A Neighbourhood-Based Approach To Collaborative Deep Learning For Job Recommendations

Shreya Venkatesh
National Institute of Technology, Karnataka
shreya.dkv@gmail.com

Balaji Balasubramanian
National Institute of Technology, Karnataka
balab23@gmail.com

Sanjana Sangameshwar
National Institute of Technology, Karnataka
ss270495@gmail.com

G. Ram Mohana Reddy*
National Institute of Technology, Karnataka
profgrmreddy@nitk.ac.in

ABSTRACT

Job recommendation has paved way to become one of the most sought after domains to deploy recommender systems. Business social networking sites have an increasing demand for accurate job recommendations that grows by the day. Deep learning for recommendation is a recent trend that comes with promising results. In this paper we present a neighborhood-based approach for job recommendations using Collaborative Deep Learning (CDL). CDL has been used in the past to achieve high quality of recommendation by solving the sparse matrix problem. We combine CDL with user clustering to achieve scalability and significant accuracy under sparse settings. Furthermore it is observed that the latent semantic information extracted from the chosen neighbourhoods leads to effective representation and gives high recall.

Categories and Subject Descriptors

H.1 [**Models And Principles**]: General; J.4 [**Computer Applications**]: Social and Behavioral Sciences

Keywords

Recommender Systems, Job Reccomendations, Deep Learning, Collaborative Deep Learning

1. INTRODUCTION

Recommender systems were developed with the intention of making information filtering an intelligent process. Over the years it has found it's use in a variety of applications to assist users in making more informed choices. The reason behind the popularity of recommender systems is due to it's ability to create a personal experience for the user. The reccomendation algorithm focuses on individual interaction between users and items to find patterns and apply those patterns in order to produce good quality recommendations.

The advent of recommender systems has created a greater scope of finding jobs online. Job recommendations are being provided as an added feature to most business social networking sites. E-recruitment is a term that has gained popularity over the past few years. There are millions of opportunities available to an individual and would be time consuming if the task of singling out jobs is a manual process. Thus the main challenge here is to select and suggest jobs as close as possible to user requirements. For this purpose most business social networking sites use implicit feedback [15] to build their recommendations.

Existing methods for recommender systems can be classified into [14] : Content-Based (CB) methods [15], Collaborative Filtering (CF) methods [16] and hybrid methods [13].

CB methods require user profile information while CF methods require past activities of users. CB can only provide recommendations based on the past experiences of the user, giving no priority to the relationship between items and thus can produce shallow recommendations. CF takes item similarities into account and produce far better recommendations than CB but has certain shortcomings of it's own. CF cannot be used to rate items that have never been seen and have thus not been rated by any user (cold start). CF and CB require a large number of ratings to make close to accurate recommendations. However when the number of items is large, very few ratings per user create a sparse situation. Both CF and CB methods do not produce accurate predictions in such sparse situations.

Hybrid approaches combine and use both CB and CF based methods to give better results than both individually. Hybrid systems can be classified into loosely coupled and tightly coupled systems based on whether two way interaction exists between auxiliary information of items and it's corresponding rating information. Tightly coupled systems which have a two-way interaction provide better performance than loosely coupled systems which only have a one-way interac-

*Professor and HOD, Department of Information Technology

tion on account of the fact that in tightly coupled systems, ratings information can guide in feature learning and the extracted features can further increase the predictive capabilities of the recommender model. However hybrid techniques are not effective enough while learning latent user representations. Deep learning has been applied in order to solve this issue.

Collaborative Deep Learning (CDL) [1] integrates deep learning techniques with CF to learn latent information effectively. CDL has been very successful in solving the sparse matrix problem with high measures of recall being achieved. But deep learning techniques usually requires computation that grows with users and items. Thus there is need to address the issue of scalability. Clustering has been used along with CF in the past to increase throughput [11]. However it has been observed that clustering produced less personalised recommendations with CF.

To overcome the above mentioned problem, in this paper we propose a neighbourhood-based approach. Instead of generalised Bayesian Stack Denoising Autoencoder, we use a generalised Bayesian Stacked Autoencoder as a slight modification to the CDL model. We have also used the TF-IDF model instead of the bag of words model used in [1]. We apply the proposed method to recommend jobs to users with implicit feedback. We also apply k-means clustering with reduced dimensionality representation to produce better quality of recommendations on an average.

