Deep Learning based approaches for Recommendation Systems

Balaji Balasubramanian[1], Pranshu Diwan[1] and Dr. Deepali Vora[1]
[1]Department of Information Technology, Vidyalankar Institute of Technology, Mumbai, India
[1]balajib26@gmail.com, pranshusdiwan@gmail.com, deepali.vora@vit.edu.in

Abstract. Recommendation systems are the most widely used Machine Learning algorithm in the industry. Recently Deep Learning algorithms have been successfully applied in fields like Computer Vision, Natural Language Processing etc. and have started being applied to Recommendation System. In this research paper different Deep Learning methods for Recommendation System have been studied.

Keywords: recommendation systems, matrix factorization, collaborative filtering, deep learning, recurrent neural network, convolutional neural network.

1 Introduction

Recommendation systems have become a part of a consumer's daily life. They help give the user personalized recommendations based on their preferences. Recommendation systems have a huge impact on how a user interacts with the content – may it be viewing or purchasing. Recommendation systems are implemented in majority of applications, from content streaming websites like Netflix or Spotify to shopping websites like Amazon. The successful implementation of recommendation systems has helped several companies to monetize the results and better understand their customers. Recently Deep Learning has witnessed a huge surge in popularity. It has mainly been used to deal with image, text and audio data. It can be seen in many applications such as Self Driving Cars [7], Language Translation [8], Speech Recognition [9] etc. The reason for this is the data explosion due to internet, better hardware for complex computation like GPU, and better algorithms for learning complex ideas. In this paper, different ways of applying Deep Learning to build Recommendation System are studied.

2 Input data types

The quality of a recommendation system depends on the type of data we use as an input. Depending on the type of data we have available, we can define two types

of systems:

2.1 Recommendation Systems having explict data

Explicit data means data which has explicit action by the user, such as a rating. Explicit data is very strong indicator of a user's preference.

2.2 Recommendation Systems having implict data

Implicit data is data consumed by the user without explicitly rating it. It may include various user actions like the number of times a song is played, or the types of movies watched, the active time of users, the click action, or how long they were on the page, and so on. While the user does not explicitly interact with the content, we can achieve confidence in user preferences based on their implicit behavior. For example, a user listening to a song 20 times has higher confidence (rating) than a song the user has listened to only thrice. The drawback is that implicit data contains a lot of noise and many times the data can be irrelevant.

3 Types of recommendation systems

3.1 Collaborative filtering based

These types of recommendation system make use of collaborative filtering methods to find user patterns and preferences and recommends items to a new user based on another user's similarity. The major advantage of collaborative filtering is that it does not take into consideration the content of the item. This is especially useful if finding similarity between items is difficult. We can perform deep representation learning using a hierarchical Bayesian model called collaborative deep learning [4].

Collaborative filtering involves filtering of information by using recommendations of other users. Based on the behavior of users, explicit user profiles are created. Collaborative filtering analyses the interdependencies between users and their behavior to generate new user-item preferences [1].

A collaborative filtering framework is broadly divided into 3 steps [2]:

Matrix factorization (data reduction): This involves factorizing a large matrix (sparse matrix) into smaller but denser matrices which represent the original matrix [3]. These smaller denser matrices contain hidden or latent features, which are used to make recommendations. The product of these denser matrices is equal to the sparse matrix.

Factorization Machine[10] is a generalization of Matrix Factorization and it combines more than two input features. It is described as:

$$y(x)=w_o + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \langle v_i, v_j \rangle x_i x_j$$

w_o is bias
w_i is the weight for i-th variable

In ⟨vi,vj⟩, two input features are described by *k* latent factors and their interaction is computed. This is pairwise for all the input features.
Factorization Machines is a widely used method because it can deal with highly sparse data and is also computationally efficient.

Collaborative Filtering (CF) method: The algorithm calculates similarities between users and items are recommended based on the user's nearest neighbors [2].
Calculating similarities: In order to calculate nearest neighbors, similarities are needed. Similarities can be calculated in several ways, like cosine similarity or Jaccard similarity.

