Unsupervised Embedding for Latent Similarity by Modeling Heterogeneous MOOC Data

Zhuoxuan Jiang[1], Shanshan Feng[2], Weizheng Chen[1],
Guangtao Wang[3], and Xiaoming Li[1]

[1]School of EECS, Peking University, China
[2]School of CSE, Nanyang Technological University, Singapore
[3]University of Michigan, Ann Arbor, USA
{jzhx,cwz,lxm}@pku.edu.cn, sfeng003@e.ntu.edu.sg, gtwang@umich.edu

Abstract. Recent years have witnessed the prosperity of Massive Open Online Courses (MOOCs). One important characteristic of MOOCs is that video clips and discussion forum are integrated into a one-stop learning setting. However, discussion forums have been in disorder and chaos due to 'Massive' and lack of efficient management. A technical solution is to associate MOOC forum threads to corresponding video clips, which can be regarded as a problem of representation learning. Traditional textual representation, e.g. Bag-of-words(BOW), do not consider the latent semantics, while recent semantic word embeddings, e.g. Word2vec, do not capture the similarity between documents, i.e. latent similarity. So learning distinguishable textual representation is the key to resolve the problem. In this paper, we propose an effective approach called No-label Sequence Embedding (NOSE) which can capture not only the latent semantics within words and documents, but also the latent similarity. We model multiform MOOC data in a heterogeneous textual network. And we learn the low-dimensional embeddings without labels. Our proposed NOSE owns some advantages, e.g. course-agnostic, and few parameters to tune. Experimental results suggest the learned textual representation can outperform the state-of-the-art unsupervised counterparts in the task of associating forum threads to video clips.

Keywords: unsupervised embedding, latent similarity, heterogeneous, MOOC

1 Introduction

As a new paradigm of online learning environments, Massive Open Online Courses (MOOCs) are rapidly developed in recent years, e.g. Coursera, edX and Udacity. Millions of learners have been benefited from the free and open courses. Compared with traditional online education, an important characteristic of MOOCs is the one-stop learning setting which integrates video clips and a discussion forum. However, due to 'Massive' and lack of efficient management, there are overload and chaos in majority of MOOC forums [21]. Based on our statistic of a MOOC forum as shown in Fig. 1, the various categories include question of

concept understanding, enquiry and advice of course arrangement, feedback, and etc. In order to archive the threads, there are several methods which have limited effect [3], e.g. defining sub-forums in advance empirically according to weeks and asking learners to tag threads. Some machine learning methods have been studied recently, e.g. content-related thread identification [21], confusion classification [1], sentiment classification [15, 20], and so on. But they are developed for specific research problems and cannot be applied for other tasks.

Another idea is to associate threads to corresponding knowledge points, i.e. video clips. The feasibility relies on some MOOCs' natures. Firstly, the pace of a MOOC is consistent to its off-line counterpart, i.e. learning contents are regularly opened to learners by each week [2]. The temporal information can be leveraged as a constraint. Secondly, learning contents are broken down to small video clips, where each one usually lasts about 10 minutes and just contains one piece of knowledge point in most situations. This makes the association result educationally meaningful. As the snapshot showed in Fig. 2, the syllabus (or knowledge points) of a MOOC is usually organized in a two-level structure. The first level corresponds to weeks and the second are in the form of several video clips. As to corresponding forums, by mining rich information, e.g. behavioral log, textual content and social relationship [8], it is probable to fulfil the task.

The association between threads and video clips is profound. Both the learners and teachers can readily seek what they want, such that the learning efficiency and teaching quality can be improved. After completing the association, the result can be used as the prerequisite of subsequential applications, such as knowledge management, learning guidance, and question answering system.

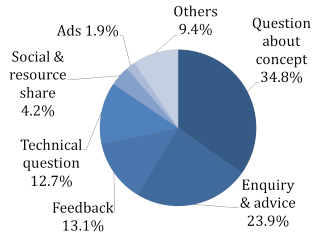


Fig. 1: Typical distribution of threads in a MOOC forum.