The three key contributions of this paper are as follows :

- Partitioning of the data into neighbourhoods by k-means clustering to increase scalability and efficiency. Items or Jobs for each neighbourhood are then filtered based on the clusters that have been formed.
- Stacked Autoencoder used for CDL instead of SDAE. For feature selection, the TF-IDF measure has been used instead of the bag of words measure.
- We have applied the modified model to a job recommendation problem. We have used the algorithm to recommend jobs to users with implicit feedback.

The rest of this paper is organized as follows: Section 2 describes our proposed model and all the work-flows in detail. In Section 3, we discuss the experiments carried out, and results. Finally we sum up our contributions in the paper in Section 4.

2. PROPOSED WORK

For the recommendation task we use implicit feedback. Given a user set $U = \{u_1, u_2, ..., u_k\}$ and Jobs $J = \{j_1, j_2, ..., j_m\}$ and feedback in the form of interactions, our aim is to recommend a subset of J to any chosen U (test user). The work-flow of our approach to job recommendation is represented by a flowchart in Figure 1.

2.1 Feature Selection And Extraction

Categorical data in the user and job dataset are converted into a vector space representation using the TF-IDF model.

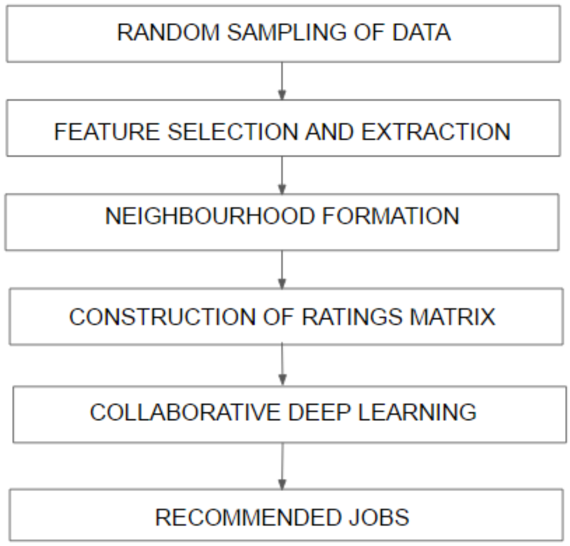

Figure 1: Proposed Model Flow

Features at this stage are converted into a lower dimension (dimensionality reduction) using Principle Component Analysis (PCA).

2.2 Neighbourhood Formation

Partially extracted features and the other selected features are used for user clustering using a simple algorithm like K-means. To find the optimal number of clusters we use silhouette score analysis [17] . Any number of clusters that achieves a Silhouette score greater than average is optimal. After the users have been clustered into say clusters C_1, C_2, ..., C_n , for each cluster filter jobs in such a way that the ratings matrix contains only jobs that the train users of that particular clusters have interacted with.

2.3 Construction of Ratings Matrix

For a test user, the cluster he/she belongs to is chosen as the neighbourhood. The ratings matrix R then comprised of users (including test user) of the neighbourhood by filtered jobs. R is then constructed by giving high scores to positive interactions (liking, bookmarking, clicking) and low scores to negative interactions (deleting a recommended job). Based on job user interactions we constructed the ratings matrix using a 4 point scale using the following rules :

- If a user clicks on a job, he/she rates it with score of 2.
- Liking a job is equivalent to him/her rating it with a score of 3.
- Bookmarking a job gives it a score of 4.
- Deleting a job shown to him/her gives it a score of 1.

Finally test user ratings are all set to 0 to illustrate severe sparsity.