For generating good recommendations, the user-item matrix sparsity should be lower than 99.5%. A number higher will have a prediction accuracy low. This will result in low quality recommendations [5]. However, at times there's a large amount of data and the sparsity of the user-item matrix can be very high. In such cases, a good accuracy score for recommendations needs to be achieved. Using association rules and mining techniques, similar patterns across users can be found, even for a very sparse matrix [5]. To reduce the size of data, clustering analysis techniques [5] are used.

Alternating Least Squares (ALS)
Alternating least squares is an optimization function used exclusively for implicit type of data. ALS is an iterative process where in each iteration, the goal is to come closer to a factorized representation of our original data. [1]

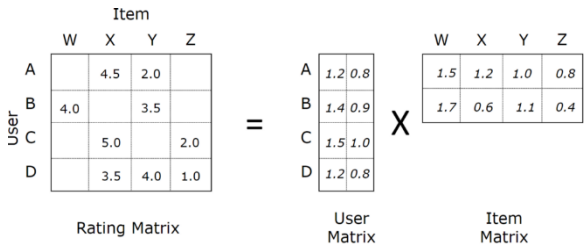

Fig 1. Matrix factorization [6]

Consider the matrix in fig.1, where M = U x V. Using ALS, the values in U and V are found. By randomly assigning values and iterating using least squares values that fit closest to M are found. ALS uses the same approach, but instead iteratively alternates between V and optimizing U and vice versa. This is done for

each iteration until a value closest to M is reached. [3]
Efficiency issue in ALS: Inverting a matrix is considered an expensive operation and for a Mx M matrix the time complexity is O(M^3). Updating one latent vector has a higher time complexity, and this is ineffective for large scale data. To speed up this process we use uniform weighting. [3]

3.2 Content based recommendation systems

In content-based recommendation system, the similarity between the items is calculated. The user is recommended items which are most similar to the items the user has interacted with.

3.3 Hybrid recommendation systems

Hybrid recommendation systems take into consideration both collaborative filtering methods to generate similar users, and content-based recommendations to generate similar items. Hybrid system ensembles the output of both systems to give a common output. Hybrid systems are generally preferred as they have the advantage of both collaborative filtering and content-based systems. Major organizations like Netflix use a hybrid recommendation system.

4 Introduction to Deep Learning

Hybrid Neural Networks are algorithms inspired by the human brain. Yann et al's [11] LeNet was one of the first successful implementations of a Neural Network. It used Convolutional Neural Network (CNN) to understand Handwritten characters. AlexNet by Alex et al. [12] was the major breakthrough for Deep Learning. It won the Imagenet Competition in 2012[13] in which it performed with a high accuracy on a dataset that contained 1.2 million images.

4.1 Convolutional Neural Network

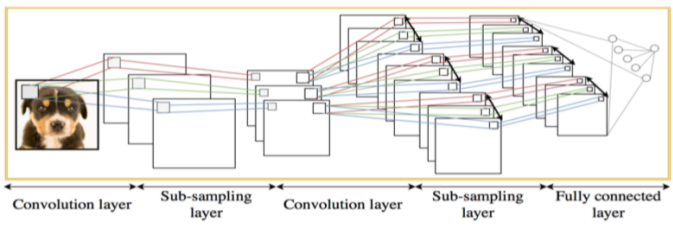

Fig 2. Basic Convolutional Neural Network (CNN) [14]

Fig 2. Shows a basic Convolutional Neural Network (CNN). It is mainly used to understand images. It has also been used to understand audio in a few applications. It has a hierarchical learning structure. The initial layers learn about

the simple features like edges and curves in the image. Those are combined in the middle layers to learn more complex features like eyes, nose etc. In the final layers the model learns very complex features like face of cat, dog etc. In the above diagram the CNN contains Convolutional layers, Sub-sampling layers and Fully connected layers. Convolutional layers understand the image to learn about it features like edges, curves, eyes, nose etc. The Sub-Sampling layers reduce the number of features of the model to simplify the learning process, remove noise and save computational resources. The fully connected layers combine all the features learnt by the previous layers to provide the output.

