International Core Journal of Engineering Volume 6 Issue 3, 2020
ISSN: 2414-1895 DOI: 10.6919/ICJE.202003_6(3).0028

Sessions Prediction for Open Educational Resources Based on Deep Learning

Yanhan Du [a], Lin Zhang[b]

Shanghai Maritime University, Shanghai 201306, China.

[a]zzduyanhan@qq.com, [b]linzhang@shmtu.edu.cn

Abstract

Open Educational Resources (OER) are increasingly welcomed by people as the demand for education. However, how to make full use of them and make learners study efficiently could be some tackle problems to be solved. In this paper, we start data processing from Yale Open Courses and propose an efficient deep learning based model capturing context similarity by using word embeddings and LSTM to predict the most relevant sessions from any text paragraph from courses so that help to recommend courses or generate learning roads for OER users. The results of the experiment indicate that the model shows high performance on OER courses dataset in the experiment and also provides ideas for sessions' prediction and recommendation.

Keywords

Open Educational Resources; Word embeddings; Word2Vec; Doc2Vec; LSTM; Cosine similarity.

1. Introduction

Today, the importance of education goes without saying and global demand for education is still increasing. However, because of the varieties of educational resource depending on the culture, the language, the domain and so on make the modalities of providing the education differently.

Open educational resources (OER) are freely accessible, openly licensed text, media, and other digital assets that are useful for teaching, learning, and assessing as well as for research purposes. There is no universal usage of open file formats in OER (Open educational resources). The European project X5GON, trying to build cross model , cross cultural, cross lingual, cross domain, and cross site global OER Network, is leading to creating a solution that will help users/students find what they need not just in OER repositories, but across all open educational resources on the web.

There are increasingly huge learners choosing OER to study courses, even obtain degrees. However, aside from the huge varieties of worldwide OER, like different languages, different culture, different domains and so on, the learning area that every learner interests or specializes in is various and users have distinct levels of knowledge, i.e. different backgrounds, so that every user has different learning path and recommended courses in the process of their learning.

Above mentioned, in the degree of OER service's providers, the problem to solve is how to provide a specific learning path to each learner and recommend relative courses to them in order to meet their demands of learning. Specifically, how to dedicate data from OER websites, clean and filter data and use deep learning methods especially neural networks algorithms to solve the problem should also be considered.

At present, many scholars use machine learning methods to deal with natural language processing (NLP), Deep learning architectures and algorithms have already made impressive advances in fields such as computer vision and pattern recognition. Following this trend, recent NLP research is now

ISSN: 2414-1895 DOI: 10.6919/ICJE.202003_6(3).0028

increasingly focusing on the use of new deep learning methods. For decades, machine learning approaches targeting NLP problems have been based on shallow models (e.g., SVM and logistic regression) trained on very high dimensional and sparse features. In recent years, neural networks based on dense vector representation have obtained good results on a variety of NLP tasks. This trend depends on the success of word embedding and deep learning methods. Deep learning makes multi-level automatic feature representation learning possible. NLP systems based on traditional machine learning rely heavily on hand-made features, which are time-consuming and often incomplete. Ronan Collobert's study[1] *Natural Language Processing (Almost) from Scratch* showed simple deep learning frameworks that outperformed best practices at the time on multiple NLP tasks, such as named entity recognition (NER), semantic role annotation (SRL) and part-of-speech tagging. After that, researchers proposed a large number of complex deep learning-based algorithms for solving difficult NLP tasks, such as Word2vec, RNN and CNN etc.…In this paper, we focus on the text data which determines the contents and themes of each course dedicating from Yale open course and use deep learning methods to predict the most relevant sessions based on paragraphs in one session of a course.

2. Related Works

2.1 Word Embeddings

The vectors that are used to represent words are called word embeddings. Word embedding is one of the most popular representation of document vocabulary. It is capable of capturing context of a word in a document, semantic and syntactic similarity, relation with other words, etc. Word embedding is a kind of vector representation of a particular word which make deep net understand for training.

