Migration to graph data structures for Big Data analysis in causal model of personal curricula

S Y Petrova[1]

[1] Department of Information Technologies and Systems, Yaroslav-the-Wise Novgorod State University Novgorod State University, Veliky Novgorod, ul. Bolshaya St. Petersburgskay, 41, office 3306, Russian Federation

Abstract. We were given a real-world problem – develop a recommender systems of planning and organization of the educational process at the university, and formation a causal model in an individual training plan for each student. Recommender system uses information about students and courses in order to attempt to recommend, through the use of some types of recommendation algorithms, educational modules that the student is likely to find useful. The major technologies used to develop the solution are: Graph database Neo4j, and recurrent neural networks using Truncated Backpropagation Through Time in Python deep learning library Keras. The whole discussion demonstrates how using new technologies for store and analyses big data allows to solve this project. The author found that the disjoint approach, with TBPTT uses, works reasonably well. Using a sliding window every time step is difficult/expensive to calculate, especially with a neural networks like TBPTT, and doesn't yield much benefit. Using a graph data model reduces the cost of executing a query to related data, which allows you to increase the productivity of the recommendation system as a whole. The prototype is flexible enough to integrate information from both internal repositories of the organization as well as external information collected from online courses networks.

1. Introduction

1.1. Project Summary

The world of educational environment has changed; old solutions are replacing by new ones, for example, blended learning. Mobility is the main motto of our time. In this regard, the entire infrastructure of information interaction between the teacher and his students is changing, and such principles as mobility and personalization of the educational process are putting at the head. The latter principle can only be achieved by having a complete information profile about each individual student, who wishing to form his personalized educational trajectory. This task includes a few subtasks:

- Identifying the student's personal characteristics and identifying the issues of professional interest that are of interest to him.
- Formation of the system of multivariate educational programs, which has necessary level of specification.
- Helping students to make a choice course, based on previous experience and global history of such decisions.

We were given a real-world problem in recommender systems and challenged to devise a novel or unique solution. Recommender system is a program which use information about students and courses

in order to attempt to recommend, through the use of some types of recommendation algorithms, educational modules that the student is likely to find useful. In general, a personalized educational trajectory is an individualized curriculum that meets the needs of a separate learning personality. In substance, it is a special program of student learning activities consisting of a logically coordinated structure of training modules. Students can work independently with this special program, using it completely or by selecting fragments from it, in accordance with their educational needs.

If we put all the courses studied in some university, on a single coordinate grid, allows us to form a system of multivariate educational programs. This system looks like net, which can be as the basis for constructing a personified educational trajectory.

1.2. What is system of multivariate educational programs

Formally, system of multivariate educational programs is database system designed to solve the problem of estimating students' preferences, for courses that the students has not yet studied. In general, knowledge sources correspond to the hierarchical structure of objects placed on the information space.

A course $\boldsymbol{p}$ is characterized by its state $\boldsymbol{\sigma}$. The state is the ensemble of information we need, to define course importance in process of estimating students' preferences. An event $\boldsymbol{e}$ is a change of state of a course. The events affecting the state of course p_1 are numbered sequentially as e_1^1, e_1^2, e_1^3 .... A course p_1 is in state σ_i^j immediately after the occurrence of event e_i^j and remains in that state until the occurrence of the next event, e_i^{j+1}. Each course can have $\boldsymbol{r} = \overline{0,m}$ relationships to others course. Relationships are directed, from an influential course to a dependent course. A curriculum is a collection of cooperating courses that presents individual learning path. Each curriculum is a finite set of courses with time constraints and other properties.

Consider a system consisting of $\boldsymbol{n}$ course, $p_1, p_2, \dots, p_i, \dots, p_n$ with σ_i^j the local state of course p_i; then the global multivariate educational program is an n-tuple of local states. This can be expressed as follows in Equation 1:

$$\Sigma^{(j_1,j_2,\dots,j_n)} = \left(\sigma_1^{j1}, \sigma_2^{j2}, \dots, \sigma_i^{ji}, \dots, \sigma_n^{jn}\right) \quad (1)$$

This is a n-dimensional net-space because we have n courses. The more relationships, the more difficult it is to determine the events leading to the desired state. A large number of paths increases the difficulty of working the recommender system.

