Embedding Based Recommender Systems, a Review and Comparison

Ahmed Hussein Ragab[*1], Passant El-Kafrawy[*2]

[*]*Computer Science Department, School of Information Technology and Computer Science, Nile University*
Giza, Egypt

[1]ah.hussein@nu.edu.eg

[2]pelkafrawy@nu.edu.eg

Abstract: ***This paper provides a summary and review of embedding based recommender systems.***
Word embedding frameworks like word2vec were originally developed for NLP tasks. However, they were quickly adopted in recommender systems to construct hybrid recommenders that incorporate side information in addition to user-item interaction to overcome common problems in recommender systems like cold start and popularity bias.
However, there are several proposed recommender systems that utilize embedding layers and each of them has its own strengths and weaknesses. A review and comparison between these different approaches is presented in this work. First, normal word embedding for NLP is introduced then different recommenders that utilize this method are presented and compared. Different evaluation metrics and standard datasets used for embedding based recommender systems are discussed afterwards and finally a unified comparison of all these datasets and evaluation metrics is presented in order to facilitate comparison between different embedding-based recommenders. Future work is then presented and discussed.

Key words: ***Word Embeddings, Recommender Systems, Deep Learning, Neural Networks***

1 INTRODUCTION

A. Traditional Recommender Systems Approaches.

Recommender systems saw a great rise in their importance with the emergence of E-Commerce platforms (like Amazon) and online content platforms (like Netflix) which have a very large number of items in their inventory and are trying to customize the experience of each user on their platform by displaying items that might be of interest to each specific user individually. Traditional recommenders can be divided mainly into **Collaborative Filtering (CF)** [1], [2], [3] methods and **content-based** methods [4]. Collaborative filtering methods rely on user – item interactions to predict what items a user will be interested in based on what similar users already like. They depend on a matrix named user-item matrix that stores the interactions between each user and all the items in store. The factorization of this matrix can be done using embeddings as detailed in [5] and [6]. This method is very powerful in uncovering hidden patterns and non-obvious interactions between items. However, they fail in recommending new items to existing users or recommending existing items to new users which is known as the cold-start problem. On the other hand, content-based methods used user and item attributes to recommend items to users which makes them able to mitigate the cold start problem as they rely on item attributes which can be added upon adding new items to the items databases or user attributes which can be collected from the user upon sign up (which explains why several music and content platforms ask their new users to select a number of items they like during sign up). **Hybrid recommenders** [7] try to capture associations between users and in the same time use side information from users and items to solve the cold start method. Several of the embedding based methods discussed in the next section are qualified as hybrid approaches as they use the user and item embeddings in addition to the user-item interaction matrix to overcome the cold-start issues.

B. Word Embeddings

The embedding concept in machine learning was first introduced in [8] and [9] which dramatically improved many NLP tasks. The idea is to use neural networks to represent each word in a vocabulary using a fixed size vector and the words that are semantically together will be closer to each other in that space because they are surrounded by similar words. The word embedding model is pre trained on a certain corpus and then this pre trained model is used to give each word a vector representation. For example, the vectors of the words ‘Man’ and ‘King’ are close to each other so are the vectors for the words ‘Woman’ and ‘Queen. There are two methods to implement word embedding. One is Continuous bag of words (CBOW) where a certain number of the preceding and following words is used to train the model or Skip gram where the word is used to predict the surrounding words. In both models, a single hidden layer is used to train the embedding layer weights to either predict the word vector based on N surrounding words or use the current word to predict N surrounding vectors. The window size and the embedding vector size are tuned hyperparameters depending on the amount of text available to train. Note that in both cases, we are not interested in the input or output layers but rather the weights of the hidden layers which are then used to return the vector representation of any word in the corpus for further tasks.

2 EMBEDDING LAYERS IN RECOMMENDERS.

A. Vanilla Implementations of Word Embeddings in NLP.

Although embedding layers were developed for NLP tasks, the idea was soon adopted in several other domains where input vectors are sparse. Most notably, in recommender systems where the user-Item matrix Is usually a very sparse matrix with many 0s are found. Some vanilla implementations like item2vec [10] and prod2vec in [11] used the same embedding layers but instead of words used products and represented each user by a product vector and used embedding layers to find users that are close to each other in this product space or similarly find embeddings of products that are close to each other in the user space. Also, Neural Personalized Embedding (NPE) in [5] tries to extend traditional matrix factorization recommenders by adding item embeddings to resolve cold-start problems. A practical implementation of this method in pharmaceutical retail recommenders is presented in [12].