4.2 Recurrent Neural Network

Recurrent Neural Network was introduced by Michael I Jordan [15] in 1986. It is used to understand natural language and has been used in applications such as Sentiment Analysis [16], language translation [17] etc.

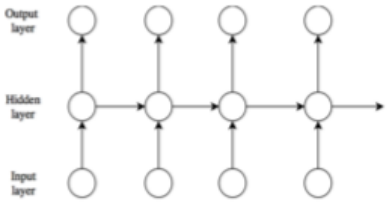

Fig 3. Basic Recurrent Neural Network [14]

Fig 3. Shows Recurrent Neural Network (RNN). It can remember sequential data. A layer at time ‘t’ receives both the input data and output of layer at time ‘t-1’. Hence the layer can keep track of data and remember a sequence. But the RNN suffers from vanishing gradient problem in which it can forget information when the sequence is too long. To deal with this LSTM [18] and GRU [19] where introduced.

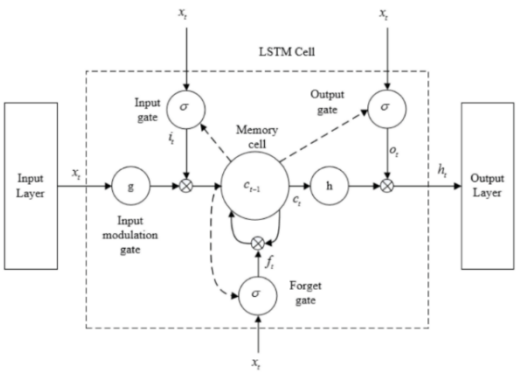

Fig 4. Structure of LSTM [20]

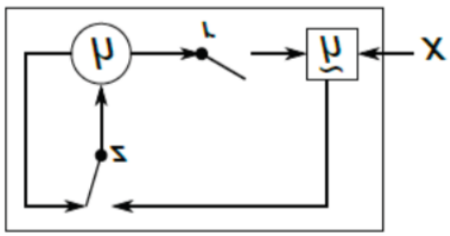

Fig 5. Structure of GRU [20]

Fig 4. shows the structure of LSTM and Fig 5. shows the structure of GRU. The main improvement over simple RNN is that these networks can control the information flow and forget information that is not required. For example, if the goal is to understand if the text review is positive or negative, it is required to focus on a few words that describe the sentiment and forget everything else.

5 Deep Learning based approaches for recommendation systems

5.1 Deep content-based music recommendation

Collaborative filtering is a popular approach for music recommendation, but it suffers from cold start problem and it cannot recommend new music because there is no user data available. Recommending music is a tough problem because music has a big variety of style and genre. Oord et al [21] uses a Convolutional Neural Network (CNN) to understand the audio signal in music. Music audio signals are very low level and hierarchical learning approach of Deep Learning can be used to understand high level features such as genre, mood and lyrics of music.

The raw audio signals are not directly passed on to CNN. An intermediate time-frequency representation is produced which is then passed onto the CNN which can learn complex features in the music. Million Song dataset [22] was used to train the CNN. The CNN is used to predict latent factors from audio signal which contains important features that describe the music. Ground truth latent factors for users and items are used to train the CNN and are produced by Weighted Matrix Factorization [23]. Authors also found that CNN outperforms bag of words approach [32].

5.2 Deep Neural Networks for YouTube recommendations

Covington et al. [25] recommends a two-stage approach for recommending YouTube videos: - Candidate Generation step and Ranking step. In Candidate

Generation step, small number of videos which are relevant to the user are selected from the huge YouTube database to speed up the recommendation process. User history and contextual data are used during this step. In Ranking step, the videos selected in the previous step are ranked.

