Session Based Recommendations using CNN-LSTM with Fuzzy Time Series

Punam Bedi[1], Purnima Khurana[1(✉)], and Ravish Sharma[2]

[1] Department of Computer Science, University of Delhi, Delhi, India
pbedi@cs.du.ac.in, pk0403@gmail.com
[2] P.G.D.A.V. College, University of Delhi, Delhi, India
ravish.sharma@pgdav.du.ac.in

Abstract. Session based Recommender systems consider change in preferences by focusing on user's short term interests that may change over a period of time. This paper proposes FS-CNN-LSTM-SR, a hybrid technique that uses CNN (Convolutional Neural Networks) and LSTM (Long Short Term Memory) deep learning techniques with fuzzy time series to recommend products to user based on his activities performed in a session. The advantage of our proposed method is that it combines the benefits of both CNN and LSTM. CNNs are capable of extracting complex local features and LSTM learn long term dependencies from sequential session data. The performance of FS-CNN-LSTM-SR is evaluated on YOOCHOOSE dataset from RecSys Challenge 2015 and is compared with three variations viz. LSTM-SR, CNN-LSTM-SR and FS-LSTM-SR. We observed that our proposed approach performed better than other three variations. The proposed technique is applicable on any E-commerce dataset where user purchasing choices need to be predicted.

Keywords: Session based recommender system · Deep learning · Fuzzy Time Series

1 Introduction

Recommender Systems are a boon in this era of huge data as they help users to make decisions from large number of available choices. Items, songs, people, places and even search results can be recommended to users. These systems provide personalized suggestions to the users which not only help users to take right decisions but also help the provider to retain their customers or users of the system [12, 19].

Traditional recommendation techniques such as Collaborative Filtering and Content based approach emphasize on long-term user preferences which are static in nature. A special class of recommender systems known as Sequential recommender systems recommends items to users considering sequential dependencies for the interactions between users and items. Most of the recommendation approaches do not take into consideration short term interest drifts and ignore user's preference shift through the time. Session-based Recommender systems are a category of recommender systems

A. Dev et al. (Eds.): AIST 2021, CCIS 1546, pp. 432–446, 2022.
https://doi.org/10.1007/978-3-030-95711-7_36

which take into account change in user preferences over the time by determining user's choices from their short term interests. These systems consider sequential interactions and also take into account user interest drifts from time to time.

Various recommendation techniques such as model based, machine learning and deep learning have evolved over the years [24]. Deep learning models are capable of handling non-linear user interactions which can be useful to determine the complex relationships between users and item interactions. The major advantage of deep learning is automatic feature extraction which reduces the manual efforts [9]. CNN (Convolutional Neural Network) is a deep neural network technique which assigns weights and biases to different aspects of input and determines important features. It is less prone to noisy data and is capable of learning crucial features independent from time [26]. RNN (Recurrent Neural Network) is the most suitable deep learning technique for session based recommender systems as it considers sequence of events within a session and is well suited when we want to predict future behavior based on the past logs or activities [3, 5, 16]. RNN suffers from vanishing gradient problem that occurs when the gradient becomes so small that there is no parameter update and the model cease to learn. LSTM (Long Short Term Memory), a variation of RNN, is the more advanced technique which maintains information in memory blocks and overcomes the vanishing gradient problem of RNN [7]. Fuzzy logic helps to deal with inconsistencies in a situation by considering an intermediate value rather than absolute true (1) or absolute false (0). It can be used with neural networks to refine the weights and help to better predict in uncertain situations. Also, fuzzy sets are used for modeling and predicting time series to obtain fuzzy time series.

Highlights of this paper:

- A hybrid technique combining the capabilities of fuzzy logic and deep learning has been utilized for generating Session based recommendations.
- A Convolutional layer is used for learning time independent important local features from input data.
- LSTM has been utilized for remembering sequence of events and providing recommendations based on user's preferences of the current session. It has been used to extract time series features.
- Various features of input data have been fuzzified to obtain fuzzy time series.
- Click patterns of users along with purchase patterns have been utilized for generating recommendations.
- The proposed technique is multivariate and generates a final list of items to be recommended to the user based on the class of predicted items, their co-occurring items as well as popularity of items.

The organization of this paper is as follows. Section 2 reviews the related work in this area. The proposed approach is explained in Sect. 3 and is subsequently evaluated and compared with three variations viz. LSTM-SR, CNN-LSTM-SR and FS-LSTM-SR in Sect. 4. Finally, Sect. 5 concludes this work.

