DOI: http://dx.doi.org/10.26483/ijarcs.v9i1.5321

ISSN No. 0976-5697

Volume 9, No. 1, January-February 2018

International Journal of Advanced Research in Computer Science

TECHNICAL NOTE

Available Online at www.ijarcs.info

RECOMMENDATION ENGINE FOR COMPETITIVE CODING QUESTIONS USING RESTRICTED BOLTZMANN MACHINES, A HYBRID APPROACH

Mohammed Arshad Siddiqui
Department of Computer Science and Engineering
PDPM Indian Institute of Information Technology Design and Manufacturing
Jabalpur, India

Harsh Agarwal
Department of Electronics and Communication Engineering
PDPM Indian Institute of Information Technology Design and Manufacturing
Jabalpur, India

Abstract:Recommendation engines have made a massive impact on every major online platform ranging from social networking to e-commerce. Recommender engines are software applications that help users by giving personalized suggestions on the services or products that are offered. They are responsible for finding relations between the provided products or services based on their inherent complementary nature of items and according to the crowd popularity. One such domain where these recommendation systems are yet to make their mark is the area of competitive coding. Competitive coding has become a major sport and selection criteria for many organizations for their candidate selection. The users engage with these websites and portals to gain valuable problem-solving skills and improve their programming abilities. Here we have presented a recommendation system for such organizations. Our approach uses vectors of weights using vector space model and TF-IDF weighting scheme for the questions. These weights are used in an unsupervised collaborative filtering process achieved using undirected graphical models, called Restricted Boltzmann Machines (RBMs) and then using the generated probabilities to predict the best questions for the users. We present efficient learning and inference procedures and demonstrate that RBM's can be successfully applied to a large dataset containing tags of questions solved by the users.

Keywords:Neural Networks, Restricted Boltzmann Machines (RBM), Recommendation System, Collaborative Filtering, Competitive Coding, Unsupervised Learning

I. INTRODUCTION

Recommender systems are software applications with the goal to generate meaningful recommendations or suggestions to a collection of users and items (products or services). For this purpose, the recommender engine uses the available data to predict generalized (same for all users) or personalized (unique for every user) recommendations depending on the goal of this engine. It is also referred as an information filtering system as it predicts the preferences of the userson any platform.

These recommendations are computed based upon certain characteristics and features of the item (content and tags) or the user (user's profile, preferences, and history) and sometimes considering both. By including both, the preference of an item and user-user similarities, based on their history to achieve a hybrid combination of content-based and collaborative recommendation gives better predictions compared to using them individually.

There has been a significant rise in recommendation engines in every field. E-commerce and social networking websites have seen the most usage where recommendations are given for items most likely to be bought by a user to increase sales and to suggest relevant articles, and other users respectively on these two types of platforms.

Competitive coding websites are platforms offering practice coding questions to their users to improve their skills and compete with other users on a global stage with millions of fellow competitors. Even with such large user-base with many being the elite in the field of information technology, these platforms have not seen advancement in recommender systems. The users required to manually find and sort questions suitable for their skills and interests, wasting valuable time.

In this paper, we have used a neural network algorithm, Restricted Boltzmann Machine to perform collaborative filtering. As [6] has proved that these can easily scale for millions of users and can include a continuous stream of data which is vital in our scenario. The RBM model is used to find similar question-solving patterns between different users and along with this, we also introduce a factor of the content-based filter in our predictions to include user's likenesstowards certain topics, giving a hybrid approach to our framework.

The rest of the paper is organized as follows. Next section presents the related work done in different fields. Section III describes the different recommendation system algorithms we used and combined. Section IV presents Restricted Boltzmann Machines. Subsequently explaining the methodology and approach used in our framework. Concluding with the results of the experiment performed on our proposed framework and Conclusion of the paper.

II. RELATED WORK

Previously, researchers [1, 2, 5, 11 and 13] have used the collaborative filtering approach for their recommendation engines. [5] used three approaches using MinHash clustering, Probabilistic Latent Semantic Indexing (PLSI), and co-visitation counts for their news predictions in Google News in a dynamic setting similar to our scenario. Tapestry [1] is an experimental mail system developed at the Xerox Palo Alto Research Center, one of the first collaborative filtering engine, but it depends on a close-knit community where every user

knows each other. [13] is a video recommendation system, [11] is a recommendation system of research papers and [2] is an item-to-item filtering on Amazon.com's products, all based on collaborative filtering procedures.