Bag-of-words (BOW) model is a Simple and straightforward method to represent text as vectors. The model is firstly used in information retrieval field and achieved decent results in image processing area and Automatic Speech Recognition recently. However, BOW model ignores the order of words not to mention the context among sentences or paragraphs which is essential in processing text.

2.2 Word2Vec

Word2Vec, developed by Tomas Mikolov [2], is one of the most popular technique using shallow neural network to learn word embeddings.

In order to have words with similar context taking close spatial positions, Word2Vec is a method to construct such an embedding, which includes two models – Skip Gram and the Continuous Bag of Words (CBOW).

CBOW is a model to put the context of each word as the input to predict the relevant word. Specifically, a one-hot encoded vector of word as the input pass through the CBOW and measure the output error compared to one-hot encoded vector of target word so that we learn the vector representation of the target word in the process of predicting the target word.

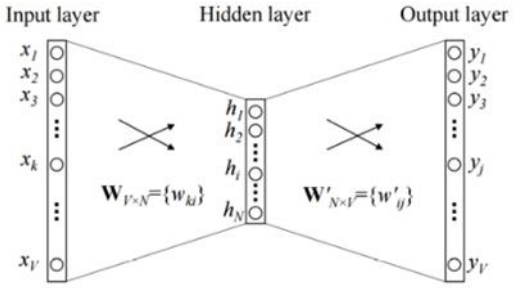

Figure 1. A simple CBOW model with only one word in the context. (Source: stokastik.in)

ISSN: 2414-1895 DOI: 10.6919/ICJE.202003_6(3).0028

CBOW can generate word representations using the context words, i.e., it use the context words to predict target words. Similarly, the method called Skip Gram model which is another variant of Word2Vec can also use the target word to predict the context.

To sum up, both of the models which use backpropagation to learn have their advantages and disadvantages. For more frequent words, CBOW is faster and has better representations. However, Skip Gram works well with small amount of data and is found to represent rare words well.

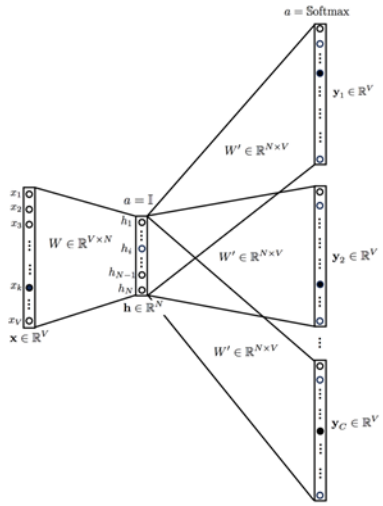

Figure 2. A Skip-Gram model with one target word as input and C context words as output. (Source: stokastik.in)

2.3 RNN and LSTM

Recurrent Neural Networks (RNNs) [3] are popular models that have shown great promise in many Natural Language Processing (NLP) tasks. In contrast to some traditional neural networks, RNNs perform the same task for every element of a sequence, the outputs are depended on the previous computations and that is why recurrent word inside. In another way to consider it, RNNs have a "memory" that could collect information of what has been calculated before. For example, if a sentence being applied into RNN is a 7 words sentence, then the network should be unfolded into a 7-layer neural network, i.e. one word for each layer.

However, Learning with recurrent networks can be very difficult because of their long range dependencies. When backpropagation makes errors across many time steps, the problem of exploding and vanishing gradients occur. In Long short-term memory (LSTM) [4], a unit of computation called memory cell replaces traditional nodes in the hidden layer of a network, which could decrease the difficulties with training encountered by earlier recurrent network.

The core concepts of LSTM are its various gates and cell state. Cell state is considered as the "memory" of the network, which transfers relative information all the way down the sequence chain. So LSTM could reduce the effects of short-term memory like traditional RNN due to the information from earlier time steps making their way to later time steps. Information will be added or removed by gates in the cell state as the cell state is working. During the training, Gates that decide which information is all owed in the cell state can learn what information is relevant to keeping or forgetting.

