**Title Page**

**Improving Song recommender using Long Short Term Memory.**

**S.Kishore, Dr. Anushya**

S.Kishore

Research Scholar,

Department of Computer Science Engineering,

Saveetha School of Engineering,

Saveetha Institute of Medical and Technical Sciences,

Saveetha University, Chennai, Tamil Nadu, India. Pin code: 602105

19211156.sse@saveetha.com

Dr Anushya

Project Guide, Corresponding Author,

Department of Product Development,

Saveetha School of Engineering,

Saveetha Institute of Medical and Technical Sciences,

Saveetha University, Chennai, Tamil Nadu, India. Pin code: 602105

Guide Email:

**Abstract:**

Long Short-Term Memory (LSTM) networks have emerged as a cornerstone in the realm of recurrent neural networks, offering a robust solution to address the challenges associated with vanishing and exploding gradients. This abstract provides a concise overview of the key aspects of LSTM networks, emphasizing their unique architecture designed to capture and retain long-term dependencies in sequential data. The discussion encompasses the historical context, the core principles underlying LSTM networks, and their widespread applications across various domains. The abstract aims to offer a glimpse into the pivotal role of LSTMs in advancing the capabilities of sequential data processing, paving the way for enhanced performance in tasks such as natural language processing, time-series analysis, and beyond.

**Keywords**: Long Short Term Memory (LSTM), Music Recommendation, Personalization, Feature Extraction, Digital Music Platforms.

**Introduction:**

In the ever-evolving landscape of artificial intelligence and machine learning, the need for models capable of effectively handling sequential data has become increasingly evident. The introduction delves into the historical evolution of recurrent neural networks (RNNs) and the limitations that led to the inception of Long Short-Term Memory (LSTM) networks. LSTMs address the challenges of preserving contextual information over extended sequences, overcoming the hurdles of vanishing and exploding gradients that plagued traditional RNNs.

The introduction outlines the fundamental principles of LSTMs, highlighting their intricate architecture, which includes memory cells, input and output gates, and a forget gate. These components collectively empower LSTMs to selectively store and retrieve information over extended time steps, facilitating the learning of long-range dependencies in sequential data.

As the narrative unfolds, the introduction also touches upon the versatility of LSTMs, showcasing their applicability in diverse domains. From natural language processing and speech recognition to time-series analysis and beyond, LSTMs have demonstrated remarkable success, underlining their significance in shaping the landscape of modern machine learning.

In essence, this introduction sets the stage for a comprehensive exploration of LSTMs, laying the groundwork for a deeper understanding of their architecture, capabilities, and the transformative impact they have had on sequential data processing.

**Materials and methods:**

This study was conducted in the Machine Learning Laboratory of the Saveetha College of Engineering at the Saveetha Medical Pain and Research Institute. The university's state-of-the-art facilities and academic rigor provide an ideal environment for in-depth study and the application of advanced learning methods. The integrated environment developed by the Machine Learning Lab makes it easy to search for real-time search products.

**LSTM:**

LSTM (Long Short-Term Memory) networks have proven pivotal in advancing the capabilities of sequence modeling, particularly in tasks involving natural language processing and time-series analysis. Their architecture, designed to capture long-term dependencies in sequential data, makes LSTMs well-suited for learning intricate patterns and contextual information.

**Input:**

The training dataset comprises sequential input data (X\_train) paired with corresponding target labels (y\_train), while the testing dataset consists of unseen sequential data (X\_test) for prediction.

**Output:**

The ultimate goal is to generate predictions for the test data, providing insights into the sequential patterns and relationships learned by the LSTM model.

**Steps for LSTM Implementation:**

1. Specify the architecture of the LSTM network, including the number of LSTM units, input dimensions, and any additional layers.

2. Prepare the sequential data for training by encoding, normalizing, or scaling features as needed.

3. Create an LSTM model with the specified architecture, incorporating input sequences and any relevant additional layers.

4. Utilize the training dataset (X\_train, y\_train) to train the LSTM model. Adjust model weights iteratively to minimize the loss function and enhance predictive accuracy.

5. Assess the model's performance on a validation set to ensure it generalizes well to unseen data and avoids overfitting.

6. Apply the traied LSTM model to the sequential data in the testing dataset (X\_test) to generate predictions for each data point.

7. Depending on the nature of the task, post-process the LSTM predictions. For example, in classification tasks, apply softmax activation for probability distribution, or for regression tasks, scale predictions if necessary.

8. Combine individual predictions, for instance, by taking the median or using an ensemble approach to improve robustness.

9. Present the aggregated predictions as the final prediction array for the test data.

10. Save the predictions and any relevant metrics for future analysis or comparison.

11. Provide the final prediction array for further evaluation or integration into downstream applications.

**LightGCN Classifier Algorithm:**

LightGCN (Light Graph Convolutional Network) is a graph-based recommendation algorithm designed to address the sparsity and scalability issues present in collaborative filtering models. It was introduced in the paper titled "LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation" by Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang, published in 2020.

