Movie Recommendation System Based on Character Graph Embeddings

Agisilaos Kounelis, Pantelis Vikatos, Christos Makris

Movie Recommendation System based on Character Graph Embeddings

Agisilaos Kounelis[1], Pantelis Vikatos[1], and Christos Makris[1]

Computer Engineering and Informatics Department
University of Patras, Patras 26504, Greece
{agis,vikatos,makri}@ceid.upatras.gr

Abstract. This paper presents a novel approach for recommending movies based on weighted Character Graphs. This approach proposes a dedicated crawler that gathers movie screenplays and a methodology of character graphs generation that contains all the necessary information needed for the representation of movie plots. A representative vector is extracted for each graph and used along with user ratings, as an input for a gradient boosting algorithm to predict movie ratings. The proposed method is tested on a publicly available MovieLens dataset and it was experimentally shown that it outperforms the fundamental collaborative filtering recommendation algorithms.

Keywords: Recommendation Systems · Character Graphs · Graph Embeddings.

1 Introduction

Creating recommendations able to be useful to humans has long been a challenge for researchers in the fields of information retrieval and AI. Broadly speaking, there are existing kinds of research efforts and platforms related to recommendations in topics such as books [18], movies [7], music [19] and university courses [15]. Recommendations Systems (RS) cover the needs of well-known online platforms i.e. Netflix, Amazon, Booking. The topic of movie recommendations gained the interest of research teams especially during the Netflix Prize competition and the award of $1M Grand Prize to team BellKor's Pragmatic Chaos.

Regarding movie suggestions, a plethora of research efforts [22] target a given number of rated user preferences or use the context and metadata of the movie to provide recommendations and predict the movie ratings. However, there are many different factors [21] which have an impact on a user's movie preference. Features such as film category, quality of the director or participation of specific actors can be determinants for selection and high rating. Online platforms such as the well-known IMDB[1] provide the basic descriptive information about each movie such as category, actors, writers and a short description. However additional features like the alternation of scenes, the plethora of characters throughout the same scenes, the frequency of monologues or common appearances of

[1] IMDb: https://www.imdb.com/

the film's heroes are missing. The research effort of Lee & Jung [11] shows that common stories of movies are strongly correlated with the efficiency of the recommendations. More specifically, they propose an innovative recommendation system using an effective and dynamic co-occurrence signed network representing the friendships between characters. This research inspires us to focus on movie plots as the factor that develops the user's movie needs.

In our paper, we represent movie plot as a weighted Character Graph (CG) whose nodes and edges correspond to the characters and the co-occurrence of characters in the same scenes respectively. An essential step in our approach constitutes the gathering of screenplays and the extraction of the characters. The identification and separation of the scenes in the screenplay relies on the well-defined structure of the script. The generated character graph contains all the necessary information needed for the depiction of the movie plot. However, for each character graph, a graph embedding is generated i.e. a single vector; which is more convenient for feeding a Machine Learning Algorithm. Our main challenge is to extract the representative vector (centroid) of each user that declares the mixture of plots that consumes and reflects his/her profile. We use the representative vector and the embedding of a candidate movie as an input for a gradient boosting algorithm to predict the movie rating. The evaluation of the recommendation system follows the common standards of the most published efforts in recommendation algorithms, using a public dataset and comparing the proposed method with the fundamental ones. We declare that after publication both the dataset and the source will be provided for re-generation of the results of the paper.

The significance of our contribution can be summarized as follows:

- Implement a dedicated crawler for gathering movie screenplays.
- Describe an efficient procedure to generate a movie character graph.
- Extract the movie embeddings and calculate the user representative vectors.
- Introduce a recommendation system architecture based on independent sub-modules.
- Train and evaluate a gradient boosting machine learning algorithm for rating prediction.

The rest of the paper is structured as follows. Section 2 overviews related work. We motivate our research from current challenges regarding related studies. In Section 3, we provide an overview of the methodology that we propose which is separated in sub-modules. Section 4 overviews details of the implementation of the system and Section 5 presents a reference to our experimental results and the discussion of our work. Finally, in Section 6, we declare the strengths and limitations of our approach as well as an outlook on future work.