B. Using Embeddings and Deep Learning for Hybrid Recommenders.

Although the previously mentioned methods are similar to latent vector factorization in the essence that they replace the latent vector with the output vector from the embedding layer. However, this opened the way to use user and item metadata in addition to user-item interactions to create hybrid recommender systems that rely on user profiles, item metadata as well as the interaction between the users with almost little to no modification to the same method. This solves the cold start problem where we have no information about new users or new items introduced to the system. This approach is presented in frameworks like meta-prod2vec and lightFM which will be discussed later. Meta-prod2vec [13] combines meta data of the input product and the products visited right before or after the item in question but does not address user metadata.
Cofactor [14] uses the matrix factorization interpretation of word2Vec by Levy and Goldber in [15] to factorize the item co-occurrence matrix in addition to the user-item matrix.
RME [16] builds on Cofactor and also adds the factorization of two other matrices the co-liked co-occurrence and the co-disliked co-occurrence matrices as well. One of the earliest applications of user embeddings in production systems is presented in [17] on YouTube video recommendations. LightFM [18] however, uses both user-item matrix as well as two other matrices which are user feature matrix and item feature matrix. The final embedding is given by the dot-product of the user and item representations adjusted by adding the user and item feature biases. This provides a robust solution to cold start problems for either new products or new items, in which case the user-item interaction for this new product or user is empty so we rely on user or item meta-data to recommend products similar in their meta-data to this new product (in the case of a new product) or recommend items most popular with users similar in their meta-data to a new user. It remains one of the most widely used recommenders in industry until now.

C. Using User Embeddings as Inputs to Neural Networks with Different Architectures.

RecDNNing[19] concatenates user and item embeddings into a fully connected layer to achieve similar output.
Wide and deep [20] combines a “Wide” component which is a generalized linear model with nonlinear feature transformations (like cross product transformation) with a “deep” component which consists of the embeddings of user and item meta-data to achieve better results as shown in Figure 1.

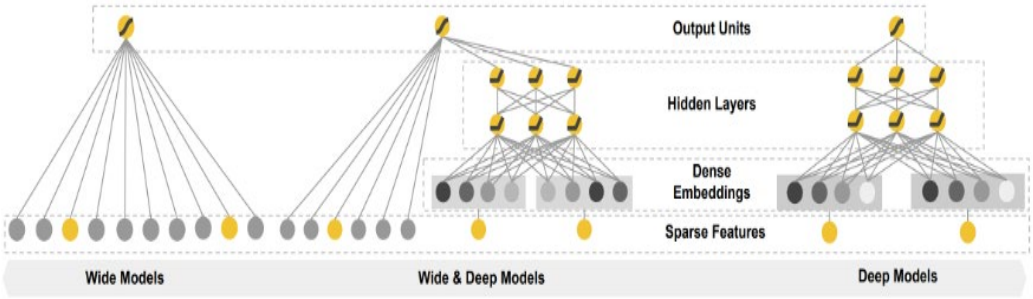

Figure 1 Wide and deep architecture [20]

Another novel architecture to wide and deep is presented in deep and cross [21] which proposes a novel cross network architecture to improve the wide and deep network performance. The network details are presented in Figure 2.