Table 1: Recall percentage for L=4

No. of Users	Cluster1	Cluster2	Cluster3	Neighbourhood-Based CDL (Average)	CDL
100	38.36	24.30	-	31.33	**43.27**
300	54.72	69.66	-	**62.19**	45.28
500	51.63	48.56	59.02	**53.07**	52.26
700	52.91	46.52	51.13	**50.26**	48.64
900	53.55	52.34	53.92	**53.27**	53.19
1100	549.63	50.53	49.38	**49.84**	47.73

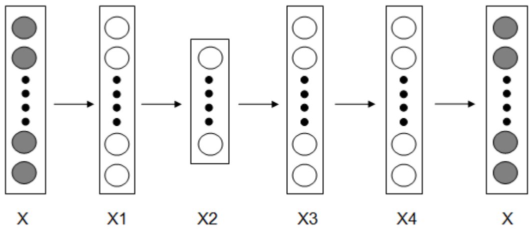

Figure 2: Stacked Autoencoder with L=4

2.4 CDL and Recommendation

Generalised Bayesian Stacked Deep Autoencoder with two layers is used for collaborative deep learning. This is equivalent to an SDAE with noise factor of 0. Figure 2 shows a two-layered stacked encoder with L=4 . The input to CDL is a J-by-S matrix X, where J stands for jobs and S represents the vocabulary set or the features, and an I-by-J Ratings matrix R with I users. Similar to [1] we use Maximum a Posteriori Estimates to maximise the joint log-likelihoods of the latent vectors U and V, weights W_l, bias b_l, ratings R and X given $\lambda_u, \lambda_v, \lambda_w, \lambda_s$, and λ_n as hyper-parameters. This algorithm is used to predict job ratings for the test users. CDL is performed separately for each of the clusters.

If u_i is the latent user vector for user i and v_j is the latent item vector for user j, coordinate ascent is used leading to the following update rules :

$$u_i \leftarrow (VC_iV^T + \lambda_u I_K)^{-1} VC_iR_i$$

$$v_j \leftarrow (UC_iU^T + \lambda_v I_K)^{-1} (UC_jR_j + \lambda_v f_e(X_{0,j}, W^+)^T)$$

where $U = (u_i)_{i=1}^I, V = (v_j)_{j=1}^J, C_i = diag(C_{i1}, ..., C_{iJ})$ is a diagonal matrix, $R_i = (R_{i1}, ..., R_{iJ})^T$ is a column vector containing all the ratings of user i, and C_{ij} reflects the confidence controlled by a and b.

Given U and V, back propagation is used to learn weights and biases W_l and b_l for each layer. They are updated using gradient ascent, the gradients of likelihood being as follows:

$$\nabla\ w_l\ L = -\lambda_w W_l$$
$$-\lambda_v \textstyle\sum_j \nabla w_l\ f_e(X_{0,j*}, W^+)^T (f_e(X_{0,j*}, W^+) - v_j)$$
$$-\lambda_n \textstyle\sum_j \nabla w_l\ f_r(X_{0,j*}, W^+)(f_r(X_{0,j*}, W^+)^T - X_{c,j*})$$

$$\nabla\ b_l\ L = -\lambda_w b_l$$
$$-\lambda_v \textstyle\sum_j \nabla b_l\ f_e(X_{0,j*}, W^+)^T (f_e(X_{0,j*}, W^+) - v_j)$$
$$-\lambda_n \textstyle\sum_j \nabla b_l\ f_r(X_{0,j*}, W^+)(f_r(X_{0,j*}, W^+)^T - X_{c,j*}).$$

Encoder function $f_e(., W^+)$ takes the corrupted content vector $X_{0,j}$ of item j as input and computes the encoding of the item, and the function $f_r(., W^+)$ also takes $X_{0,j}$ as input, computes the encoding and then the reconstructed content vector of item j. If the number of layers L=4, $f_e(X_{0,j}, W^+)$ is the output of the second layer while $f_r(X_{0,j}, W^+)$ is the output of the fourth layer.

We approximate the predicted rating as :

$$R_{ij} \approx (\mathrm{u}_i)^T \mathrm{v}_j$$

After we obtain predicted ratings for test users, we pick the jobs with above average ratings for each test user as final recommended jobs.

3. EXPERIMENTS AND RESULTS

We have used Octave as the platform for CDL and all experiments have been carried out on an Intel Core i5 - 3230M CPU with 4GB RAM. For CDL, stacked autoencoder was fixed with L=4 (hidden layers) and with 0.01 as the value for hyperparameters $\lambda_u, \lambda_w, \lambda_n, \lambda_v, \lambda_s$. Confidence parameters a and b were defined as 0.01 if the rating for job j by user i is 0 (unknown), or 1 otherwise. 300 epochs runs was required for satisfactory recommendations.

