

Real-time Recommendation System for Stock Investment Decisions

Artur Bugaj and Weronika T. Adrian^a

AGH University of Science and Technology, al. A. Mickiewicza 30, 30-059 Krakow, Poland

Keywords: Recommendation Systems, Decision Making, Collaborative Filtering, Prediction of User Behavior Patterns, Recommendation Engine, Real-time, Knowledge Graphs, Semantic Processing.

Abstract: Recommendation systems have become omnipresent, helping people making decisions in various areas. While most of the systems can give accurate recommendations, their learning procedures can be time-consuming. In some cases, this is not permissible; for example when the information about the items and users changes very fast in time. In this paper, we discuss a new recommendation engine, based on labelled property graph knowledge representation and attributed network embeddings, which calculates real-time recommendations for stock investment decisions. In particular, we demonstrate an application of the DANE (dynamic attributed network embedding) framework proposed by Li et al. and show the promising results of the system.

1 INTRODUCTION

Making choices among available options – regardless if we look for an interesting book or want to make an investment decision – is significantly determined by the context in which we operate: the possibilities we have, our interests, and the previous decisions we made. Sometimes, our own history of choices may be limiting, so recommendations based on other people behaviour and outcomes may be a valuable option that widens the horizons of our analysis. A particular decision setting is a situation in which we have to make decisions quicker than usual, and at the same time, we do not have enough resources to determine what will be the optimal choice, or if the choice we claim the best will be optimal in the near future. To obtain a fast and relatively accurate recommendations, we need a system that adapts quickly to changing data.


Recommendation systems (Bouraga et al., 2014) have been around for a few decades, but the rapid growth of the information stored on the Web, sparked a renewed interest in their improvements. Ever-changing data to be analyzed poses a challenge for real-time systems that must balance the accuracy of recommendations with the speed of response. In the field of stock market, prices and forecasts can change daily and the investors' activity is dynamic in its nature – they buy or sell stocks or even change fields of interest. It is thus important to recognize which and how these factors should be taken into consid-

eration to deliver fast and accurate recommendation for a user that wants to make a good investment decision (Hernández-Nieves et al., 2020).

In this paper, we propose a recommendation engine for stock market that integrates semantic similarity assessment among investors, to incorporate their behaviour and areas of interest, with a technical analysis of the companies considered for investment. By bringing in the benefits of collaborative filtering and content-based recommendation, we improve the accuracy of the system's suggestions. This paper is organized as follows: in Section 2, we present the basic concepts and background, Section 3 describes the proposal and recent results, and we conclude the paper in Section 4.

2 PRELIMINARIES

Recommendation systems can be generally categorized into collaborative filtering (CF)-based (ones that take into consideration similarities among the system's users), content-based (that rely on considered items' attributes) and hybrid that aim to bring in together the advantages of both approaches and minimize their limitations. In recent years, knowledge graphs have been widely adopted as a side information resource for recommendation engines, as they on the one hand minimize the threat of a "cold start" in the systems, and on the other – provide a useful mean for recommendation explanations (for a detailed sur-

^a  <https://orcid.org/0000-0002-1860-6989>

vey see (Guo et al., 2020; Liu and Duan, 2021)).

Graph representations, although intuitive, expressive and flexible, do not constitute a good input to various data mining tasks, such as node classification, link prediction or community detection. To this end, various methods for learning numerical representations of nodes, edges and subgraphs have been put forward (Perozzi et al., 2014; Tang et al., 2015). The assumptions of the underlying graphs differ: some of these embedding methods are only suitable for homogeneous networks (i.e., graphs with nodes “of the same kind”), other allow for various node types, but only one sort of edges etc. Most of the methods assume a static structure to be learned and only some of the recent proposals have considered dynamically changing networks. In particular, Li et al. (Li et al., 2017) have proposed a framework for dynamic attributed networks embedding that addresses both the rich set of attributes being associated with nodes in a graph and a dynamic nature of it (addition deletion of nodes and edges).

