Structuring Lectures in Massive Open Online Courses (MOOCs) for Efficient Learning by Linking Similar Sections and Predicting Prerequisites

Sheng-syun Shen¹, Hung-yi Lee¹, Shang-wen Li², Victor Zue², Lin-shan Lee¹

¹Graduate Institute of Communication Engineering, National Taiwan University
²MIT Computer Science and Artificial Intelligence Laboratory

Abstract

The increasing popularity of Massive Open Online Courses (MOOCs) has resulted in huge number of courses available over the Internet. Typically, a learner can type a search query into the look-up window of a MOOC platform and receive a set of course suggestions. But it is difficult for the learner to select lectures out of those suggested courses and learn the desired information efficiently. In this paper, we propose to structure the lectures of the various suggested courses into a map (graph) for each query entered by the learner, indicating the lectures with very similar content and reasonable sequence order of learning. In this way the learner can define his own learning path on the map based on his interests and backgrounds, and learn the desired information from lectures in different courses without too much difficulties in minimum time. We propose a series of approaches for linking lectures of very similar content and predicting the prerequisites for this purpose. Preliminary results show that the proposed approaches have the potential to achieve the above goal.

Index Terms: Massive Open Online Courses (MOOCs)

1. Introduction

The increasing popularity of Massive Open Online Courses (MOOCs) has resulted in huge number of courses available over the Internet under various MOOC platforms such as edX and Coursera. When a learner wishes to learn a certain subject, he can simply type a search query on the look-up window of a MOOC platform and receive a series of course suggestions for him to select. But a course may not cover all the topics important for him. Even if he spends time to take one course, he may still miss some information important for him taught in other courses or lose the global picture for the whole subject he wishes to learn. On the other hand, going through all related courses is impossible for a learner, and especially wasteful when the courses have a good portion of overlap. Also, very often it will be easier to learn a lecture after some other lectures because the content of the former is based on the concepts mentioned in the latter.

With the thoughts mentioned above, it may be much more efficient if the learner can choose from a whole set of lectures collected from different courses, shown on a map (graph) for the global picture of the subject indicating the lectures with overlapped content and reasonable sequence order of learning. Such a graph would pave the roads towards efficient personalized learning, because different learners may select different learning paths over the same graph due to different interests and backgrounds. There are two key technologies necessary for constructing such a graph, i.e., linking lectures with overlapped or very similar content, and predicting prerequisites between lectures so as to give them a good sequence order. This paper presents a series of approaches towards these directions and some initial results obtained. A MOOC interface towards the above goal is currently being developed. It is a prototype system called Cangjie, which organizes the lectures from more than 50 courses on edX and Coursera. The video demonstration of the prototype system can be found at [2]. Although there are already many approaches proposed for helping learners browse on-line courses including retrieving relevant lectures [3–7], key term extraction [5,7], summarizing the audio/video recordings [7,8] and visualizing interaction history [9], they primarily aimed at managing the information from a single course, instead of considering the relationships among the content of lectures of different courses as considered here. Moreover, some previous works proposed for automatically linking objects to help people navigate unfamiliar territory [10,11] and observing prerequisite relations among courses to learn a directed universal concept graph [12,13]. However, constructing a map for the lectures to enhance MOOC learning as in this paper has not been widely studied yet.

2. Structuring Lectures with a Map

Fig. 1(a) is an example of the retrieval results after the learner enters a query for the subject he wishes to learn, including 3 courses $X$, $Y$ and $Z$ each with 5 lectures in sequence order as $\{x_i, i=1,2,...,5\}$ and so on. Fig. 1(b) indicates that the approaches proposed in this paper may find out that lecture 2 of $X$ ($x_2$) has content very similar to that of lecture 3 of $Y$ ($y_3$), while $z_1$ is the prerequisite of $y_2$ and $y_1$ is the prerequisite of $z_5$. This produces the map (or a graph) as shown in Fig. 1(b). A learner may then choose the learning path as shown red in Fig. 1(c), $\{z_1 \rightarrow y_2 \rightarrow (y_1, x_3) \rightarrow y_3 \rightarrow y_4 \rightarrow z_5\}$, so as to learn the selected parts of three courses without too much difficulties in minimum time. The approaches used to link lectures in different courses but with very similar content is introduced in Section 4. The approaches for predicting the prerequisite relationships between lectures is introduced in Section 4. These approaches make it possible to produce a lecture map for each query entered by a learner, so the learner can easily design his personalized learning path accordingly and learn efficiently.

![Figure 1: Structuring lectures in multiple courses with a map](image-url)
3. Linking Lectures with Similar Content

3.1. Individual Pair Similarity

A simple approach is to compute the similarity $S(x_i, y_j)$ between the lectures $x_i$ and $y_j$ for courses $X$ and $Y$ as in Fig. 2(a) based on their audio transcriptions or lecture titles, and then choose the pairs with similarity exceeding a threshold.

