Building a Recommender System based on Perceived Emotions

Subba Rao Illa
Computer Science
Clemson University
Clemson, United States of America
C16280847
silla@g.clemson.edu

Chirayu Sharma
Computer Science
Clemson University
Clemson, United States of America
C77139645
chirays@g.clemson.edu

Shivam Panwar
Computer Science
Clemson University
Clemson, United States of America
C70182808
spanwar@g.clemson.edu

Sundaresh Narayanan
Computer Science
Clemson University
Clemson, United States of America
C73923755
sundarn@g.clemson.edu

***Abstract*— Emotions play a significant role in our life. Our very next action depends on our current emotion or the state of our mind. We have proposed to build a recommender-based system for certain emotions trying to improve and enhance them by recommending songs. To understand and study the emotions, we have gathered user data from a Twitter data set of 1.6 million tweets. We also conducted a small survey to help build our system better in recommending with respect to what people actually want to be recommended during a certain state of emotion. Four Data sets have been put into use which includes Tweets for user data, IMDB for movies and Spotify for songs. With the help of recurrent neural networks, we have built model which uses the Long-short-term-memory concept to study the data and recommend accordingly. The model has been designed to handle a large data set and work efficiently to provide optimal results. For novelty we have made significant changes in parameters to drastically reduce the runtime. The experimental results proved to be phenomenal and when asked for the opinion about it, people were satisfied with the recommendations provided by the model. In this paper, we will be illustrating the architecture, working and approach in detail of our system. The system grasps the user data that we provided and presents suggestions which found to be optimal for almost all users.**

Keywords—Recommender system, Emotions, Happiness, sadness, RNN, Long-short-term memory, IMDB, Spotify.

I. INTRODUCTION

A good recommender system catches the eye of customers and attracts new ones if it proves it's worth in terms of accuracy and customer satisfaction. Applications like Netflix and Spotify have excellent recommendation systems where they suggest movies and songs based on the customer's previously accessed content in respective platforms. The major aspect of our project will be to suggest movies and songs to users based on their emotional state(Happy or sad). In order to know the emotional state of our customers, we were planning to analyze the user data with the help of sources such as browsing history, Facebook posts, twitter data etc. Since twitter data is free and readily available, we will be proceeding with the twitter data to develop the recommendation system and determine the emotional state of the user. With the help of user tweets data set, we will be performing sentiment analysis using Natural Language Processing algorithms to determine the nature of the tweet that is positive or negative[1]. Based on nature of tweets, we will be recommending movies and songs to users. Let us assume that a user tweet about him being sad or depressed. Our recommendation system will identify the tweet's nature as negative using the Natural Language processing algorithm. Based on the weight of the tweet a suggestion of a movie or a song with a genre of comedy/Family drama or Pop/rock respectively to improve the emotional state of the user[3]. So basically, our recommendation system will suggest movies and songs of a genre depending on the weight of the tweet. The recommendation systems built till date only suggest content based on user's history. Our system will try to change or improve the emotional state of our user.

A. Motivation

There is no recommender system in the current scenario that helps people improve or enhance their emotions without the use of sensor apparels. Emotions are not being addressed at all. They are so underrated in some societies; we ignore them like it is nothing. Emotions play a key role in our lives with respect to what actions we take next. These actions can be harmful or helpful to us as well as others around and close to us. People who are sad and depressed need immediate help from others but because they don't reach in time leads to bad conclusions. People who are happy should be kept happy as always[3]. Keeping all these reasons in mind, we decided to build a recommender based system which would understand the emotions of people through user data such as twitter and help them recommend something soothing like songs and movies which would help improve their emotional state if sad and enhance it if happy. We all love a good song or a movie irrespective of time and place. Our goal is to engage people in that song or movie to help the person come out of his emotional darkness. A good song and movie recommended at the right time to the right person might do the trick. Our main goal is to develop a recommendation system based on human's perceived emotions. The major aspect of our project will be to suggest movies and songs to users based on their emotional state.

B. Architecture

We have used 3 data sets for building our system. The twitter dataset which consists of 1.6 million rows of user data helps us understand the emotional state of the user. We have used this data for training our model and help it learn all kinds of emotions hidden behind a simple tweet. This dataset is passed into a simple recurrent neural network which segregates the tweets into two different categories namely: Happy and Sad. The recurrent neural network studies the tweets one by one and breaks down the sentences into simple individual words. The sigmoid function inside the network calculates the weight of each word by comparing vector values[2].