Syllabus	Threads (Title only)
Week 1 Introduction	**Thread 1:** Question about incrementing and reducing weights in the binary neural network example (A simple example of learning)
-Why do we need machine learning?	
-What are neural networks?	
-Some simple models of neurons	
-A simple example of learning	**Thread 2:** What is the implementation of Neural Network.
-Three types of learning	
Week 2 The Perceptron learning procedure	
-Types of neural network architectures	**Thread 3:** Why different dimensionality? Training hyperplanes vs. weight vectors
-Perceptrons: The first generation of neural networks	
-A geometrical view of perceptrons	
-Why the learning works?	**Thread 4:** How to understand the weight space?
-What perceptrons can't do?	
.....	.....

Fig. 2: A snapshot of video clips and threads.

The task of association between threads and course video clips, actually, can be regarded as a document ranking problem in the view of information retrieval. An intuitive idea is to use the weighted Bag-of-words (BOW) [16] as features to calculate the similarities between threads and video clips, and then rank them by the computed similarities. However, BOW cannot effectively capture semantic knowledge. Moreover, recent studied semantic word embeddings, e.g. Word2vec [13], cannot capture the similarity of documents, and we call it latent similarity. The latent similarity is crucial to distinguish whether a document can be associated to the right target. In our task, the textual representation is expected to preserve both the latent semantics and latent similarity.

On the other hand, as a kind of real-world data, it is very expensive to obtain labeled training dataset, since associating MOOC threads to video clips requires expertise. So we have to leverage more information to compensate the lack of labels during learning word embeddings.

In this paper, we focus on the technical problem that how to discover latent similarity knowledge from heterogeneous data. There are several challenges to solve this problem: 1) how to leverage characteristics of data to compensate the lack of labels, 2) how to build a model which can integrate heterogeneous information, and 3) how to learn the latent similarity from the new data model.

By our observation, we find four kinds of information helpful: 1) word sequence in threads and posts, 2) word sequence in subtitles of video clips, 3) learners' behavioral sequence of clickstream about videos and threads, and 4) timestamp sequence of video clips (publishing time) and threads (posting time). The reason that they are chosen will be discussed in Section 3.1.

Our basic idea is to first model all the sequential information into a heterogeneous textual network, and then embed the nodes of the network, i.e. words and documents, to low-dimensional vectors which can preserve both latent semantics and latent similarity. At last the learned vectors are used to rank video clips by calculating similarities between threads and them under a constraint of time. Inspired by methods of distributional representation which have been proven successful in a mass of natural language processing tasks, e.g. word analogy [13], text classification [10], sentiment analysis [11] and POS tagging [5], we propose an approach called No-label Sequence Embedding (NOSE).

To summarize, our contributions in this paper include:

- We design a novel embedding model to integrate heterogeneous sequential information, e.g. word sequence and clickstream sequence, in order to learn latent semantics and latent similarity simultaneously.
- We develop a new computing framework to learn the textual representation based on the proposed model.
- We collect two real-world MOOC data and conduct thorough experiments. Results confirm that our proposed approach can effectively learn the latent similarity without labels by modeling heterogeneous sequential information. And our novel textual embeddings can outperform the state-of-the-art counterparts in our task.

2 Related Work

It is a fundamental research question that how to better represent text. Existing approaches can be generally classified into two aspects: unsupervised and supervised. Supervised embeddings, such as deep neural network, need to be fed with labels [9, 12]. However, in the association task, we do not have the label information. Thus, we review related work merely within this field of unsupervised learning textual representation.

Unsupervised word embeddings are usually universal and can be applied to various tasks. The process of learning them is often efficient to scale up to

millions of documents. Other advantages of unsupervised embeddings include no label required and few parameters to tune. However, unsupervised embeddings are lightly under-performed compared with supervised ones in most specific tasks [18]. Among most learning methods of unsupervised embeddings, the information of textual co-occurrence in local context at different levels are leveraged. For example, CBOW and Skip-gram [13] learn word embeddings based on word co-occurrence. Para2vec [10] utilizes word and document co-occurrence to learn their embeddings. Hierarchical Document Vector (HDV) [6] leverages both document co-occurrence and contents to achieve better representation. Latest Predictive Text Embedding (PTE) [18] models text to a uniform heterogeneous network and obtains the state-of-the-art performance on textual classification. However, it leverages labels to guarantee the performance of classification. It cannot be adopted directly to solve the proposed problem in this paper.