2 Related Work

In this section, review of literature related to session based recommender systems and motivation for our work is presented.

Deep learning techniques provide better prediction results as they consider sequence of user's actions, automate feature extraction and extrapolate patterns [4]. Various deep learning techniques such as Recurrent Neural Networks (RNN) and Convolutional Neural Networks (CNN) have been used for development of social recommender systems [14] and session based recommendations. Session based recommendation has been utilized in various domains including music, e-commerce websites, travel destinations and hotels [6, 8, 13]. Researchers have argued that by modeling the whole session, more accurate recommendations can be provided and proposed an RNN-based approach for session-based recommendations [6]. RNN is used for time series data analysis and it takes into consideration sequential dependencies in the input data. LSTM, a variation of RNN, maintains information in the form of memory blocks and thus solves the problem of vanishing gradient that occurs in RNN [7]. LSTM is also capable of remembering longer sequences and is generally operated for text manipulation tasks where larger input needs to be remembered. So, many researchers have utilized them in a variety of domains including machine reading, financial market prediction, traffic speed prediction, trajectory prediction and even session based recommendations [2, 3, 10–12]. Srilakshmi et al. have applied LSTM on E-commerce datasets and the items are embedded using Graph based Embedding Technique [22] whereas Hidasi utilized GRU for session recommendations in E-commerce domain [6].

One or more deep learning techniques are often combined by researchers to exploit the advantages of every technique. CNN layers and LSTM layers can be cascaded one after another for sequential time series data. The convolutional layers are used for learning internal representation of time series data i.e. local features of the input data and the learned features are further fed into LSTM layer for identifying sequential dependencies and then making final predictions. Zhang et al. has also combined RNN and CNN for session based recommendation on E-commerce dataset [25]. Gabreil et al. has utilized both LSTM and CNN for session recommendation on news data [21]. We have implemented LSTM, CNN and combination of LSTM, CNN on YooChoose Dataset which is an Ecommerce dataset from RecSys Challenge 2015.

Uncertainty in user's behavior can be very well handled using fuzzy logic. Some features like amount of time spent on a product before it was bought, day of the week or month when the product was bought may not have crisp values. Such features account for uncertainty in user's behavior which can be handled by mapping their values to fuzzy values. Conventional time series has crisp values but cannot handle uncertain situations. Fuzzy time series works on the principle of fuzzy logic but the data is organized in the form of time series. Researchers have used deep learning techniques to model fuzzy Logical Relationships [15] and combined fuzzy time series with CNN for short-term load forecasting [17]. Wang, *et al.* has combined attention mechanism- a deep learning technique with fuzzy logic to create a fuzzy neural network for session based recommendation [23]. But, to the best of our knowledge no evidence has been found for utilization of both LSTM and CNN with fuzzy logic for session recommendations.

In this paper, we have proposed a hybrid technique which combines fuzzy time series and multivariate CNN-LSTM considering both numerical and categorical features for session based recommendations. Fuzzy logic deals with handling uncertainty in user's preferences, Convolutional layer for extracting deep and important features and multivariate LSTM learns patterns from both non-fuzzy and fuzzy features and also sequential dependencies from input data. LSTM alone takes longer time to train but when combined with CNN the number of epochs required to train the model are reduced. We have chosen LSTM over Recurrent Neural Networks (RNN) and Attention Mechanism due to its ability to handle sequential data with longer sequence. RNN suffer from vanishing gradient problem and even, the efficiency of attention mechanism for longer sequences is greatly reduced as more weight parameters add to training time.

3 Proposed FS-CNN-LSTM-SR (Fuzzy Time Series-CNN-LSTM-Session Based Recommendations)

A hybrid technique FS-CNN-LSTM-SR (Fuzzy Time Series-CNN-LSTM-Session Based Recommendations) is proposed that combines Convolutional layer of CNN and LSTM with fuzzy logic for session based recommendations.

The approach works in two phases viz. Preprocessing phase and Recommendation phase. The architecture of our proposed technique is shown in Fig. 1 below. Preprocessing phase is required to clean the data and transform it to be correctly processed by the proposed technique. This step includes filling missing values, retrieving required fields from data, extracting new features, converting original time series data into fuzzy time series, data normalization and padding. In Recommendation phase, data is prepared in a format to be worked upon by CNN-LSTM, prediction of user purchasing behavior is made and finally a list of items is recommended to user.