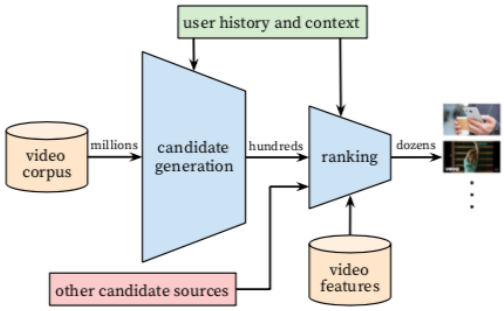

Fig 6. 2-step process for recommending YouTube videos [25]

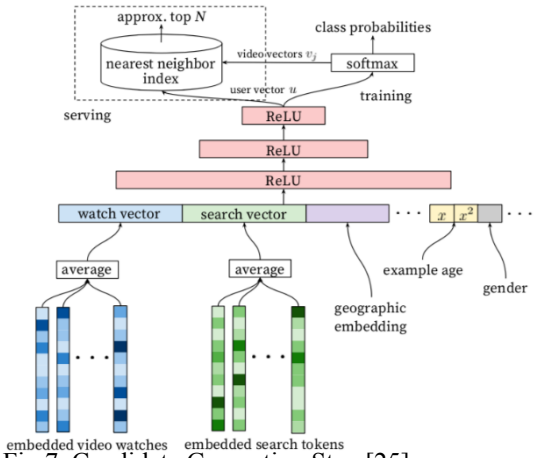

Fig 7. Candidate Generation Step [25]

Candidate Generation Step- There are millions of videos and a particular user has only seen a few of those, so the dataset is very sparse. Neural Networks cannot deal with sparse data effectively so embeddings are used to create dense representation of this data which can then be passed onto the Neural Network. The embeddings are trained on watched video data, search history, geographic location of user, age, gender etc and separate embeddings are created for each of these input features. Any number of features can be and they are then concatenated and passed onto Fully Connected layer with Rectified Linear Units (ReLU) activation function. Output layer contains Softmax activation. The output of the SoftMax layer is then passed to a nearest neighbor algorithm which generates hundreds of

videos that are relevant to the user.

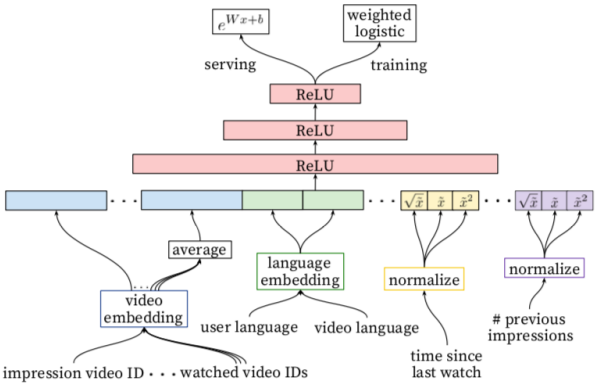

Fig 8. Ranking step [25]

Ranking Step- In this step, the few hundred videos selected in the previous step are ranked. The Neural Network architecture is similar to that of the previous step, only difference is that in the final step every videos is scored using logistic regression after which they are ranked and presented to the user. Hundreds of features such as video impressions, interaction of the user with the topic of the video and the channel that uploaded the video are used to train the model. The number of features used to train the model is far more than the previous step to get a more accurate answer and the number of videos is just a few hundred and it won't be computationally expensive to use complex data.

5.3 Session based recommendations with recurrent neural networks

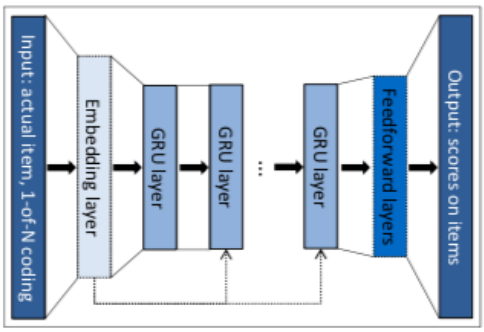

Fig 9. GRU based architecture for Session-based Recommendations [26]

Hidesi et al [26] used Deep Learning for session-based recommendation. Gated Recurrent Units (GRU) was used to keep track of long session of user

interaction data which can be used to understand the mood of the user and make recommendations. GRU was preferred to LSTM because it provides similar accuracy while consuming less computational resources. Fig m shows the Neural Network architecture that was used. Input to the GRU contains a sequence of past events in the session and the model predicts the future events. The input data is 1-of-N encoded. 1-of-N encoding contains a vector whose length is equal to the number of items and each item has a place in the vector. For each item, the place in the vector corresponding to the item is one and rest is zero. This is then passed to embedding layers which converts sparse input to a dense output and is then passed to GRU layer. There are multiple GRU layers and all the layers receive input from the previous GRU layer and also the input embedding because it is found to provide better accuracy.