Each event e_i^j occurring at the moments $t = 0,1,2,\dots m$, corresponds to the function f_{ij}, the result of which is the activation or conservation of the course. The subsequent state of the module depends on the previous and the results of the action:

$$S = \begin{cases} 1, S > \theta - activation. \\ 0, S < \theta - conservation. \end{cases} f(t) = F\left(\varphi(t), \Psi(t)\right) \quad (2)$$

where Ψ – statements of state formation; F – statement of action; θ – threshold module activity.

1.3. Problem Description

The amount of data available in system of multivariate educational programs is huge. This amount of description of education modules is much too overwhelming and time consuming for a student to go through.

Conventional search can help a student find courses; however, this does not help the students find new courses that they may not have previously known about. Leveraging recommender systems to make this data more organized and visible is beneficial to both the students and the university. A recommendation system with a well-implemented algorithm can help alleviate this problem by picking out courses that are most relevant to the given student and displaying them to the student in a special interface.

Given a set of students $\boldsymbol{S}$, a student $\boldsymbol{s}$ such that $s \in S$, a states rating scale $\boldsymbol{\sigma}$, and a set of courses $\boldsymbol{P}$, a recommender system aims to solve the following problem: Based on the knowledge of prior preferences of student $\boldsymbol{s}$ and students similar to $\boldsymbol{s}$, predict the courses, p_1, p_2 , ... , p_i, where, p_1, p_2 , ... $p_i \in P$, that $\boldsymbol{s}$ would indicate highest preference for based on ratings state scale $\boldsymbol{\sigma}$.

This can be also viewed as a formal optimization problem where the quantity to be optimized is the state. This can be expressed as follows in Equation 3:

$$\forall s \in S, p_u = \text{argmax}_{p \in P}\, \sigma(s, p) \tag{3}$$

Recommendation system requires building several base components: source dataset, filtering technique, data model, response time and memory consumption of system. And apply important considerations on these issues: sparsity of dataset [1], scalability of data storage to the dataset [2], cold start problem [3], diversity vs. accuracy problem [2], security and privacy [4] [5], evaluation metrics of results [6]. The decisions in these areas will greatly affect the behavior and performance of the recommendation system.

The problem is stated as follows: the data model should have set relationships ability and provide fast join seek and adequate scalability to solve the given optimization problem in increasing data volume.

2. Project

2.1. Technologies

There are several ways to implement a personified educational trajectory. Among them we can distinguish the following: artificial intelligence or semi-automated system.

The method of a semi-automated selection system means that both the computer and the person participate in the decision. A semi-automated selection system can have varying degrees of user influence on decision making.

The first way, when the students chooses all the courses that they want to study themselves, but the system, only builds the trajectory of learning. The implementation of this method is carried out with the help of a given algorithm, at the entrance of which the courses for learning and relations with other courses are necessary for further training, and at the output a hierarchical tree of courses. In addition, whoever performs the selection does not know every courses, so it is likely that his or her pick will be biased, because one single person cannot rapidly explore the whole network of the courses. Their decision making could be (too) slow, and a way to deal with inconsistent judgments by different people must be provided at the end of the consultation.

The second way is when the student using the help of system, chooses the course. So, the system offers a list of courses that can be studied further, based on the first chosen course. A student from them chooses the necessary ones, and the program forms the following list of courses, considering the chosen ones, etc. This is how the student's educational path is built. The algorithm of this method is not much more complicated than the previous one. At the entrance we have the chosen course, and at the output – all the courses that can be studied knowing this.

And the third way, when a student only sets the ultimate goal of learning, to which he will seek. The program forms a student's educational path according a given goal and algorithm.