3. Distributed Context Paragraph Vectors LSTM

As for the particularity of OER courses, we proposed a model called *Distributed Context Paragraph Vectors LSTM* that adapt to the Open Educational Resources text corpus, so that the relationship between paragraphs in the corpus could be learnt. Through the model, the most relevant sessions from any text paragraph from courses could be predicted. The core steps are shown in the following figure.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202003_6(3).0028

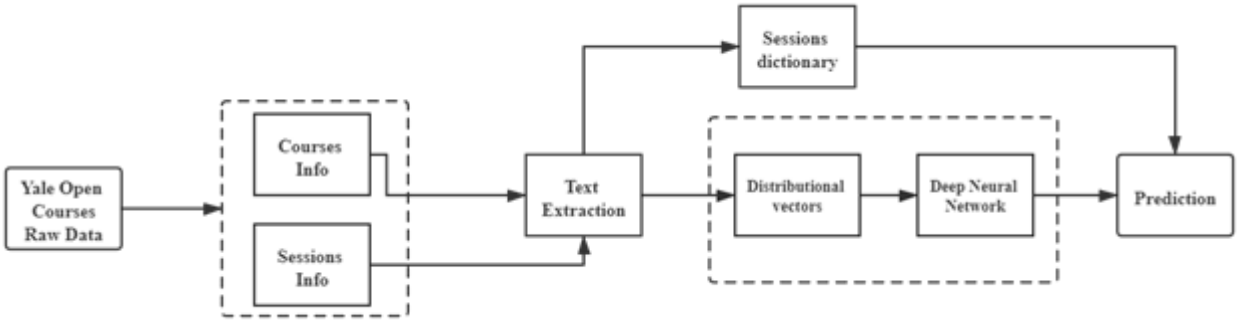

Figure 3. The process figure of the Distributed Context Paragraph Vectors LSTM

The main process of the model is: Firstly, for the data collected from Yale Open Courses, we kept the most necessary information for the model and organized them in two parts: *Courses info* and *Sessions info*. Secondly, we extracted the transcript data from *Sessions info*, separated it into paragraphs with labels from *Courses info* and save them as *Sessions dictionary*. Thirdly, the text was transformed into *Distributional vectors* by Doc2vec as the input of the LSTM model and got the output of the model. Finally, combining *Sessions dictionary* and the output, the most relevant sessions were generated.

3.1 Data Source

The data source is from Yale Open Courses which "provides free and open access to a selection of introductory courses taught by distinguished teachers and scholars at Yale University" and the aim of the project is to "expand access to educational materials for all who wish to learn." The data are combined in many courses which could be collected in several features. The structure of the data is explained in the following table:

Table 1. Courses' structure in Yale Open Courses

Features	Contents
Descriptions	Course number, About the course, Course materials
Syllabus	/
Sessions	Lectures (audio, videos, transcripts, other resources), Assignments, Exams
Suggested books	/
Other materials (problem sets)	/

The table shows the essential features and contents that represent each course. Sessions in courses are those lectures including English transcripts which are important for building our model and sessions prediction.

3.2 Distributed Context Paragraph Vectors LSTM

In the model, we combined two advanced methods together to make the model make high performance. Based on Word2Vec, Doc2Vec is a great technique to use, giving good results. The goal of Doc2Vec is to generate a vector of a document i.e. numeric representation. However, documents don't have a common logical structure such as words. Therefore, using the Word2Vec model, Doc2Vec adds another vector – Paragraph ID. [5]

