hidden State (ht-1)
3. Current Hidden State(ht)

Initially, a vector is initiated by operating the tan h rule. Then, the data is handled employing a sigmoid rule and diverged through the weights to be recollected using references of info h_t-1 and x_t. Then, weights and vector weights product are estimated and later transmitted to the following cell.

FLOW CHART

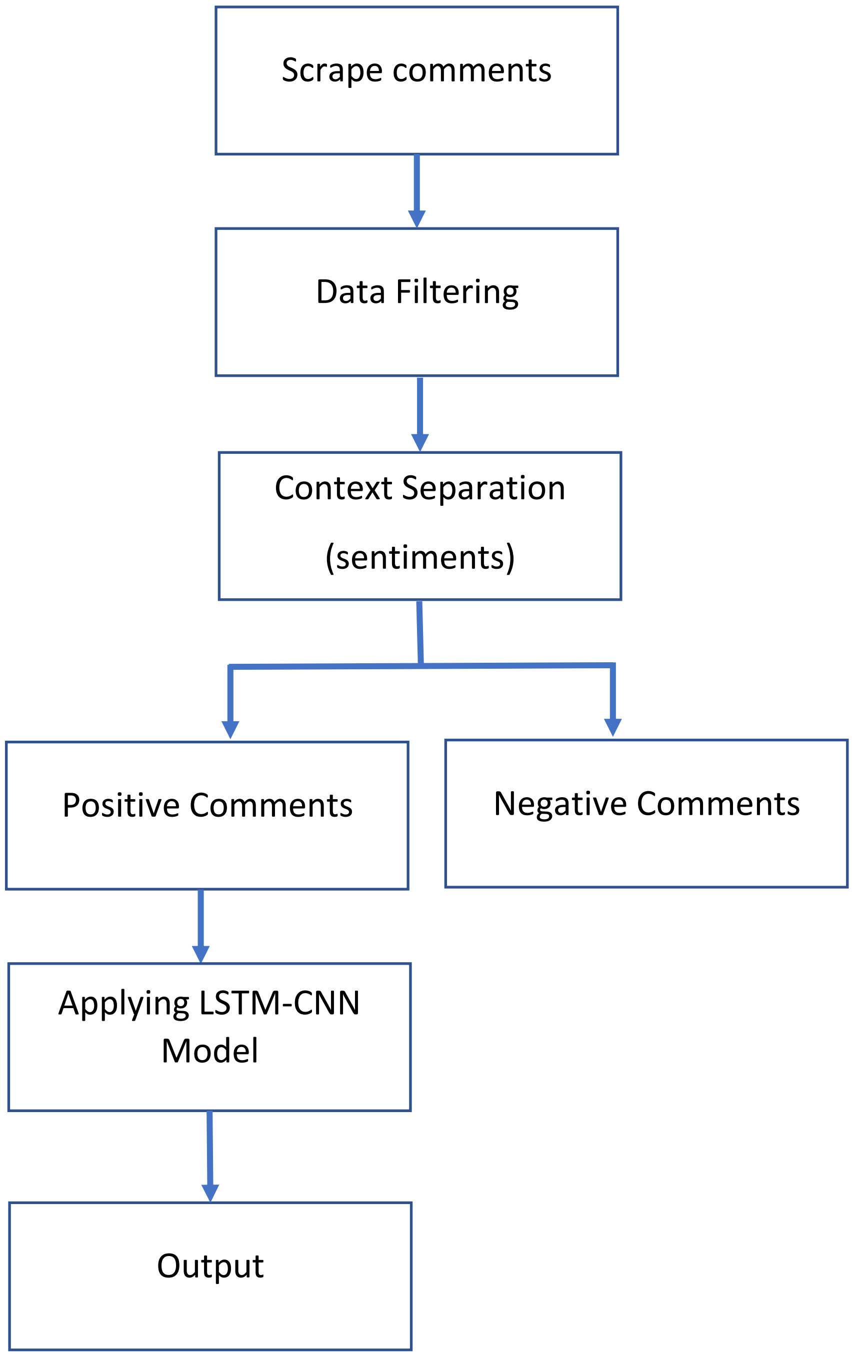

Fig (3.6) Flow Chart

Data Flow Diagram

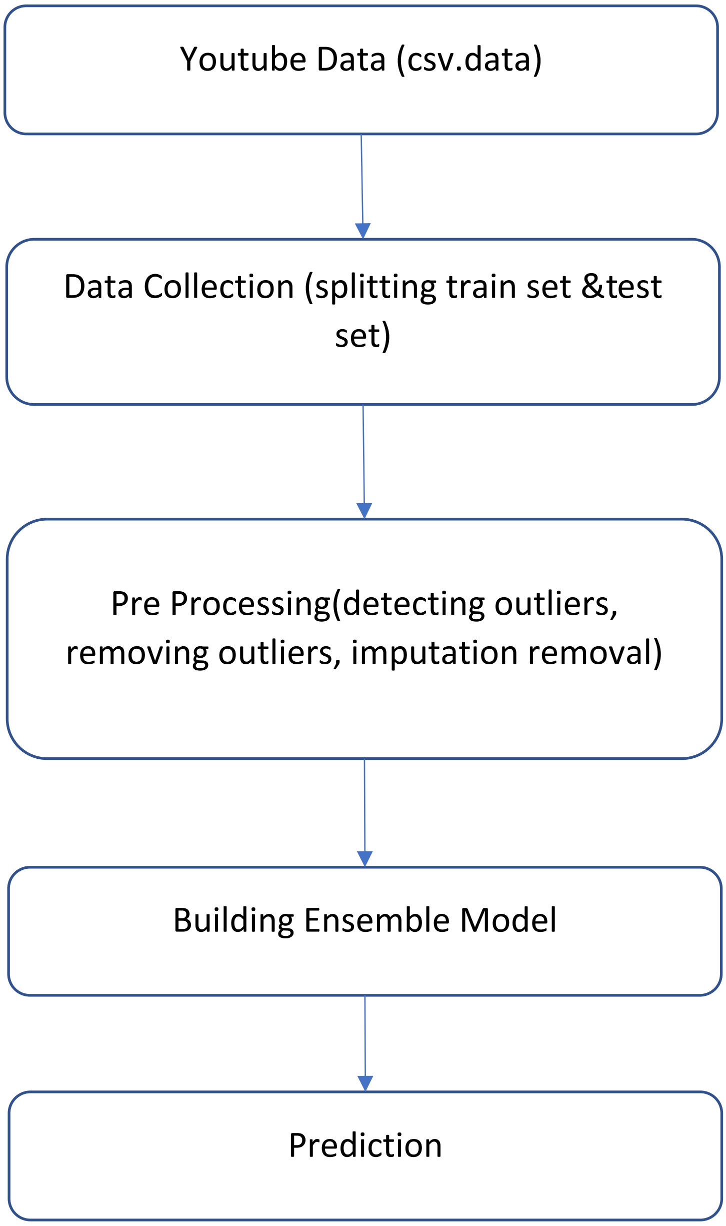

Fig (3.7): data flow diagram for ensemble model

Work Flow Diagram

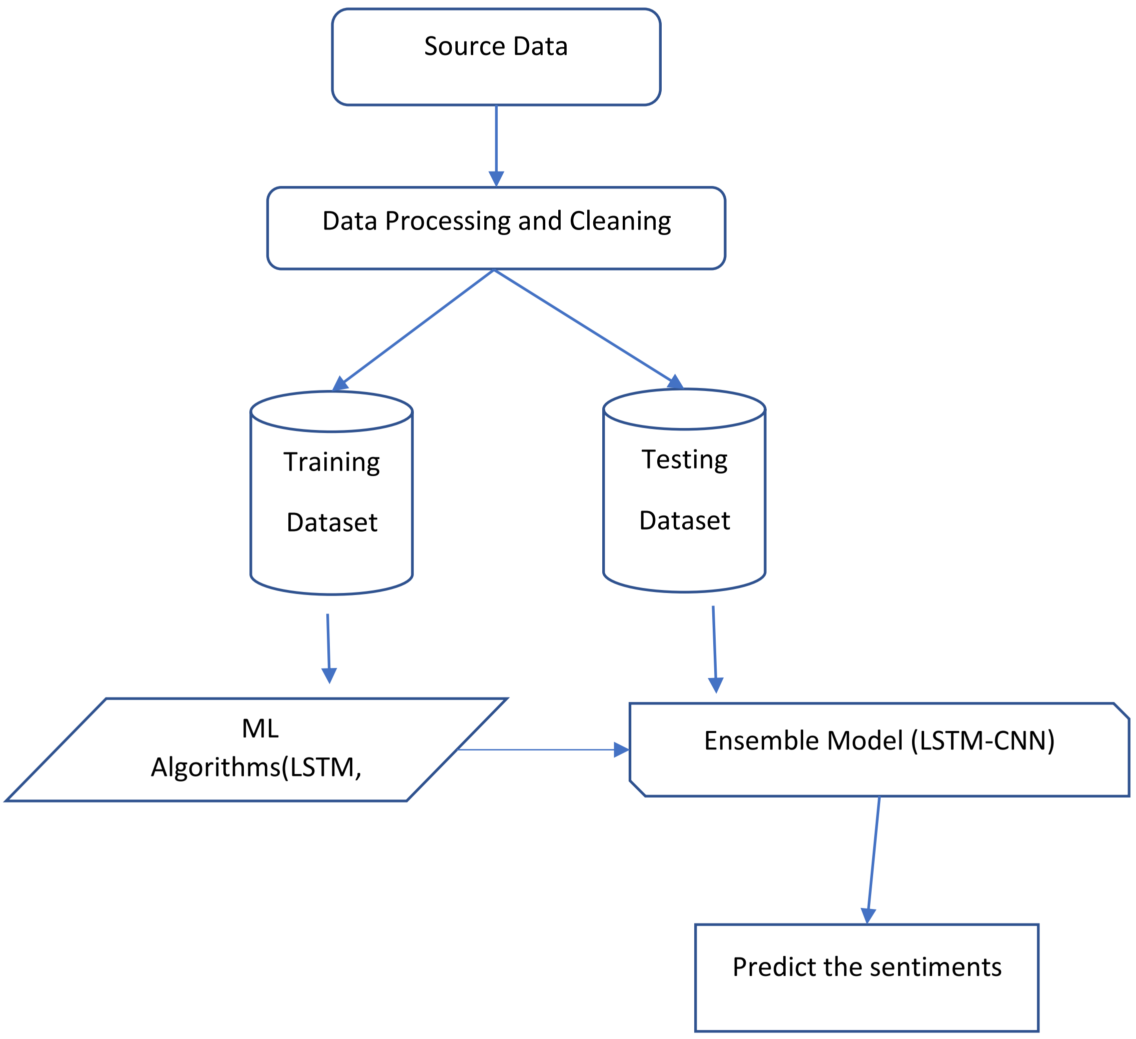

Fig (3.8) Work-flow Diagram

3.3 UML Diagrams:

3.3.1 Use Case Diagram

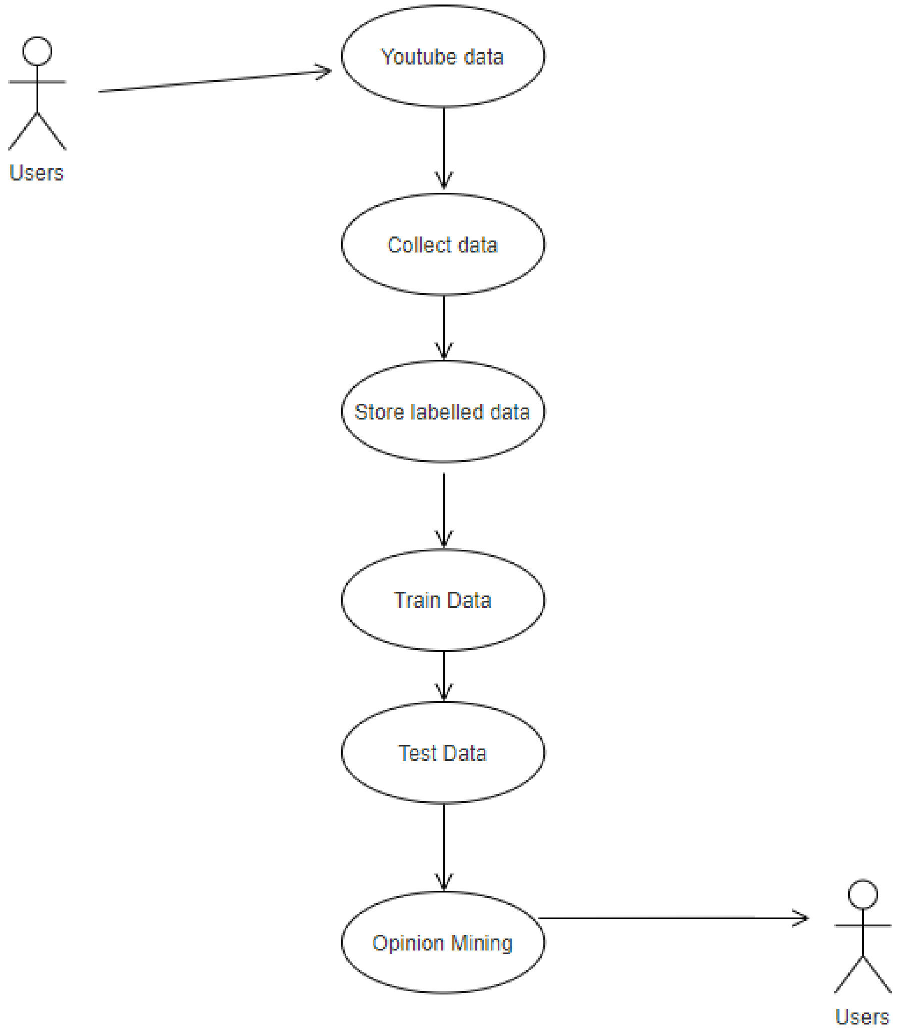

Fig (3.9) Use Case Diagram

3.3.2 Sequence Diagram