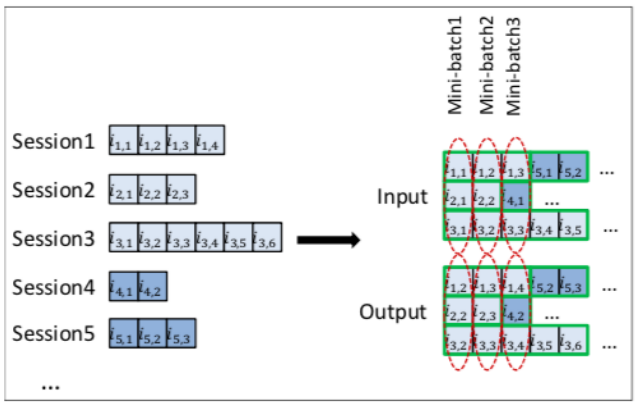

Fig 10. Session-Parallel Mini Batches [26]

The authors came up with Session-Parallel mini batch to feed the input to the Neural Network. Each session is of different length, some sessions may contain a handful of event whereas some sessions may contain hundreds of events.
So we pass first event of X sessions and then the second event and so on. Ranking loss function is used to train the Neural Network.

5.4 Wide and Deep Learning for recommendation systems

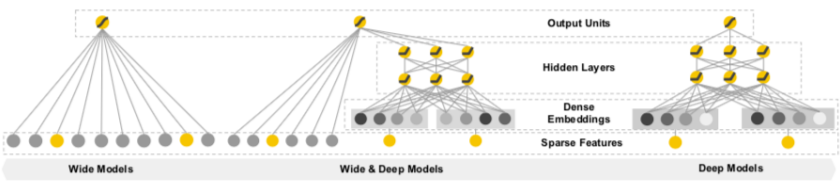

Fig 11. Wide and Deep Learning model [27]

Cheng et al [27] combined Deep Learning (deep) and linear models (wide). Linear models memorize the frequently co-occurring pairs and Neural Network can learn complex features and have good generalization power and it is jointly trained. The linear model uses a simple function $y= w^Tx + b$ where $x[x_1, x_2,..]$ is a vector of features and w is the model parameters and b is the bias. The Deep Learning model has a Dense Embedding followed by dense layer of neurons with ReLU activation units. The wide and deep part are jointly trained by backpropagating the gradients.

5.5 Wide Content2Vec: Specializing joint vector representations of product images and text for product recommendations

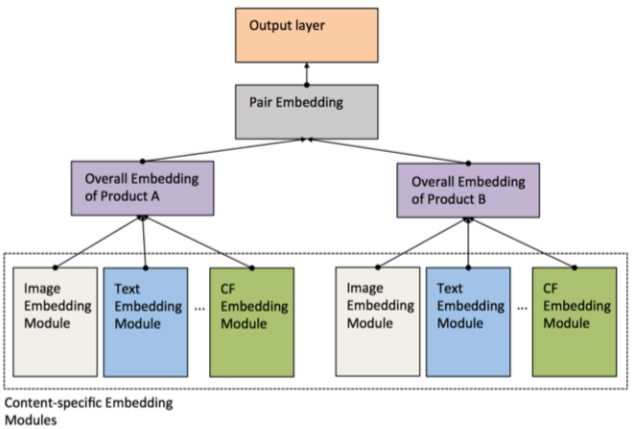

Fig 12. Combining multiple content-specific Embedding modules [28]

Content2Vec by Nedelec et al. [28] combines the features learnt from multiple data sources such as image, text, user interaction etc. The model is flexible, and we can easily plugin other data sources. Figure a. shows the model. In the first stage there are multiple content-specific data modules that are trained on a specific data source such as image, text, etc. The embedding for the image uses AlexNet [12]. Pretrained AlexNet is used which was trained on the ImageNet dataset [13] and the features are transferred to another task. Word2Vec was used for text embedding module. The Word2Vec was trained on the entire product catalog and it contains a good representation of the important features of the product and Text CNN [29] was used on top of it for the task for text classification. In addition to this, embeddings are used for product purchase sequence, product metadata etc.