ISSN: 2414-1895 DOI: 10.6919/ICJE.202003_6(3).0028

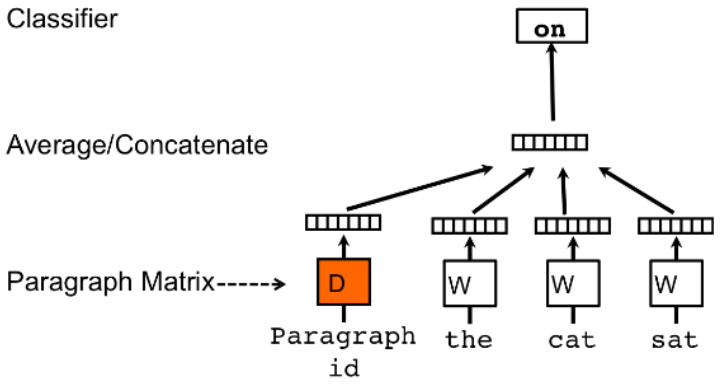

Figure 4. The paragraph id indicates the missing information from the current context and could act as a memory of the generality of the paragraph

As for LSTM, every standard LSTM model has time step t with its corresponding input sequence $X = \{x_1, x_2, x_3 \ldots x_t\}$, input gate i_t , forget gate f_t and output gate o_t [6]. The memory cell c_t could choose data to be memorized or forgotten by controlling those gates.

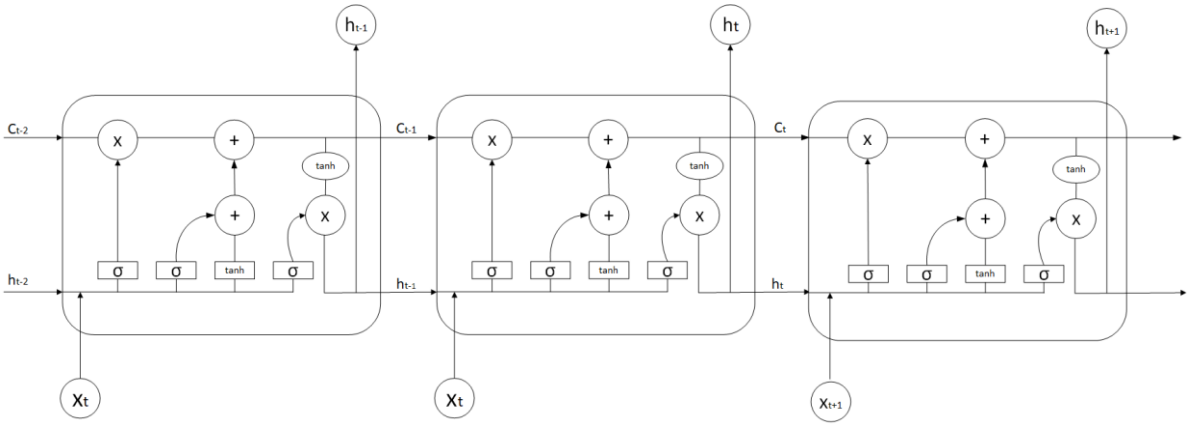

Figure 5. The LSTM structure when time step = t, circles including operators represent pointwise operation, rectangles represent neural network layer.

$$i_t = \sigma(W_i x_t + U_i h_t) \tag{1}$$

$$f_t = \sigma(W_f x_t + U_f h_t) \tag{2}$$

$$o_t = \sigma(W_o x_t + U_o h_t) \tag{3}$$

$$\hat{c}_t = \tanh(W_c x_t + U_c h_t) \tag{4}$$

Unit j, for time step t, memory unit c_t^j follows the formula:

$$c_t^j = i_t^j \square \hat{c}_t + f_t^j \square c_{t-1}^j \tag{5}$$

When the memory cell is updated, current hidden layer h_t^j is also calculated:

$$h_t^j = o_t^j \square \tanh(c_t^j) \tag{6}$$

For the formulas above, W is the weight matrix for the input, U is internal state transition matrix, σ is sigmoid function, $\tanh$ is the hyperbolic tangent function, h_t is the hidden state vector of the output, $\hat{c}_t$ is new content after adjustment and update, $\square$ means dot multiplication. Input gate, forget gate and output gate control information inputting and outputting memory cells. The input gate adjusts the new information inputting a memory cell; the forget gate decides how much information should be

ISSN: 2414-1895 DOI: 10.6919/ICJE.202003_6(3).0028

saved; the output gate could determine the output information. The gate structure of LSTM makes the information on the series form a balanced long-term and short-term dependence.