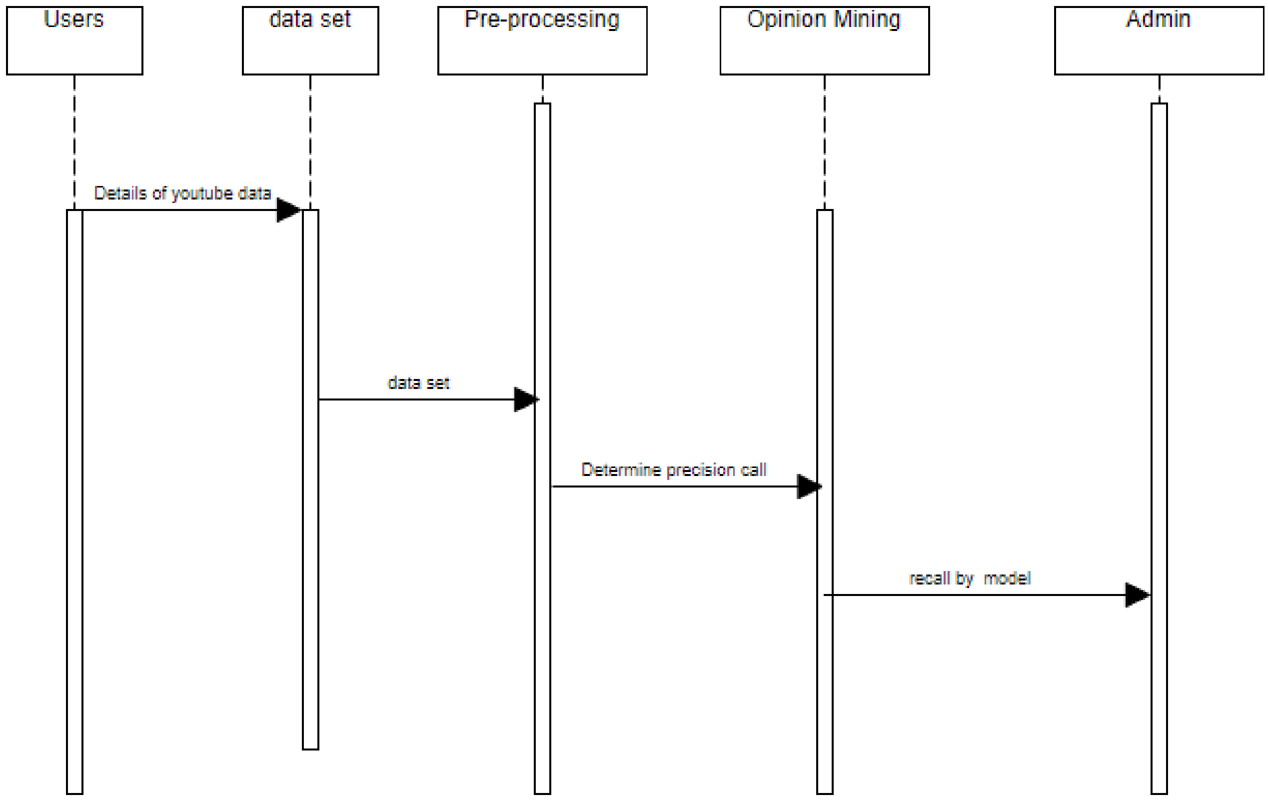

Fig (3.10) Sequence Diagram

3.3.3 Activity Diagram

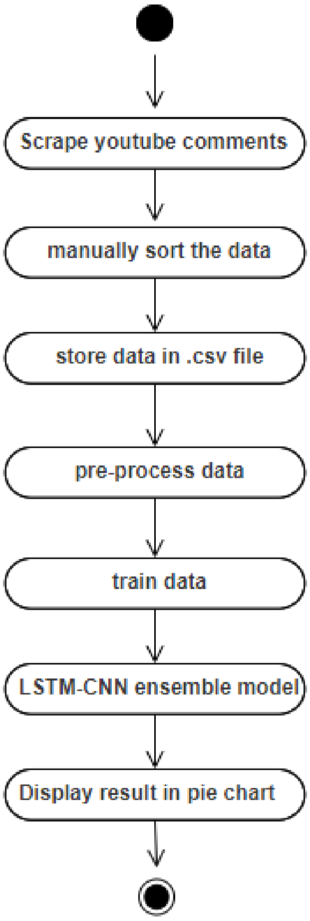

Fig (3.11) Activity Diagram

3.3.4 Class Diagram

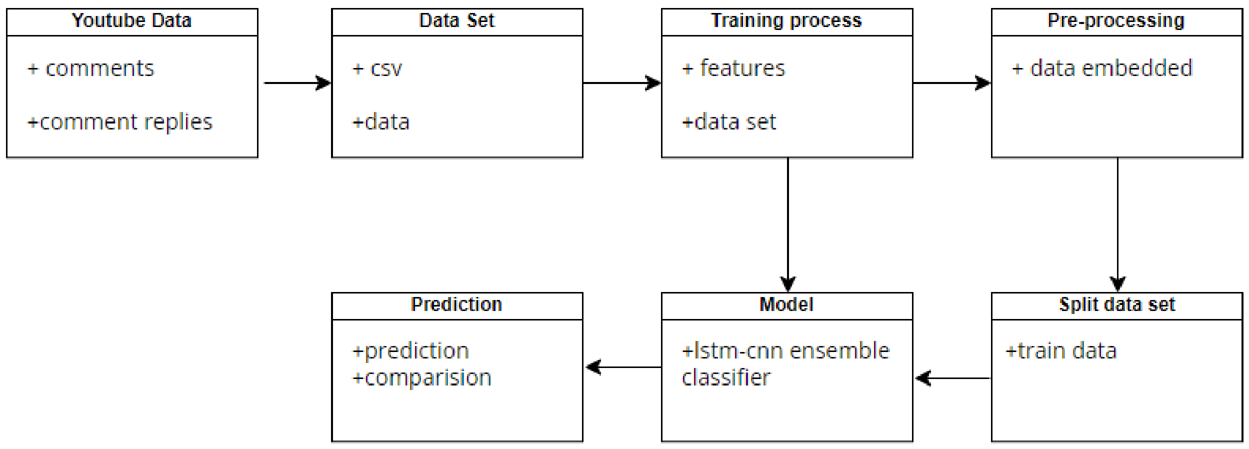

Fig (3.12) Class Diagram

3.3.5 Interface Diagram

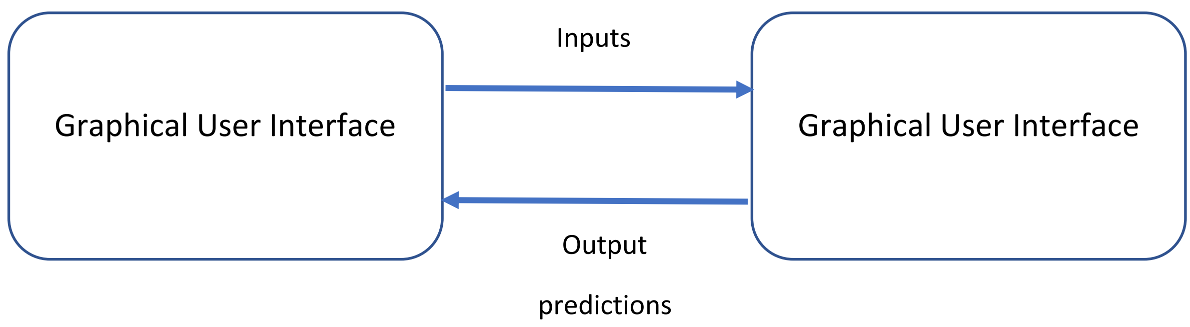

Fig (3.13) Interface Diagram

Entity Relationship Diagram (ERD)