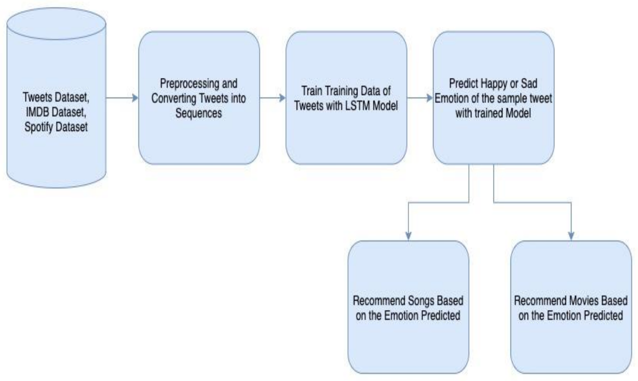
Tweets Dataset, IMDB Dataset, Spotify Dataset
Preprocessing and Converting Tweets into Sequences
Train Training Data of Tweets with LSTM Model
Predict Happy or Sad Emotion of the sample tweet with trained Model
Recommend Songs Based on the Emotion Predicted
Recommend Movies Based on the Emotion Predicted

Fig 1: Framework of the system

The RNN network creates a gradient value which varies if it predicts the correct value of the word. The gradient value becomes stable after n iterations and then we can say that we have trained the model successfully. We made sure that the model we designed is not overfitting the dataset we used. The training part was time consuming since the dataset was very large. We then have given access to two different datasets containing a list of songs and movies of different genres to the model[2]. Two conditions have been set up where a certain list of songs and movies will be recommended if the tweet depicts happiness or sadness as a result. If the tweets depict happiness, we would want to enhance user's emotions and keep him happy. For sadness, we would want to cheer the user by distracting him or motivating him to do well and progress ahead[1].

II. RELATED WORK

People have proposed numerous recommendation systems which suggest different multimedia options to handle the emotional state of the user but with the use of sensor apparels. Users need to wear physiological sensors which would take user's pulse rate as an input to understand what kind of feeling the user is going through. This input is passed onto the system which will then have access to the database of recommendations. There is a recommendation system which suggests multimedia content based on the facial recognition of the user. A neural network which maps the user's facial image to learn the emotional state. Depending on the facial expression, user may get satisfied or unsatisfied with the suggestions because facial expressions do not tell us about the long-lasting emotion. Deep learning has become the hottest topic in the market to study any attribute about any use[5]. A simple dataset is enough to help the neural network learn all the patterns there can ever be about a user. These networks are widely used to learn almost all attributes a user has so that when suggesting or offering something to the user, it should be the right match since every user is different and has a different taste in given choices[2].

Our methodology uses simple set of tweets to study the emotional state of the user. To train the model we have used a static dataset but for future work we may link it to the live flowing tweets daily. We try to improve and enhance user's emotions so that he stays happy and satisfied. A good song or movie can change the mood instantly. Our system will play a key role in changing the user's mood by recommending both songs and movies which will be successful to the user's liking. Unlike many other approaches we do not use any kind of sensor apparels to study user's emotions[4].

III. METHODOLOGY

A. *Problem Statement*

Our project involves a model which takes tweets as input to predict and depict whether the tweet shows Happiness or sadness. It involves many problems and complexities. Training a model should include a very large dataset and a Natural language processing algorithm which could help decode the dataset according to our set of conditions. Training a small dataset is not a difficult task but training a dataset with 1.6 million rows is. Our next step in the methodology is recommending songs and movies to a person depending on his mood[3].

Training a dataset this huge is challenging and time consuming. One iteration can take several hours to run and if this is the case with a static dataset, we could imagine how hard it could be with live flowing data. We need the right and high-quality machines to tackle this problem plus smart selection of neural networks which will help us save time and work efficiently under an environment full of load and pressure. Note that everyone has a different taste in songs and movies. It will be difficult for us to choose the kind and type of song or movie to recommend without knowing the person's actual choice and taste[3]. A recommender system will only catch an eye of a user only if it does something special and be unique every time to everyone in a different way. We cannot build a system which suggests the same thing to all because then User's wouldn't seem to like it all. Every user needs to be treated royally and only provide suggestions which proves to his liking and taste.

B. *Approach*

To begin with the approach, we managed to shortlist 3 datasets for our system. A dataset which consists 1.6 million tweets for training our model. A dataset which has 85854 movies from 1970 to 2020 with all genres. A Spotify dataset with 1994 songs suitable for all tastes. In order to handle such large datasets, we need an algorithm which could help decode the emotion behind the tweet quickly and is able to do it again if given a random sentence. After careful inspections and study, we decided to go with the recurrent neural network to help build a model which will depict the emotion behind a tweet[2].