All of the above methods are effective only if the user knows the ultimate goal of his training or at least the direction of training. But sometimes a student doubts the choice and provides the opportunity for the system to choose for him, and for this, a method of implementation using the recommendation system and artificial intelligence are needed. One and best of the methods for implementing artificial intelligence is the use of artificial neural networks. Artificial neural networks have become widespread over the past 30 years and have allowed to solve complex data processing problems. Therefore, the method of machine learning with the use of the capabilities of a semi-automated system best suited to develop a system of multivariate educational programs.

In order to store data with inter-object links, it is necessary to use the appropriate type of storage system. Today we can distinguish two classes of databases, which suitable for this application, these are: relational databases and graph databases.

The relational database model is most simple for software implementation. It is on it that the most common DBMS for personal computers is based. The relational database model is based on the concept of relationship. The advantages of a relational approach to creating a database, represented by a set of relationships, are:

- The ability to select a fairly simple data model in the form of two-dimensional tables that satisfy the requirements for modifying the database.
- Store highly-structured data in tables with predetermined columns of specific types and many rows of those defined types of information.
- Providing domain relations, and producing a cross-relationships based on the JOIN operations.
- High accuracy of data manipulation tools, achievable by dint of the using of relational algebra and the calculus of relations.

However, the relational model has some important disadvantages relative to this project [8]:

- Due to the rigidity of their organization, relational databases require developers and applications to strictly structure the data used in their applications.
- The simplicity of the relational model can be its main drawback, it does not allow in a number of applications to satisfactorily perform the functions of identification, structuring and classification of processed objects.
- Joins are computed at query time by matching primary and foreign keys of all rows in the connected tables. These operations are compute-heavy and memory-intensive and have an exponential cost.
- When many-to-many relationships occur in the model, you must introduce a JOIN table that holds foreign keys of both the participating tables, further increasing join operation costs.

Contrariwise, we have graph database where both relationships and data are equal importance to the subject itself. This means we are not required to infer connections between entities using special properties such as foreign keys or out-of-band processing like map-reduce. By assembling nodes and relationships into connected structures, graph databases enable us to build simple and sophisticated models that map closely to our problem domain. The data stays remarkably similiar to the its form in the real world – small, normalized, yet richly connected entities [8].

Thus, the major technologies used to develop the solution are: Graph database Neo4j, and recurrent neural networks using Truncated Backpropagation Through Time in Python deep learning library Keras. All technologies used were freely available for download on the Internet.

2.2. Recurrent neural networks approach for rank computing

According to the received characteristics of the student, the neural network determines the best courses for further education. And the student, in turn, chooses the final goal based on the decision made by the system and his preferences. As a result, the program forms a curriculum for this student. We divide the problem into part of the subtasks, each of which will correspond to a separate act of work. Three acts of neural work are selected in Figure 1:

- Determination of the general rate by a separate studied course.
- Determination of the general rate for an individual potential course.
- Determination of list some recommendations for further training.

Each subtask is solved by a separate neural network. An integral neural network is a set of interconnected neural networks. In the developed software product, all three neural networks are neural networks with back propagation. Recurrent neural networks are able to learn the temporal dependence across multiple timesteps in sequence prediction problems. Modern recurrent neural networks like the Long Short-Term Memory, or LSTM, network are trained with a variation of the Backpropagation algorithm called Backpropagation Through Time. This algorithm has been modified

further for efficiency on sequence prediction problems with very long sequences and is called Truncated Backpropagation Through Time (TBPTT) [9].

The major technologies used to develop the solution are: Graph database Neo4j, and recurrent neural networks using Truncated Backpropagation Through Time in Python deep learning library Keras. The core data structure of Keras is a model, a way to organize layers. The simplest type of model is the Sequential model, a linear stack of layers. For more complex architectures, you should use the Keras functional API, which allows to build arbitrary graphs of layers [11].

Here is the Sequential model:

```
1. from keras.models import Sequential
2. model = Sequential()
```

Stacking layers is as easy as .add():

```
1. from keras.layers import Dense
2. model.add(Dense(units=64, activation='relu', input_dim=100))
3. model.add(Dense(units=10, activation='softmax'))
```

Then we configure its learning process with .compile():

```
1. model.compile(loss='categorical_crossentropy',
2. optimizer='sgd',
3. metrics=['accuracy'])
```

A core principle of Keras is to make things reasonably simple, while allowing the user to be fully in control when they need to (the ultimate control being the easy extensibility of the source code).