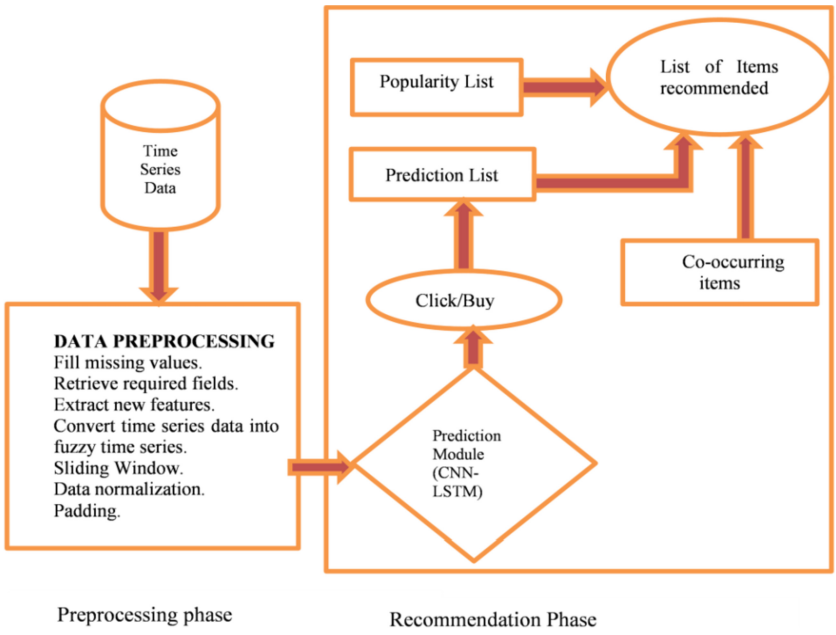

Fig. 1. Architecture of FS-CNN-LSTM-SR

3.1 Preprocessing Phase

The data has been preprocessed to be operated upon by proposed FS-CNN-LSTM-SR technique. The preprocessing steps are as follows:

Preprocessing

- Retrieve m attributes $\{x_1, x_2, \ldots, x_m\}$ from input data files.
- Create a single file by merging different files on session id.
- Sort every session on the basis of timestamp (t, t+1, … , t+n) to obtain sequential records in a session.
- Derive new domain specific features such that attribute set expands to $\{x_1, x_2, \ldots, x_m, x_{m+1}, x_{m+2}, \ldots, x_{m+n}\}$.
- Convert Time Series data into Fuzzy Time Series data $\{fx_1, fx_2, \ldots, fx_m, fx_{m+1}, fx_{m+2}, \ldots, fx_{m+n}\}$.

Steps to Convert Input Time Series Data Into Fuzzy Time Series Data

- Define Universe of Discourse (UoD) for every input attribute x_i that needs to be mapped to fuzzy sets. x_i can be numerical or categorical.

- Partition UoD for every individual attribute into several overlapping intervals. The membership function can be Triangular, Gaussian or Trapezium. The number of partitions is different for individual attribute depending on the domain size of attribute.
- Convert Numerical values of each input feature x_i at timestamp 't', denoted by $x_i(t)$ into fuzzy values $fx_i(t)$ which results into a fuzzy time series.
- **Normalization:** Normalize all the attribute values between range 0 and 1. This step is performed so that no attribute has more impact on prediction due to its higher values.
- **Sliding Window:** Convert fuzzy time series data into a supervised learning problem such that the data of two previous timestamps '$t-2$' and '$t-1$' can be considered to predict user purchasing choice for the current time stamp 't'.
- **Data Padding:** Each session is appended with some rows so that all the sessions are same in length. The padded value is a value different from values not in any instance of the data (say -1).

3.2 Recommendation Phase

The preprocessed data is converted into a format so that it is further processed by CNN-LSTM model for prediction. The Recommendation Phase comprises of two modules viz. Prediction module and Recommendation module. Prediction module is used to predict the items which can become eligible for recommendation and Recommendation module determines the actual set of items to be recommended to user. Pred_List_User is the list of items maintained by prediction phase and Rec_List_User is the list of items to be recommended to user that is maintained by recommendation phase.

The items which are predicted in '*buy*' category by the prediction module and were not earlier bought by user are eligible for recommendation and stored in *Pred_List_user* for every user. The maximum prediction probability is a deciding factor to determine whether the user clicked (0) or bought (1) the product. The possible choices of prediction are shown below in Table 1.