[6] Compares different RBM models: conventional RBM, RBM with Gaussian hidden units and conditional RBM. Their model demonstrated that these undirected graphical models are suitable for modeling tabular or count data. They presented learning and inference procedures for this class of models, proving that RBM's can be successfully applied to a large dataset containing over 100 million users and ratings. [7] argues that RBM can and should be used in various classification problems and evaluated their performance in various scenarios.

III. RECOMMENDATION ENGINES

According to [8] a recommendation engine or system is an information filtering system as it predicts the preferences of the users on any platform.Recommender systems are used to produce a rated list of recommendations using broadly two approaches [9] – through collaborative filtering or through content-based filtering.

A. Content-based Filtering

Content-based filtering methods are based on the description or content of item and profile of the user which consists of preferences. In such system, keywords or tags are used to describe an item and a user's profile gives idea about the type of item the user likes. In other words, the previous rating of items by the user is used to generate new recommendations for that user.TF-IDF [4], Bag-of-Words model (CBOW) and the Skip-Gram model [12] are examples of such algorithm used to rate abstract features of the items.

B. Collaborative Filtering

Collaborative filtering arevarious techniques and algorithms used to filter information of users and find similarity patterns amongst them.

Collaboration based filtering aims to achieve similarity between users to recommend items based on them. According to [3] the three prerequisites for this approach are:

1. presence of abundant users to make it more likely that any given user matches with preferences of others,
2. a metric or basis for users to demonstrate their interests, and
3. An algorithm which can find the correlation between these metrics to make recommendations.

C. Hybrid Recommender Systems

Hybrid systems,as the name suggests are the hybrid between content-based and collaborative filtering processes combined together to achieve better recommendations. This approach makes separate recommendations (both content-based and collaborative) for the users and then techniques are applied to combine the results. The hybrid systems are seen to outperform the other systems in most cases.

IV. RESTRICTED BOLTZMANN MACHINES

A Restricted Boltzmann Machine (RBM) [10] is an artificial neural network which is used as a generative stochastic model for diverse data including classification of labeled or unlabeled images,the bag of words that represent documents and user ratings of movies [6].

RBMs are called shallow neural networks.These are two-layered neural netsto construct adeep-belief network (DBN). The first layer of this RBM model is known as the visible, or inputlayer, and the second is called hidden layer.

The nodes are connected to each other across layers, but there are no connections between nodes within a group (intra-layer communication is restricted), this is the restriction in restricted Boltzmann machines.

V. METHODOLOGY

We have adopted unsupervised hybrid filtering approach for the design and implementation of competitive coding recommendation system. The recommendation engine is based on the past question attempts of an active user (Collaborative filtering approach) and the vector of weights of the tags each question has, weighed separately for each user (content-based filtering approach). Then, these two approaches are combined to give suitable recommendations. Thus, this approach depends on both, the users'history of solved questions and tags as well as the similar trends of questions and tags solved by other users on the same platform.

Vector space model (as an information retrieval model) and TF-IDF weighing schemeareused to represent question tags as vectors of weight.

The undirected graphical model, RBM is trained with the vectored representation of the questions' tags solved by each user and given as input to the visible layer to determine the most relevant questions to the users.

VI. OUR APPROACH

A. Dataset for the system

Competitive coding questions and tags along with the anonymous user data are acquired from open sources on the internet using various APIs. The corpus used to train model contains 5,500 unique questions having a total of 1,500 unique tags which are solved by 700 users.

The corpus is represented by two data sets, one containing questions and associated tags and the other containing users and questions solved by them.

Table I. Questions – Tags dataset

Index	***qid***	***tid***
0	1	3, 4, 5, 7
1	2	9, 10, 12, 13, 14
2	3	11, 14, 15, 17, 24
3	4	13, 14, 16, 17, 18
4	5	25, 26, 27, 29, 30

Where, qid is the question ID, tid contains associated tag IDs

Table II. Users – Questions dataset

Index	***user***	***qSolved***
0	1	9, 10, 31, 33, 34…
1	2	18, 23 16, 17, 31…
2	3	9, 10, 11, 12, 14…
3	4	17, 25, 27, 28, 29…
4	5	7, 13, 14, 15, 16…

Where, qSolved contains questions attempted by a user

B. Data Transformation

The user – questions dataset is transformed to construct a user – tags dataset. This new data frame contains a matrix of size *M*×*N*, ***M***is the total number of users and ***N***is the total number of tags. This data frame, thus contains, the information about the count of attempts a user has made on each tag.