2 Related Work

We present the related work in three fields. The first refers to topics of character graphs. The second provides studies related to graph embeddings. The third includes literature of recommendation systems and methodologies.

Study [14] presents a first approach in the field of character graphs, mapping out texts according to geography, social connections and other variables. A study to extract social networks from nineteenth-century British novels and serials is presented in research [4], in which the networks have been constructed by dialogue interactions. A similar study is presented in effort [2], related to social event detection and social network extraction from a literary text and particularly the book Alice in Wonderland. An expansion to movie screenplay, as a source of unstructured literary, is presented in studies [27,5,1,2]. In effort [27] the extraction of the social network by parsing the screenplay is proposed in order to investigate communities, hidden semantic information and automate story segmentation. Another study [5] is focused on character interaction and networks between characters from plays and movies. In research effort [1] it is presented a formalization of the task of parsing movie screenplays as well as the extraction of the social network of all characters having a dialogue with each other in a scene.

Study [13] provides the well known word2vec technique. It is arguably the most famous face of the neural network natural language processing revolution, taking as input a text corpus and returning a set of vectors as the output. The contribution in study [10] is a generalization of the word2vec method representing entire documents as vectors known as doc2vec. graph2vec [17] proceeds the method of doc2vec to the next stage, representing a graph as a vector facilitating comparisons between graphs.

The most common technique used by recommendation systems is Collaborative Filtering (CF) [22]. CF systems aggregate user interactions, discover similar users based on preferences and rank items group-wise. Matrix factorization [8] constitutes a well-known technique for collaborative filtering, providing dimensionality reduction of the sparse user-item matrix. Content-based systems recommend items based on similarity to content information. Recent research attempts of Suglia et al. [23] and Musto et al. [16] describe the use of embeddings for comparison, generating a representation for each user by averaging the description embeddings of the items that the user has interacted with.

3 Model

We depict the overall model architecture in Figure 1. The proposed model introduces the concept of a movie plot to create recommendations. The fundamental step in the methodology is the automated plot extraction from the film screenplay and the representation in a form that will be ready for use by machine learning algorithms. Given the existence of the script and the characters of the film from heterogeneous sources, we represent the plot as a weighted undirected graph which is called Character Graph (CG). Graph's nodes and edges correspond to the characters and the co-occurrence of the character pairs in the same scene respectively. An embedding is generated for each character graph though a Graph to Vector approach. The set of graph embeddings, movie ratings and a user's representative centroids of their highest rated movies constitute user

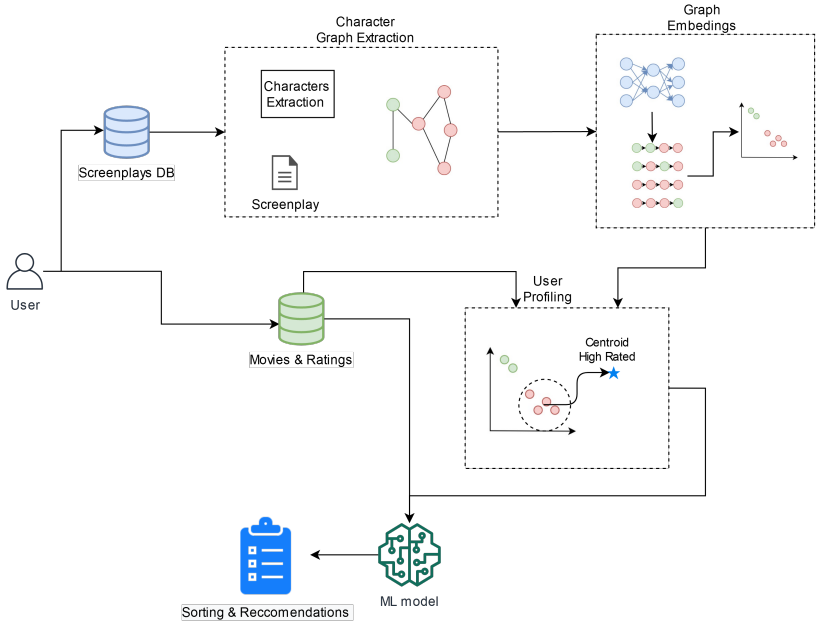


Fig. 1: Model Architecture

profiling. A trained model uses the combination of the centroid embedding and the embedding of a movie target to predict the rating. The ordered list of the candidate's targets forms the recommendations for our system.