4. Experiment and Results Analysis

4.1 Data Collection

As for the collection of information from the data source, we use a crawler script generated by Python which is an efficient language to analyze and crawl elements from webpages. After running the crawler script files, there are two .csv files generated: Courses.csv and Sessions.csv. One file includes the information of Courses, and the other includes the information of sessions (lectures, exams and so on that belong to each course).

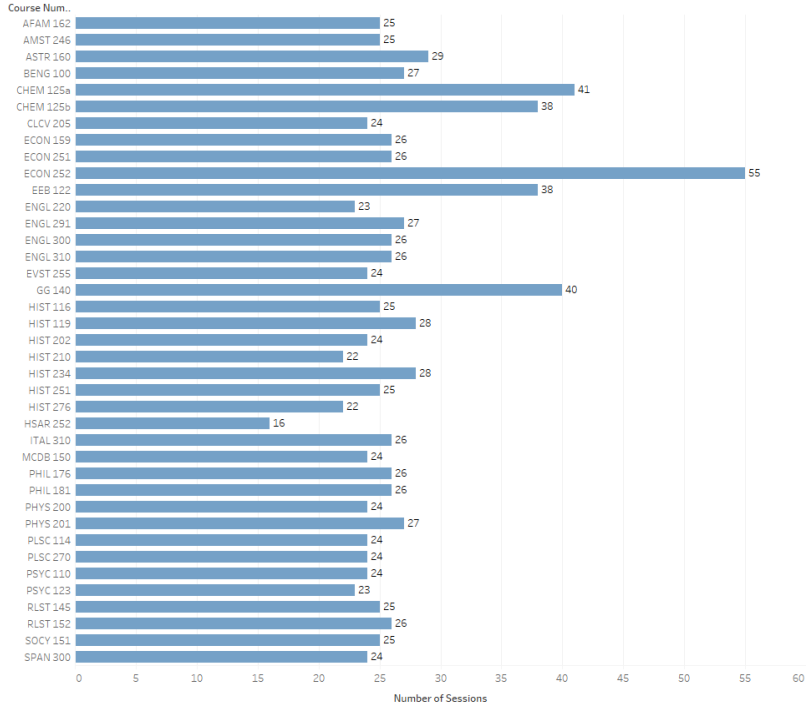

Figure 6. The relationship between courses and sessions. The figure shows the number of sessions in each course.

If we regard the number of sessions as a measure to decide which are big courses and which are small courses, and we can see that the biggest course is "ECON252" which has 55 sessions and "HSAR 252", holding 16 sessions, is the smallest course of the 40 courses.

In the whole dataset, there are total 40 courses and 1058 sessions that includes lectures, updates, exams and so on. We chose *Transcription* column, extracting corresponding text and storing them seperately in each session in order to start word embeddings.

Besides, considering the effect of LSTM model training ,we processed the dataset and generated training set and test set as the ratio 4:1. After data processing, the two training sets have 800 records each and the two test sets have 217 records each.

4.2 Evaluation Criteria

In this paper, the model is evaluated by the *cosine similarity* [7] which measures the difference between two individuals. It is known to us that for two different vectors, if the angle between them is smaller, then the two vectors are more similar which means that the text which represents two vectors is similar. Based on the common sense, cosine similarity measures the similarity value between vectors by calculating the cosine of the angle between two vectors.