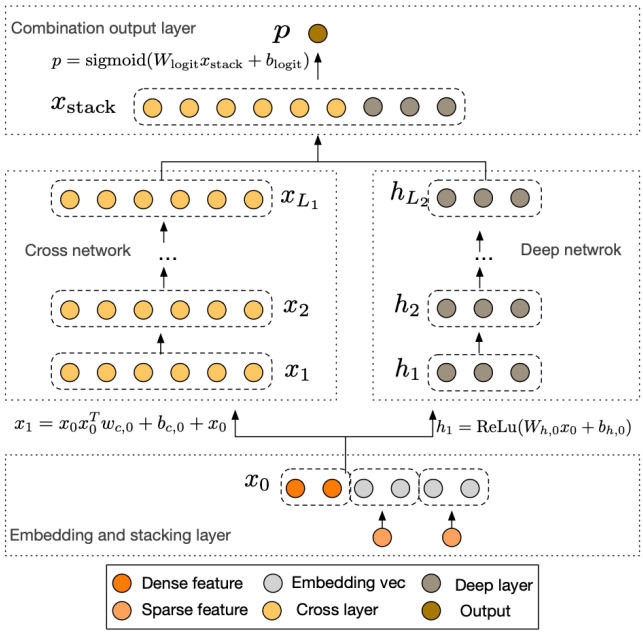

Figure 2 Deep and Cross architecture [21]

Similarly, a compressed interaction network (CIN) is presented in xDeepFM [22] presented in Figure 3.

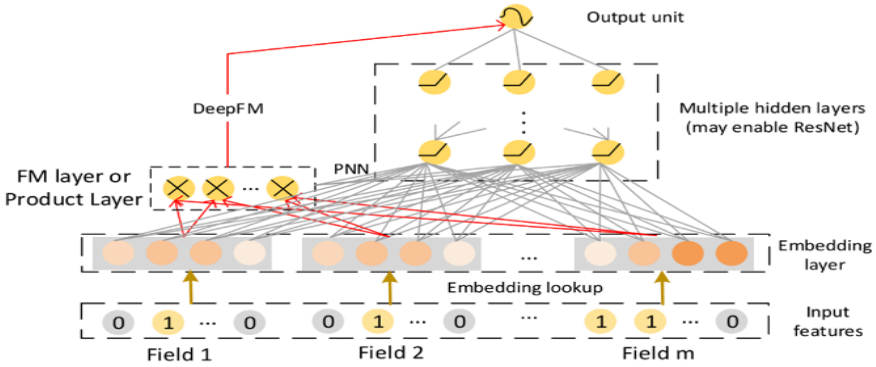

Figure 3 CIN architecture [22]

3 DATASETS

This section discusses some of the most used open-source recommender system datasets used in the mentioned recommender systems frameworks as well as other recommender systems in literature as well. Some work uses proprietary datasets from the company / institute where the work is done that are not published or shared with the community. This makes it more difficult to reproduce and benchmark these methods against other methods using the same datasets.

A. Movielens [23]

This dataset is the most widely used dataset in recommender system literature.
It was collected from the website (movielens.umn.edu) by the GroupLens research project at the university of Minnesota between September 19th, 1997 and April 22, 1998.
It contains 100,000 ratings between 1-5 from 943 users to 1682, where each user at least rates 20 movies and basic user information like age, gender and occupation is described. Later, a larger version of the dataset was released with 25 million records recorded between 1995 and 2019 [23], [24].

B. Million Songs [25]

Similar to the Movielens dataset, this dataset contains user ratings for almost 1 million songs. It was released by the website last.fm in 2011. It has different variations with more values like the 10M songs, 25M songs and 30M songs datasets.

C. Book Crossings [26]

This dataset Contains 278,858 users (anonymized but with demographic information) providing 1,149,780 ratings (explicit / implicit) about 271,379 books.

D. Dianping

Dianping is a famous user review website in China. Authors in [22] used the last 3 visited places for each user to predict his next most likely place to visit. Other datasets provided from Amazon, Good Reads and steam gaming platform are summarized in the repository in [27].

E. Arxiv [28]

Authors in [14] used the Arxiv website's log data between January and December of 2012 as a user-paper (or user-item) clicks dataset.

F. Cross Validated [29]

Authors in [18] used the cross validated dataset from Stack-Exchange website network data dump to recommend questions to users so they can answer them.

G. Criteo [30]

Authors in [22] used the Criteo Dataset which is a famous advertising click through prediction dataset to predict whether a user will click an advertisement given the page he is currently in.

H. Online Retail [31]

This dataset contains E-commerce transactions for an online retailer during the period between Dec 1,2010 to Dec 9, 2011.

I. Taste Profile [32]

This dataset contains users' song playing activity on the Echo Nest online song platform it is listed under the Million songs dataset [25] Umbrella.