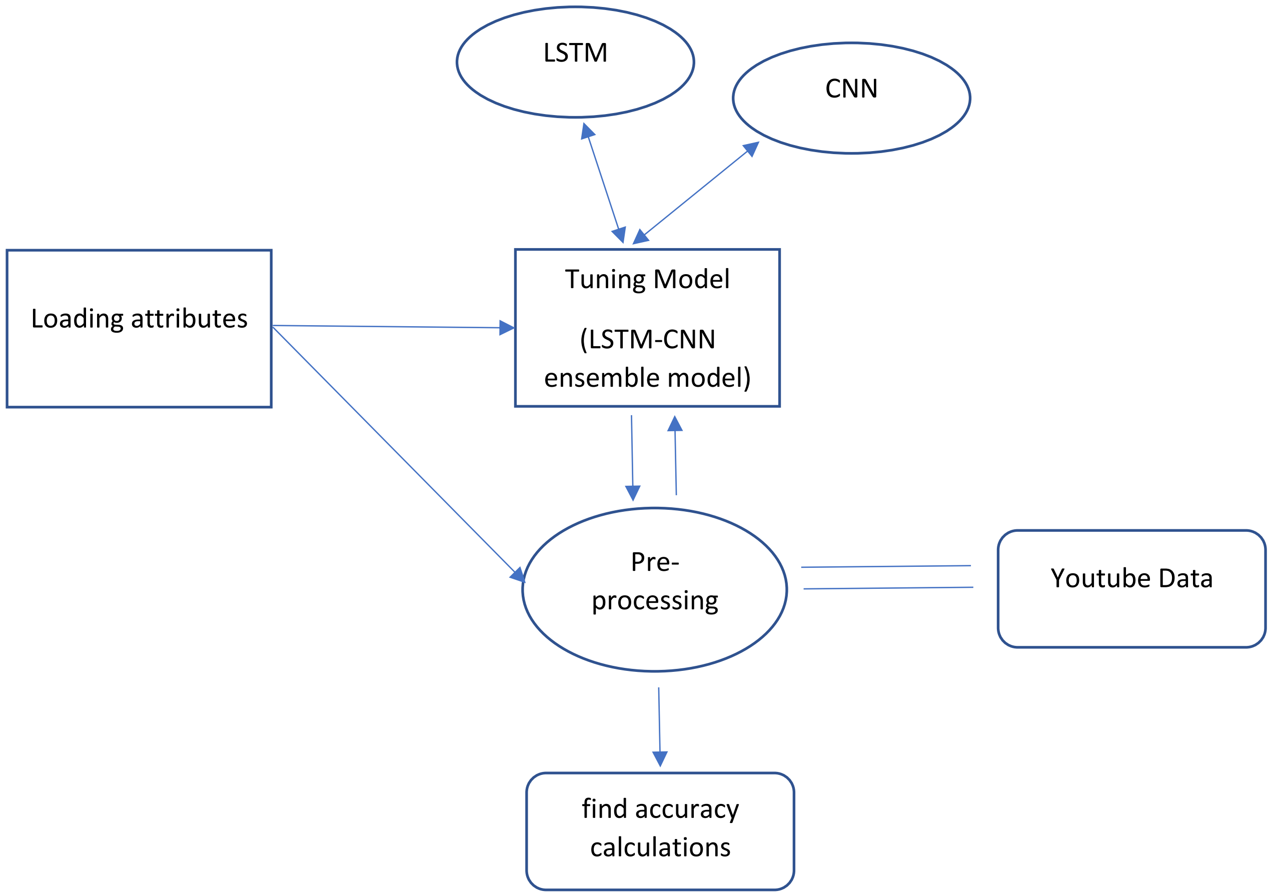

Fig (3.14) Entity Relationship Diagram (ERD)

3.4 Implementation and Coding

This section states the role of every subsystems/module and with the code implementations.

3.4.1 Operational Details:

The proposed classification methods outlined in the steps:

1. Collection of Data: We have collected the datasets from the YouTube comments by using the YouTube API. The name, comment, reply author, reply, these four columns are collected into spreadsheet using Spreadsheet App. Then the reply author and reply columns are manually sorted into the name and comments columns.

2. Pre-processing: The raw data is an unsupervised data with noise and inconsistency. Batchgen is used to clean the datasets. With the help of regular expressions all the unwanted characters and emojis are removed from the text. The datasets are divided into “goodfile” and “badfile.”

3. Vocabulary: By using Glove from TensorFlow sentiments of the comments are polarized and vocabulary is built. Then the vocabulary is processed.

4. Model Selection: Selecting a model is a crucial step while working in a machine learning field. According to the requirements of the project the wise choice must be made to make better predictions. We chose LSTM-CNN ensemble model. So, the datasets are trained on LSTM-CNN ensemble model. Datasets are split into batches and each batch are trained individually.

LSTM-CNN Model

Firstly, an embedded layer is created and the data is passed to LSTM layer. So, at this level of lstm layer the whole sentence is captured without losing its original sentiments as it can remember the statements. For example, “I prefer Ott but now I want to watch in theatres.” Illustration states that “they said that they prefer Ott” initially but at the end of the sentence their viewpoints have changed. Now they want to watch a movie in theatre. Hence, LSTM layer captures this positive sentiment, “they like Movie theatre.” In CNN networking, a convolution layer and maxpooling layer is applied for each filter. After applying those layers, the pooled features are combined and final predictions are made.

3.4.2 Code Listing:

CODE for Collecting raw data from Youtube:

```
function scrapeCommentsWithReplies(){
  var ss = SpreadsheetApp.getActiveSpreadsheet();
  var result=[['Name','Comment','Time','Likes','Reply Count','Reply Author','Reply','Published','Updated']];
  var vid = ss.getSheets()[0].getRange(1,1).getValue();
  var nextPageToken=undefined;

  while(1){

    var data = YouTube.CommentThreads.list('snippet', {videoId: vid, maxResults: 100, pageToken:
nextPageToken})
    nextPageToken=data.nextPageToken
    for (var row=0; row<data.items.length; row++) {
       result.push([data.items[row].snippet.topLevelComment.snippet.authorDisplayName,
          data.items[row].snippet.topLevelComment.snippet.textDisplay,
          data.items[row].snippet.topLevelComment.snippet.publishedAt,
          data.items[row].snippet.topLevelComment.snippet.likeCount,
          data.items[row].snippet.totalReplyCount,'','','','']);
      if(data.items[row].snippet.totalReplyCount>0){
       parent=data.items[row].snippet.topLevelComment.id
       var nextPageTokenRep=undefined

while(1){
         var data2=YouTube.Comments.list('snippet', {videoId: vid, maxResults: 100,
pageToken: nextPageTokenRep,parentId:parent})
         nextPageTokenRep=data2.nextPageToken;
         for (var i =data2.items.length-1;i>=0;i--){
           result.push(['','','','','',
                data2.items[i].snippet.authorDisplayName,
                data2.items[i].snippet.textDisplay,
                data2.items[i].snippet.publishedAt,
                data2.items[i].snippet.updatedAt]);
         }
```

```
        if(nextPageTokenRep =="" || typeof nextPageTokenRep === "undefined"){
          break
        }
      }
    }
  }
  if(nextPageToken =="" || typeof nextPageToken === "undefined"){
    break;
  }
}

var newSheet=ss.insertSheet(ss.getNumSheets())
newSheet.getRange(1, 1,result.length,9).setValues(result)
}
```