Table 1. Actual and predicted values of item purchase status - Click/Buy

Actual	Predicted	Action
Click	Buy	Eligible for recommendation
Click	Click	Model learnt correctly
Buy	Click	Ignore
Buy	Buy	Model learnt correctly

The algorithm for the proposed technique is explained below. The algorithm is applicable for any E-Commerce dataset where user's purchasing behavior needs to be determined.

Algorithm

1. // Preprocessed data(obtained from section 3.1) is converted into a format to be operated upon by CNN-LSTM model in steps 2 to 5
2. $No\_of\_input\_instances,\ No\_of\_inp\_timestamps,\ No\_of\_features \leftarrow Reshape(Preprocessed\_Data)$ // the shape requirement is for CNN-LSTM
3. $Num\_data, Categorical\_data \leftarrow Split(Reshaped\_Data)$
4. $Processed\_cat\_data \leftarrow Embedding\_Layer(Categorical\_data)$ *// to deal with high dimensionality*
5. $Final\_processed\_data \leftarrow Data\_masking(Num\_data + Processed\_cat\_data)$ //to ignore padded values during training
6. // Steps 7 to 9 computes prediction of class(Click/Buy)
7. $z \leftarrow Conv\_layer, LSTM,\ Activation, Dropout(Final\_processed\_data)$ //combines three sets of four consecutively connected layers (Convolution, LSTM, Activation, Dropout)
8. $\sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}}$ // Softmax Activation function where z_i is the input vector, z_j is the output vector and k is the number of classes.
9. $Click\ or\ Buy \leftarrow Predict(Activation\_layer)$ // Predicts that the item would be either clicked or bought
10. Create *Pred_List_User* //Pred_List_User consists of those items which are predicted in buy category but not bought by user earlier.
11. Create $PopL_C$ // List of Items sorted in decreasing order of popularity grouped by Item Category
12. Create *GPopL* // Global Popularity List
13. Create *Rec_List_user* // Ranked Item Recommendation List of n items
14. M ← Length(*Pred_List_User*)
15. IF M=n then
16. *Rec_list_user←Pred_list_user*
17. ELSE IF M>n then
18. Top_n ← $I_1, I_2, \ldots\ldots, I_n$ // Retrieve Top-n items from Pred_list_user
19. Rec_list_user← Top_n
20. ELSE IF M<n then
21. M_items←Pred_List_user
22. Rem_items← $PI_1, PI_2, \ldots\ldots PI_{n\text{-}M}$ *//Retreive top n-M items from* $PopL_C$
23. Rec_list_user← M_items+Rem_items
24. ELSE IF M=0 then
25. Top_n ← $GI_1, GI_2, \ldots.., GI_n$ // Retrieve top-n items from GPopL
26. Rec_list_user ← Top_n
27. ENDIF
28. Recommend Rec_list_user //This list consists of n-items

4 Experimental Study

The proposed hybrid technique FS-CNN-LSTM-SR has been implemented in Python using Keras Framework. Keras is a deep neural-networks API developed on top of Tensorflow or Theano, the open source libraries for extensive numerical computation. The input time series data has been converted into fuzzy time series using python pyFTS library which provides various methods of partitioning and membership functions [20].

4.1 Dataset

YOOCHOOSE dataset - It is an e-commerce dataset obtained from RecSys Challenge 2015 which consists of multiple independent temporal sequences in the form of sessions and here one session pertains to one user [1].

Number of unique sessions in the original data is 9249729 and total number of records is 34154697. The number of sessions of length one and length two are 1259711 and 3558087 respectively. These sessions have been filtered out by sliding window. The sessions left to be evaluated by our proposed technique are 4431931 with 14778812 records. Number of unique categories and number of unique items are 340 and 52739 respectively.

4.2 Evaluation Metrics

Precision, Recall and F1-measure [18] evaluation metrics have been utilized to evaluate the efficacy of the prediction module in recommendation phase of our proposed technique.

Precision, Recall and F1-measure for class 'Buy' are defined in Eqs. (1), (2) and (3). In the similar manner, Precision, Recall and F1-measure for another class 'Click' can be defined. The weighted average F1-score is defined below in Eq. (4). Let Act_{Buy}, $Pred_{Buy}$ denotes Actual Buy instances and Predicted Buy instances respectively.