J. Bing News [33]

This dataset is part of Bing News MIND dataset for news recommendation.

4 EVALUATION METRICS

This section explains the most common evaluation metrics used to asses recommender systems.

A. Normalized Discounted Cumulative Gain (NDCG) [*34]*

Normalized Discounted Cumulative Gain (NDCG) is one of the main metrics used to asses recommender systems. It is a measure that was first used in information retrieval that assesses the ranking quality of returned results. It is calculated as:

$$NDCG = \frac{DCG}{Ideal\ DCG}$$

where

$$DCG = \frac{1}{N}\sum_{u=1}^{N}\sum_{j=1}^{J}\frac{g_{u,i_j}}{\log_b(j+1)} \quad (1)$$

where $g_{u,i}$ is the gain from recommending item i to user u and *Ideal DCG* is the DCG when every item is recommended to the user in the correct order. NDCG @ N (or sometimes NDCG @ K) is the NDCG calculated by recommending the Top N items only. Usually N is equal to 5 or 10 as these are the items usually recommended to users.

B. Precision and Recall [34]

In the context of recommender systems, precision and recall are used when the target of the assessment of the recommender is to predict whether a user will like a certain item being recommended to him or not. They are calculated as normal precision and recall (referred to as hit rate in some contexts) would be calculated however they are usually calculated on the top N items like NDCG as well.

C. RMSE, MAE and MAPE [34]

Other recommenders try to predict the user's rating for certain items based on his rating of previous items. These websites usually ask the customer to give a rating to each item and usually in the form of 5-star rating system. In these cases, normal regression metrics are used to assess the recommender's ability to correctly predict user's taste RMSE (Root Mean Square Error), MAE (Mean Absolute Error) and MAPE (Mean Absolute Percentage Error) are used in such cases and the recommended items are ranked in a descending order according to their predicted user rating.

5 RESULTS

This section summarizes the results reported on the discussed recommenders as well as the datasets used in each one. Results are either reported by the authors of the work where the recommender was first introduced or authors of later work in comparison to the original work for benchmarking. All results are detailed in Table 1, where every recommender is listed along with the datasets it was used on and the metrics that were used to assess the recommender either in the original work where the recommender is first published, or in later work where other authors benchmark their work against SOTA recommenders at the time of publication. It is evident that most assessments either use NDCG@K or Recall@K (which are explained in the previous sections) and the value of K is either 10 or 20. This is related to the fact that usually the industrial applications of recommender systems involve placing these top recommended items in a special section of the homepage of the website or application that is recommending these items.

TABLE I

DETAILED RESULTS OF DIFFERENT EMBEDDING-BASED RECOMMENDER-SYSTEMS

Recommender	Dataset	Metric	result	reported in	Notes
item2vec	Microsoft Xbox Music service	accuracy	0.82	[10]	private dataset, reported on the top 10K popular artist , other top N results are detailed in [10]
NPE	ML-10M	Recall @20	0.1497	[5]	
		NDCG @20	0.1449	[5]	
	Online Retail	Recall@20	0.2296	[5]	
		NDCG@20	0.1742	[5]	
	Taste Profile	Recall @20	0.1788	[5]	
		NDCG @20	0.1594	[5]	

TABLE I:

DETAILED RESULTS OF DIFFERENT EMBEDDING-BASED RECOMMENDER-SYSTEMS-(CONTINUED)

prod2vec	e-mail receipts sent to users who voluntarily opted-in	CTR vs control group of normal ads	9.81% uplift	[11]	prediction accuracy was compared but no numerical results only that prod2vec and its variants outperformed baseline models
	e-mail receipts sent to users who voluntarily opted-in	YR vs control group of normal ads	7.63% uplift	[11]	Private dataset not available online
	30 Music Dataset	Hit Rate at 10	0.017	[13]	
	30 Music Dataset	Hit Rate at 20	0.0101	[13]	
	30 Music Dataset	NDCG at 10	0.105	[13]	
	30 Music Dataset	NDCG at 20	0.113	[13]	
meta-prod2vec	30 Music Dataset	Hit Rate at 10	0.0292	[13]	ensemble of meta-prod2vec and Co-Count based item pair similarities
	30 Music Dataset	Hit Rate at 20	0.018	[14]	
	30 Music Dataset	NDCG at 10	0.144	[14]	
	30 Music Dataset	NDCG at 20	0.161	[14]	
Cofactor	ArXiv	Recall at 20	0.067	[16]	
	ArXiv	recall at 50	0.11	[16]	
	ArXiv	NDCG at 100	0.079	[16]	
	ArXiv	MAP at 100	0.021	[16]	
	Movie Lense 20M	Recall at 20	0.145	[16]	
	Movie Lense 20M	recall at 50	0.177	[16]	
	Movie Lense 20M	NDCG at 100	0.172	[16]	
	Movie Lense 20M	MAP at 100	0.055	[16]	