3 SYSTEM PROPOSAL AND IMPLEMENTATION

Our proposal utilizes the attributed network embedding concept described in (Li et al., 2017). We model the domain of investors and their stock decision with a labelled property graph that is later appropriately queried for a given user, and the obtained subgraph is embedded onto a low-dimensional vector space with the so-called *consensus embedding*. Once we obtain the suggestions for potential companies to invest in, we incorporate the technical analysis of the candidate stocks to produce the final recommendations in near real-time. The system consist of two modules:

- *recommendation-engine*: a module that calculates recommendations based on users similarity (common relationships and attributes), and
- *data-provider*: a module that calculates current stock movement score (i.e., if a price is predicted to grow, to fall, or to stay the same).

Investors and their data, gathered from Investor Hunt ¹ and Nasdaq ² services, are stored in a graph database (see Fig. 1). The model contains four types of nodes:

1. *Investors*,
2. *Categories*,

3. *Industries*,
4. *Stocks*,

linked with the following relationships:

1. *(Investor)-[:POSSESS]-(Stock)*,
2. *(Investor)-[:INTERESTED]-(Category)*,
3. *(Category)-[:INCLUDES]-(Industry)*,
4. *(Industry)-[:COMPANY]-(Companies)*,

The recommendation system works in three consecutive phases that are:

1. **Candidate Selection**, in which the system fetches a subgraph related to the target user from the database. It ensures that further processing is done on most similar investors.
2. **Scoring**: similarity matrices for nodes in the fetched subgraph are created, in order to create their embeddings and to find correlations between them (here goes the actual implementation of consensus embedding)
3. **Re-ranking**: after computing the recommendations, we add scores dependent on other factors, which are important, but could not be retrieved with previous steps.

In the following subsection, we discuss the details of the process.

3.1 Candidate Selection

The first phase of the process is finding the most similar investors according to the graph database structure. Consequently, we reduce the number of similar investors and stocks significantly, using only a single query that reduces searching to subgraph of related nodes. Such a subgraph is shown on Figure 1. It is a result of a parametrized query, where the parameter is a target investor ID. We expect most similar investors as those, which we can find as having some common companies as well as some common interests.

We fetch only the necessary data, which optimizes the performance of the database; the common categories with the target investor and the stocks they have. To simplify a whole process, graph data is “mapped” into set of records, which are sorted by amount of common companies. We can fetch all investors, or limit a number of retrieved record to some constant number. It can potentially reduce some valuable data, but taking into account further calculations, we get much more speed by sacrificing only a little of data (in Subsec. 3.2, we describe a part of the recommendation engine algorithm, where we create similarity matrices, calculation of which is $O(n^2)$).

¹See <https://investorhunt.co/>.

²See <https://www.nasdaq.com/>.

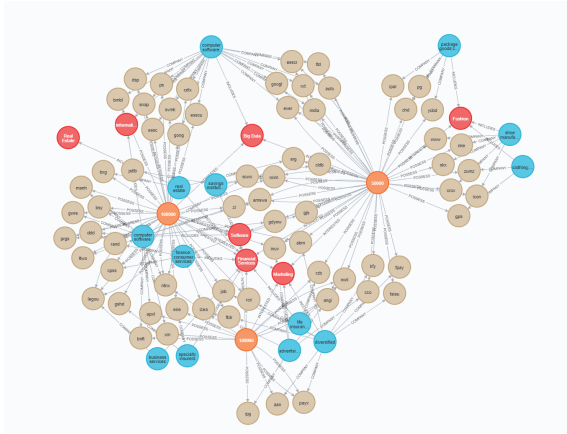


Figure 1: Subgraph selected in **Candidate selection** phase.