Recurrent Neural Networks is a deep learning approach which involves a class of artificial networks showing the connection between nodes and directed graph to understand temporal sequence behavior. This approach can be used for sentiment analysis using

certain modifications to the network. This approach has been used by many multinational companies like Google, android and amazon for speech recognition, Image recognition and other applications. It can be widely used to feed data in a sequential manner. For example: If you ask a chat bot " What is the age of Bill Gates? " After being answered the current question you can ask the chat bot a question related to Bill Gates without mentioning him in your question. " What is his company's approximate value? ". This is majorly because the neural network remembers the previous question and can answer all the questions in that chain. RNN has the ability to re-access the hidden variable where the input variables are stored. This makes it efficient to answer all the queries if at all is somehow connected to the previous one. We decided to go forward with this approach because to break down the sentiment analysis in one or more sentences[2].
The only drawback with RNN is while performing a back propagation the gradient value that is being calculated among the layers of the network degrades and it takes time for the model to understand the temporal dynamic sequence. For those reasons, we use a better approach within the RNN known as Long-Short-Term memory.

Long short-term memory is an artificial recurrent neural network used in deep learning. The major advantage of LSTM over an ordinary RNN is the memory unit. RNN gets rid of the data points of the sequence quickly than LSTM. LSTM can backtrack to its hidden variables. The hidden variables have multiple output and input gates to filter the storing of selective parts of the sentence instead of the whole[2].
LSTM networks are well-suited to classifying, processing and making predictions based on time series data, since there can be lags of unknown duration between important events in a time series. LSTMs were developed to deal with the vanishing gradient problem that can be encountered when training traditional RNNs. Relative insensitivity to gap length is an advantage of LSTM over RNNs, hidden Markov models and other sequence learning methods in numerous applications[2].

A common architecture is composed of a cell (the memory part of the LSTM unit) and three "regulators", usually called gates, of the flow of information inside the LSTM unit: an input gate, an output gate and a forget gate. Some variations of the LSTM unit do not have one or more of these gates or maybe have other gates. These networks produce a gradient value which helps decide which side the prediction should be pointing out. The gradient value changes according to the number of predictions it gets right. The prediction is then compared with the actual value so that the model understands the data correctly and learns the emotion behind the sentence or word[2].