TABLE I:

DETAILED RESULTS OF DIFFERENT EMBEDDING-BASED RECOMMENDER-SYSTEMS-(CONTINUED)

	Taste Profile	Recall at 20	0.208	[16]	
	Taste Profile	recall at 50	0.3	[16]	
	Taste Profile	NDCG at 100	0.268	[16]	
	Taste Profile	MAP at 100	0.111	[16]	
YouTube	YouTube Watch data (no reference of data availability online for reproduction)	holdout MAP	0.13	[17]	example age (when did the user watch this video) is used during training , among other metadata features as well
LightFM	Cross Validated	ROC AUC	0.695	[19]	warm start
	Cross Validated	ROC AUC	0.696	[19]	cold start
	Movie Lens	ROC AUC	0.763	[19]	warm start
	Movie Lens	ROC AUC	0.716	[19]	cold start
RecDNNing	MovieLens-100K	RMSE	0.62	[19]	
wide and deep	Google play app download data	offline AUC	0.728	[20]	Private dataset not available online for reproduction
	Google play app download data	online acquisition gain	3.90%	[20]	
	Criteo	AUC	0.8	[20]	
	Criteo	Log Loss	0.449	[20]	Depth=3
	Dianping	AUC	0.8361	[20]	
	Dianping	log Loss	0.3364	[20]	Depth=2
	Bing News	AUC	0.8377	[20]	

TABLE I:

DETAILED RESULTS OF DIFFERENT EMBEDDING-BASED RECOMMENDER-SYSTEMS-(CONTINUED)

	Bing News	log Loss	0.2668	[22]	Depth=2
xDeepFM	Criteo	AUC	0.8052	[22]	Depth=3
	Criteo	log Loss	0.4418	[22]	Depth=2
	Dianping	AUC	0.8639	[22]	Depth=3
	Dianping	log Loss	0.3156	[22]	Depth=3
	Bing News	AUC	0.84	[22]	Depth=3
	Bing News	log Loss	0.2649	[22]	Depth=2

6 CONCLUSION

Although embeddings were originally developed for NLP tasks, it was proven to be instrumental in solving other problems and tasks. Most notably in recommender systems as discussed in this work. This makes NLP breakthroughs like embeddings, transformers, GPT, etc.; instrumental not only for the advancement of NLP itself but also for other domains and Machine learning tasks as well prompting NLP to be the spearhead of innovation in the machine learning and Data science domain in general by drawing parallels between NLP tasks and other tasks in different domains like the case discussed in this work. However, unlike NLP where work is done mainly on publicly available datasets for benchmarking and comparison, we see a lot of work in recommender systems being done on proprietary datasets that cannot be shared for reproduction, comparison and benchmarking. This is mainly found in research coming from industrial companies as they face difficulty sharing datasets that contain user behavior even if anonymized due to legal or competitive reasons.
This stresses the need for more publicly available datasets especially ones that contain both user and item side information for the use in hybrid recommenders as they are quickly becoming the state of the art in recommender systems. Finally, this work summarizes different Recommender systems that utilize embedding layers as the basis of their architecture and summarizes their results against one another on different datasets and across the different metrics reported either by the recommender's authors or by other authors comparing their work to it for benchmarking.

7 Future Work

With the move to transformer-based networks in NLP, recent literature in NLP is trying to use transformers for recommendation like bert4rec [35] a similar survey is needed to compare and survey transformer-based recommender systems as well. Also, each of these frameworks uses a different dataset and a separate set of evaluation metrics to assess their performance, a unified benchmarking framework would help directly comparing these frameworks and identify the areas of relative strength and weaknesses. The industrial implementation presented in [36] by Microsoft is a very representative example of such efforts where several recommenders from the aforementioned ones are implemented using Python programming language. However, these algorithms are not yet compared using the same metrics and datasets using these implementations.