The calculation formula for cosine similarity is:

ISSN: 2414-1895 DOI: 10.6919/ICJE.202003_6(3).0028

$$\cos(\theta) = \frac{\sum_{i=1}^{n}(x_i \times y_i)}{\sqrt{\sum_{i=1}^{n}(x_i)^2} \times \sqrt{\sum_{i=1}^{n}(y_i)^2}} \tag{7}$$

In the formula, x_i and y_i are two vectors representing two different paragraphs. $\cos(\theta)$ indicates the similarity between two vectors, as the value is close to 1, the two vectors have higher degree of similarity.

In our model, we decide to use *matching matrix*, a table that is often used to describe the performance of an unsupervised model. In the row of the matrix, it shows every lecture of all the courses, and is the same as the column. The value of each cell in the matrix is the cosine similarity between two vectors.

$$S_{m\times n} = \begin{bmatrix} S_{11} & S_{12} & \cdots & S_{1n} \\ S_{21} & S_{22} & \cdots & S_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ S_{m1} & S_{m2} & \cdots & S_{mn} \end{bmatrix} \tag{8}$$

In the matrix $S_{m\times n}$, S_{ij} indicates different values of cosine similarity, which means that the value represents the similarity between two vectors.

4.3 Result Analysis

After data processing, word embeddings, model building and training, we calculated cosine similarity between every two paragraphs and generated the matching matrix as the result of the model prediction. For intuitive understanding, we visualized the matrix as a figure, of which the color reflects the value of cosine similarity and two axes shows the same records of the dataset. The records in the dataset are all arranged orderly according to normal learning sequence.

In the following figure, the values of the cosine similarity of 217 records in the test set have been calculated before being trained.

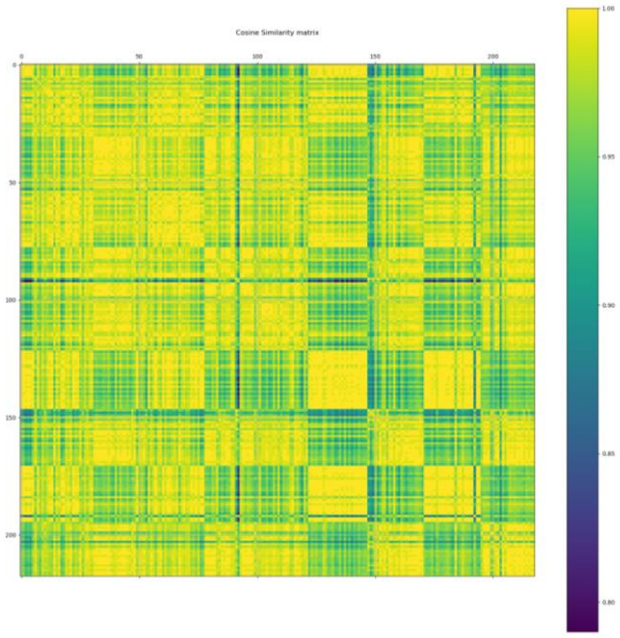

Figure 7. The visualization of the matching matrix before the test set being predicted by the model.

There is no doubt that for diagonal in the figure has the highest cosine similarity equal to 1. However, it seems that most two of the vectors have high similarity since the figure is full of yellow color which

ISSN: 2414-1895 DOI: 10.6919/ICJE.202003_6(3).0028

didn't indicate any valuable information. In other words, before put into the model, sessions each other are difficult to distinguish, the relationship between different sessions, i.e. Context, has not been learnt.

After trained by the Distributed Context Paragraph Vectors LSTM model, we calculated the cosine similarity of the test set and generated the matching matrix.