Another series of methods for representing text without labels are discussed in the task of Dataless Classification [4,17]. These methods commonly require large-scale world knowledge, e.g. Wiki data, to extract textual features. But world knowledge is hard to obtain and process in many cases. Also similar to BOW and Word2vec, the latent similarity still cannot be embodied in the embeddings. By the way, this is also a classification problem, which is different from ours.

3 No-label Sequence Embedding and Ranking

In this section, we introduce our approach for learning word embeddings and document ranking. Firstly, we introduce the motivation to extract the information of latent similarity from four kinds of sequential data. Then we state the data model to integrate heterogeneous information. Then the method of embedding the heterogeneous network to low-dimensional vectors is described. At last we introduce the algorithm to rank documents within a constraint of timestamps sequence. We can place the problem definition after we have introduced the data. Otherwise, people may not understand the definition.

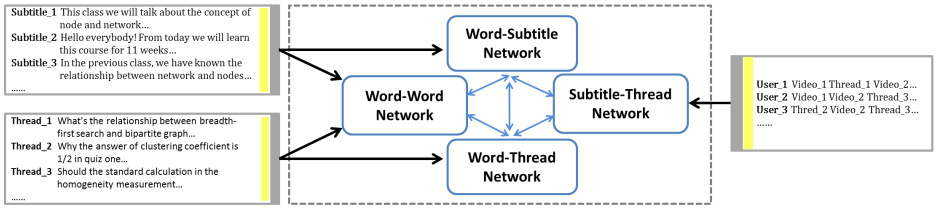
3.1 Data Model

In order to compensate the missing of labels, the data model should integrate as much as information. Besides, it also should capture the information of latent similarity. Based on our observation of multiform MOOC data, subtitles of video clips and contents of threads are essential for associating them. The subtitles of video clips are well-organized and formal, since they are generated by instructors. While the contents of threads are written by various learners, thus they are colloquial and informal. We should separately learn their embeddings because this way can preserve their linguistic peculiarity, i.e. latent similarity. This intuition is also confirmed by our experimental analysis.

In addition, we find learners' clickstream logs, that is records of watching videos, reading threads and posting threads in chronological order, indicate that

the adjacent entities, especially a video clip and a thread, are semantically relevant. An intuitive explanation is that a learner may jump between videos and threads either he wants to seek some further discussion when he is watching a video, or he wants to review the relevant video contents when he is reading a thread. So behavioral logs contain the latent similarity.

In order to integrate all these different sources of text, we design a heterogeneous network which can facilitate learning embeddings both separately and cooperatively. We leverage the co-occurrence of different textual levels and construct four sub-networks which comprise a comprehensive heterogeneous network as Fig. 3 shows.



(a) Raw text without labels (b) Heterogeneous Network (c) Clickstream records

Fig. 3: Data model constructed from raw sequential MOOC data.

Word-Word Network. A word-word network is based on word co-occurrence in word sequence and denoted as $\mathcal{G}_{ww} = (\mathcal{V}, \mathcal{E}_{ww})$. The $\mathcal{V}$ is the word vocabulary and $\mathcal{E}_{ww}$ is the edge between words. The edge weight w_{ij} is defined as the co-occurrence times of words v_i and v_j within a fixed-size window. Word co-occurrence is basic information to capture semantics between words.

Word-Subtitle Network. A word-subtitle network is based on word sequence in document level contexts and denoted as $\mathcal{G}_{ws} = (\mathcal{V} \cup \mathcal{S}, \mathcal{E}_{ws})$. The $\mathcal{V}$ is the word vocabulary and $\mathcal{S}$ is the set of subtitles. $\mathcal{E}_{ws}$ is the edge between words and subtitles. The edge weight w_{ij} is defined as the occurrence times of word v_i in subtitle s_j. Word-document co-occurrence is to capture semantics between words and documents.

Word-Thread Network. Similar to word-subtitle network, a word-thread network is also based on word sequence in document level contexts and denoted as $\mathcal{G}_{wt} = (\mathcal{V} \cup \mathcal{T}, \mathcal{E}_{wt})$. The only difference is that $\mathcal{T}$ is the set of threads and $\mathcal{E}_{wt}$ is the edge between words and threads. We treat them as two different networks due to their different latent linguistic styles as mentioned before.