The task of the neural network is to decide on the most successful further education of the student, based on the result of the courses he has studied previously and the statistics of other students in the courses on which further education of this student is possible. These are "by choice" for a given specialty. At the input of the neural network we have a list of courses studied by the student with a certain number of points in three categories: knowledge, skills, experience.

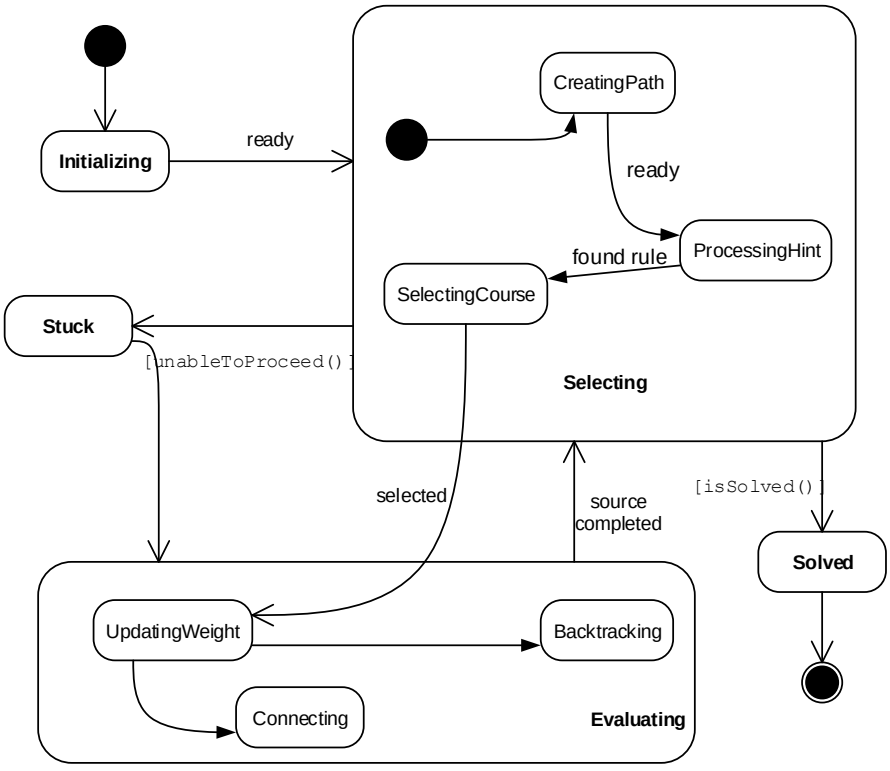

Figure 1. Process of determining the best courses for further education.

The weight of the course $\boldsymbol{V}$ characterizes by its possibility of inclusion in individual learning path and can be described by means of a vector (4).

$$V = (G, I, J, \tau, P) \tag{4}$$

G – parameter of systemic nature of the module; I– level of abstraction of educational information; J – degree of depth of educational information; τ – time coordinate; P– probability of an activation.

To calculate the systemic nature of the module, the degree of depth of the training information, and the level of abstraction of the educational information, we use the formulas described in [7].

$$G = \frac{1}{2+M}\left[\sum_{i=1}^{Q}\frac{N_{bi}}{N_{b0}} + \sum_{i-1}^{R}\frac{N_{ti}}{N_{t0}}\right] \tag{5}$$

where Q – the number of generalized sections, i.e. those who have more than average references to other academic disciplines;

R – the number of base topics, i.e. those whose sum of links exceeds the average number of links;

N_{bi} – the number of links to the *i*-th generalizing section;

N_{ti}– the number of links to the *i*-th base topic.

$$I = \frac{1}{N}\left[N_3 + N_{(2.4)} + N_{(2.3)}\right] \tag{6}$$

where N – the total number of courses in system of multivariate educational programs;

N_3 – the number of courses of the third type;

$N_{(m,\,n)}$ – the number of courses located on the *n*-th level of the *m*-th type.

$$J = \frac{1}{N}\sum_{n=1}^{3}\sum_{m=1}^{3}\left[3(m-1)+n\right]\times N_{(m.n)} \tag{7}$$

where m – type number of the courses;

n – level number of system of multivariate educational programs.