3.2 Scoring

After selecting a subset of the most similar investors, we prepare the data to be used in further recommendations, which are done according to the method proposed in (Li et al., 2017). We take into account the data that we have fetched in the previous step, that is, common categories of the stocks that investors are interested in and the companies they currently have. We create similarity matrices for them. Since data we have are actually collections, we can calculate similarity of collections (sets) with Jaccobi similarity metric. In our case, where stocks and categories are represented as nodes, this is the best way to map them into matrices. Let us denote similarity matrices for categories and stocks as $C^{n \times n}$ and $S^{n \times n}$. Let us also denote D_C , D_S and L_C , L_S as diagonal and Laplacian matrices created from C and S , respectively. We calculate D_C (and D_S) as

$$D_C = \begin{bmatrix} \sum_{i=1}^n M_C(1,i) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sum_{i=1}^n M_C(n,i)' \end{bmatrix}$$

and L_C (and L_S) as

$$L_C = D_C - M_C$$

Then, we normalize Laplacian matrix, from which we will calculate eigen decomposition problem.

$$L_{Anorm} = \begin{bmatrix} \frac{\sum_{i=1}^n A(1,i) - A(1,1)}{\sum_{i=1}^n A(1,i)} & \cdots & \frac{A(1,n)}{\sum_{i=1}^n A(1,i)} \\ \vdots & \ddots & \vdots \\ \frac{A(n,1)}{\sum_{i=1}^n A(n,i)} & \cdots & \frac{\sum_{i=1}^n A(n,i) - A(n,n)}{\sum_{i=1}^n A(n,i)} \end{bmatrix} \quad (1)$$

Then, we make other calculations of Offline Model of DANE (Li et al., 2017), which is the eigen

decomposition problem mentioned earlier, and intermediate embeddings, which we denote as Y_C (for categories) and Y_S (for stocks):

$$L_{Anorm} v = \lambda v \quad (2)$$

$$Y_C, Y_S = [c_2, \dots, c_k, c_{k+1}], [s_2, \dots, s_k, s_{k+1}]$$

Result of Y_C and Y_S is a $[c_2, \dots, c_k]$ and $[s_2, \dots, s_k]$, where c_k and s_k is a vector representing similarity between target investor, and k^{th} investor according to companies, and categories, respectively. As those vectors have been obtained with eigen decomposition, they are eigenvectors of normalized Laplacian matrix. After eigen decomposition and getting top k eigen vectors we maximize correlation between intermediate embeddings Y_C and Y_S , and therefore we solve eigen decomposition problem for:

$$\begin{bmatrix} Y_C Y_C' & Y_C Y_S' \\ Y_S Y_C' & Y_S Y_S' \end{bmatrix} \begin{bmatrix} p_C \\ p_S \end{bmatrix} = \gamma \begin{bmatrix} Y_C Y_C' & Y_C Y_S' \\ Y_S Y_C' & Y_S Y_S' \end{bmatrix} \begin{bmatrix} p_C \\ p_S \end{bmatrix} \quad (3)$$

We obtain the final result with

$$Y = [Y_C, Y_S] \times P \quad (4)$$

where $P = [p_C; p_S]$ is a top l eigenvectors from Eq. 3. Once we have calculated the most similar investors, we can check which companies could be potentially most interesting for target investor. Let us denote the similarity of investor i as s_i and investor i 's stocks set as $I_i = \{s_1, s_2, \dots, s_l\}$. Then:

$$r_{investor} = \sum_{i=1}^n \begin{cases} 0 & s \notin I_i \\ s_i & s \in I_i \end{cases} \quad (5)$$

where $r_{investor}$ is a similarity result for stock basing on investors' similarity.

3.3 Re-ranking

After calculation of the most similar stocks according to similarity of investors ($r_{investors}$), we re-rank those stocks, according to the price forecasts. The price forecasts are calculated in another module, in which we predict the price movements with *technical analysis*. Let us denote this as $r_{forecast}$. We calculate $r_{forecast}$ based on several technical analysis algorithms of current stock movement, such as: Detection of support/resistance of price, RSI, and OBV. Such a calculation allows us to estimate (with some probability) future movement of specified stock price. We want to recommend stocks which can grow in near future, and advice against those, whose price can fall. Therefore, the final recommendation is calculated as:

$$r_{stock} = r_{investors} + r_{forecast} \quad (6)$$

In this way, target user can get recommendations not necessarily from their target of interest, but they can see more stocks, which can lead to profits.