$$Precision_{Buy} = \frac{Total\ \#\ of\ Act_{Buy}\ predicted\ as\ buy}{Total\ \#\ of\ Pred_{Buy}} \tag{1}$$

$$Recall_{Buy} = \frac{Total\ \#\ of\ Act\_Buy\ predicted\ as\ buy}{Total\ \#\ of\ Act_{Buy}} \tag{2}$$

$$F_{1Buy} = \frac{2 \times Precision_{Buy} \times Recall_{Buy}}{Precision_{Buy} + Recall_{Buy}} \tag{3}$$

$$Weighted_{avg}(F1) = \frac{Total\ \#\ of\ Act_{Buy} \times F_{1Buy} + Total\ \#\ of\ Act_{Click} \times F_{1Click}}{Total\ instances} \tag{4}$$

Another evaluation criteria for determining the performance of a classifier is AUC-ROC i.e. Area under the Receiver Operating Characteristic curve. ROC curve depicts performance of classifier on minority or positive class. AUC summarizes an ROC curve and it is used to measure the distinguishing capability of classifier. It is computed by considering the true outcomes for both the classes (click, buy) from the test set and the

probabilities predicted for the buy class (minority class). The AUC value lies between 0.0 and 1.0. More the AUC value better is the performance of classifier [27].

The performance of a recommender system is determined based on the rank of the predicted items and it has been evaluated using metrics Recall@K and MRR@K for our technique. We have taken 'K' as 20. Recall@20 is the proportion of cases in which the predicted item lies in top-20 items. MRR@20 (Mean Reciprocal Rank) is defined as the mean of the reciprocal ranks of the predicted items and takes into consideration actual rank of the items. The reciprocal rank of the items with rank greater than 20 has been considered to be zero. Recall@20 and MRR@20 are shown in Eqs. (5) and (6) below where '*T*' is the total number of predicted items. All the predicted items are relevant for recommendation.

$$Recall@20 = \frac{\#\ of\ \ items\ recommended\ \ @20\ that\ are\ relevant}{Total\ \#\ of\ \ relevant\ items} \tag{5}$$

$$MRR@20 = (1/T)\sum_{n=1}^{T}\left(\frac{1}{rank(n)}\right) \tag{6}$$

4.3 Prediction

The prediction module was tested on three membership functions for fuzzy partitioning viz. Triangular, Trapezium and Gaussian membership functions as depicted in Fig. 2. Weighted-Average F1-score was calculated for each of the membership functions and trapezium membership outperformed the other two membership functions. For each interaction of every session, the prediction module predicts whether the item will be clicked or bought.

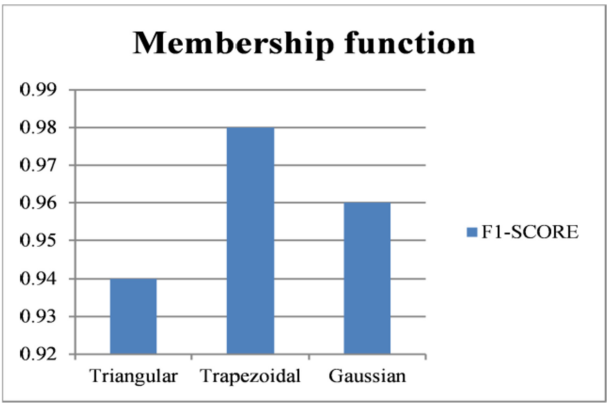

Fig. 2. F1-score for membership functions (Triangular, Trapezoidal, Gaussian)

We implemented four variations to generate top-K session recommendations and it has been observed that our proposed technique outperformed all other variations. Four variations are as follows:

a) LSTM-SR: The traditional LSTM was operated on original time series data.
b) CNN-LSTM-SR: Conv1D layer of Convolutional Neural Network was combined with LSTM in order to use the capabilities of both the models on original time series data.
c) FS-LSTM-SR: The traditional LSTM was operated on fuzzy time series data consisting of both fuzzy and non-fuzzy features.
d) FS-CNN-LSTM-SR: In this variation, CNN and LSTM models were combined and operated on fuzzy time series data consisting of both fuzzy and non-fuzzy features.

In literature, LSTM and CNN-LSTM have been utilized in various domains including Ecommerce, News, Music for session recommendations. We have implemented these existing techniques on YooChoose dataset and Keras Embedding layer has been utilized for embedding purpose. It was observed that fuzzy time series enhances the prediction accuracy when applied with LSTM in FS-LSTM-SR and CNN-LSTM in FS-CNN-LSTM-SR technique for both the categories: Click and Buy as shown in Table 2.