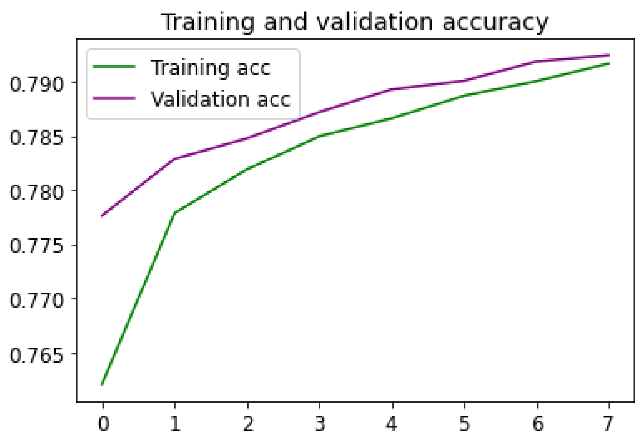

Fig 2: Validation and Training Accuracy

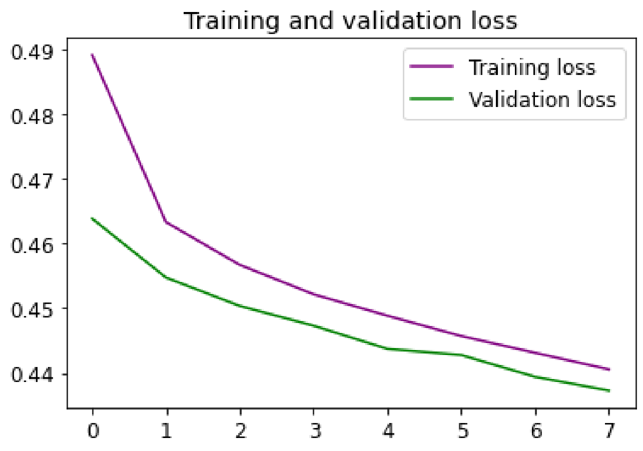

Fig 3: Training and Validation Loss

To start with the datasets we shortlisted, we had to preprocess since there were numerous NaN (Not a number) and null values all over. After cleaning the datasets, we loaded them into the Jupiter notebook. With the help of Keras Tensor flow Long-short-term memory model we imported, we started training it with the twitter dataset. For the activation function, we have used the SoftMax classifier. The SoftMax classifier uses the cross-entropy loss. The SoftMax classifier gets its name from the SoftMax function, which is used to squash the raw class scores into normalized positive values that sum to one, so that the cross-entropy loss can be applied[2].
There are many other classifiers, but we found this one to fit in our model perfectly. Out of the whole dataset we have used 80% of the data for training and the rest 20% for testing. The batch size for training has been set to 1024. The number of iterations in one cycle which is also known as Epoch has been set to 8. We have included early stopping parameters which helps us monitor validation accuracy. If the accuracy starts slipping or fading, we ensure to stop them so that we can help the model to move onto the next chunk of data once it reaches its highest accuracy. The dataset of twitter has been split into two parts where the minor part of 1 lakh tweets is used for validation testing. Training samples are different from validation samples to train the model in an ideal way and also avoid overfitting[3].

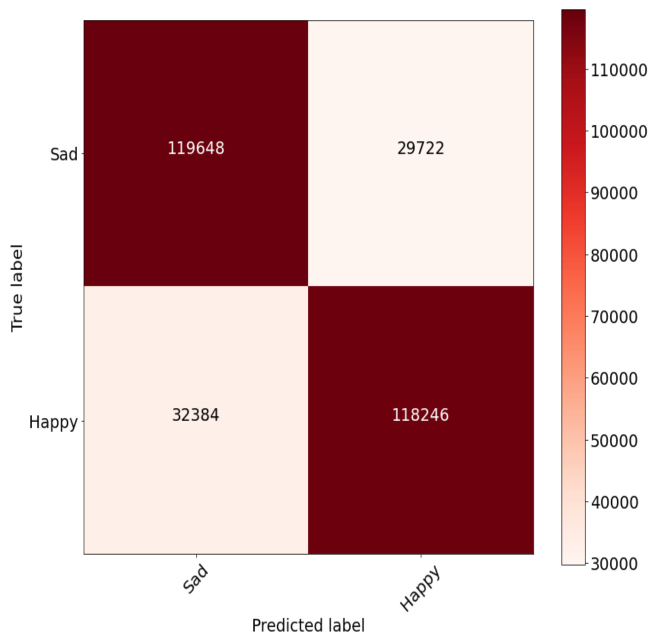

Fig 4: Confusion Matrix of model without chunks

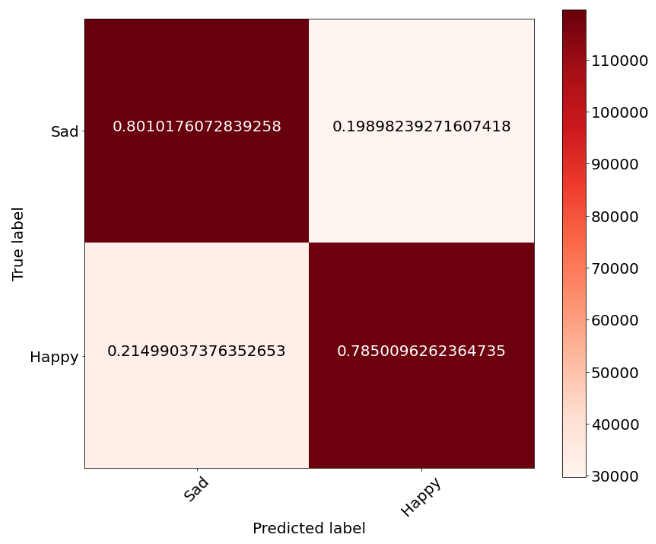

Fig 5: Normalized Confusion Matrix of the model without chunks

C. Novelty

We observed that the runtime for training our model with a static dataset goes up to 7070 seconds in total which is roughly 1 hour and 57 minutes. In order to reduce the runtime, we made some changes in the parameters. Firstly, we split up the data of 1.2 million into 10 equal slices. Now instead of training the whole dataset together like before, we started training the slices one after the other like a patch with the same model. Once the first slice gets trained, we move onto the next one by initiating the early stopping criteria. Instead of 8 epochs, there are 80 epochs running in this scenario, but the major difference is that it runs for a lot less time compared to the traditional methodology. We initialize the slice by setting up the start and end index which changes after every cycle of epochs. The early stopping criteria makes sure that the model moves onto the next slice of data if the validation accuracy starts dropping. A threshold has been set to 1 * e^{-4}. If the improvement of validation accuracy drops less than the threshold for two consecutive epochs or if the validation accuracy proves to be lesser than the previous one for two consecutive iterations, we would stop the current training and move onto the next chunk with best weights available. This helps us save a lot of time. Although our accuracy dropped by a percent for this methodology, but if we look at the bigger picture this methodology can be used to train live flowing data in less time. After comparing runtimes from the both the approaches we found this approach to achieve a runtime of 3582 seconds which is almost half of what we achieved previously.