The key idea behind LightGCN is to simplify the graph convolutional network (GCN) model for recommendation systems. Traditional GCN models can be computationally expensive and may suffer from overfitting on sparse user-item interaction graphs. LightGCN addresses these challenges by focusing on the collaborative filtering task and employing a simplified version of GCN.

**Input:**

Training data (X-train, y-train): learning features and text

Test data (X-test): estimated features

Number of neighbours (k): parameters of the algorithm

< BR>

**Output:**

Prediction: Array containing predictions for each data point in x-test

Step 1: Initialize empty prediction

Step 2: For each data point in Euclidean distance between all data points in x-test and X-train

Step 4: Record the distance and get the form bar in X-train

Step 5: Calculate the distance from the top point to

Step 6: Select initial neighbours

Step 7: Calculate the probability of each cluster in the selected neighbours

Step 8: Check the most common neighbours category

Step 9: Save the predicted product as the most common category for x-test

Step 10: Repeat steps 2-9 for all points in test X.

**Statistical Analysis:**

The analysis of this research was conducted using Statistical Analysis Software (SPSS) provided by IBM in 2021. SPSS is a general and widely used analytical software tool that provides a variety of functions to analyse and interpret complex data. The use of SPSS allows a critical evaluation of the prediction performance of the Graph Convolutional Network(LightGCN) classifier and the LSTM algorithm in the context of Song Recommendation. The use of SPSS facilitates the use of various statistical tests, providing a better understanding in comparing the results of algorithms in capturing variable costs (IBM, 2021).

**Result:**

Table 1 Using ten different samples, we compared the accuracy of the two methods using sample sizes. The library algorithm for two different models is explained in detail. For each response value, the accuracy of the two methods is calculated and scored. Measure and record the final average. The last column shows the actual average of the two methods. The average accuracy of the LSTM algorithms is 48.20%, while the average accuracy of the LIGHTGCN Classifier method is 68.88%.

**Table** **2** represents the comparative performance metrics of LightGCN Classifier and Long Short Term Memory in predicting Music Recommendations. Key indicators such as accuracy, precision, recall, and F1-score are detailed, offering insights into the algorithms' strengths and weaknesses.

**Table** **3** defines the correlation coefficients between input features and song recommendation movements for both GCN Classifier and Long Short Term Memory. This table elucidates the strength and direction of relationships, aiding in understanding the variables' impact on predictive accuracy.

**Figure** **1** visually represents the distribution of predicted vs. actual stock prices for both GCN Classifier and LSTM. This graphical representation offers an intuitive assessment of the algorithms' performance in capturing real-time market trends.

**Discussion:**

The research paper titled "Music Recommendation Using LSTM" introduces a groundbreaking approach to music recommendation systems by harnessing the power of Long Short-Term Memory (LSTM) networks. In contrast to conventional methods relying on matrix factorization, this study delves into the efficacy of LSTMs in elevating music recommendation performance. Leveraging LSTMs, well-regarded for their ability to capture temporal dependencies, represents a departure from traditional techniques. The study underscores the superior performance and interpretability of LSTMs compared to conventional recommendation systems. The LSTM model's proficiency in learning intricate sequential patterns within music features, facilitated by its memory cells and gating mechanisms, contributes to its effectiveness in delivering personalized music recommendations. One of the key strengths of the LSTM model lies in its ability to address challenges posed by sparse recommendation tasks. The model demonstrates promising potential to surpass existing methods in handling scenarios where user-item interactions are limited. The efficiency of the LSTM model is attributed to its nuanced architecture. By incorporating memory cells and gating mechanisms, the LSTM model achieves effective feature extraction without the need for extensive feature transformation and nonlinear activation functions. This streamlined architecture not only enhances computational efficiency but also contributes to the model's interpretability. The study extends the practical applications of LSTMs to the domain of ubiquitous personalized music recommendations, showcasing their adaptability to diverse datasets, including unconventional sources such as smart bracelet data. This emphasizes the versatility of LSTMs in accommodating various user interactions and preferences, paving the way for innovative approaches to music curation. While acknowledging the study's limitations, such as a small sample size and the need for further validation, the findings provide valuable insights into the potential of LSTMs to revolutionize music recommendation systems. The research opens avenues for future exploration, encouraging investigations into the scalability and robustness of LSTMs in large-scale datasets and their applicability in diverse recommendation systems.

**Conclusion:**

This research significantly contributes to advancing the landscape of recommendation systems by highlighting the potential of LSTMs in tailoring personalized music suggestions. The implications of this study transcend the confines of the music industry, indicating broader applications of LSTMs in recommendation systems across a myriad of domains.

**DECLARATIONS Conflict of Interests:**

The authors of this paper declare no conflict of interest.

**Authors Contributions**

Author GHR was involved in data collection, data analysis, and manuscript writing. Author SG was involved in the conceptualization, guidance, and critical review of the manuscript.

**Acknowledgments**

I want to express my gratitude towards Saveetha School of Engineering, Saveetha Institute of Medical and Technical Sciences (Formerly Known as Saveetha University) for providing the necessary infrastructure to carry out this work successfully.