Table 2. Comparison of LSTM-SR, FS-LSTM-SR, CNN-LSTM-SR and FS-CNN-LSTM-SR for session based recommendations

Evaluation metric	LSTM-SR	FS-LSTM-SR	CNN-LSTM-SR	FS-CNN-LSTM-SR
Precision (CLICK)	0.86	0.90	0.88	0.91
Recall (CLICK)	0.95	0.93	0.96	0.92
Precision (BUY)	0.85	0.91	0.87	0.93
Recall (BUY)	0.68	0.72	0.65	0.75

Each categorical variable was dealt with separately and has been first encoded using label encoder and then embedding layer before fitting and evaluating. Embedding expects the categories to be ordinal encoded, but no relationship between categories is assumed. Embedding dimension of each categorical attribute has been chosen empirically as per the size of each attribute as shown in Table 3.

Table 3. Embedding dimension for individual attributes

Categorical attribute	Embedding size
ItemId	500
Category	50
Weekday	5
Time_of_the_day	5

Data has been reshaped to feed into CNN-LSTM since LSTM expects three dimensional inputs [samples, timestamps, features]. Three CNN-LSTM layers have been stacked to improve the performance of our proposed hybrid technique.

These variations have been tested on chunks of various sizes. For every chunk, 80% of the data has been selected for training and remaining 20% for testing. With the increase in data, the technique learnt to predict well. The weighted average F1-score for sessions of different sizes is shown below in Fig. 3. The x-axis represents session size and y-axis corresponds to F1-score. The graph shows that the weighted average F1-score improves as the data size increases since the model has more data to learn upon and FS-CNN-LSTM-SR technique outperformed all the techniques.

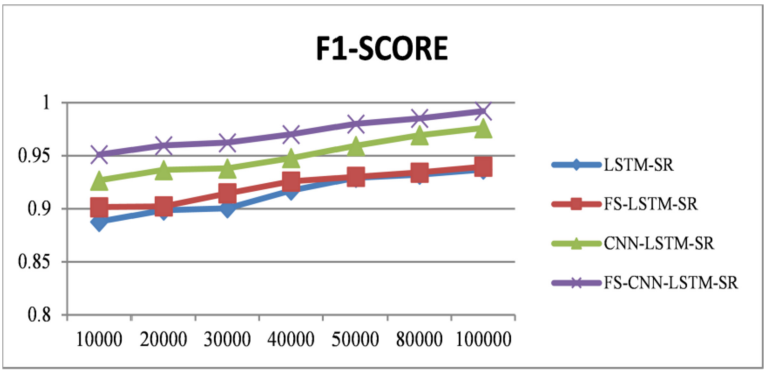

Fig. 3. F1-score for different number of sessions

Items are recommended based on the prediction results. The items in the *Rec_list_user* are the relevant top-n items to be considered by the user.

4.4 Hyperparameter Tuning

There are multiple hyperparameters which have been tuned to improve the performance of our proposed technique. With increase in number of epochs, the validation loss keeps on decreasing and it converges to a value for 100 epochs. It's value is minimum for the proposed technique FS-CNN-LSTM-SR and is shown below in Fig. 4. The x-axis represents number of epochs and y-axis represents validation loss computed using categorical cross entropy for our multi-class classification problem (Click (0), Buy (1), Padded value (-1)). The third category padded value (-1) does not interfere with the original categories and learning of the model is not affected at all.

The ROC-AUC curve for the minority class (i.e. buy class) is shown below in Fig. 5. The classifier was made to run on 100000 sessions for 100 epochs. The Area under the curve value is 0.94 which means that the prediction module is 94% capable of distinguishing between majority class (click) and minority class (buy).

Keras tuner has been used for hyper-parameter tuning in which optimum set of parameters determined were learning rate = 0.01, number of neurons = 16, activation function *'relu'* for the intermediate layers and 32 filters of size 2 for CONV1D layer. Dense, CNN-LSTM and Dropout layers have been used for processing input data. Dense layer is a deeply connected neural network layer. CNN layer is used to extract time independent important features which are further processed by LSTM Layer. LSTM