REFERENCES

[1] G. Linden, B. Smith and J. York, "Amazon.com recommendations: Item to-item collaborative filtering," *IEEE Internet Computing Journal*, no. 1, pp. 76- 80, 2003.

[2] A. Karatzoglou, X. Amatriain, L. Baltrunas, and N. Oliver, "Multiverse Recommendation: N-Dimensional Tensor Factorization for Context-Aware Collaborative Filtering," *in Proc. of the Fourth ACM Conference on Recommender Systems*, New York, NY, USA, 2010, pp. 79–86.

[3] X. Ning and G. Karypis, "SLIM: sparse linear methods for top-n recommender systems," *in Proc. of 11th IEEE International Conference on Data Mining, ICDM*, Vancouver, Canada, December 11-14, 2011.

[4] P. Brusilovsky, A. Kobsa, and W. Nejdl, *The Adaptive Web Methods and Strategies of Web Personalization*,1st ed, Springer Berlin Heidelberg, 2007.

[5] T. Nguyen and A. Takasu, "NPE: Neural Personalized Embedding for Collaborative Filtering," *in Proc. of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, Stockholm, Sweden, Jul. 2018, pp. 1583–1589.

[6] T. Huang, D. Zhang, and L. Bi, "Neural embedding collaborative filtering for recommender systems," *Neural Comput & Applic Journal*, vol. 32, no. 22, pp. 17043–17057, Nov. 2020.

[7] Çano, Erion, and Maurizio Morisio, "Hybrid Recommender Systems: A Systematic Literature Review." *Intelligent Data Analysis Journal vol* 21, no. 6 (January 1, 2017): 1487–1524.

[8] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," *in Proc. of 1st International Conference on Learning Representations, ICLR 2013*, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings, 2013.

[9] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and Their Compositionality," *in Proc. of the 26th International Conference on Neural Information Processing Systems* - Volume 2, Red Hook, NY, USA, 2013, pp. 3111–3119.

[10] O. Barkan and N. Koenigstein, "ITEM2VEC: Neural item embedding for collaborative filtering," in *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*, Vietri sul Mare, Salerno, Italy, Sep. 2016, pp. 1–6.

[11] M. Grbovic et al., "E-commerce in Your Inbox: Product Recommendations at Scale," *in Proc. of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Sydney NSW Australia, Aug. 2015, pp. 1809–1818.

[12] L. Piciu, A. Damian, N. Tapus, A. Simion-Constantinescu, and B. Dumitrescu, "Deep recommender engine based on efficient product embeddings neural pipeline," *in Proc. of the 17th RoEduNet Conference: Networking in Education and Research (RoEduNet)*, Cluj-Napoca, Sep. 2018, pp. 1–6.

[13] F. Vasile, E. Smirnova, and A. Conneau, "Meta-Prod2Vec: Product Embeddings Using Side-Information for Recommendation," *in Proc. of the 10th ACM Conference on Recommender Systems*, Boston Massachusetts USA, Sep. 2016, pp. 225–232.

[14] D. Liang, J. Altosaar, L. Charlin, and D. M. Blei, "Factorization Meets the Item Embedding: Regularizing Matrix Factorization with Item Co-occurrence," *in Proc. of the 10th ACM Conference on Recommender Systems*, Boston Massachusetts USA, Sep. 2016, pp. 59–66.

[15] O. Levy and Y. Goldberg, "Neural Word Embedding as Implicit Matrix Factorization," *in Proc. of the 27th International Conference on Neural Information Processing Systems - Volume 2,* Cambridge, MA, USA, 2014, pp. 2177–2185.

[16] T. Tran, K. Lee, Y. Liao, and D. Lee, "Regularizing Matrix Factorization with User and Item Embeddings for Recommendation," *in Proc. of the 27th ACM International Conference on Information and Knowledge Management*, Torino Italy, Oct. 2018, pp. 687–696.