We obtained our data by sampling from the RecSys Challenge 2016's XING datasets. We have conducted extensive experiments on multiple samples with varied number of users. Each sample has been split into 70% training and 30% test. Silhouette analysis yielded 2 and 3 as optimal 'k' values for less than 500 users and more than 500 users respectively.

To observe the performance of our algorithm we have used the recall measure. Results recorded are the average recalls of all users for three runs. Table 1 contains figures that represent the recall measures that were achieved after neighbourhood-based CDL. Results have also been compared with the previous model. For recall we have considered scores 2,3,4 as positive and 1 as negative. Recall per test users has been calculated using the following formula:

$$\text{Recall} = \frac{\text{Total number of predicted positive ratings}}{\text{Total number of actual positive ratings}}$$

Figure 3 is a graph that compares throughput for different users observed the two methods. Neighbourhood-based CDL outperforms CDL by a wide margin. For comparison, the throughput of d CDL has been calculated using the following formula:

$$\text{Throughput}_{CDL} = \frac{\text{Size(Ratings matrix+Job Matrix)}}{\text{Time taken}}$$

The denominator for throughput of neighbourhood-based CDL is then the sum of the individual run times for each cluster.

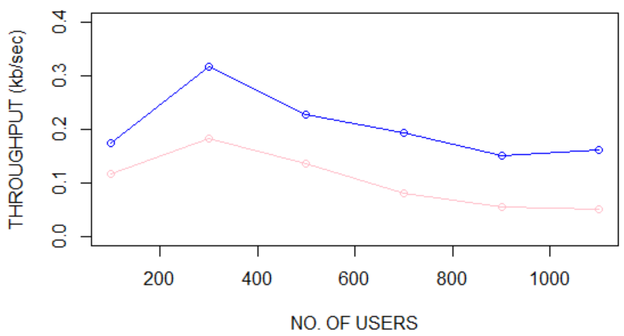

Figure 3: Throughput Vs Users Graph

Figure 4 is a graph which depicts the recall trends with varying users. Recall depicted is scaled to a range of [0-1]. From the graph it is seen that the performance remains consistent irrespective of change in size of dataset.

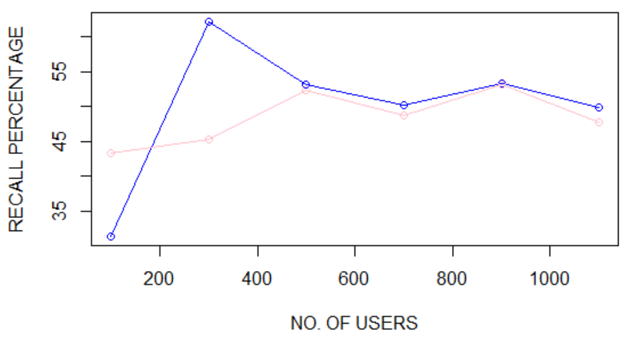

Figure 4: Recall Vs Users Graph

Figure 5 shows a visualisation of the ratings matrix before and after CDL for a sample of 500 users. It is clear that test user ratings have been set to 0 to represent severe sparsity.