CODE SNIPPET

for Cleaning the Raw-Dataset:

```
def clean_str(string):
    #EMOJIS
    string = re.sub(r":\)","emojihappy1",string)
    string = re.sub(r":P","emojihappy2",string)
    string = re.sub(r":p","emojihappy3",string)
    string = re.sub(r":>","emojihappy4",string)
    string = re.sub(r":3","emojihappy5",string)
    string = re.sub(r":D","emojihappy6",string)
    string = re.sub(r" XD ","emojihappy7",string)
    string = re.sub(r" <3 ","emojihappy8",string)
    string = re.sub(r":\(","emojisad9",string)
    string = re.sub(r":<","emojisad10",string)
    string = re.sub(r":<","emojisad11",string)
    string = re.sub(r">:\(","emojisad12",string)
    #MENTIONS "(@)\w+"
    string = re.sub(r"(@)\w+","mentiontoken",string)
```

```
#WEBSITES
string = re.sub(r"http(s)*:(\S)*","linktoken",string)
#STRANGE UNICODE \x...
string = re.sub(r"\\x(\S)*","",string)
#General Cleanup and Symbols
string = re.sub(r"[^A-Za-z0-9(),!?\'\`]", " ", string)
string = re.sub(r"\'s", " \'s", string)
string = re.sub(r"\'ve", " \'ve", string)
string = re.sub(r"n\'t", " n\'t", string)
string = re.sub(r"\'re", " \'re", string)
string = re.sub(r"\'d", " \'d", string)
string = re.sub(r"\'ll", " \'ll", string)
string = re.sub(r",", " , ", string)
string = re.sub(r"!", " ! ", string)
string = re.sub(r"\(", " \( ", string)
string = re.sub(r"\)", " \) ", string)
string = re.sub(r"\?", " \? ", string)
string = re.sub(r"\s{2,}", " ", string)
return string.strip().lower()
```

CODE SNIPPET

Split data with mixed positive and negative data into two different files.

```
def separate_dataset(filename):
  good_out = open("good_"+filename,"w+");
  bad_out  = open("bad_"+filename,"w+");
  seen = 1;
  with open(filename,'r') as f:
    reader = csv.reader(f)
    reader.next()
    for line in reader:
      seen +=1
      sentiment = line[1]
      sentence = line[3]
      if (sentiment == "0"):
        bad_out.write(sentence+"\n")
      else:
        good_out.write(sentence+"\n")
      if (seen%10000==0):
        print(seen);
  good_out.close();
  bad_out.close();
```

```
#Load Dataset
def get_dataset(goodfile,badfile,limit,randomize=True):
    good_x = list(open(goodfile,"r").readlines())
    good_x = [s.strip() for s in good_x]

    bad_x  = list(open(badfile,"r").readlines())
    bad_x  = [s.strip() for s in bad_x]
    if (randomize):
        random.shuffle(bad_x)
        random.shuffle(good_x)
    good_x = good_x[:limit]
    bad_x = bad_x[:limit]
    x = good_x + bad_x
    x = [clean_str(s) for s in x]
    positive_labels = [[0, 1] for _ in good_x]
    negative_labels = [[1, 0] for _ in bad_x]
    y = np.concatenate([positive_labels, negative_labels], 0)
    return [x,y]
```

CODE SNIPPET

generating random batches for training and testing the data:

```
def gen_batch(data, batch_size, num_epochs, shuffle=True):
    """
    Generates a batch iterator for a dataset.
    """
    data = np.array(data)
    data_size = len(data)
    num_batches_per_epoch = int((len(data)-1)/batch_size) + 1
    for epoch in range(num_epochs):
        # Shuffle the data at each epoch
        if shuffle:
            shuffle_indices = np.random.permutation(np.arange(data_size))
            shuffled_data = data[shuffle_indices]
        else:
            shuffled_data = data
        for batch_num in range(num_batches_per_epoch):
            start_index = batch_num * batch_size
            end_index = min((batch_num + 1) * batch_size, data_size)
            yield shuffled_data[start_index:end_index]
```

Initializing Parameters

```
# Data loading params
dev_size = .10
# Model Hyperparameters
embedding_dim  = 32     #128
max_seq_legth = 70
filter_sizes = [3,4,5]  #3
num_filters = 32
dropout_prob = 0.5 #0.5
l2_reg_lambda = 0.0
use_glove = True
# Training parameters
batch_size = 128
num_epochs = 10 #200
evaluate_every = 100 #100
checkpoint_every = 100000 #100
num_checkpoints = 1 #Checkpoints to store
# Misc Parameters
allow_soft_placement = True
log_device_placement = False
```

CODE SNIPPET

DATA PREPARATION- build vocabulary

```
filename = "youtube_raw_comments.csv"
goodfile = "good_tweets.csv"
badfile = "bad_tweets.csv"
# Load data
print("Loading data...")
x_text, y = batchgen.get_dataset(goodfile, badfile, 5000) #TODO: MAX LENGTH
# Build vocabulary
max_document_length = max([len(x.split(" ")) for x in x_text])
if (not use_glove):
    print ("Not using GloVe")
    vocab_processor = learn.preprocessing.VocabularyProcessor(max_document_length)
    x = np.array(list(vocab_processor.fit_transform(x_text)))
else:
    print ("Using GloVe")
    embedding_dim = 50
    filename = 'pre-trainer/glove/glove.6B.50d.txt'
    def loadGloVe(filename):
        vocab = []
        embd = []
        file = open(filename,'r')
        for line in file.readlines():
            row = line.strip().split(' ')
            vocab.append(row[0])
            embd.append(row[1:])
        print('Loaded GloVe!')
        file.close()
        return vocab,embd
```

```
vocab,embd = loadGloVe(filename)
vocab_size = len(vocab)
embedding_dim = len(embd[0])
embedding = np.asarray(embd)
W = tf.Variable(tf.constant(0.0, shape=[vocab_size, embedding_dim]),
        trainable=False, name="W")
embedding_placeholder = tf.placeholder(tf.float32, [vocab_size, embedding_dim])
embedding_init = W.assign(embedding_placeholder)
session_conf = tf.ConfigProto(allow_soft_placement=True, log_device_placement=False)
sess = tf.Session(config=session_conf)
sess.run(embedding_init, feed_dict={embedding_placeholder: embedding})
from tensorflow.contrib import learn
#init vocab processor
vocab_processor = learn.preprocessing.VocabularyProcessor(max_document_length)
#fit the vocab from glove
pretrain = vocab_processor.fit(vocab)
#transform inputs
x = np.array(list(vocab_processor.transform(x_text)))
#init vocab processor
vocab_processor = learn.preprocessing.VocabularyProcessor(max_document_length)
#fit the vocab from glove
pretrain = vocab_processor.fit(vocab)
#transform inputs
x = np.array(list(vocab_processor.transform(x_text)))
```