In the second stage there is an overall embedding that combines several content-specific modules into a unified embedding that describes the product using all its data sources. Authors introduce Pairwise Residual Unit which is

inspired by the original Residual unit [24]. It can preserve the features learnt by the Content-specific Embedding modules and make incremental improvements on the contents learnt by the previous stage.

In the third stage there is a pair embedding that computes similarity scores between the different products, which is then used to make product recommendations.

5.6 Wide DeepFM: A Factorization-Machine ased Neural Network for CTR Prediction

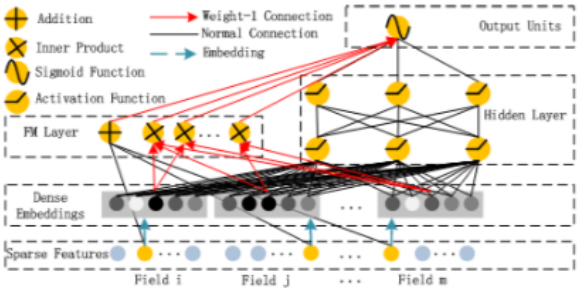

Fig 13. Wide and Deep Architecture for DeepFM [30]

DeepFM [30] is an improvement over Wide and Deep Model [27]. It uses the same input and embedding for wide and deep portion of the model and it doesn't require any feature engineering. The wide part is a Factorization Machine [10] that models the order-2 feature interaction. Deep portion of the model contains a dense neural network. ReLU activation function is found to work better than tanh and sigmoid. Dropout [31] is used in which a small number of neurons are randomly set to zero. Each neuron has a probability (for example 0.4) of being set to zero. This forces the Neural Network to represent a complex function with a smaller number of neurons and helps deal with overfitting. The final output layer combines the output from the wide and deep part. The wide and deep part is jointly trained and can learn complementary features.

6 Conclusion

Deep Learning methods for Recommendation System have been studied in this paper. It complements the traditional Recommendation System algorithms by learning from complex data sources like image, text and audio. There is a need for better models that can combine all these data sources and build a multimodal algorithm.

7 Acknowledgement

The authors would like to thank all our anonymous critics for their feedback on this paper, along with the faculty of Vidyalankar Institute of Technology for their unconditional support.