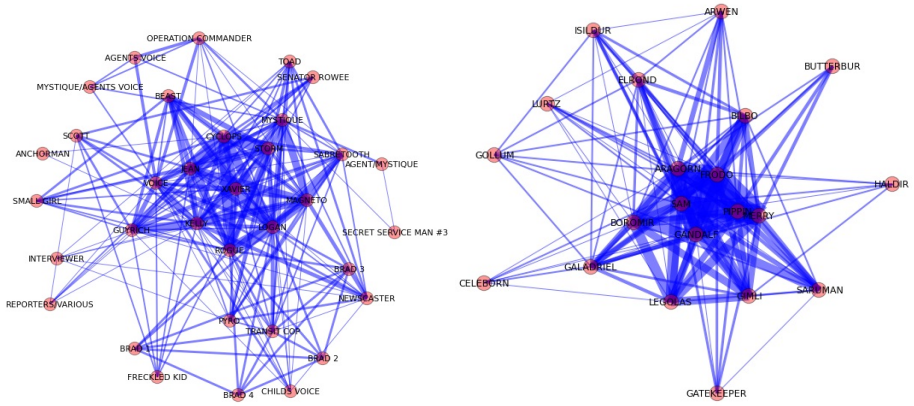
A summary of the independent tasks that our methodology consists of is given below:

- **Character Graphs Construction**. The prerequisite export of the characters is generated by utilizing external sources that list information about the film characters. The screenplay is modeled as an undirected graph G, where nodes and edges are characters and co-occurrence in scenes respectively. Each edge includes a weight that represents the frequency of the co-occurrence of the two characters in the same scenes.
- **Generation of Graph Embeddings**. This task is dedicated to the transformation of a character graph into a single vector. Using the graph2vec methodology, graph embeddings are learned in an unsupervised manner given a dimension size.
- **User Profiling**. The user profiling contains the set of the movie vectors as well as the rating of each movie for a particular user. The mixture of most preferable movie plots constitutes a user signature.
- **Rating Prediction & Recommendations**. An efficient classification model is trained and used for rating prediction. The model uses the centroid of a user's top rated movies and the extracted embedding of the movie for which

the prediction is made. The recommendations are produced by sorting the predicted ratings.

In the following subsection tasks and modules of our model are described in detail.

3.1 Character Graph Extraction



(a) X-Men (b) The Lord of the Rings

Fig. 2: Social Networks of Co-occurrence Characters in Screenplay Scenes

We consider that this process consists of four main steps: 1) the identification of scenes; 2) the identification of characters; 3) the quantification of their interactions; 4) the extraction of the proper graph.

Due to the structure of screenplays, scenes can be easily separated. Afterwards, there are several direct and indirect methods for identification of characters in a text [12,3,26]. Authors in [9] mention that there are various ways to eliminate possible errors related characters recognition. Those methods include removing: infrequent names, characters that appear not frequent enough, a character if he appears three times or fewer, or a character when he amounts to 1% or less of all occurrences. We collect the top-10 characters per movie keeping only the most significant characters. We declare that the characters that interact with each other should have at least one co-occurrence in the same scene. Every co-occurrence detected in a scene increases the weight of the edge that connects the corresponding pair of characters. Each link contains a weight related to the frequency of the co-occurrence in the whole screenplay. In Figure 2 sam-

ples of movie character graphs are depicted. The Graph Extraction algorithm is described in Algorithm 1.