CODE SNIPPET

shuffling data and split into training and test set

```
# Randomly shuffle data
np.random.seed(42)
shuffle_indices = np.random.permutation(np.arange(len(y)))
x_shuffled = x[shuffle_indices]
y_shuffled = y[shuffle_indices]
# Split train/test set
# TODO: This is very crude, should use cross-validation
dev_sample_index = -1 * int(dev_size * float(len(y)))
x_train, x_dev = x_shuffled[:dev_sample_index], x_shuffled[dev_sample_index:]
y_train, y_dev = y_shuffled[:dev_sample_index], y_shuffled[dev_sample_index:]
print("Vocabulary Size: {:d}".format(len(vocab_processor.vocabulary_)))
print("Train/Dev split: {:d}/{:d}".format(len(y_train), len(y_dev)))
#embed()
```

CODE SNIPPET

TRAINING STEP

```
def train_step(x_batch, y_batch,save=False):
      feed_dict = {
        model.input_x: x_batch,
        model.input_y: y_batch,
        model.dropout_keep_prob: dropout_prob
      }
      _, step, summaries, loss, accuracy = sess.run(
          [train_op, global_step, train_summary_op, model.loss, model.accuracy],
          feed_dict)
      time_str = datetime.datetime.now().isoformat()
      print("{}: step {}, loss {:g}, acc {:g}".format(time_str, step, loss, accuracy))
      if save:
          train_summary_writer.add_summary(summaries, step)
#CREATE THE BATCHES GENERATOR
    batches = batchgen.gen_batch(list(zip(x_train, y_train)), batch_size, num_epochs)

    #TRAIN FOR EACH BATCH
    for batch in batches:
      x_batch, y_batch = zip(*batch)
      train_step(x_batch, y_batch)
      current_step = tf.train.global_step(sess, global_step)
      if current_step % evaluate_every == 0:
          print("\nEvaluation:")
          dev_step(x_dev, y_dev, writer=dev_summary_writer)
          print("")
      if current_step % checkpoint_every == 0:
          path = saver.save(sess, checkpoint_prefix, global_step=current_step)
          print("Saved model checkpoint to {}\n".format(path))
    dev_step(x_dev, y_dev, writer=dev_summary_writer)
```

CODE SNIPPET

LSTM-CNN ENSEMBLE MODEL

```
class LSTM_CNN(object):
    def __init__(self, sequence_length, num_classes, vocab_size, embedding_size, filter_sizes,
num_filters, l2_reg_lambda=0.0,num_hidden=100):
        # PLACEHOLDERS
        self.input_x = tf.placeholder(tf.int32, [None, sequence_length], name="input_x")    # X - The Data
        self.input_y = tf.placeholder(tf.float32, [None, num_classes], name="input_y")      # Y - The Lables
        self.dropout_keep_prob = tf.placeholder(tf.float32, name="dropout_keep_prob")       # Dropout


        l2_loss = tf.constant(0.0) # Keeping track of l2 regularization loss
        #1. EMBEDDING LAYER ################
        with tf.device('/cpu:0'), tf.name_scope("embedding"):
            self.W = tf.Variable(tf.random_uniform([vocab_size, embedding_size], -1.0, 1.0),name="W")
            self.embedded_chars = tf.nn.embedding_lookup(self.W, self.input_x)
            #self.embedded_chars_expanded = tf.expand_dims(self.embedded_chars, -1)
        #2. LSTM LAYER ######################
        self.lstm_cell = tf.contrib.rnn.LSTMCell(32,state_is_tuple=True)
        #self.h_drop_exp = tf.expand_dims(self.h_drop,-1)
        self.lstm_out,self.lstm_state =
tf.nn.dynamic_rnn(self.lstm_cell,self.embedded_chars,dtype=tf.float32)
        #embed()
        self.lstm_out_expanded = tf.expand_dims(self.lstm_out, -1)
        #2. CONVOLUTION LAYER + MAXPOOLING LAYER (per filter) ###############
        pooled_outputs = []
        for i, filter_size in enumerate(filter_sizes):
            with tf.name_scope("conv-maxpool-%s" % filter_size):
                # CONVOLUTION LAYER
                filter_shape = [filter_size, embedding_size, 1, num_filters]
                W = tf.Variable(tf.truncated_normal(filter_shape, stddev=0.1), name="W")
```

```
            b = tf.Variable(tf.constant(0.1, shape=[num_filters]), name="b")
            conv = tf.nn.conv2d(self.lstm_out_expanded, W,strides=[1, 1, 1,
1],padding="VALID",name="conv")
            # NON-LINEARITY
            h = tf.nn.relu(tf.nn.bias_add(conv, b), name="relu")
            # MAXPOOLING
            pooled = tf.nn.max_pool(h, ksize=[1, sequence_length - filter_size + 1, 1, 1], strides=[1, 1, 1, 1],
padding='VALID', name="pool")
            pooled_outputs.append(pooled)
# COMBINING POOLED FEATURES
      num_filters_total = num_filters * len(filter_sizes)
      self.h_pool = tf.concat(pooled_outputs, 3)
      self.h_pool_flat = tf.reshape(self.h_pool, [-1, num_filters_total])

      # #3. DROPOUT LAYER ###################
      with tf.name_scope("dropout"):
         self.h_drop = tf.nn.dropout(self.h_pool_flat, self.dropout_keep_prob)
      # Final (unnormalized) scores and predictions
      with tf.name_scope("output"):
         W = tf.get_variable(
            "W",
            shape=[num_filters_total, num_classes],
            initializer=tf.contrib.layers.xavier_initializer())
         b = tf.Variable(tf.constant(0.1, shape=[num_classes]), name="b")
```

```
    l2_loss += tf.nn.l2_loss(W)
    l2_loss += tf.nn.l2_loss(b)
    self.scores = tf.nn.xw_plus_b(self.h_drop, W, b, name="scores")
    self.predictions = tf.argmax(self.scores, 1, name="predictions")
# CalculateMean cross-entropy loss
with tf.name_scope("loss"):
    losses = tf.nn.softmax_cross_entropy_with_logits(logits=self.scores, labels=self.input_y)
    self.loss = tf.reduce_mean(losses) + l2_reg_lambda * l2_loss
# Accuracy
with tf.name_scope("accuracy"):
    correct_predictions = tf.equal(self.predictions, tf.argmax(self.input_y, 1))
    self.accuracy = tf.reduce_mean(tf.cast(correct_predictions, "float"), name="accuracy")
print("(!!) LOADED LSTM-CNN! :)")
```

CODE SNIPPET

EVALUATE MODEL

```
def dev_step(x_batch, y_batch, writer=None,save=False):
    feed_dict = {
      model.input_x: x_batch,
      model.input_y: y_batch,
      model.dropout_keep_prob: 0.5
    }
    step, summaries, loss, accuracy = sess.run(
        [global_step, dev_summary_op, model.loss, model.accuracy],
        feed_dict)
    time_str = datetime.datetime.now().isoformat()
    print("{}: step {}, loss {:g}, acc {:g}".format(time_str, step, loss, accuracy))
    if save:
        if writer:
            writer.add_summary(summaries, step)
```

CHAPTER 4

RESULTS AND DISCUSSIONS

4.1 Scraping YouTube Comments:

Raw Dataset

The below table (4.1) represents the raw datasets that were directly collected from YouTube.