Now, TF-IDF algorithm is applied to this vector space. The algorithm is applied for each user to get the weight of every tag for all users uniquely. This allows us to include the content, the tags, of questions into account while making the predictions. Allowing us to include the factor of content-based filtering in our recommendation. The resulting values are then converted to integral percentages out of 100 to enable their usage in multinomial distribution.

The undirected graphical model, RBM is trained with the vectored representation of the questions' tags solved by each user and given as input to the input (visible) layer to determine the most relevant questions to the users. The methodology followed is as given below:

$$tf - idf\ (T,U,C)\ =\ tf\ (T,U)\ \times idf\ (T,C) \tag{1}$$

Where,*T* is the number of tags the user *U*attempted,*U* is the user in the corpus and *C* is the *M* × *N* data-frame developed.

Term Frequency (TF) is given by:

$$tf\ (T,U)\ =\ \frac{N_{T,U}}{N_U} \tag{2}$$

Inverse-document frequency (IDF) is given by:

$$idf\ (T,C)\ = \log \frac{N}{|U \in C : T \in C|)} \tag{3}$$

Where:
N = number of users in the collection
N_U= Number of tags attempted by the user *U*
$N_{T,U}$= Number of times tag T is attempted by user *U*

C. The Model

For training the undirected RBM with two layers, we have *M* users and *N* tags with integer rating values from 0 to *K*. In our case, *K = 100*.

Case 1: All *M* users solved the same set of *N* questions:

We can treat each user as a separate training model for an RBM which had *N* softmax visible units (tags) connected symmetrically to a set of binary hidden units. Each hidden unit can then learn to model a dependency between the tags of solved questions by different users.

Case 2: Most of the questions are not solved by the users:

Considering the case that a user *m*, from the set of total users *M*, have solved a set of questions *qSolved* from the total available questions. Therefore, a different RBM must be constructed for every available user. Every RBM will have the same number of hidden units, but an RBM will only have visible softmax units for the tags solved by that user. So, an RBM has only a few connections if that user solved few questions only. Each RBM will only have single training case, but all the corresponding weights and biases are tied together, so if two users have solved questions with similar tags, their two RBM's must use the same weights between the softmax visible unit for that set of tags and the hidden units. Though, the binary states of the hidden units will be different for different users.

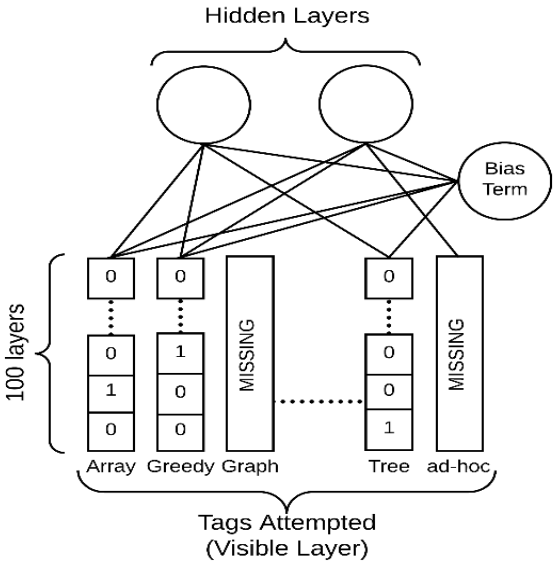


Figure 1. Restricted Boltzmann Model

Here we used a conditional multinomial distribution for modellingcolumns of visible binary matrix (***V***) and Bernoulli distribution for hidden user features ***h***(see *FigureI)*

In the equations below $\mathbf{v_i^k}$ is 1, if the*i*th tag is *k*, $\mathbf{b_i^k}$is the bias of rating k for tag i,$\mathbf{W_{ij}^k}$ is a symmetric interaction parameter between feature *j* and rating *k* of tag *i*. and ***F***is the number of hidden layers.