The core of the model consists of an LSTM cell that processes one course at a time and computes probabilities of the possible values for the next course in the sentence. The memory state of the network is initialized with a vector of zeros and gets updated after reading each course.

We use TBPTT supervised learning algorithm, which adjust the weights to minimize the error.

Recurrent neural networks LSTM can use their internal state to remember over very long input sequences. Such as over thousands of timesteps. This means that the configuration of TBPTT does not necessarily define the memory of the network that you are optimizing with the choice of the number of timesteps. You can choose when the internal state of the network is reset separately from the regime used to update network weights. Instead, the choice of TBPTT parameters influences how the network estimates the error gradient used to update the weights. More generally, the configuration defines the number of timesteps from which the network may be considered to model your sequence problem.

We can state this formally as something like [9]:

$$yhat(t) = f(X(t), X(t-1), X(t-2), \ldots X(t-n))$$

Where yhat is the output for a specific timestep, f(…) is the relationship that the recurrent neural network is approximating, and X(t) are observations at specific timesteps.

Suppose we are using a vanilla RNN and are given some categorical sequence x of length T:

$$= [x_1, \ldots, x_T]$$

To fit the parameters, compute a cross-entropy loss as follows. First, I compute the network outputs at all time-steps:

$$h_t = tanh(W_{hx}x_t + W_{hh}h_{t-1})$$
$$y_t = W_{yh}h_t \quad \text{for} \quad t < T$$

Each output is passed through a *softmax* to get a distribution vector P_t over the categorical output space. We used the loss

$$loss = -\frac{1}{T-1}\sum_{t=1}^{T-1} logP_t[x_{t+1}]$$

For some models, where T is small, training this model by unfolding the RNN over the whole time series and doing SGD works well.

Now, for large T we try to use truncation, i.e. essentially only consider the past up to some point $k \ll T$, say $k = 50$. We tried two variations:

- the first approach is to split my sequence x into disjoint sub-sequences of length k and then doing a descent step for each of these sub-sequences;
- second, a sliding window approach, where the window size is k, thus processing all $O(T)$ sub-sequences of length k.

Also, one could imagine intermediate methods, where the sliding window overlaps by some larger amount.

2.3. Graph data approach to store big data

The major technologies used to store big data is Graph database Neo4j. Graph databases address one of the great technological trends of today: leveraging complex and dynamic relationships in highly connected data to generate scalability and high performance computations advantages. For data of any significant size or value, graph databases are the best way to represent and query connected data. Authors [10] shown, that connected data is data whose interpretation and value requires us first to understand the ways in which its constituent elements are related. More often than not, to generate this understanding, we need to name and qualify the connections between things. Since the complete system of multivariate educational programs consists of more than million relationships, we present here a small illustrative subnetwork. The Figure 2 below shows the data model for the subnetwork, which contains four node labels and five relationship types.