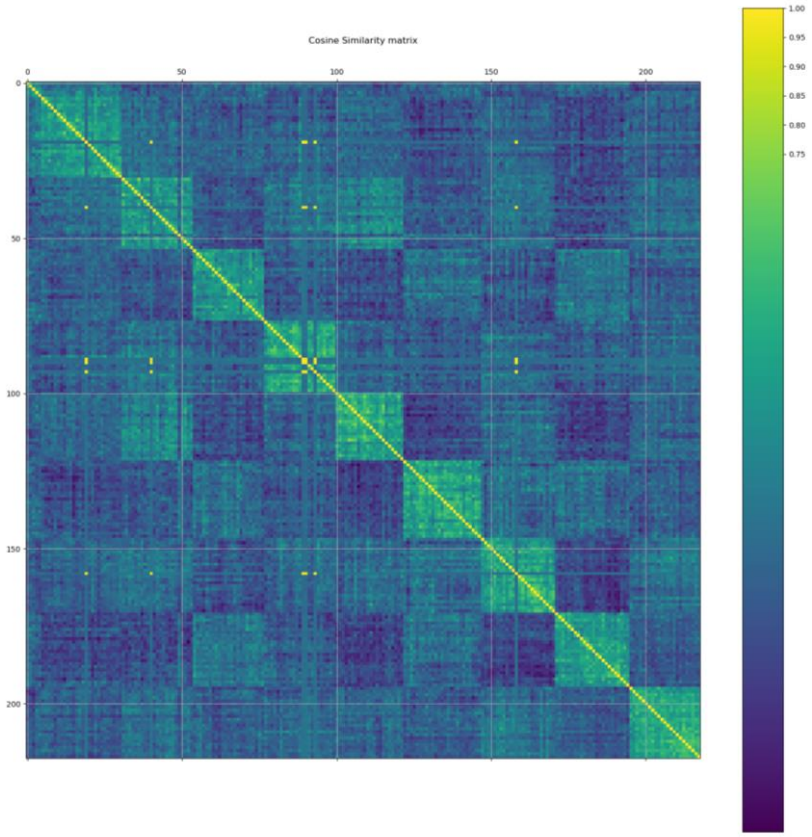

Figure 8. The visualization of the matching matrix for the test set after trained by the model.

In the figure above, the diagonal in the figure has the highest cosine similarity equal to 1 which is stable comparing to last figure. However, the we can also see that some blocks around diagonal has lighter color and they have obvious differences compared with other area in the figure, which means the context of the sessions is considered in our model and it is completely different with Figure 7. We could infer that 9 yellow blocks on the diagonal represent the 9 courses in the dataset. Each session has high similarities with other sessions in the same course.

Compared with Figure 7 and Figure 8, the model could consider the context of sessions and make sessions learn it, which shows the decent performance in the matching matrix.

5. Conclusion

In this paper, we proposed the Distributed Context Paragraph Vectors LSTM model and only used text transcriptions from OER courses to learn the context of sessions each other in each course. Through our model, two adjacent sessions each other have higher similarities. It is instrumental to predict relevant sessions using text paragraphs. The model shows high performance on OER courses dataset in the experiment. The model also provide ideas for sessions' prediction and recommendation so that a learning path could be generated for a learner studying on the OER courses.

References

[1] Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011). Natural language processing (almost) from scratch. Journal of machine learning research, 12(Aug), 2493-2537.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202003_6(3).0028

[2] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In Advances in neural information processing systems (pp. 3111-3119).

[3] Williams, Ronald J.; Hinton, Geoffrey E.; Rumelhart, David E. (October 1986). "Learning representations by back-propagating errors". Nature. 323 (6088): 533–536.

[4] Gers, F. A., Schmidhuber, J., & Cummins, F. (1999). Learning to forget: Continual prediction with LSTM.

[5] Le, Q., & Mikolov, T. (2014, January). Distributed representations of sentences and documents. In International conference on machine learning (pp. 1188-1196).Q. D. Zeng, Q. E. Li: Progress in Civil Engineering, Vol. 32 (2012) No. 9, p. 3077-3080.

[6] Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. Advances in NIPS.

[7] Nguyen, H. V., & Bai, L. (2010, November). Cosine similarity metric learning for face verification. In Asian conference on computer vision (pp. 709-720). Springer, Berlin, Heidelberg.