Subtitle-Thread Network. A subtitle-thread network is based on document sequence in local context of learners' clickstream records and denoted as $\mathcal{G}_{st} = (\mathcal{S} \cup \mathcal{T}, \mathcal{E}_{st})$. The $\mathcal{S}$ is the set of subtitles and $\mathcal{T}$ is the set of threads. $\mathcal{E}_{st}$ is the edge between subtitles and threads. The edge weight w_{ij} is defined as the co-occurrence times of subtitle s_i and thread t_j in learners' clickstream logs within a window size. This network may capture knowledge of both latent semantics and latent similarity between subtitles and threads.

Heterogeneous Network. Combining all the four sub-networks via common nodes, we can get a heterogeneous network. Note that we only study the heterogeneous network with four sub-network by considering the latent semantics information. The heterogeneous network can be extended by adding more sub-networks.

Based on the heterogeneous networks, we define the No-label Sequence Embedding problem as follows.

No-label Sequence Embedding Given multiform sequential data, e.g. raw behavioral sequence and textual sequence, the problem of No-label Sequence Embedding is to construct a data model from all the data and unsupervisedly learn low-dimensional word embeddings which can preserve both latent semantics and latent similarity.

3.2 Learning Embedding

Except the word-word network, the other sub-networks are all directive graphs. By replacing the edges with bidirectional ones in the word-word network, all the four sub-networks can be treat as bipartite networks. Then we decompose the task of heterogeneous network embedding to four sub-tasks of bipartite network embedding.

There are several network embedding, such as DeepWalk [14],LINE [19] and node2vec [7]. However, all of them are proposed for homogeneous network, and cannot be simply used for the heterogeneous network. Inspired by LINE, we develop a new algorithm to learn the embedding of heterogeneous networks.

Bipartite Network Embedding Given an arbitrary bipartite network $\mathcal{G} = (\mathcal{V}_A \cup \mathcal{V}_B, \mathcal{E})$, where $\mathcal{V}_A$ and $\mathcal{V}_B$ represent two separate sets of nodes and $\mathcal{E}$ is the set of edges between nodes. The conditional probability of observing node v_i in set $\mathcal{V}_A$ given the node v_j in set $\mathcal{V}_B$ is defined as:

$$p(v_i|v_j) = \frac{\exp(\boldsymbol{u}_i^T \cdot \boldsymbol{u}_j)}{\sum_{i' \in A} \exp(\boldsymbol{u}_{i'}^T \cdot \boldsymbol{u}_j)} \tag{1}$$

where $\boldsymbol{u}_i$ is the embedding vector of node v_i in $\mathcal{V}_A$, and $\boldsymbol{u}_j$ is the embedding vector of node v_j in $\mathcal{V}_B$. Equation (1) actually computes the conditional probability of any v_j in $\mathcal{V}_\mathcal{B}$ over all nodes in $\mathcal{V}_A$. Thus the second-order proximity can be preserved if let $p = (\cdot|v_j)$ close to its empirical distribution $\hat{p} = (\cdot|v_j)$. So the objective function to be minimize is:

$$O = \sum_{j \in B} \lambda_j d(\hat{p}(\cdot|v_j), p(\cdot|v_j)) \tag{2}$$

where $d(\cdot, \cdot)$ is the KL-divergence between two probability distribution, λ_j is the importance of vertex v_j in the network and can be defined as the degree $degree_j = \sum_i w_{ij}$, and the empirical distribution can be defined as $\hat{p}(v_i|v_j) = \frac{w_{ij}}{degree_j}$. Equation (2) can be simplified by removing some constants as:

$$O = -\sum_{(i,j)\in\mathcal{E}} w_{ij} \log p(v_j|v_i) \tag{3}$$

To optimize (3), we utilize stochastic gradient descent method of edge sampling [19] and negative sampling [13] which are fast and efficient, instead of computing the summation of the entire set of nodes in $\mathcal{V}_A$. Firstly a positive sample is randomly selected based on probability proportional to its weight w_{ij}, and then a fixed size of negative samples are selected based on a noise distribution $p_n(j)$. By using negative sampling, Equation (3) is in detail replaced by:

$$O = \sum_{(i,j)\in\mathcal{E}} \left\{ \log \sigma(\boldsymbol{u}_i^T \cdot \boldsymbol{u}_j) + \sum_{j=1}^{K} E_{v_n\sim P_n(v)}[\log \sigma(-\boldsymbol{u}_n^T \cdot \boldsymbol{u}_j)] \right\} \tag{4}$$

where $\sigma(x) = 1/(1+\exp(-x))$ is the sigmoid function; the first term means positive samples and the second term means negative samples of which K is the fixed size. Noise distribution $P_n(v) \propto d_v^{3/4}$ is the same as in [13] where d_v is the out degree of node v.