4. CONCLUSION AND FUTURE WORK

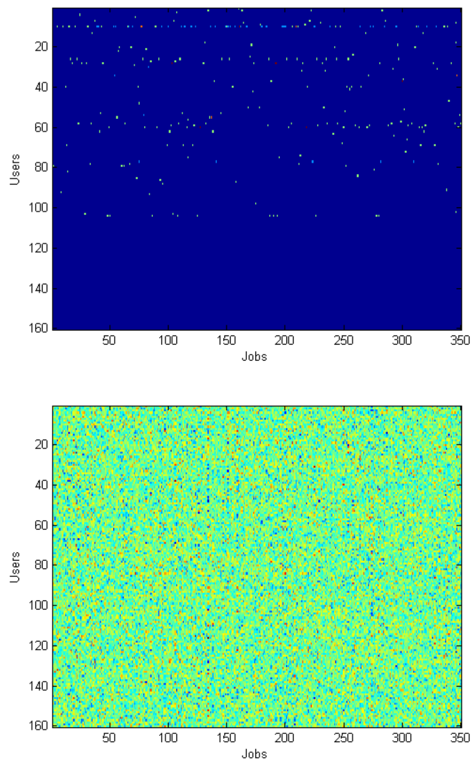

Figure 5: Ratings matrix for 500 users before and after CDL

In this paper we have demonstrated that our algorithm, apart from being scalable, performs well under sparse settings. It is observed that clustering with reduced dimensional representation produces more accurate positive recommendations than just CDL. Furthermore the use of more powerful weighting methods during feature extraction like TF-IDF has helped in effective representation. Future work is required to understand exactly why low dimensional representation works for this particular situation. Several modifications such as testing the algorithm using a better architecture, i.e. L=6, applying other powerful deep learning models like convolution networks can be made and tested. Powerful algorithms for clustering such as hierarchical clustering and density clustering can be applied in order to improve results. Scalability can be improved further using parallel programming using CUDA or distributed computing using Hadoop to parallelly run CDL on the obtained neighbourhoods.

5. REFERENCES

[1] Wang, Hao, Naiyan Wang, and Dit-Yan Yeung. "Collaborative deep learning for recommender systems." Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 2015.

[2] Salakhutdinov, Ruslan, AndriyMnih, and Geoffrey Hinton. "Restricted Boltzmann machines for collaborative filtering." Proceedings of the 24th international conference on Machine learning. ACM, 2007.

[3] Lu, Yao, Sandy El Helou, and Denis Gillet. "A

recommender system for job seeking and recruiting website." Proceedings of the 22nd international conference on World Wide Web companion. International World Wide Web Conferences Steering Committee, 2013.

[4] Al-Otaibi, Shaha T., and MouradYkhlef. "A survey of job recommender systems." International Journal of the Physical Sciences 7.29 (2012): 5127-5142.

[5] Al-OtaibiShaha, T., and YkhlefMourad. "Job Recommendation Systems for Enhancing E-recruitment Process." (2013): 433-440

[6] Diaby, Mamadou, Emmanuel Viennet, and Tristan Launay. "Toward the next generation of recruitment tools: an online social network-based job recommender system." Advances in Social Networks Analysis and Mining (ASONAM), 2013 IEEE/ACM International Conference on. IEEE, 2013.

[7] Hong, Wenxing, Siting Zheng, and Huan Wang. "Dynamic user profile-based job recommender system." Computer Science and Education (ICCSE), 2013 8th International Conference on. IEEE, 2013.

[8] Bell, Robert M., and Yehuda Koren. "Scalable collaborative filtering with jointly derived neighborhood interpolation weights." Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on. IEEE, 2007.

[9] Al-Otaibi, Shaha T., and MouradYkhlef. "An Artificial Immune System for job recommendation." Bio-inspired Intelligence (IWOBI), 2014 International Work Conference on. IEEE, 2014.

[10] Malinowski, Jochen, et al. "Matching people and jobs: A bilateral recommendation approach." System Sciences, 2006. HICSS'06. Proceedings of the 39th Annual Hawaii International Conference on. Vol. 6. IEEE, 2006.

[11] Sarwar, Badrul M., et al. "Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering." Proceedings of the fifth international conference on computer and information technology. Vol. 1. 2002.

[12] Gupta, Arpan, and Deepak Garg. "Applying data mining techniques in job recommender system for considering candidate job preferences." Advances in Computing, Communications and Informatics (ICACCI, 2014 International Conference on. IEEE, 2014._Agarwal, Deepak, and Bee-Chung Chen. "Regression-based latent factor models." Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2009.

[13] Bobadilla, JesÃžs, et al. "Recommender systems survey." Knowledge-Based Systems 46 (2013): 109-132.

[14] Kelly, Diane, and Jaime Teevan. "Implicit feedback for inferring user preference: a bibliography." ACM SIGIR Forum. Vol. 37. No. 2. ACM, 2003.

[15] Purushotham, Sanjay, Yan Liu, and C-C. Jay Kuo. "Collaborative topic regression with social matrix factorization for recommendation systems." arXiv preprint arXiv:1206.4684 (2012).

[16] Rousseeuw, Peter J. "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis." Journal of computational and applied mathematics 20 (1987): 53-65.

[17] Sarwar, Badrul, et al. "Analysis of recommendation algorithms for e-commerce." Proceedings of the 2nd ACM conference on Electronic commerce. ACM, 2000.