Algorithm 1: GetGraph ($Screenplay$, $Characters$)

input : $Screenplay$: The Screenplay of the Movie in text format
$Characters$: The list of the Movie's Characters
output: $G = (V, E)$: The Character Network Graph of $Screenplay$
$V = C$, $E = \varnothing$
$Scenes = \{$text sections between EXT and INT tags of $Screenplay\}$
begin
  foreach $Scene \in Scenes$ **do**
    $SceneCharacters = FindCharacters(Scene)$
    $Cooccurrences = \{(a, b) \mid a, b \in SceneCharacters$ and $a \neq b\}$
    foreach $(a, b) \in Cooccurrences$ **do**
      if $(a, b) \notin E$ **then**
        $E \leftarrow E \cup (a, b)$
      $(a, b).weight \leftarrow (a, b).weight + 1$
  return $G = (V, E)$

3.2 Generation of Graph Embeddings

The main objective of this module is the transformation of the character graphs to a form that applies to machine learning approaches. Vector representations of data range from feature vectors, via automatically constructed graph kernels, to learned representations, either computed by dedicated embedding algorithms or implicitly computed by learning architectures, like graph neural networks. The performance of machine learning methods crucially depends on the quality of the vector representations. We introduce the methodology of graph2vec as implemented in [20]. In graph2vec, Graph Embeddings are learned in an unsupervised manner given a dimension size. The purpose of graph2vec technique is to transform a graph to a vector. In this way, the similarity percentage of two graphs is the same as the similarity percentage of their two corresponding vectors. This is a way to approximate the graph isomorphism (GI) problem [6] for the Character Graphs.

3.3 User Profiling

An active user in online movie platforms e.g. IMDb; has the option to rate a movie based on his/her own criteria. We consider that one important factor which influences the rating is the plot of the movie. Thus, we formulate a user profile based on the high rated movies and the graph embeddings extracted from the screenplays. We extract as user signature, the centroid of the graph embeddings of user's highest rated movies.

3.4 Rating Prediction & Recommendations

The methodology provides recommendations using rating estimations for each movie that a particular user has not rated. We trained and used the XGBoost classifier since it was the one achieving the best score compared with other classifiers. The model makes use of the user's centroid vector as well as the embedding of the target movie. The Rating Prediction algorithm is described in Algorithm 2, taking advantage of the *GetGraph* function described above.

Algorithm 2: GetPredictions ($Screenplays$, $Ratings$)

input : $Screenplays$: The Screenplays of the Movies in text format
$Ratings$: The ratings of users in $(user, movie, rating)$ format
output: $Predictions$: The missing ratings predictions in $(user, movie, rating)$ format

$Graphs = \varnothing$, $Centroids = \varnothing$, $Predictions = \varnothing$
begin
  foreach $Screenplay \in Screenplays$ **do**
    $Graph = GetGraph(Screenplay)$
    $Graphs \leftarrow Graphs \cup Graph$
  $Embeddings = Graph2Vec(Graphs)$
  foreach $User \in Ratings$ **do**
    $UserCentroid = FindCentroid(User, Embeddings, Ratings)$
    $Centroids \leftarrow Centroids \cup (User, UserCentroid)$
  $Predictions = XGB(Centroids, Embeddings, Ratings)$
  return $Predictions$

4 Implementation

A multi-threaded IMSDb crawler using Python was implemented and a set of 1061 movie screenplays was gathered from The Internet Movie Script Database (IMSDb)[2]. Studies [1,24] mention that scenes are bounded by $INT.$ and $EXT.$ tags, as shown in Figure 3. Regarding scenes separation and characters recognition, the HTML-formatted screenplays that these tags occur were only retained. The name of each movie was the same as the name of the script, with the only exception being movies whose title starts with "The" or "A". Such an example is "The Theory of Everything" whose script name is "Theory of Everything, The". We handled such cases using the proper movie names for collecting the characters. We collected movie's characters using data from The Movie Database (TMDb) API[3], which guarantees data consistency API. For each movie the top-10 characters along with their TMDb id were stored.