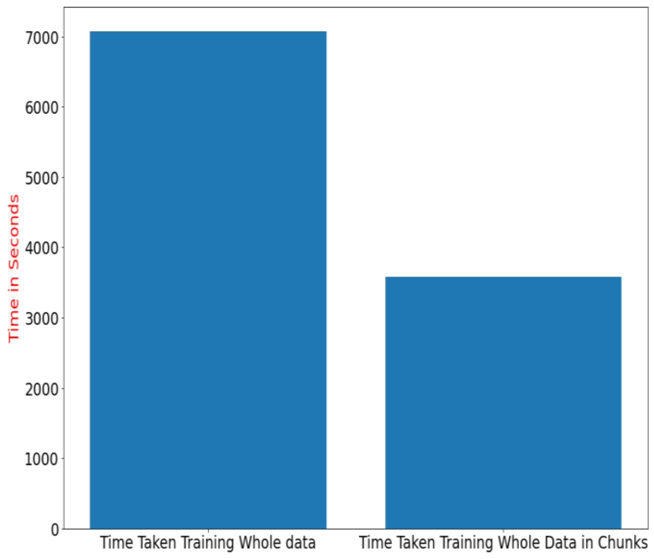

Fig 6: Runtime of initial methodology vs Novel methodology

This approach is better in our opinion. When the accuracy starts dropping, it would be meaningless to train with the same chunk. Due to such reason, we move onto the next chunk of data which saves time.

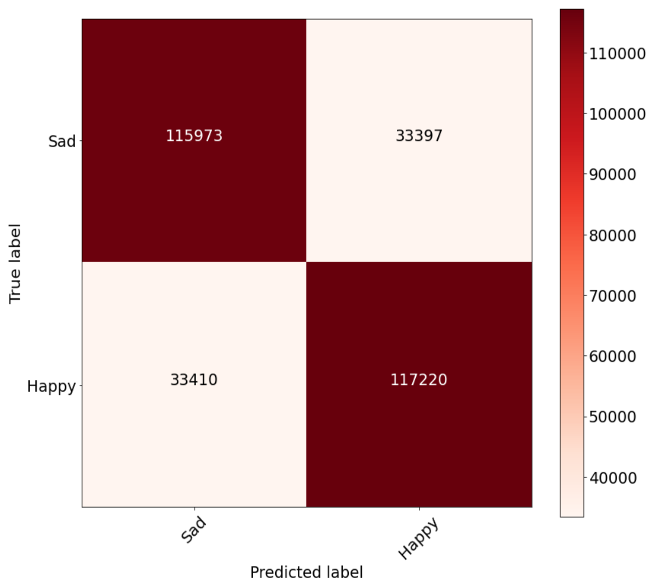

Fig 7: Confusion Matrix of the model with chunks

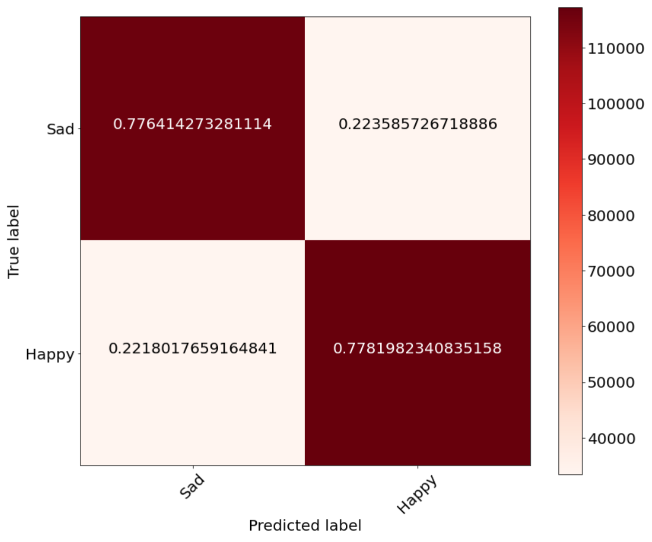

Fig 8: Normalized Confusion Matrix with chunks

D. User Data Case Study

Recommending songs and movies to the user's liking is a great challenge for us. Before assuming what user may like, we wanted to know a general idea of what people actually like to listen or watch during certain kind of emotion. We conducted a survey and collected 104 responses from different users. We learned that people usually want to listen or watch things when they are sad which might help them change their mood or keep their mind off of the problem. For example, a sad person might watch a comedy movie or listen to calm and soothing song. A happy user will watch a movie which entertains him or listen to a song which is his favorite. According to the responses we collected we will be making changes in our recommending criteria and help keep the users satisfied and happy[2].