$$P\left(v_i^k = 1 \middle| h\right) = \frac{exp(\,b_i^k + \sum_{j=1}^{F} h_j W_{ij}^k)}{\sum_{l=1}^{K} exp(b_i^l + \sum_{j=1}^{F} h_j W_{ij}^l)} \tag{4}$$

$$P\left(\mathrm{h}_j = 1 \middle| v\right) = \frac{1}{1+\exp(-(\mathrm{b}_j + \sum_{i=1}^{m}\sum_{k=1}^{K} \mathrm{v}_i^k \mathrm{W}_{ij}^k))} \tag{5}$$

The marginal distribution over the visible ratings *V* is:

$$P(V) = \sum_h \frac{exp(-E(V,h))}{\sum_{V',h'} exp(-E(V',h'))} \tag{6}$$

While, the "energy" term is given by:

$$E(V,h) = -\sum_{i=1}^{m}\sum_{j=1}^{F}\sum_{k=1}^{K} \mathrm{v}_i^k \mathrm{W}_{ij}^k h_j - \sum_{i=1}^{m}\sum_{k=1}^{K} \mathrm{v}_i^k \mathrm{b}_i^k - \sum_{j=1}^{F} b_j h_j \tag{7}$$

Therefore, we will get the gradients for the parameters of a single user-specific RBM. The full gradients will then be obtained by averaging over the *M* users.

D. Predictions

Given the observed ratings *V*, we can predict a rating for a new question *q*. For this, we perform one iteration of the mean

field updates to get the probability distribution over *K* ratings for a tag *q*:

$$\widehat{p_j} = p(h_j = 1|V) = \sigma(b_j + \sum_{i=1}^{m}\sum_{k=1}^{K} v_i^k W_{ij}^k) \quad (8)$$

$$P(v_q^k = 1|\hat{p}) = \frac{exp(\, b_q^k + \sum_{j=1}^{F} \widehat{p_j} W_{qj}^k)}{\sum_{l=1}^{K} exp(b_q^l + \sum_{j=1}^{F} \widehat{p_j} W_{qj}^l)} \quad (9)$$

Though making predictions using time linear in the number of hidden units gives slightly better results, but the mean field updates method is more computationally efficient and can easily be deployed for a very large number of users.

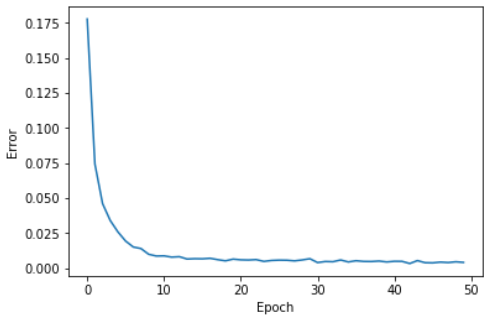
VII. Experimental Results

We trained the RBM model using 4 different values of hidden units 5, 10, 15 and 20, to analyze the optimal size of the hidden layer. The weights were updated using a learning rate of 0.01 and were initialized with small random values sampled with a zero-mean normal distribution with standard deviation of 0.01. To speed-up the training, we subdivided the whole dataset into small mini-batches, each of 25 cases (users), and updated the weights after each minibatch was computed. All models were trained for 50 passes (epochs) through the complete training dataset.

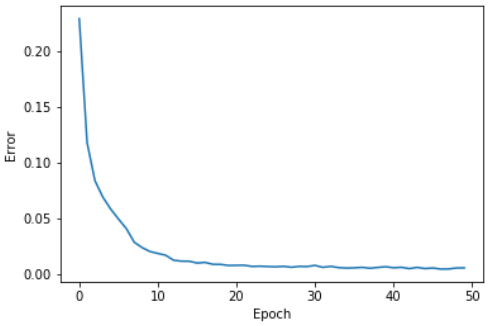
From *Figure II (a)*, we can infer that the optimal value to get the least root mean square error in our model is 10 hidden units.