The graph representations were generated using the NetworkX[4]. The automatic extraction of necessary entities using was occurred by the Named-entity

[2] IMSDb: https://imsdb.com/
[3] TMDb: https://www.themoviedb.org/
[4] NetworkX: https://networkx.github.io

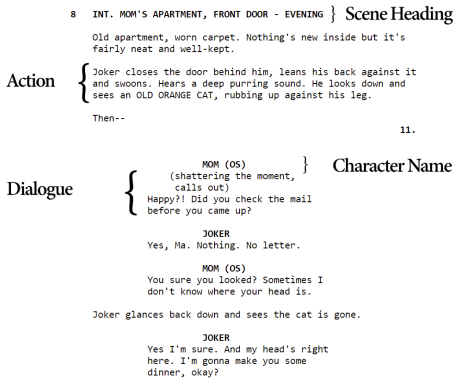


Fig. 3: Screenplay Structure

recognition (NER) open-source library, spaCy[5]. We used fuzzywuzzy[6] to match names using the Levenshtein Distance from the collected movie characters, with a minimum threshold of 80%. The weight of the edge that connects two actors were increased representing the co-occurrence. Each isolated node was removed from the graph. That practically means that our technique was unable to identify a co-occurrence of him/her with another character in the entire screenplay.

Regarding to the type of the graph, the graph2vec approach of the generation of graph embeddings, uses graphs with node weights. The generated character graph contains only edge weight. Thus, we created a label for each node using the Normalized Degree Centrality ($C_D(u) = \frac{degree(u)}{\sum_{e \in E} weight(e)}$) rounded to one decimal place. Intuitively, each label represents the percentage of participation of the corresponding character in the movie network. After parameter tuning, we concluded to setting the dimensionality of the embeddings to 70, the number of Weisfeiler-Lehman iterations to 2 and the minimal count of graph feature occurrences to 1. We used the XGBoost classifier [7] for machine learning purposes.

5 Evaluation & Results

The number of screenplays downloaded from IMSDb was 1061. The extracted characters from the plots were found from TMDB API for 1029 movies. Characters were recognized using our method in 986 movies and the same number of graphs was generated. The MovieLens 1M Dataset[8], containing 6040 users and 3952 movies, was used in an attempt to test our recommendation system. 311658 ratings were found for 969 of the downloaded screenplays. At first, for

[5] spaCy: https://spacy.io/
[6] fuzzywuzzy https://pypi.org/project/fuzzywuzzy/
[7] XGBoost: https://pypi.python.org/pypi/xgboost
[8] MovieLens: https://grouplens.org/datasets/movielens/1m/

each user, a centroid was extracted from the embeddings of his highest rated movies (more than 4 out of 5).

Procedure	**#movies**
Crawling IMSDB	1061
Merging TMDB	1029
Creating Character Graphs	986
Merging Movielens	969

Table 1: Number of movies in preprocessing steps

The machine learning model was trained and tested using K-Fold Cross-Validation (K=10). We evaluated the classifier in the performance metrics of precision, recall and F-measure (F1 score) which is the harmonic mean of precision and recall. Figure 4 presents the performance of F-measure XGB in comparison to the increasing number of training samples. We used as input for training and evaluation a percentage of the available dataset. The F-measure reaches the highest value, 0.8, using 90% of the available dataset. The lowest value of F-measure is 0.6 using 10% of the dataset. The performance seems to have a linear correlation with training samples until 60% but it continues at a high peak of 70%.

Trainset (%)	**Precision**	**Recall**	**F1**
10	0.63	0.61	0.6
20	0.65	0.63	0.62
30	0.66	0.64	0.63
40	0.67	0.65	0.65
50	0.69	0.67	0.66
60	0.7	0.68	0.68
70	0.77	0.76	0.75
80	0.79	0.78	0.78
90	0.81	0.8	0.8

Table 2: Evaluation Metrics of gradient boosting algorithm (XGB)

The remaining part of the dataset constitutes the evaluation set in the following procedure of recommendations.

Our method is evaluated using the average Normalized Discounted Cumulative Gain (nDCG) of the users. We simulate 10 times the production of evaluation results and the score is calculated as the average of nDCG measurements.

The Discounted Cumulative Gain functions used are $DCG_p^1 = \sum_{i=1}^{p} \frac{rel_i}{2^i}$ and $DCG_p^2 = \sum_{i=1}^{p} \frac{rel_i}{\sqrt{i}}$ as presented in [25].