Name	Comment	Time	Likes	Reply Count	Reply Author	Reply	Published	Updated
IGN	Have movie thea	2020-10-14T21:	80	61				
					Lego Dude5000	IGN idk? I would	2020-10-14T21:	2020-10-14T21:48:08Z
					The Hil_Figures	No not yet	2020-10-14T21:	2020-10-14T21:47:39Z
					5hr _	No very sad 😢	2020-10-14T21:	2020-10-14T21:47:55Z
					Oath Panda	mine closed	2020-10-14T21:	2020-10-14T21:47:59Z
					Robbie Harris	Theaters around	2020-10-14T21:	2020-10-14T21:48:25Z
					PHALINVIJ	Not yet	2020-10-14T21:	2020-10-14T21:48:31Z
					Destructus 86	Nope. Still close	2020-10-14T21:	2020-10-14T21:48:56Z
					A . B Shah	Nah	2020-10-14T21:	2020-10-14T21:49:23Z
					MrLegobro	I have been goin	2020-10-14T21:	2020-10-14T21:50:27Z
					RetroScopeZ	yess for like 3 m	2020-10-14T21:	2020-10-14T21:51:13Z
					Iwan Kellerman	Yes	2020-10-14T21:	2020-10-14T21:51:16Z
					Lachlan Filmer	Mine in Sydney /	2020-10-14T21:	2020-10-14T21:52:36Z
					Tom	Just one and the	2020-10-14T21:	2020-10-14T21:53:34Z
					vhw	They're op	2020-10-14T21:	2020-10-14T21:54:23Z
					Dylan Tackett	Idk	2020-10-14T21:	2020-10-14T21:54:08Z
					Paul Bauman	Can't say they h	2020-10-14T21:	2020-10-14T22:02:20Z
					Lumify | Gil	Yes, then they cl	2020-10-14T22:	2020-10-14T22:00:00Z
					Victor Zhargalov	Yes	2020-10-14T22:	2020-10-14T22:01:56Z
					Bricks, Studs, Ar	No, looks like the	2020-10-14T22:	2020-10-14T22:08:59Z
					Jacobi-vision	Yeah I saw tenet	2020-10-14T22:	2020-10-14T22:13:44Z
					Aashish Yelchur	Yes	2020-10-14T22:	2020-10-14T22:15:55Z
					Vegeta	No	2020-10-14T22:	2020-10-14T22:20:48Z
					Sebastián Aguila	Yes	2020-10-14T22:	2020-10-14T22:34:48Z
					Spirit of Omega	Yes actually. The	2020-10-14T22:	2020-10-14T22:39:11Z
					Spirit of Omega	@RetroScopeZ	2020-10-14T22:	2020-10-14T22:39:46Z
					Genesis Ambroc	they are going to	2020-10-14T22:	2020-10-14T22:47:14Z
					Skull_boi	Nope	2020-10-14T22:	2020-10-14T22:51:10Z
					Spencer E	Not yet	2020-10-14T22:	2020-10-14T22:54:36Z
					Allwin Issac	Nope	2020-10-14T22:	2020-10-14T22:55:34Z
					Ben Sweeney	NYC...NOPE	2020-10-14T23:	2020-10-14T23:11:28Z
					jay eazy	Toronto movie th	2020-10-14T23:	2020-10-14T23:12:30Z
					echtalion	No, cineworld ha	2020-10-14T23:	2020-10-14T23:12:42Z
					Ghost711 G	0	2020-10-14T23:	2020-10-14T23:18:45Z

Table (4.1) raw dataset

Pre-processed Dataset:

	Unnamed: 0	Name	Comment	Comments
0	0	Study IQ education	Download our app - <a href="https://play.googl...	download our app a href
1	1	Gaur.971	Waiting for only two movies The kashmir fil...	waiting for only two moviesbrthe kashmir files...
2	2	MR□KARAN SHUKLA	I also follow WION.... ❤	i also follow wion
3	3	Dulal Hansda	Teach something to Mr. Siddhanth Sirji, sansan...	teach something to mr siddhanth sirji sansani ...
4	4	Piyush 1034	Movie kahi bhi release karo par hum to telegra...	movie kahi bhi release karo par hum to telegra...
...	...	...	...	...
6226	6218	Sam	yes but i was born in 1988 think of how many t...	yes but i was born in think of how many times...
6227	6219	Dalton Bush	Yeah I 100% agree I never want movie theaters ...	yeah i agree i never want movie theaters to g...
6228	6220	Mario Kart Fan	Firstly I choose theaters over ott	firstly i choose theaters over ott
6229	6221	Dashinglucifer	ott	ott
6230	6222	Imaad Maqsood	Wow theaters are really awesome bruh	wow theaters are really awesome bruh

Table (4.2) snapshot of pre-processed dataset.

4.2 Code Execution:

Here, *train.py* is the main file.

The code is executed by writing the commands in the VS Code terminal

Code Snippet:

$ python3 train.py

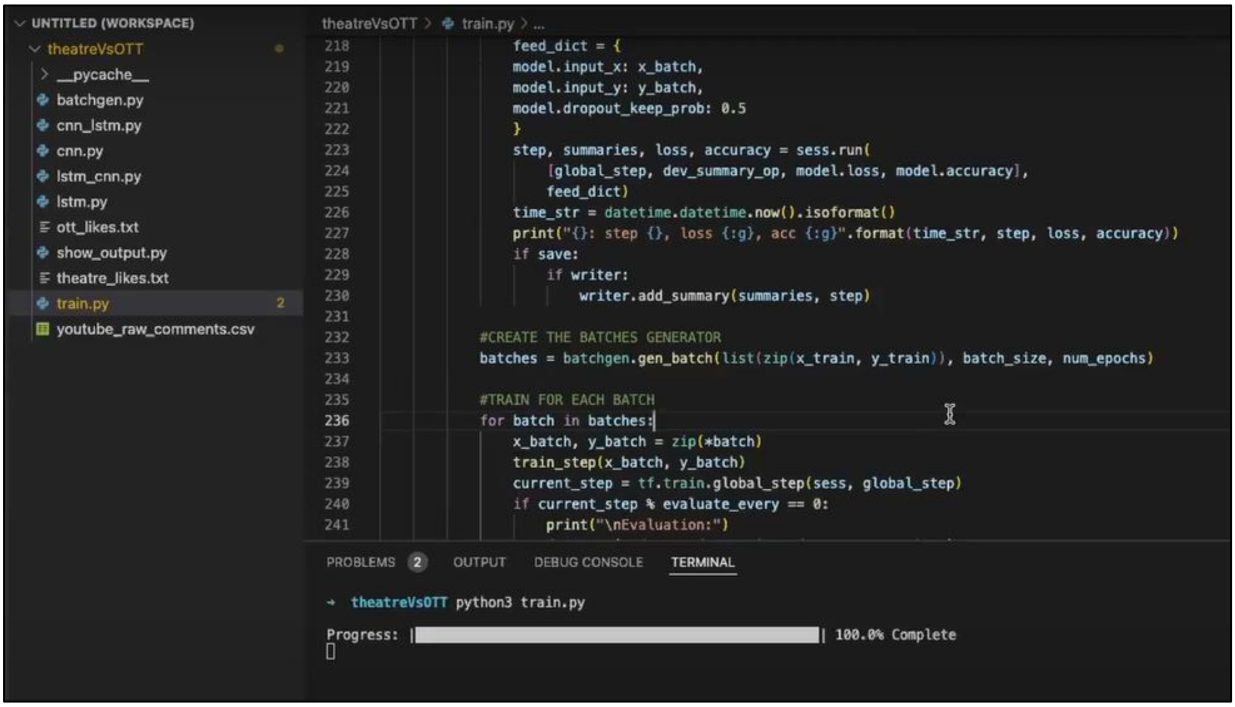

Fig (4.1) Executing the main file (train.py)

4.3 Plotting results on the Chart:

The pie chart displays the percentage of “LIKE OTT” and “LIKE THEATRE.” In the figure (4.2), 43 percent of people are theatre biased and 57 percent of people are Ott biased.