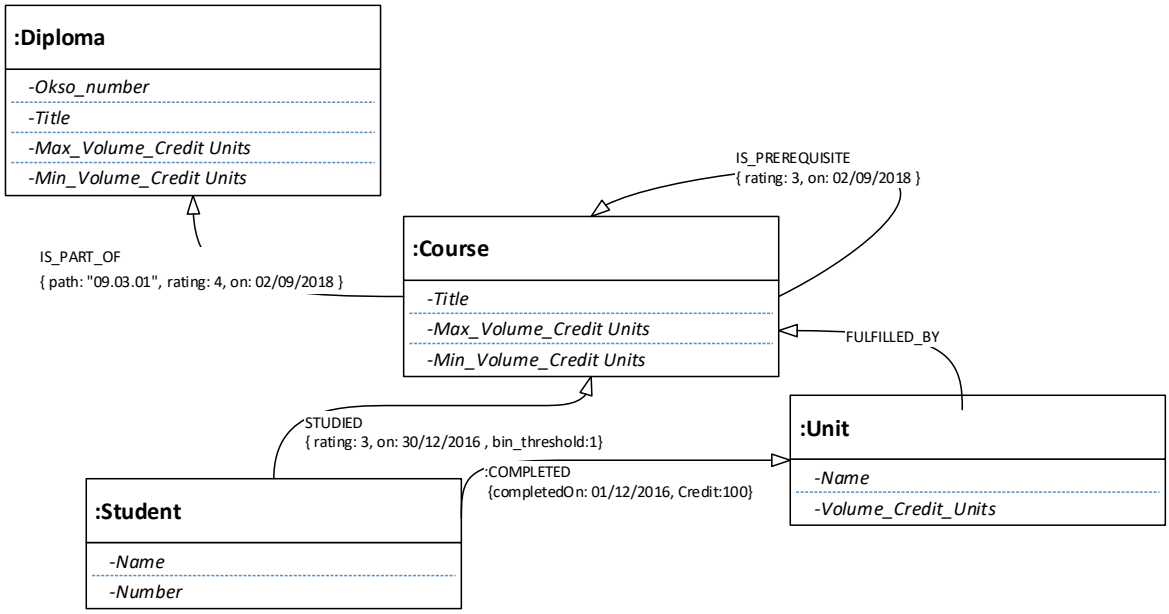

Figure 2. Initial graph data model.

This network visualization in Figure 1 shows several interesting data relationships. Related to the Course internal data, note that:

- STUDIED, points to the passed Courses of the Student. This relationship has few properties, in Figure 2 shows three important for our investigation: rating (estimation of the Course the

student has given), on (evaluation date), bin_threshold (binary statement about the capabilities of the Student: if the student has already performed successfully a certain course, then the link property is bin_threshold=1; if instead the Student is assumed to be capable of taking the course but never tried, in the graph it is set bin_threshold=-1 ; and finally, if it is known that the Student is not capable to learn the related course, the property bin_threshold=0).

- The Course node itself is linked to a collection of Unit nodes, that represent a certain activities (lectures, labs, practise, and etc.) inside the course. A learning item IS_FULFILLED by a course.
- A learning path consists of a few ordered courses; some of these can be studied in parallel. Since a diploma can have more than one path, and items on the path can be common, we need a way to identify which items constitute one learning path. Therefore, a learning path is modelled as a chain where each IS_PART_OF relation is qualified by the learning path name, for example { path: "09.03.01"}.
- Some Course node can be connected from 0 to *n* another Course nodes via the IS_PREREQUISITE link, to implement dependence of one subject on another. This connection may be very important especially for knowledge learning path.

Finally, each Student node can be connected to Union node, via the COMPLETED link. COMPLETED, indicates that the student has a specific skill. It is highly equivalent to the previous relationship STUDIED, with the only difference that it also illustrates a Student's knowledge, competence and experience in credits. The second property could be used to infer the level of expertise acquired by the student in a certain area, and may be used to understand possibilities learning next course.

2.4. Filtering out Courses

The classical approach to modeling such problems proves to be inadequate for the reason that the representation of the parameters of this problem by precise numerical values turns out to be in principle inadequate due to their weak structuredness, variability in time and uncertainty.

We saw in Figure 2 a part complex network (or graph) of nodes and links, that has the appearance of an emergent self-organizing structure.

Excluding some courses from the search is the first task to complete. This requirement may derive from common-sense reasoning, internal regulations, or requests. Moreover, it will minimize the number of nodes and links to traverse in subsequent queries, resulting in improved performance.

First step, we can use Label for filtering. A node can have zero, one or more labels. Nodes that share the same label are grouped into a collection that identifies a subset of nodes in the database graph to perform queries against it. In this example, it is a Diploma label. Therefore, a first query will search for courses that have joined the subject area, and filter another labeled courses out of the prospective pool.

Second step, we use a path, that specifies a traversal of part of the graph. It is typically used as part of a query to specify a pattern, where the query will retrieve from the graph data that matches the pattern. A path is typically specified by a start node, followed by one or more relationships, leading to one or more end nodes that satisfy the pattern. The sequences of courses are specified by path expressions.