VIII. Conclusion

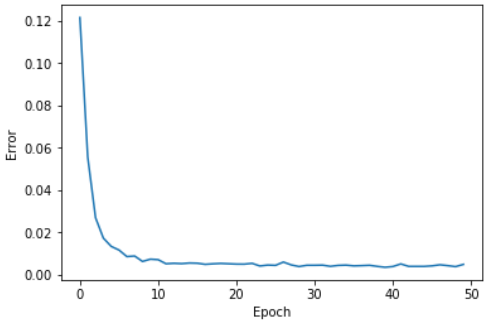
This paper presents a model which can predict relevant competitive coding questions that a user must attempt based on the questions attempted by the user and other users on the platform. The model uses a hybrid approach for filtering, wherein the Restricted Boltzmann Machine is deployed to make precise recommendations (predictions), while the Term Frequency-Inverse Document Frequency (TF-IDF) algorithm on a vectorized dataset is performed to normalize the tags associated with the questions and rate them accordingly as a percentage of plausibility to be solved by a user. This model has the potential to revolutionize the way competitive coding questions are attempted. Until now, the user (solver) needed to choose the coding questions on their own which weredone randomly or based on intuition, in turnreduces their productivity, as a lot of time is often utilized in search of the questions which are suitable for them, according to the question type and difficulty. As the 'mean field updates' algorithm is very computationally efficient, it can be used on anenormous number of users with an enormous number of questions and the model can be updated in real-time making it suitable for the continuous stream of data to be utilized efficiently.



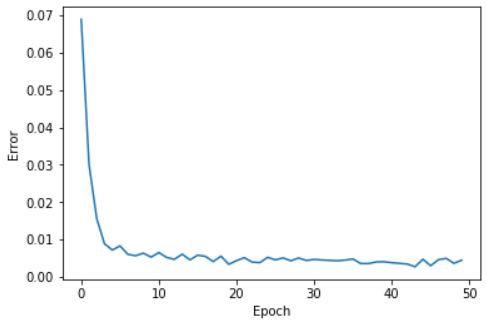
(a)



(b)



(c)



(d)

Figure 2.Graphs of error vs. epochs with different values of hidden units(a) 5 hidden units (b) 10 hidden units (c) 15 hidden units (d) 20 hidden units

IX. References

[1] Goldberg, David, David Nichols, Brian M. Oki, and Douglas Terry. "Using collaborative filtering to weave an information tapestry." Communications of the ACM 35, no. 12 (1992): 61-70.

[2] Terveen, Loren, and Will Hill. "Beyond recommender systems: Helping people help each other." HCI in the NewMillennium 1, no. 2001 (2001): 487-509.

[3] Sarwar, Badrul, George Karypis, Joseph Konstan, and John Riedl. "Item-based collaborative filtering recommendation algorithms." In Proceedings of the 10th international conference on World Wide Web, pp. 285-295. ACM, 2001.

[4] Ramos, Juan. "Using tf-idf to determine word relevance in document queries." In Proceedings of the first instructional conference on machine learning, vol. 242, pp. 133-142. 2003.

[5] Das, Abhinandan S., Mayur Datar, Ashutosh Garg, and Shyam Rajaram. "Google news personalization: scalable online collaborative filtering." In Proceedings of the 16thinternational conference on World Wide Web, pp. 271-280. ACM, 2007.

[6] Salakhutdinov, Ruslan, Andriy Mnih, and Geoffrey Hinton. "Restricted Boltzmann machines for collaborative filtering." In Proceedings of the 24th international conference on Machine learning, pp. 791-798. ACM, 2007.

[7] Larochelle, Hugo, and Yoshua Bengio. "Classification using discriminative restricted Boltzmann machines." In Proceedings of the 25th international conference on Machine learning, pp. 536-543. ACM, 2008.

[8] Rokach, Francesco Ricci Lior. "Introduction to Recommender Systems Handbook Francesco Ricci Lior Rokach And Bracha Shapira."

[9] Melville, Prem, and Vikas Sindhwani. "Recommender systems." In Encyclopedia of machine learning, pp. 829-838. Springer US, 2011.

[10] Hinton, Geoffrey E. "A practical guide to training restricted Boltzmann machines." In Neural networks: Tricks of the trade, pp. 599-619. Springer, Berlin, Heidelberg, 2012.

[11] Beel, Joeran, Stefan Langer, Marcel Genzmehr, and Andreas Nürnberger. "Introducing Docear's research paper recommender system." In Proceedings of the 13th ACM/IEEE-CS joint conference on Digital libraries, pp. 459-460. ACM, 2013.

[12] Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. "Distributed representations of words and phrases and their compositionality." In Advances in neural information processing systems, pp. 3111-3119. 2013.

[13] Hongliang, Cui, and Qin Xiaona. "The video recommendation system based on DBN." In Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/ IUCC/ DASC/ PICOM), 2015 IEEE International Conference on, pp. 1016-1021. IEEE, 2015.