From the chart statistics, it is evident that most of the people are in favour of OTT platforms than Theatres.

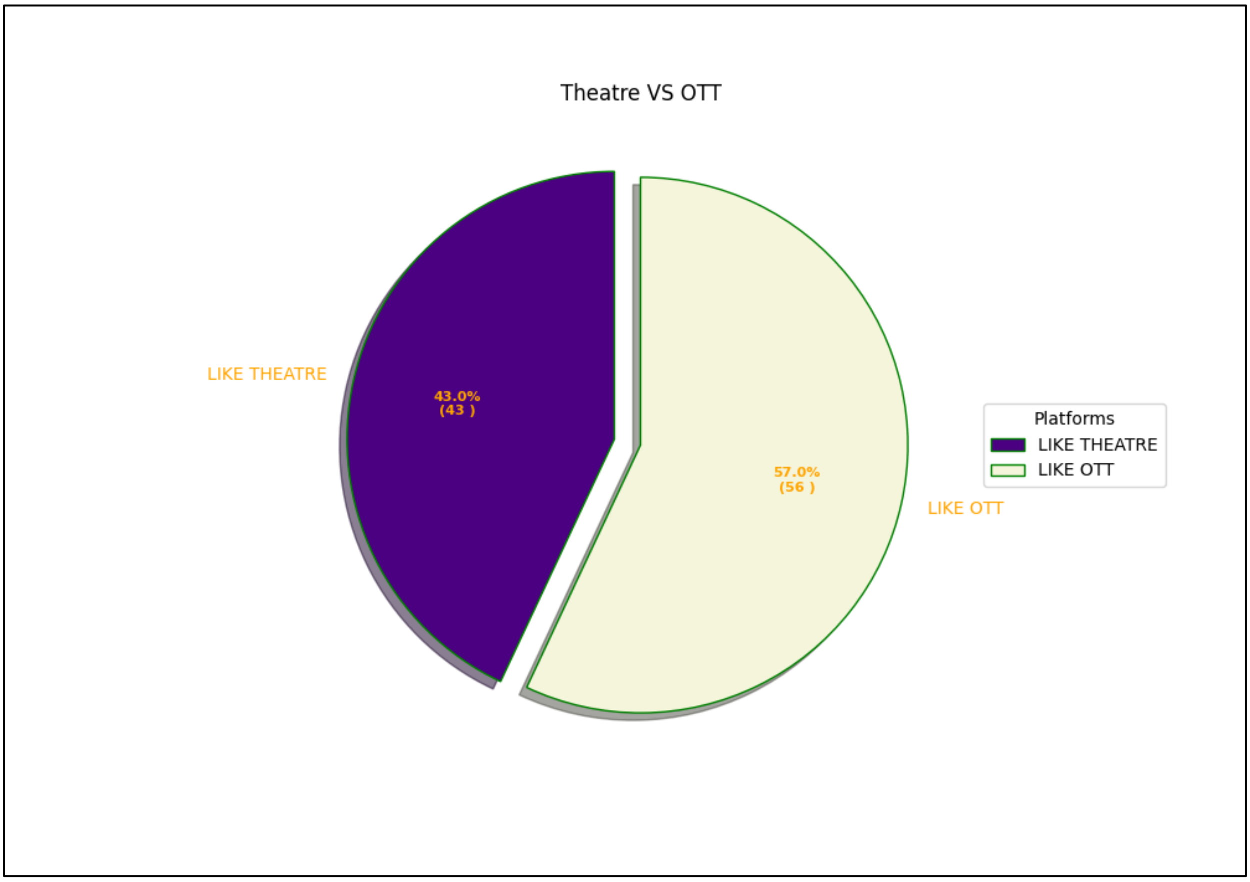

Fig (4.2) Results

CHAPTER 5

CONCLUSION AND FUTURE SCOPE OF STUDY

5.1 Conclusion

The pursuit of this paper is to study the integrated model using LSTM and CNN models to design an ensemble model to enhance the performance of two individual machine learning models namely convolutional neural network (CNN) and Long-Term Short-Term Memory (LSTM). As mentioned, LSTM-CNN ensemble model yields effective outcomes. The methodology that we presented has some disadvantages are as follows:

- The datasets are collected from a lone source. These data are the comments scrapped from YouTube.
- Analysis is done only on the positive insights. For example, this paper only talks about the people, who are biased towards either Ott or theatres.

5.2 Future Scope of Study

Ensemble learning is still new. There are many learning concepts. In the future work, datasets will be collected from many available sources to get even more accurate predictions. Also, add other models such as SVM, ME, Regression models to create an advanced ensemble learning for better forecasts. And along with positive insights, even the negative insights can be included in the upcoming work.

REFERENCES

1. Milene Dias Almeida and Vinicius Mothé Maia and Roberto Tommasetti and Rodrigo de Oliveira Leite “Sentiment analysis based on a social media customised dictionary” (July 2021)
2. Sarlan, A., Nadam, C., & Basri, S., “Twitter sentiment analysis ” (November 2014)
3. Özgür Ağrali and Ömer Aydın , “Tweet Classification and Sentiment Analysis on Metaverse Related Messages” (2021)
4. Alsayat A, “ Improving Sentiment Analysis for Social Media Applications Using an Ensemble Deep Learning Language Model” (2022) (“Dr. Ahmed Alsayat - Google Scholar”)
5. Ghosh M and Sanyal G, “An ensemble approach to stabilize the features for multi-domain sentiment analysis using supervised machine learning.” (“Amharic Character Recognition Based on Features Extracted by CNN and ...”)
6. Matthew Whitehead and Larry Yaeger, “Mining Using Ensemble Classification Models” (2018)
7. Vohra M.S. and Teraiya J, “Applications and challenges for sentiment analysis” (2013)
8. Sushree Das, Ranjan Kumar Behera, Mukeshkumar, Santanu Kumar Rath, “Real-Time Sentiment Analysis of Twitter Streaming data for Stock Prediction” (2018)
9. Rui Xia, Chengqing Zong, and Shoushan Li, “Ensemble of feature sets and classification algorithms for sentiment classification” (2011) (“Sentiment Lexicon Enhanced Neural Sentiment Classification”)
10. Pedro M. Sosa, “Twitter Sentiment Analysis using lstm-cnn models” (June 2017)