The reason for adopting the edge sampling is to reduce the large variance of edge weights, because very-large-weight edges may dominate the optimization process. An alternative idea is to duplicate the edges proportional to their weights. However, this will deteriorate the efficiency greatly due to the great increase of edges.

Heterogeneous Network Embedding In order to learn the word embeddings of four sub-networks separately and cooperatively, a simple strategy is that each time one bipartite network is selected to learn eambeddings and the intermediate results are transited to other bipartite networks. Although subtitle ids and thread ids are two different kinds of nodes, for simplicity, we regard them as the same one since there is no overlapping between them. Thus, in total we set two kinds of nodes, document id (d) and words (v). They are shared between bipartite networks. The whole objective function is:

$$O_{NOSE} = O_{ww} + O_{ws} + O_{wt} + O_{st} \tag{5}$$

where

$$O_{ww} = -\sum_{(i,j)\in E_{ww}} w_{ij} \log p(v_i|v_j) \tag{6}$$

$$O_{ws} = -\sum_{(i,j)\in E_{ws}} w_{ij} \log p(v_i|d_j) \tag{7}$$

$$O_{wt} = -\sum_{(i,j)\in E_{wt}} w_{ij} \log p(v_i|d_j) \tag{8}$$

$$O_{st} = -\sum_{(i,j)\in E_{st}} w_{ij} \log p(d_i|d_j) \tag{9}$$

Although we can assign weights to each bipartite network for proportionally selecting edges to learn each once, we choose to equally and sequentially learn embeddings for simplicity. As shown in Algorithm 1, the algorithm is very efficient because the time complexity is just $O(KN)$ and K is a small number.

3.3 Document Ranking

After learning word embeddings, we simply use them to calculate the similarity between threads and subtitles with metric of Cosine distance. Considering the trait of MOOC data that the time of threads posted are usually behind their corresponding video clips. So we have a constraint to make the ranking more reasonable. The algorithm detail is shown in Algorithm 2. The reason for why we do not integrate the subtitle sequence and thread sequence based on timestamps into the heterogeneous network is that we believe they may contain little semantics between any two entities.

Algorithm 1: Learning embeddings of heterogeneous network

INPUT: $\mathcal{G}_{ww}$, $\mathcal{G}_{ws}$, $\mathcal{G}_{wt}$, $\mathcal{G}_{st}$, number of samples N, number of negative samples K
OUTPUT: word embeddings $\boldsymbol{w}$
while $iteration \leq N$:
  sample an edge e_{ij} and K negative edges from E_{ww}, update $\boldsymbol{u}_{v_i}$ and $\boldsymbol{u}_{v_j}$
  sample an edge e_{ij} and K negative edges from E_{ws}, update $\boldsymbol{u}_{d_i}$ and $\boldsymbol{u}_{v_j}$
  sample an edge e_{ij} and K negative edges from E_{wt}, update $\boldsymbol{u}_{d_i}$ and $\boldsymbol{u}_{v_j}$
  sample an edge e_{ij} and K negative edges from E_{st}, update $\boldsymbol{u}_{d_i}$ and $\boldsymbol{u}_{d_j}$
return word embeddings

Algorithm 2: Document Ranking

INPUT: thread embeddings T, subtitle embeddings S, timestamps $Time$
OUTPUT: rankings of subtitles for each thread L
for t_i in T:
  for s_j in S
    if $Time_{t_i} > Time_{s_j}$:
      $L_{t_i}.add(s_j, \cos(t_i, s_j))$
  $L_{t_i}.rank()$
return L

4 Experiment

4.1 Data sets

We collect the sequential data of two MOOCs from Coursera[1] and China University MOOC[2] respectively. The former is an interdiscipline called *People and Network* which involves computer science, social science and economics, while the other is a conceptual course called *Introduction of MOOC* which introduces a new concept. From both course, we collect subtitles of video clips, forum contents, learners' sequential clickstream log and timestamps sequence. Since our textual data is mostly in Chinese, we preprocess the raw data with a word segment tool consistently before evaluating our method.