[17] P. Covington, J. Adams, and E. Sargin, "Deep Neural Networks for YouTube Recommendations," *in Proc. of the 10th ACM Conference on Recommender Systems*, Boston Massachusetts USA, Sep. 2016, pp. 191–198.

[18] M. Kula, "Metadata Embeddings for User and Item Cold-start Recommendations," *in Proc. of the 2nd Workshop on New Trends on Content-Based Recommender Systems co-located with 9th ACM Conference on Recommender Systems (RecSys 2015)*, Vienna, Austria, September 16-20, 2015, 2015, vol. 1448, pp. 14–21.

[19] H. Zarzour, Z. A. Al-Sharif, and Y. Jararweh, "RecDNNing: a recommender system using deep neural network with user and item embeddings," *in Proc. of 10th International Conference on Information and Communication Systems (ICICS)*, Irbid, Jordan, Jun. 2019, pp. 99–103.

[20] H.-T. Cheng et al., "Wide & Deep Learning for Recommender Systems," *in Proc. of the 1st Workshop on Deep Learning for Recommender Systems*, Boston MA USA, Sep. 2016, pp. 7–10.

[21] R. Wang, B. Fu, G. Fu, and M. Wang, "Deep & Cross Network for Ad Click Predictions," in *Proc. of the ADKDD'17*, Halifax NS Canada, Aug. 2017, pp. 1–7.

[22] J. Lian, X. Zhou, F. Zhang, Z. Chen, X. Xie, and G. Sun, "xDeepFM: Combining Explicit and Implicit Feature Interactions for Recommender Systems," *in Proc. of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London United Kingdom*, July 2018, pp. 1754–1763.

[23] F. M. Harper and J. A. Konstan, "The MovieLens Datasets: History and Context," *ACM Trans. Interact. Intell. Syst. Journal*, vol. 5, no. 4, pp. 1–19, Jan. 2016.

[24] Movie Lens 25M Website : https://grouplens.org/datasets/movielens/25m/ (accessed 1 Oct. 2021).
[25] T. Bertin-Mahieux, D. Ellis, B. Whitman, and P. Lamere, "The Million Song Dataset.," *in Proc. of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*, Jan. 2011, pp. 591–596.
[26] C.-N. Ziegler, S. M. McNee, J. A. Konstan, and G. Lausen, "Improving Recommendation Lists through Topic Diversification," *in Proc. of the 14th International Conference on World Wide Web*, New York, NY, USA, 2005, pp. 22–32.
[27] J. McAuley, Recommender Systems and Personalization Datasets. Available From https://cseweb.ucsd.edu/~jmcauley/datasets.html , (accessed 1 Oct. 2021).
[28] C. B. Clement, M. Bierbaum, K. P. O'Keeffe, and A. A. Alemi, "On the Use of ArXiv as a Dataset," ArXiv, vol. abs/1905.00075, 2019.
[29] Stack Exchange Data Dump Website: https://archive.org/details/stackexchange , (accessed 1 Sep 2021).
[30] Kaggle Display Advertising Challenge Dataset Website:https://labs.criteo.com/2014/02/download-kaggle-display-advertising-challenge-dataset/ , (accessed 1 Oct. 2021).
[31] D. Chen, S. L. Sain, and K. Guo, "Data mining for the online retail industry: A case study of RFM model-based customer segmentation using data mining," *J Database Mark Cust Strategy Manag*, vol. 19, no. 3, pp. 197–208, Sep. 2012.
[32] The Echo Nest Taste Profile Subset Website: http://millionsongdataset.com/tasteprofile/ , (accessed Oct. 2021).
[33] F. Wu et al., "MIND: A Large-scale Dataset for News Recommendation," *in Proc. of the 58th Annual Meeting of the Association for Computational Linguistics*, Online, July 2020, pp. 3597–3606.
[34] A. Gunawardana and G. Shani, "Evaluating Recommender Systems," *in Recommender Systems Handbook*, F. Ricci, L. Rokach, and B. Shapira, Eds. Boston, MA: Springer US, 2015, pp. 265–308.
[35] F. Sun et al., "BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer," *in Proc. of the 28th ACM International Conference on Information and Knowledge Management,* Beijing China, Nov. 2019, pp. 1441–1450.
[36] A. Argyriou, M. González-Fierro, and L. Zhang, "Microsoft Recommenders: Best Practices for Production-Ready Recommendation Systems," *in Proc. of the Web Conference 2020*, Taipei Taiwan, Apr. 2020, pp. 50–51.