We will be recommending songs and movies with a mixture of all genres but with a majority of the category that came in the responses. After collecting the data from the responses, we have made the recommending criteria much suitable and feasible for all users. The choice of song and movie are different for everyone. Based on the responses we collected, we found that a user might want to listen to a song when he is happy whereas the same song could be played by someone else when he is sad. So, the recommendation always depends on the user's personal choice. In order to make the recommendation eye-catchy, it would be feasible for us to get our hands on some personal preferences of each user. Just like Apple music asks a few questions on personal choices before subscribing to the service, we would like to do the same wherever our framework is being used so that we could recommend based on user preference[6].
Here are responses we collected from our survey.
If we look at the responses of what kind of movies people watch when they are happy or sad, majority of the users prefer to watch Comedy and thriller movies no matter what their mood is. This proves our point that it is up to each user of what kind of movie to watch or which song to listen to. In order to make our recommendation system better, we should be suggesting according to the user's personal preference[6].

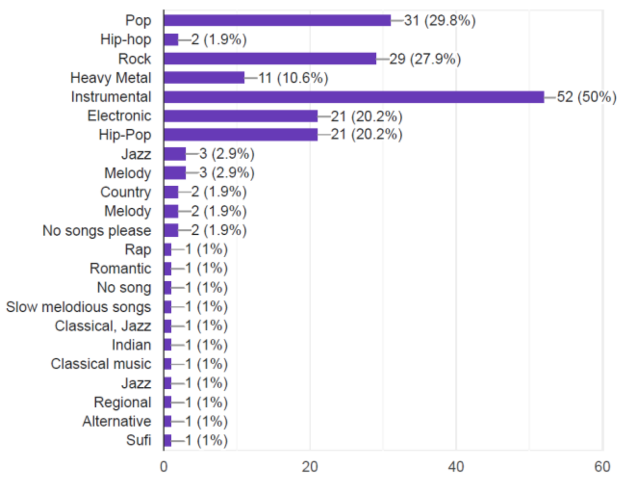

Fig 9: Songs people would listen to when they are sad

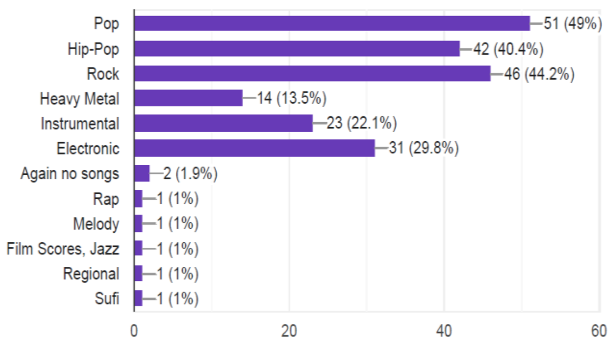

Fig 10: Songs people would listen to when they are happy

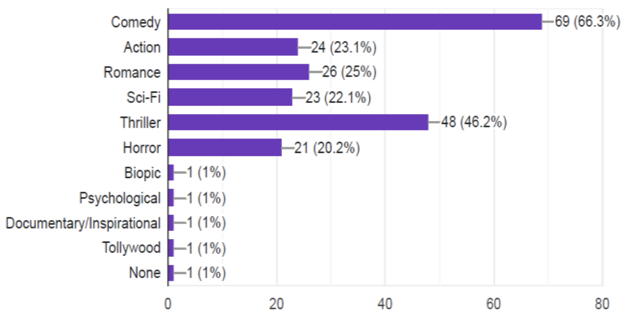

Fig 11: Movies people would watch when they are sad

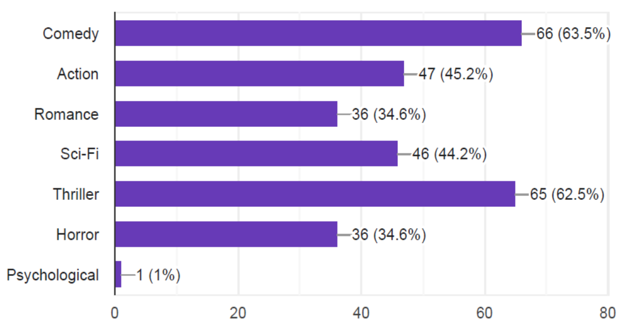

Fig 12: Movies people would watch when they are happy

IV. EXPERIMENTAL RESULTS

The three datasets namely Twitter (1.6 million rows), IMDB (85854), Spotify(1994) are used in designing our model. After filtering out the three datasets, we started cleaning it.