[1] https://www.coursera.org, which is an educational technology company that offers MOOCs worldwide.
[2] http://www.icourse163.org, which is a leading MOOCs platform in China. Supported by Ministry of Education of the People's Republic of China and NetEase, Inc.

As to evaluation, we invite the course TAs to tag some threads in advance. Due to the chaos of MOOC forum threads as mentioned before, we discard the threads about technical operation, social contact, advice to the course, thoughts and others that are irrelevant to course contents. At a result we have 103 valid threads of *People and Network* and 254 valid threads of *Introduction to MOOC* in total. They are our whole available test samples. In some situations, there may be more than one topics being asked in an initial post, we allow to tag the test sample with the two most possible labels. The numbers of double-labeled test samples in each course are 8/103 and 119/254 respectively. Table 1 shows the course information.

Table 1: Statistics of two MOOC datasets.

Course Name	#users	#video clips	#threads	#posts
People and Network	10,807	60	219	1,206
Introduction to MOOC	3,949	19	557	7,177

4.2 Baselines and Parameters

We only compare our embeddings with unsupervised rivals. Consistently, the representation of subtitles is set as the average of all the word vectors which belong to that document. While the representation of threads only use the word vectors which belong to the title and the initial post of a thread. However, all posts are kicked in for learning word embeddings.

- BOW: the classical text representation which firstly builds a vocabulary V with the whole textual contents and then each document is represent as $|V|$-dimensional vector like a bag of words. We remove stop words from the vocabulary and TFIDF weights are set to each dimension.
- CBOW: the state-of-the-art word embeddings proposed by [13] which uses context to predict the target word embedding.
- Skip-gram: another version of the state-of-the-art word embeddings proposed by [13] which uses the target word to predict its context.
- Pare2vec: the state-of-the-art word embeddings which considers the document-level context information [10].
- LINE: the large-scale information network embeddings which can be used to textual network, but only homogeneous network. Here we simply combine different kinds of bipartite networks to one and see how LINE performs without separately treating them.
- NOSE: our proposed no-label sequence embeddings which leverages all information. We also try to remove one sub-network ordinally to see their contribution degree.

The dimension of word vectors is empirically set as 100. In CBOW, Skip-gram, Para2vec, the window sizes are all set as 5. In LINE and NOSE, the number of negative samples are also set as 5, while the window size used in constructing $\mathcal{G}_{st}$ is set as 3 because this can get best performance shown in later experiments. Especially, in LINE and NOSE, the total number of edge samples, N, is set 50 million since we find larger number may cause over-fitting.

4.3 Results and Analysis

As a ranking result, the measure metric we use is precision averaged by 10 times of runs. From Table 2, we can find traditional sematic word embeddings, CBOW, Skip-gram and Para2vec, are comparable to BOW in terms of precision @1. Separate learning the four sub-networks plays a crucial role since NOSE performs better than LINE. This suggests latent similarity is captured by our learning approach. Among the four sub-networks, $\mathcal{G}_{st}$ contributes the most while $\mathcal{G}_{ww}$ degrades the performance both in the series of LINE and NOSE, confirming that $\mathcal{G}_{st}$ contains the latent similarity while $\mathcal{G}_{ww}$ only contains latent semantics. Building a data model in such a form of textual network may also strengthen the distinctive relationship between documents, comparing LINE and NOSE with CBOW, Skip-gram and Para2vec, indicating that the network can capture both information of word-word semantics and word-document semantics simultaneously. Although BOW is competitive on results of @3 and @5, we concern more about precision @1 because our task is to associate threads to videos rather than a real search problem. We find NOSE($\mathcal{G}_{ws}+\mathcal{G}_{wt}+\mathcal{G}_{st}$) algorithm can achieve the best result @1. Table 3 shows that the constraint of time plays another crucial role during document ranking.

Table 2: Ranking precision result of two MOOC datasets.