BIOGRAPHY

Ahmed Hussein Ragab Received his BSc in 2014 from Cairo University Faculty of Engineering with a major in Electrical and Communication engineering (with honors). He then joined IBM Egypt in 2015, working as an Infrastructure engineer and obtained a diploma in Big Data and Data science from Nile University in 2017 and then started pursuing his MSc. degree in informatics from Nile University with a focus on Deep learning and Recommender systems. In 2018, He joined Vodafone Egypt as a member of the Big Data analytics Team and he is currently working as a Data Scientist since March 2020 at MNT-Halan ; a Fintech super-App that provides access to different financial services for under-served communities in Egypt including mobile wallets , E-commerce and Buy Now Pay Later (BNPL). Ahmed's research interests include deep learning, Natural language processing and Recommender systems.

Passant Elkafrawy, Professor, joined Information Technology and Computer Science School, Nile University in 2019. Dr. Passent M ElKafrawy is a Professor since 2018, she got her Ph.D. from the University of Connecticut in the United States in 2006 in Computer Science and Engineering; in the field of Computational Geometry, Artificial Intelligence. Then she taught at Eastern State University of Connecticut for one year. In 2007, she worked as an Assistant Professor in Faculty of Science, Menoufia University, Mathematics, and Computer Science department. In 2011, she joined the American University in Cairo as an Adjunct Professor in the Computer Science and Engineering Department. Then was appointed as an Associate Professor in 2013.
She has over 45 publications and editor in 3 books. Supervising several research studies between Ph.D. and MSc in the field of Natural Language Processing, Semantic Knowledge, Bioinformatics, Big Data Analytics, and Knowledge mining and acquisition. She has joined IBRO school for neurodegenerative physician training organized by ENND and Personalized medicine workshop organized by Zewail Univ. Her Research interests are in Software Engineering, Bioinformatics, Big data analytics, Machine Learning, and Cloud computing.

أنظمة الترشيح باستخدام طبقات التضمين

أحمد حسين رجب[1] ، بسنت الكفراوي[2]
قسم علوم الحاسب، كلية الحاسبات ونظم المعلومات، جامعة النيل
[1]ah.hussein@nu.edu.eg
[2]pelkafrawy@nu.edu.eg

ملخص:

(Embedding Layers) تقدم هذه الورقة البحثية تلخيصاً وإستعراضاً لانظمة الترشيح التي تعتمد على طبقات التضمين للكلمات

إن أطر عمل تضمين الكلمات مثل (WORD2VEC) طورت خصيصاً من أجل مهام التعلم الآلي للغات.

إلا أنه سرعان ما تم تطويعها واستخدمها لبناء أنظمة ترشيح هجينة تعتمد على معلومات إضافية إلى جانب معلومات تفاعل المستخدم مع العناصر أو المنتجات التي يقوم النظام بترشيحها لتجنب مشاكل معروفة مثل البداية الباردة للعناصر الجديدة أو الانحياز للعناصر المشهورة.

هناك العديد من الانظمة المقترحة التي تستخدم طبقات التضمين في ترشيح الاشياء إلى أنه لكل منها نقاط قوة وضعف تستعرضها هذه الورقة بالتفصيل.

أولاً، يتم تعريف أنظمة تضمين الكلمات المستخدمة في التعلم الآلي للغات. بعد ذلك يتم تعريف ومقارنة أنظمة الترشيح المختلفة التي تستخدم تضمين الكلمات.

بعد ذلك يتم تعريف المقاييس المختلفة المستخدمة لتقييم أنظمة الترشيح ومجموعات البيانات القياسية المستخدمة في تقييم هذه الانظمة.

وفي النهاية يتم وضع مقارنة موحدة لكل هذه الانظمة وتقييماتها لتسهيل مقارنة وتقييم هذه الانظمة.

الكلمات المفتاحية:
أنظمة الترشيح، تضمين الكلمات، الشبكات العصبية، التعلم العميق