Whenever you run the equivalent of a JOIN operation, the graph database uses this path expressions, directly accessing the associated nodes and eliminating the need for costly search and match calculations. This ability to preset relationships in a database structure allows us to provide performance several orders of magnitude higher than in other types of databases, especially for queries using JOIN ones. As you can see, the structural differences between relational and graph data bases are very large. The rectilinear structure of the graph leads to much simpler and more expressive data models than the models created using traditional relational or other NoSQL databases.

3. Conclusion

This research will be used as a basis for developing a system of planning and organization of the educational process at the Novgorod State University, and formation of a model and an individual training plan for each student.

The sample curriculum recommends a course, if this amount has not been determined by the state educational standard, and the time of its study in the general set of courses of the curriculum.

An approximate program of course includes requirements for knowledge, skills and skills acquired as a result of its study. The sample program gives the distribution of the total labor intensity of the study of the course to the student's classroom and out-of-class work, as well as sets the types of classroom activities, gives recommendations on course design.

The major technologies used to develop the solution are: Graph database Neo4j, and recurrent neural networks using Truncated Backpropagation Through Time in Python deep learning library Keras. The whole discussion demonstrates how using new technologies for store and analyses big data allows to solve this project. The author found that the disjoint approach, with TBPTT uses, works reasonably well. Using a sliding window every time step is difficult/expensive to calculate, especially with a neural networks like TBPTT, and doesn't yield much benefit. Using a graph data model reduces the cost of executing a query to related data, which allows you to increase the productivity of the recommendation system as a whole. The prototype is flexible enough to integrate information from both internal repositories of the organization as well as external information collected from online courses networks.

4. References

[1] Sarika Jain, Anjali Grover, Praveen Singh Thakur, and Sourabh Kumar Choudhary 2015 Trends, problems and solutions of recommender system. In *Computing, Communication & Automation (ICCCA), 2015 International Conference on* pp 955–958 IEEE

[2] Jie Lu, Dianshuang Wu, Mingsong Mao, Wei Wang, and Guangquan Zhang 2015 Recommender system application developments: a survey. *Decision Support Systems*, 74 pp 12–32

[3] Lucas Bernardi, Jaap Kamps, Julia Kiseleva, and Melanie JI Mueller 2015 The continuous cold start problem in e-commerce recommender systems. *arXiv preprint arXiv:1508.01177*.

[4] Robin Burke, Michael P Mahony, and Neil J Hurley 2015 Robust collaborative recommendation. In *Recommender Systems Handbook* pp 961–995 Springer

[5] Arjan JP Jeckmans, Michael Beye, Zekeriya Erkin, Pieter Hartel, Reginald L Lagendijk, and Qiang Tang 2013 Privacy in recommender systems. In *Social media retrieval* pp 263–281 Springer

[6] Linyuan Lu, Mat¨u´˘s Medo, Chi Ho Yeung, Yi-Cheng Zhang, Zi-Ke Zhang, and Tao Zhou 2012 Recommender systems. *Physics Reports*, 519(1) pp 1–49

[7] Makarov A A 1999 Methodology and methods of the system organization of integrated monitoring of the quality of education \ *thesis for obtaining the scientific degree of Doctor of Technical Sciences on specialty 05.13.10 - Management in social and economic systems.* - Moscow.

[8] Concepts: Relational to Graph 2018 [Electronic resource]. – *Access https://neo4j.com/developer/graph-db-vs-rdbms/*

[9] Jason Brownlee 2017 How to Prepare Sequence Prediction for Truncated Backpropagation Through Time in Keras, [Electronic resource]. - *Access https://machinelearningmastery.com/truncated-backpropagation-through-time-in-keras/*

[10] Ian Robinson, Jim Webber and Emil Eifrem 2015 Graph Databases. *Neo Technology,* 220 p. Published by O'Reilly Media

[11] Keras: The Python Deep Learning library. 2018, [Electronic resource]. - *Access https://keras.io/*