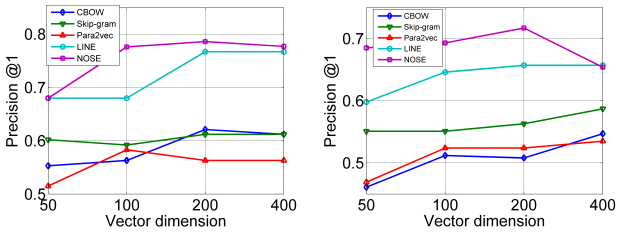
Algorithm	*People and Network*			*Introduction to MOOC*		
	@1	@3	@5	@1	@3	@5
BOW	0.583	0.845	**0.903**	0.449	0.811	**0.906**
CBOW	0.563	0.718	0.786	0.512	0.697	0.795
Skip-gram	0.592	0.738	0.786	0.551	0.744	0.866
Para2vec	0.583	0.670	0.777	0.524	0.713	0.827
LINE($\mathcal{G}_{ws}+\mathcal{G}_{wt}$)	0.621	0.845	0.883	0.535	0.807	0.898
LINE($\mathcal{G}_{ws}+\mathcal{G}_{wt}+\mathcal{G}_{ww}$)	0.592	0.786	0.854	0.394	0.728	0.846
LINE($\mathcal{G}_{ws}+\mathcal{G}_{wt}+\mathcal{G}_{st}$)	0.680	0.825	0.883	0.646	0.799	0.902
LINE(all)	0.650	0.767	0.845	0.406	0.728	0.862
NOSE($\mathcal{G}_{ws}+\mathcal{G}_{wt}$)	0.738	0.805	0.874	0.654	0.803	0.890
NOSE($\mathcal{G}_{ws}+\mathcal{G}_{wt}+\mathcal{G}_{ww}$)	0.699	0.835	0.874	0.657	**0.827**	0.886
NOSE($\mathcal{G}_{ws}+\mathcal{G}_{wt}+\mathcal{G}_{st}$)	**0.776**	**0.845**	0.883	**0.693**	0.803	0.870
NOSE(all)	0.767	0.825	0.874	0.685	0.803	0.874

Table 3: Ranking precision @1 result without time constrain by NOSE.

Algorithm	*People and Network*	*Introduction to MOOC*
NOSE($\mathcal{G}_{ws}+\mathcal{G}_{wt}$)	0.631	0.520
NOSE($\mathcal{G}_{ws}+\mathcal{G}_{wt}+\mathcal{G}_{ww}$)	0.621	0.512
NOSE($\mathcal{G}_{ws}+\mathcal{G}_{wt}+\mathcal{G}_{st}$)	0.670	0.567
NOSE(all)	0.641	0.547

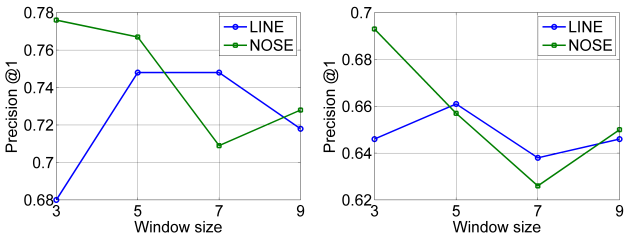
Parameter Sensitivity In this part, LINE and NOSE utilize information of $\mathcal{G}_{ws}$, $\mathcal{G}_{wt}$ and $\mathcal{G}_{st}$ consistently. Fig. 4 shows the results of different vector dimensions used during learning word embeddings. We find NOSE is better than

the others with various number of dimensions. Considering the trade-off between learning efficiency and precision, number of dimensions is set as 100. Fig. 5 shows the results of different window sizes used when constructing $\mathcal{G}_{st}$. We set window sizes as 3, 5, 7 and 9, and find NOSE performs best with size of 3 while LINE is best with 5. This result may stem from the guess that small local context already can capture the latent similarity during separately learning embeddings, while LINE needs larger contexts.



(a) People and Network (b) Introduction to MOOC

Fig. 4: Precision @1 of different vector dimensions. $\mathcal{G}_{ws}$, $\mathcal{G}_{wt}$ and $\mathcal{G}_{st}$ are utilized.