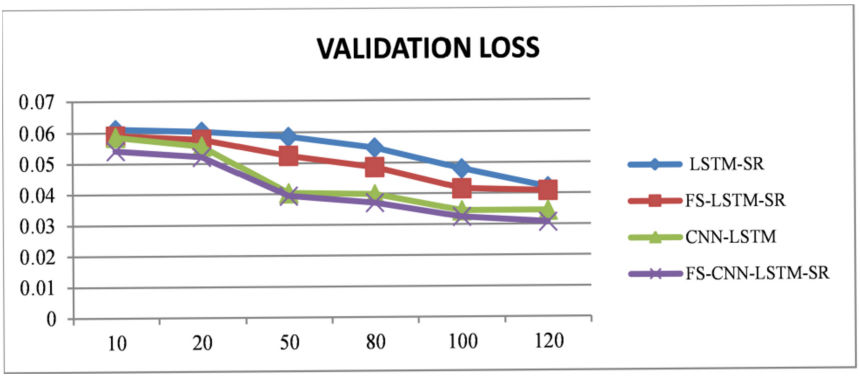

Fig. 4. Validation loss for different epochs

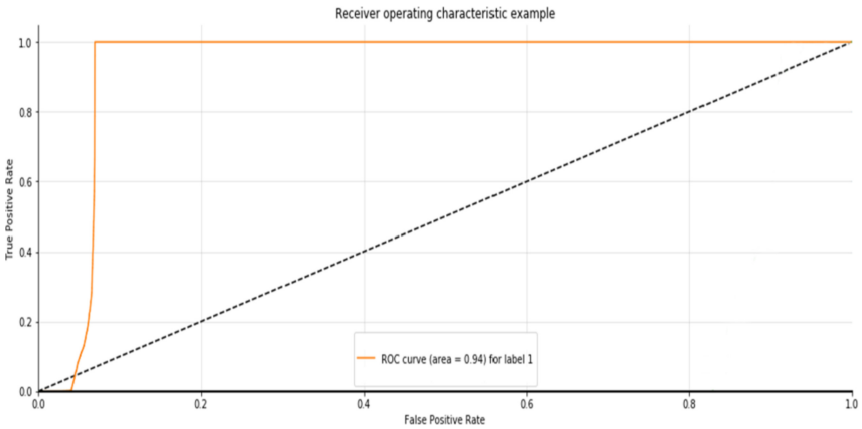

Fig. 5. ROC-AUC curve

layer is used to process sequential time series data. To prevent overfitting, dropout layer is used after every stacked LSTM layer. Categorical cross entropy and Softmax activation functions were used at the last layer. Random search in *Keras tuner* is utilized which does not search exhaustively and thus provide required set of parameters in lesser time and comparatively requires less computation power.

4.5 Recommendation

Items are recommended based on the prediction results. The items in the *Rec_list_user* are the relevant top-n items to be considered by the user. Session based recommender systems are evaluated using different metrics including Precision, Recall and MRR (Mean Reciprocal Rank). We have used Recall@20 and MRR@20 for evaluation of our proposed approach [13]. Recall@20 determines the proportion of relevant items in top-20 items and MRR@20 considers the reciprocal rank of the relevant items with rank less than 20.

Experiments have shown that the proposed technique performed better and is depicted in Table 4. Recall@20 and MRR@20 values are nearly same for LSTM-SR, FS-LSTM-SR, CNN-LSTM-SR and FS-CNN-LSTM-SR for smaller sessions. But, for larger session sizes FS-CNN-LSTM-SR outperformed all the above mentioned techniques.

Table 4. Recall@20 and MRR@20 for LSTM-SR, FS-LSTM-SR, CNN-LSTM-SR, FS-CNN-LSTM-SR

# of sessions	Evaluation metric	LSTM-SR	FS-LSTM-SR	CNN-LSTM-SR	FS-CNN-LSTM-SR
10000	Recall@20	0.1484	0.1469	0.1470	0.1471
	MRR@20	0.0961	0.0843	0.0922	0.0890
50000	Recall@20	0.2955	0.3451	0.3476	0.3492
	MRR@20	0.1288	0.1692	0.1697	0.1710
100000	Recall@20	0.3562	0.4633	0.4883	0.4971
	MRR@20	0.1720	0.2176	0.2232	0.2474

5 Conclusions

Session based Recommender systems recommend next item keeping in view a set of user interactions with the system that happened within a certain time frame. Such systems do not explicitly take user preferences rather auto-learn such things on the basis of user interactions only. In this paper, we have developed a hybrid technique FS-CNN-LSTM-SR that combines CNN-LSTM model with fuzzy-time series for session based recommendations. The input time series data has been converted into fuzzy time series by converting crisp sets of input attributes to fuzzy attributes. The fuzzy time series is then worked upon by CNN-LSTM. The technique considers predicted items, co-occurring items and popular items in creation of final set of recommended list for every user. The experimental results demonstrate that proposed hybrid technique provides better prediction results as compared to LSTM-SR, FS-LSTM-SR and CNN-LSTM-SR techniques. The proposed technique FS-CNN-LSTM-SR outperforms all the above techniques for larger chunks of input data selected from YOOCHOOSE dataset-RecSys challenge 2015.