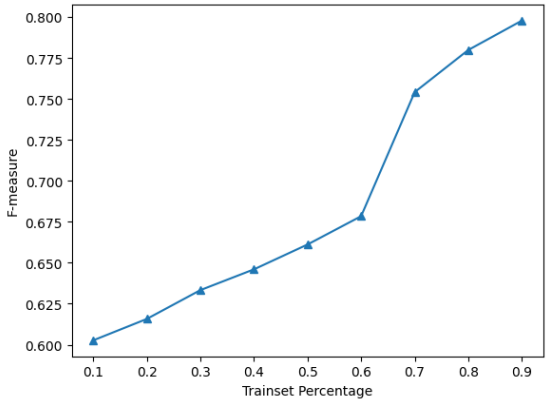


Fig. 4: Performance of XGB model

Our method outperforms other fundamental approaches, such as kNN (user-based nearest neighbor approach) and SVD (matrix factorization approach). Figure 5 presents the nDCG measurement in each train set percentage. Our method, Graph Embeddings, reaches the highest nDCG value, 0.98, increasing 0.03% the performance from methods kNN and SVD. We add the random selection and we produce the nDCG score shuffling the movie scores on a user level. It's clear that our ranking function converges to a different limit and distinguishes well from the other ranking functions.

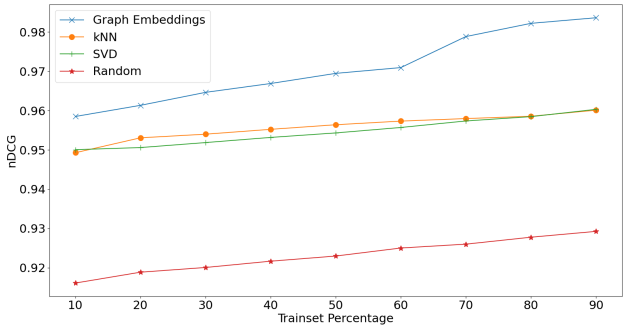


Fig. 5: A comparison based on nDCG between the proposed and other popular methods

6 Conclusions

In this paper, we present an innovative methodology for movie recommendations based on the automated movie character graph and a dedicated machine learning model for rating prediction. We focus on a model architecture initiating from the gather of the necessary information of screenplays and ending to movies rating prediction, which constitutes the basic component of our work.

We propose a way to use movie social network graphs which represent the co-occurrence of characters in scenes in order to use the plot of the movies in our recommendation system. We substitute the character graphs to their embeddings which seems to be an efficient way to handle them. Then, we evaluate the performance of some well-known collaborative filtering methods and compared their performance with our approach, which outperforms. The limitation of this approach is usually associated with the lack of publicly available screenplays. Also, some scripts follow their format, something that prevents the recognition of co-occurrences because our proposed method needs the declaration of the boundaries of the scene. Besides, our method relies on the specification of the cast from external sources. According to experimental results, our novel methodology ultimately aims to provide a good basis for future works that want to establish a possibly iterative procedure through which they can continually train and improve on the results and generalization ability of an existing predictor.

As future work, we are interested in extending our methodology beyond movies (e.g., books, series) and recognize the factors that influence the outcomes of our algorithm at a finer granularity level. Furthermore, our current interests are directed on utilizing cloud computing technologies in our method and the study of scalability issues that will help in reducing the computational time needed.

References

1. Agarwal, A., Balasubramanian, S., Zheng, J., Dash, S.: Parsing screenplays for extracting social networks from movies. In: Proceedings of the 3rd Workshop on Computational Linguistics for Literature (CLFL). pp. 50–58 (2014)
2. Agarwal, A., Kotalwar, A., Rambow, O.: Automatic extraction of social networks from literary text: A case study on alice in wonderland. In: Proceedings of the Sixth International Joint Conference on Natural Language Processing. pp. 1202–1208 (2013)
3. Bonato, A., D'Angelo, D.R., Elenberg, E.R., Gleich, D.F., Hou, Y.: Mining and modeling character networks. In: International workshop on algorithms and models for the web-graph. pp. 100–114. Springer (2016)
4. Elson, D.K., Dames, N., McKeown, K.R.: Extracting social networks from literary fiction. In: Proceedings of the 48th annual meeting of the association for computational linguistics. pp. 138–147. Association for Computational Linguistics (2010)
5. Gil, S., Kuenzel, L., Caroline, S.: Extraction and analysis of character interaction networks from plays and movies. Retrieved June **15**, 2016 (2011)
6. Grohe, M.: word2vec, node2vec, graph2vec, x2vec: Towards a theory of vector embeddings of structured data. In: Proceedings of the 39th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems. pp. 1–16 (2020)