(a) People and Network (b) Introduction to MOOC

Fig. 5: Precision @1 of different window sizes for constructing $\mathcal{G}_{st}$.

5 Conclusion

Archiving forum threads to video clips is meaningful both for instructors and learners. This task is technically regarded as a problem of document ranking. In order to solve the problem, we propose an approach to learn latent semantics and latent similarity simultaneously. Our approach can also overcome the constraints of no label and heterogeneous data, which often happen in real-world datasets. The experimental results perform well and confirm the effectiveness of our approach for learning latent similarity. Last but not the least, there is still room to improve the precision by leveraging more effective information, e.g. in the view of instructor's behavior.

Acknowledgments. This research is supported by the National Research Foundation, Prime Ministers Office, Singapore under its IDM Futures Funding Initiative, China NSFC with Grant No.61532001 and No.61472013, and China MOE-RCOE with Grant No.2016ZD201. We thank the anonymous reviewers for their insightful comments.

References

1. Agrawal, A., Venkatraman, J., Leonard, S., Paepcke, A.: Youedu: Addressing confusion in mooc discussion forums by recommending instructional video clips. In: EDM. pp. 297–304 (2015)
2. Anderson, A., Huttenlocher, D.P., Kleinberg, J.M., Leskovec, J.: Engaging with massive online courses. In: WWW. pp. 687–698 (2014)
3. Anderson, A., Huttenlocher, D.P., Kleinberg, J.M., Leskovec, J.: Language independent analysis and classification of discussion threads in coursera mooc forums. In: IRI. pp. 654–661 (2014)
4. Chang, M.W., Ratinov, L.A., Roth, D., Srikumar, V.: Importance of semantic representation: Dataless classification. In: AAAI. pp. 830–835 (2008)
5. Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P.: Natural language processing (almost) from scratch. Journal of Machine Learning Research 12, 2493–2537 (2011)
6. Djuric, N., Wu, H., Radosavljevic, V., Grbovic, M., Bhamidipati, N.: Hierarchical neural language models for joint representation of streaming documents and their content. In: WWW. pp. 248–255 (2015)
7. Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: KDD. pp. 855–864 (2016)
8. Huang, J., Dasgupta, A., Ghosh, A., Manning, J., Sanders, M.: Superposter behavior in mooc forums. In: L@S. pp. 117–126 (2014)
9. Kim, Y.: Convolutional neural networks for sentence classification. In: EMNLP. pp. 1746–1751 (2014)
10. Le, Q.V., Mikolov, T.: Distributed representations of sentences and documents. In: ICML. pp. 1188–1196 (2014)
11. Mesnil, G., Mikolov, T., Ranzato, M., Bengio, Y.: Ensemble of generative and discriminative techniques for sentiment analysis of movie reviews (2014), arXiv preprint arXiv:1412.5335
12. Mikolov, T., Karafit, M., Burget, L., Cernocký, J., Khudanpur, S.: Recurrent neural network based language model. In: INTERSPEECH. pp. 1045–1048 (2010)
13. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: NIPS. pp. 3111–3119 (2013)
14. Perozzi, B., Al-Rfou', R., Skiena, S.: Deepwalk: Online learning of social representations. In: KDD. pp. 701–710 (2014)
15. Ramesh, A., Kumar, S.H., Foulds, J.R., Getoor, L.: Weakly supervised models of aspect-sentiment for online course discussion forums. In: ACL. pp. 74–83 (2015)
16. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Information Processing & Management 24(5), 513–523 (1988)
17. Song, Y., Roth, D.: On dataless hierarchical text classification. In: AAAI. pp. 1579–1585 (2014)
18. Tang, J., Qu, M., Mei, Q.: Hierarchical neural language models for joint representation of streaming documents and their content. In: KDD. pp. 1165–1174 (2015)
19. Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., Mei, Q.: Line: Large-scale information network embedding. In: WWW. pp. 1067–1077 (2015)
20. Wen, M., Yang, D., Rosé, C.P.: Sentiment analysis in mooc discussion forums: What does it tell us? In: EDM. pp. 130–137 (2014)
21. Wise, A.F., Cui, Y., Vytasek, J.: Bringing order to chaos in mooc discussion forums with content-related thread identification. In: LAK. pp. 188–197 (2016)