References

1. Ben-Shimon, D., et al.: Recsys challenge 2015 and the YOOCHOOSE dataset. In: Proceedings of the 9th ACM Conference on Recommender Systems, pp. 357–358 (2015)
2. Cheng, J., et al.: Long short-term memory-networks for machine reading. arXiv Preprint arXiv:1601.06733 (2016)
3. Fischer, T., Krauss, C.: Deep learning with long short-term memory networks for financial market predictions. Eur. J. Oper. Res. **270**(2), 654–669 (2018)
4. Goodfellow, I. et al.: Deep Learning. MIT Press (2016)
5. Gregor, K., et al.: DRAW: a recurrent neural network for image generation. In: International Conference on Machine Learning, pp. 1462–1471 (2015)
6. Hidasi, B., et al.: Session-based recommendations with recurrent neural networks. arXiv Preprint arXiv:1511.06939 (2015)
7. Hochreiter, S.: The vanishing gradient problem during learning recurrent neural nets and problem solutions. Int. J. Uncertain. Fuzziness Knowl. Based Syst. **6**(02), 107–116 (1998)
8. Kouki, P., et al.: From the lab to production: a case study of session-based recommendations in the home-improvement domain. In: 14th ACM conference on Recommender Systems, pp. 140–149 (2020)
9. LeCun, Y., et al.: Deep learning. Nature **521**(7553), 436–444 (2015)
10. Lenz, D., Schulze, C., Guckert, M.: Real-time session-based recommendations using LSTM with neural embeddings. In: Kůrková, V., Manolopoulos, Y., Hammer, B., Iliadis, L., Maglogiannis, I. (eds.) ICANN 2018. LNCS, vol. 11140, pp. 337–348. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01421-6_33
11. Li, M., et al.: Predicting future locations of moving objects with deep fuzzy-LSTM networks. Transp. A Transp. Sci. **16**(1), 119–136 (2020)
12. Lu, J., et al.: Recommender system application developments: a survey. Decis. Support Syst. **74**, 12–32 (2015)
13. Ludewig, M., Jannach, D.: Evaluation of session-based recommendation algorithms. User Model. User-Adap. Inter. **28**(4–5), 331–390 (2018)
14. Nisha, C.C., Mohan, A.: A social recommender system using deep architecture and network embedding. Appl. Intell. **49**(5), 1937–1953 (2019)
15. Panigrahi, S., Behera, H.S.: A study on leading machine learning techniques for high order fuzzy time series forecasting. Eng. Appl. Artif. Intell. **87**, 103245 (2020)
16. Quadrana, M., et al.: Personalizing session-based recommendations with hierarchical recurrent neural networks. In: Proceedings of the 11th ACM Conference on Recommender Systems, pp. 130–137 (2017)
17. Sadaei, H.J., et al.: Short-term load forecasting by using a combined method of convolutional neural networks and fuzzy time series. Energy **175**, 365–377 (2019)
18. Sammut, C., Webb, G.I. (eds.): Encyclopedia of Machine Learning. Springer, Boston (2010). https://doi.org/10.1007/978-0-387-30164-8
19. Ben Schafer, J., et al.: Recommender systems in e-commerce. In: Proceedings of the 1st ACM Conference on Electronic Commerce, pp. 158–166 (1999)
20. Silva, P.C.L., et al.: An open source library for Fuzzy Time Series in Python (2018)
21. de Souza Pereira Moreira, G., et al.: News session-based recommendations using deep neural networks. In: Proceedings of the 3rd Workshop on Deep Learning for Recommender Systems, pp. 15–23 (2018)
22. Srilakshmi, M., et al.: Improved Session based Recommendation using Graph-based Item Embedding (2020)
23. Wang, C.-S., Chiang, J.-H.: FuzzAttention on session-based recommender system. In: 2019 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), pp. 1–6 (2019)

24. Zhang, J., et al.: Recurrent convolutional neural network for session-based recommendation. Neurocomputing **437**, 157–167 (2021)
25. Zhang, S., et al.: Deep learning based recommender system: a survey and new perspectives. ACM Comput. Surv. **52**(1), 1–38 (2019)
26. A Comprehensive Guide to Convolutional Neural Networks – the EL15 way. https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53. Accessed 15 Sept 2021
27. Understanding AUC-ROC curve. https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5. Accessed 10 Oct 2021