References

1. Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In IEEE International Conference on Data Mining (ICDM 2008), pages 263–272, 2008.
2. Stijn Geuens , Kristof Coussement , Koen W. De Bock , A framework for configuring collaborative filtering-based recommendations derived from purchase data, European Journal of Operational Research (2017), doi: 10.1016/j.ejor.2017.07.005
3. X. He, H. Zhang, M.-Y. Kan, and T.-S. Chua. Fast matrix factorization for online recommendation with implicit feedback. In SIGIR, pages 549–558, 2016
4. H. Wang, N. Wang, and D.-Y. Yeung. Collaborative deep learning for recommender systems. In KDD, pages 1235–1244, 2015.
5. M. K. Najafabadi, M. N. Mahrin, S. Chuprat, and H. M. Sarkan, "Improving the accuracy of collaborative filtering recommendations using clustering and association rules mining on implicit data," Computers in Human Behavior, vol. 67, no. Supplement C, pp. 113 – 128, 2017.
6. Jae Duk Seo. "Matrix Factorization Techniques for Recommender Systems". November 22, 2018. Towards Data Science, https://towardsdatascience.com/paper-summary-matrix-factorization-techniques-for-recommender-systems-82d1a7ace74
7. L. Fridman, D. E. Brown, M. Glazer, W. Angell, S. Dodd, B. Jenik, J. Terwilliger, J. Kindelsberger, L. Ding, S. Seaman et al., "Mit autonomous vehicle technology study: Large-scale deep learning based analysis of driver behavior and interaction with automation," arXiv preprint arXiv:1711.06976, 2017
8. Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. CoRR abs/1409.0473 (2014). Retrieved from http://arxiv.org/abs/1409.0473.
9. D. Amodei, R. Anubhai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, J. Chen, M. Chrzanowski, A. Coates, G. Diamos, et al., Deep speech 2: End-to-end speech recognition in english and mandarin, 2016, pp. 173–182.
10. S. Rendle. Factorization machines. In ICDM, pages 995–1000, 2010
11. LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11):2278–2324, November 1998a.
12. Alex Krizhevsky, Sutskever I, and Hinton G.E, Imagenet classification with deep convolutional neural networks. In NIPS, 2012.
13. A. Berg, J. Deng, and L. Fei-Fei. Large scale visual recognition challenge 2010. www.image- net.org/challenges. 2010.
14. Pouyanfar, S., Sadiq, S., Yan, Y., Tian, H., Tao, Y., Reyes, M. P., & Iyengar, S. S. (2018). A Survey on Deep Learning: Algorithms, Techniques, and Applications. ACM Computing Surveys (CSUR), 51(5), 92
15. Michael I. Jordan. 1986. Serial order: A parallel distributed processing approach. Advances in Psychology 121 (1986), 471–495.
16. Richard Socher, Alex Perelygin, Jean Y. Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christo- pher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In Confer- ence on Empirical Methods in Natural Language Processing. Citeseer, Association for Computational Linguistics,

1631–1642.
17. Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. CoRR abs/1409.0473 (2014). Retrieved from http://arxiv.org/abs/1409.0473.
18. S. Hochreiter and J. Schmidhuber. Long short-term memory. Neural computation, 9(8):1735–1780, 1997
19. Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In The Conference on Empirical Methods in Natural Language Processing. 1724–1734 .
20. R. Fu, Z. Zhang, and L. Li, "Using lstm and gru neural network methods for traffic flow prediction," in 2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC), Nov. 2016, pp. 324–328.
21. A. V. D. Oord, S. Dieleman, and B. Schrauwen. Deep content-based music recommendation. In NIPS , pp. 2643−2651, 2013.
22. Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. The million song dataset. In Proceedings of the 11th International Conference on Music Information Retrieval (ISMIR), 2011.
23. Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In Proceedings of the 2008 Eighth IEEE International Conference on Data Mining, 2008.
24. K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," arXiv:1512.03385, 2015.
25. Covington, P.; Adams, J. & Sargin, E. Deep Neural Networks for YouTube Recommendations. Proceedings of the 10th ACM Conference on Recommender Systems, ACM, 2016, 191-198
26. Francois B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk. Session-based recommendations with recurrent neural networks. arXiv preprint arXiv:1511.06939, 2015.
27. H.-T. Cheng, L. Koc, J. Harmsen, et al. Wide & deep learning for recommender systems. In 1st Workshop on Deep Learning for RecSys, pages 7–10, 2016
28. T. Nedelec, E. Smirnova, and F. Vasile, "Specializing joint representations for the task of product recommendation," arXiv preprint arXiv:1706.07625, 2017.
29. Assari Yoon Kim. Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882, 2014.
30. I. Huifeng Guo, Ruiming Tang, et al. 2017. DeepFM: A Factorization-Machine based Neural Network for CTR Prediction. In Proceedings of the 26th International Joint Conference on Articial Intelligence. 1725–1731.
31. G.E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R.R. Salakhutdinov. Improving neural networks by preventing co- adaptation of feature detectors. arXiv preprint arXiv:1207.0580, 2012.
32. Y. Zhang, R. Jin, and Z.-H. Zhou. Understanding bag-of-words model: a statistical framework. International Journal of Machine Learning and Cybernetics, 1(1-4):43–52, 2010.