Since there were many null and NaN values in the dataset, we had to eradicate them in order to start working on them. Datasets got loaded successfully and we moved on to begin working with the model. We managed to successfully implement the Long-Short-Term memory model from Keras Tensor Flow. The batch size for training data was kept 1024 and the epoch size as 8. We hoped the iterations wouldn't take much time, but it took around 2 hours 10 minutes to complete[9].

After training the model successfully, we wanted to check whether our model is predicting the emotion behind a sentence correctly. We checked it with multiple examples and the model predicted all of them correctly. There were some issues with the model understanding sentences with sarcasm or two negatives but after looking closely, we managed to address the problem and solve it. The LSTM does have the ability to remove or add information to the cell state, carefully regulated by structures called gates. Gates are a way to optionally let information through. They are composed out of a sigmoid neural net layer and a pointwise multiplication operation. The sigmoid layer outputs numbers between zero and one, describing how much of each component should be let through. A value of zero means "let nothing through," while a value of one means "let everything through!" An LSTM has three of these gates, to protect and control the cell state. We used the LSTM approach to train the dataset of tweets over 1 million[7]. We tested the trained model by feeding the random tweets having positive or negative impact and it has predicted everything correctly so far. Once we get an output of the tweet being negative or positive, the data is being passed to the function where the recommender system then recommends songs and movies from the dataset we have linked[7].

Epoch	8
Batch Size	1024
Dropout rate	0.2
Recurrent dropout rate	0.2
Baseline	562,782
Early stopping threshold	1 * e^{-4}

Table 1: Hyperparameters

Below are the two tables which describe our classification report of the two different models, one with training on complete training set and other with data chunks.

	precision	recall	F1-score	Support
Sad tweet	0.79	0.80	0.79	149370
Happy tweet	0.80	0.79	0.79	150630
Accuracy			0.79	300000
Macro avg	0.79	0.79	0.79	300000
Weighted avg	0.79	0.79	0.79	300000

Table 2: Classification report of model without chunks

	precision	recall	F1-score	Support
Sad tweet	0.78	0.78	0.78	149370
Happy tweet	0.78	0.78	0.78	150630
Accuracy			0.78	300000
Macro avg	0.78	0.78	0.78	300000
Weighted avg	0.78	0.78	0.78	300000

Table 3: Classification report of model ran with chunks

For tweets depicting happiness: Since we need to enhance the emotions, people who have tweeted happy tweets will be recommended movies of genre- Drama and Thriller majorly along with other interesting genres like Action, Comedy and Sci-Fi. Songs that will be recommended will be of the genre- Pop and Hip-pop[5].

ID	Title	Genre	Rating
1820	All the small things	Pop punk	79
759	Always remember us this way	Dance pop	81
751	Someone You loved	Pop	85
691	Castle on the hill	Pop	79
505	When I was your man	Dance pop	82
747	Shotgun	Folk-pop	82
577	Photograph	Pop	84
726	Shallow	Dance pop	88
669	24K magic	Dance pop	78
687	Shape of You	pop	87

Table 4 : Songs recommended for the positive tweet.(Pop and Hip-pop)

ID	Title	Genre	Rating
67185	Nobody's perfect	Action, Drama, Romance	8.1
70744	Logan	Action, Drama	8.1
71114	Adios Vaya Con Dios	Crime, Drama, thriller	8.2
68015	The two pamelas	Crime, Drama, Mystery	8.2
49557	Loyalty and Respect	Action, Crime, Drama	8.2
56563	Warrior	Action, Crime, Sport	8.2
84775	Poeta	Comedy, drama, romance	8.3
57651	Prisoners	Crime, Drama, Mystery	8.1
67006	It's such a beautiful day	Animation, Drama	8.3
84348	Hamilton	Biography, Drama, History	8.7

Table 5: Movies recommended for the positive tweet. (Drama, Romance and Thriller)

For tweets depicting sadness: For people who posted negative tweets will be recommended movies of genre- Sci-fi, Action-comedy. The movies will be a mixture of action and comedy so that we can help the person calm down and

relax using humor. Only movies with both action and comedy as their genre will be recommended to the user so that we do not enhance or encourage violence[10]. Songs of the genre Instrumental, Pop and Rock will be recommended so that we can treat the emotion of sadness through catharsis.
For the recommendation filter we have fit a condition which will scrim the top 50 rated movies and suggest random 10 to the user. The same condition goes for songs as well.