7. Harper, F.M., Konstan, J.A.: The movielens datasets: History and context. Acm transactions on interactive intelligent systems (tiis) **5**(4), 1–19 (2015)
8. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer **42**(8), 30–37 (2009)
9. Labatut, V., Bost, X.: Extraction and analysis of fictional character networks: A survey. ACM Computing Surveys (CSUR) **52**(5), 1–40 (2019)
10. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: International conference on machine learning. pp. 1188–1196 (2014)
11. Lee, O.J., Jung, J.J.: Modeling affective character network for story analytics. Future Generation Computer Systems **92**, 458–478 (2019)
12. Makris, C., Vikatos, P.: Community detection of screenplay characters. In: Ifip international conference on artificial intelligence applications and innovations. pp. 463–470. Springer (2016)
13. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
14. Moretti, F.: Graphs, maps, trees: abstract models for a literary history. Verso (2005)
15. Morsy, S., Karypis, G.: Will this course increase or decrease your gpa? towards grade-aware course recommendation. arXiv preprint arXiv:1904.11798 (2019)
16. Musto, C., Semeraro, G., de Gemmis, M., Lops, P.: Learning word embeddings from wikipedia for content-based recommender systems. In: European Conference on Information Retrieval. pp. 729–734. Springer (2016)
17. Narayanan, A., Chandramohan, M., Venkatesan, R., Chen, L., Liu, Y., Jaiswal, S.: graph2vec: Learning distributed representations of graphs. arXiv preprint arXiv:1707.05005 (2017)
18. Núñez-Valdéz, E.R., Lovelle, J.M.C., Martínez, O.S., García-Díaz, V., De Pablos, P.O., Marín, C.E.M.: Implicit feedback techniques on recommender systems applied to electronic books. Computers in Human Behavior **28**(4), 1186–1193 (2012)
19. Van den Oord, A., Dieleman, S., Schrauwen, B.: Deep content-based music recommendation. In: Advances in neural information processing systems. pp. 2643–2651 (2013)
20. Rozemberczki, B., Kiss, O., Sarkar, R.: An api oriented open-source python framework for unsupervised learning on graphs (2020)
21. Su, J.: Content based recommendation system (Jan 5 2016), uS Patent 9,230,212
22. Su, X., Khoshgoftaar, T.M.: A survey of collaborative filtering techniques. Advances in artificial intelligence **2009** (2009)
23. Suglia, A., Greco, C., Musto, C., De Gemmis, M., Lops, P., Semeraro, G.: A deep architecture for content-based recommendations exploiting recurrent neural networks. In: Proceedings of the 25th conference on user modeling, adaptation and personalization. pp. 202–211 (2017)
24. Turetsky, R., Dimitrova, N.: Screenplay alignment for closed-system speaker identification and analysis of feature films. In: 2004 IEEE International Conference on Multimedia and Expo (ICME)(IEEE Cat. No. 04TH8763). vol. 3, pp. 1659–1662. IEEE (2004)
25. Wang, Y., Wang, L., Li, Y., He, D., Liu, T.Y., Chen, W.: A theoretical analysis of ndcg type ranking measures (2013)
26. Weng, C.Y., Chu, W.T., Wu, J.L.: Movie analysis based on roles' social network. In: 2007 IEEE International Conference on Multimedia and Expo. pp. 1403–1406. IEEE (2007)
27. Weng, C.Y., Chu, W.T., Wu, J.L.: Rolenet: Movie analysis from the perspective of social networks. IEEE Transactions on Multimedia **11**(2), 256–271 (2009)