3.4 Update

In (Li et al., 2017), only eigenvalues and corresponding eigenvectors are updated, which increases the efficiency of the solution. This is possible, when we take into account whole knowledge graph. We are guaranteed, that indexes of columns on intermediate and consensus embedding will not change. However, with such an approach, the first step will be incredibly slow, as creation of similarity matrix, and eigen decomposition are computationally complex calculations.

As we mentioned in Sect. 3.1, we fetch a subgraph. If we assume, that relations between nodes changes in time, we can get different set of investors, in different order (a record mentioned in 3.1 refers to one investor's data), what implies the fact, that the similarity matrices can have different sizes, and the calculated properties can be related to different nodes in consecutive iterations. Therefore, we cannot make any updates, as it is proposed in (Li et al., 2017). Due to this setting, we have to repeat the whole procedure of calculation of eigendecomposition per each iteration, however, it is still a fast computation.

3.5 Results and Discussion

We have evaluated our proposal on the data scrapped from Investor Hunt and Nasdaq services: in total, 1368 investors with their business categories interests, amount of investments and average transaction value, and 545 companies. As an evaluation criterion, we state that a recommended stock have to occur minimum 6 times in top 40% most similar investors. Lowering the relevancy criterion would decrease this accordingly. We compared our results to ones obtained with alternative embedding algorithms, such as DeepWalk (Perozzi et al., 2014) and LINE (Tang et al., 2015). The results are shown in Tables 1 and 2. They could be better, if we consider whole graph in calculations instead of subgraph, but it would affect the response speed simultaneously.

Table 1: Comparison of precision@k.

	Top@10	Top@25	Top@50
DANE	63,33%	52,67%	31,67%
DeepWalk	25%	24%	21%
LINE	30%	32%	19%

Table 2: Comparison of recall@k.

	Top@10	Top@25	Top@50
DANE	63,33%	85,47%	97,9%
DeepWalk	25%	27,67%	49,17%
LINE	30%	58,88%	71,67%

4 CONCLUSION

In this paper, we addressed the challenge of real-time recommendation for stock market. We discussed a new system for investment recommendation based on attributed network embeddings and technical analysis of stocks. Our recommendation engine provides fast and robust computation of recommendations for the investment decisions, based on joint analysis of similarity of investors and stock predictions. The results obtained so far are promising and motivating for further exploration and a broader evaluation of the approach, covering also the technical analysis.

REFERENCES

- Bouraga, S., Jureta, I., Faulkner, S., and Herssens, C. (2014). Knowledge-based recommendation systems: A survey. *International Journal of Intelligent Information Technologies (IJIT)*, 10(2):1–19.
- Guo, Q., Zhuang, F., Qin, C., Zhu, H., Xie, X., Xiong, H., and He, Q. (2020). A survey on knowledge graph-based recommender systems. *IEEE Transactions on Knowledge and Data Engineering*.
- Hernández-Nieves, E., del Canto, Á. B., Chamoso-Santos, P., de la Prieta-Pintado, F., and Corchado-Rodríguez, J. M. (2020). A machine learning platform for stock investment recommendation systems. In *International Symposium on Distributed Computing and Artificial Intelligence*, pages 303–313. Springer.
- Li, J., Dani, H., Hu, X., Tang, J., Chang, Y., and Liu, H. (2017). Attributed network embedding for learning in a dynamic environment. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 387–396.
- Liu, J. and Duan, L. (2021). A survey on knowledge graph-based recommender systems. In *2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, volume 5, pages 2450–2453.
- Perozzi, B., Al-Rfou, R., and Skiena, S. (2014). Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710.
- Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., and Mei, Q. (2015). Line: Large-scale information network embedding. In *Proceedings of the 24th international conference on world wide web*, pages 1067–1077.