ID	Title	Genre	Rating
1886	Brown eyed girl	Classic Rock	79
1113	Highway to hell	Album Rock	83
943	Sweet Home alabama	Album rock	82
1241	I'm still standing	Glam rock	79
839	Stairway to heaven	Album rock	79
262	Numb	Alternative metal	81
524	Do I wanna know?	Garage rock	82
112	How to Save a Life	Modern Rock	80
706	Thunder	Modern Rock	86
1602	Creep	Alternative Rock	82

Table 6: Songs recommended for the negative tweet. (Rock and Heavy metal)

ID	TITLE	Genre	Rating
41787	Spider-Man 2	Sci-Fi, Action	7.3
84110	Janelle Monae	Action, Comedy, Sci-Fi	7.3
64556	New Prime	Sci-Fi, Drama	7.9
68970	I origins	Sci-Fi, Adventure	7.4
62334	Black Panther	Sci-Fi, Action, Adventure	7.3
42569	Eternal sunshine of the spotless mind	Sci-Fi, Drama	8.3
37662	Equilibrium	Drama, Sci-Fi	7.4
63736	The Bang-Bang Brokers	Action, Comedy	7.4
65697	The history of future folk	Comedy, Musical	7.2
38072	Donnie Darko	Sci-fi, Drama	8.0

Table 7: Movies recommended for the Negative tweet. (Sci-fi, Comedy and Action-Comedy)

V. CONCLUSION

Considering emotions as a significant part of our daily life, we designed a recommender-based system which will suggest songs and movies to the users according to their emotional state of mind. We achieved a better runtime with the model after making significant changes in parameters. For future work we are planning to implement changes on semantic analysis as well as topic modelling where our model will be to understand sentences with sarcasm and double meaning. We only considered two emotions in this approach, but we will be definitely considering other emotions as well. We learned how to use Keras tensor flow and other packages which helped us build this model. We figured out an approach to reduce the runtime with the inclusion of early stopping criteria.

References

[1] A. Go, R. Bhayani, and L. Huang. Twitter sentiment classification using distant supervision. CS224N
project report, Stanford, 1(12):2009, 2009.

[2] Kumar, J., Goomer, R., & Singh, A. K. (2018). Long short-term memory recurrent neural network (lstm-rnn) based workload forecasting model for cloud datacenters. *Procedia Computer Science*, *125*, 676-682.

[3] Ouyang, Xi, et al. "Sentiment analysis using convolutional neural network." *2015 IEEE international conference on computer and information technology; ubiquitous computing and communications; dependable, autonomic and secure computing; pervasive intelligence and computing*. IEEE, 2015.

[4] Can EF, Ezen-Can A, Can F. Multilingual sentiment analysis: An RNN-based framework for limited data. arXiv preprint arXiv:1806.04511. 2018 Jun 8.

[5] Wang, Jin, et al. "Dimensional sentiment analysis using a regional CNN-LSTM model." *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. 2016.

[6] Chen, Minghai, et al. "Multimodal sentiment analysis with word-level fusion and reinforcement learning." *Proceedings of the 19th ACM International Conference on Multimodal Interaction*. 2017.

[7] Alzaidy, Rabah, Cornelia Caragea, and C. Lee Giles. "Bi-LSTM-CRF sequence labeling for keyphrase extraction from scholarly documents." *The world wide web conference*. 2019.

[8] Bouchard, Guillaume. "Efficient bounds for the softmax function, applications to inference in hybrid models." *Presentation at the Workshop for Approximate Bayesian Inference in Continuous/Hybrid Systems at NIPS-07*. 2007.

[9] Yu, Dejian. "Softmax function based intuitionistic fuzzy multi-criteria decision making and applications." *Operational Research* 16.2 (2016): 327-348.

[10] Shim, Kyuhong, Minjae Lee, Iksoo Choi, Yoonho Boo, and Wonyong Sung. "Svd-softmax: Fast softmax approximation on large vocabulary neural networks." In *Advances in Neural Information Processing Systems*, pp. 5463-5473. 2017.

GITHUB REPOSITORY LINK: https://github.com/subbuilla/Mining.git