**References:**

[ 1] Mayer R, Rauber A. Music Genre Classification by Ensembles Of Audio And Lyrics Features.

In: Proceedings pf the 12th International Society for Music Information Retrieval Conference

(Ismir 2011). Miami, Fl (2011).

[ 2] Sturm Bl. Classification Accuracy Is Not Enough: On the Evaluation of Music Genre

Recognition Systems. J Intell Inform Syst. (2013)

[ 3] Yang Yh, Chen Hh. Machine Recognition of Music Emotion: A Review. Trans Intell Syst Techn.

(2013)

[ 4] Huq A, Bello Jp, Rowe R. Automated Music Emotion Recognition: A Systematic Evaluation. J

New Music Res. (2010–11) 39:227–44.

[ 5] Knees P, Schedl M. Music Similarity and Retrieval — An Introduction to Audio- And Web-Based

Strategies. Berlin; Heidelberg: Springer (2016).

[ 6] Karydis I, LidaKermanidis K, Sioutas S, Iliadis L. Comparing Content and Context Based

Similarity for Musical Data. Neurocomputing. (2013)

[ 7] Schedl M, Knees P, Mcfee B, Bogdanov D, Kaminskas M. Music Recommender Systems. In: Ricci

F, Rokach L, Shapira B, Kantor Pb, Editors. Recommender Systems Handbook, 2nd Ed. Boston,

Ma: Springer (2015).

[ 8] Ricci F, Rokach L, Shapira B, Kantor Pb Editors. Recommender Systems Handbook, 2nd Ed.

Boston, Ma: Springer (2015).

[ 9] Bell Rm, Koren Y. Lessons from The Netflix Prize Challenge. AcmSigkdd Exp Newslett. (2007)

9:75–9. Doi: 10.1145/1345448.1345465

[ 10] De Gemmis M, Lops P, Musto C, Narducci F, Semeraro G. In: Ricci F, Rokach L, Shapira

B, Kantor Pb, Editors. Semantics-Aware Content-Based Recommender Systems, 2nd Ed. Boston,

Ma: Springer (2015).

**Tables and Figures:**

**Table 1**

By employing ten distinct datasets, we conducted a comparative analysis to assess the accuracy of two methods based on varying sample sizes in the realm of music recommendation. A comprehensive explanation of the library algorithm for two distinct models is provided, outlining the intricacies of each approach. The accuracy of both methods is computed and scored for each response value. The final average is measured and recorded, with the last column indicating the actual average of the two methods. Notably, the average accuracy of the LSTM-based algorithms stands at 63.22%, whereas the LIGHTGCN Classifier method achieves an average accuracy of 75.16%. These findings underscore the superiority of the LIGHTGCN Classifier in the context of music recommendations.

| Iterations | Accuracy of LSTM | Accuracy of LightGCN |
| --- | --- | --- |
| 1 | 48.00 | 69.29 |
| 2 | 48.25 | 69.28 |
| 3 | 48.96 | 69.27 |
| 4 | 47.94 | 69.24 |
| 5 | 47.24 | 69.21 |
| 6 | 47.31 | 69.07 |
| 7 | 49.36 | 68.97 |
| 8 | 49.86 | 68.83 |
| 9 | 48.15 | 68.45 |
| 10 | 17.00 | 68.21 |

**Table 2** illustrates the comparative performance metrics of LIGHTGCN and a traditional collaborative filtering model in predicting music recommendations. Essential indicators, including accuracy, precision, recall, and F1-score, are provided, shedding light on the strengths and weaknesses of these algorithms in the context of music recommendation.

| Algorithm | | N | Mean | Std. Deviation | Std. Error Mean |
| --- | --- | --- | --- | --- | --- |
| Accuracy | LightGCN | 10 | 68.8820 | .43194 | .13659 |
| LSTM | 10 | 48.2070 | .93954 | .29711 |

**Table 3** presents the correlation coefficients between input features and music preference predictions for both the LIGHTGCN Recommender and the LSTM-based model. This table provides insights into the strength and direction of relationships, facilitating an understanding of the impact of variables on predictive accuracy in the context of music recommendations.

|  | | Leven’s Test for Equality of Variances | | t-test for Equality of Means | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| F | Sig. | t | DF | Sig(2-tailed) | Mean Difference | Std. Error Difference | Lower | Upper |
| Accuracy | Equal Variance  Assumed | 3.730 | .069 | 63.226 | 18 | .000 | 20.67500 | .32700 | 19.98799 | 21.36201 |
| Equal Variance  Not Asuumed |  |  | 63.226 | 12.642 | .000 | 20.67500 | .32700 | 19.96651 | 21.38349 |

**Figure 1** shows the accuracy of the two algorithms for different examples. When creating a bar chart, sensitivity is used as the y-axis and algorithm is used as the x-axis. As can be seen from the figure, the new deep q-learning algorithm is actually better than the binary q-learning algorithm