The similarity $S(x_i, y_j)$ based on audio transcriptions can be simply obtained by the feature vectors in the vector space model with tf-idf weighting and the cosine similarity. This can also be performed on all the key terms extracted from the course transcriptions, rather than on all words in the lexicon. On the other hand, we can also perform latent topic analysis and obtain a topic distribution vector for a lecture $x$, $\{P(t_k|x), k = 1, ..., T\}$, in which $P(t_k|x)$ is the probability of observing topic $t_k$ from $x$, and $T$ is the number of latent topics. The topic similarity between two lectures $x_i$ and $y_j$ is then,

$$S(x_i, y_j) = \sum_{k=1}^{T} P(t_k|x_i)P(t_k|y_j). \quad (1)$$

Lecture titles usually carry key information so should also be carefully used. In addition to similarly computing $S(x_i, y_j)$ based on the titles just as based on audio transcriptions mentioned above, we can also leverage the grammatical information obtained from the syntactic parsing tree of each title, representing each title by a feature vector of grammatical rules used

$$S(x_i, y_j) = \sum_{k=1}^{|L|} \sum_{s} P(w_s|x_i)P(w_s|y_j). \quad (2)$$

where $|L|$ is the number of grammatical rules. Each dimension is a rule, and it is 1 if used and 0 otherwise.

3.2. Global Lecture Structure Considerations

There always exists sequence orders for lectures in a course because usually one concept follows another. As a result, when lectures with similar content in different courses are linked, these links should not cross over each other frequently because they should follow a certain order in their own courses. In Fig. 2(b) for $(x_i, y_j)$ and $(x_{i'}, y_{j'})$ are two pairs of linked lectures in two courses $X$ and $Y$ and $i > i'$ and $j > j'$ as in Fig. 2(a). These two links seem reasonable because $x_i$ follows $x_{i'}$ in $X$ and $y_j$ follows $y_{j'}$ in $Y$. On the other hand, if $i > i'$ but $j < j'$ as in Fig. 2(b), this is a crossover and less likely to be correct, because if $x_i$ is similar to $y_{j'}$, $x_{i'}$ before $x_i$ is less likely to be similar to $y_j$ after $y_j$. This leads to the consideration for global lecture structures.

Let $\mathcal{L}$ represent a set of linked similar lecture pairs for courses $X$ and $Y$.

$$\mathcal{L} = \{(x_{i_1}, y_{j_1}), ..., (x_{i_k}, y_{j_k}), ..., (x_{i_{|\mathcal{L}|}}, y_{j_{|\mathcal{L}|}})\}, \quad (2)$$

where $(x_{i_k}, y_{j_k})$ represents the k-th pair of similar lectures with $x_{i_k}$ in $X$ and $y_{j_k}$ in $Y$. We can define an objective function $F(\mathcal{L})$ to be maximized,

$$F(\mathcal{L}) = \sum_{(x_i, y_j) \in \mathcal{L}} S(x_i, y_j) - \lambda_1 C(\mathcal{L}) - \lambda_2 |\mathcal{L}|, \quad (3)$$

where $\lambda_1$ and $\lambda_2$ are parameters to be determined by a development set. The first term on the right hand side of (3) is to accumulate the similarity values obtained in Section 3.1 for all linked pairs in $\mathcal{L}$. From (3), the second term on the right hand side of (3) represents "accumulated degree of crossover" in the set $\mathcal{L}$ because the larger the value of $|i - i'| + |j - j'|$, the more serious the crossover for this pair of links, and these values are accumulated in (4). The last term in (3), $-\lambda_2 |\mathcal{L}|$, prevents from including too many links, since any linked pair $(x_i, y_j)$ may contribute to the first term in (3) as long as $S(x_i, y_j)$ is positive.

A greedy algorithm can be used to solve the problem of maximizing $F$. In each iteration, the system links the pair of lectures that increases $F$ the most. Although this algorithm can only find an approximate solution, it works reasonably well in our experiments.

4. Prerequisite Prediction

The goal here is to determine which of the two given lectures in the same domain should be taken first.

4.1. Feature vector extraction

4.1.1. Framework based on Keywords

The purpose here is to construct a feature vector $u$ for each lecture.

4.1.2. Semantic Weights for Keywords

We select the top-n most frequently used words (with stop words deleted first) as the keywords $W = \{w_1, w_2, ..., w_n\}$, based on which the feature vectors can be built.

$$u = [tf(w_1), tf(w_2), ..., tf(w_n)], \quad (6)$$

where $tf(w_i)$ is the number of times the word $w_i$ appears in the lecture.

4.1.3. Feature Vector Representation Schemes

Different schemes are proposed here to construct the feature vectors $u$ for the lectures based on the keyword set $W$ and the corresponding semantic weights for keywords mentioned above.

- **Bag-of-word representation (BOW):**

$$u = [tf(w_1), tf(w_2), ..., tf(w_n)], \quad (6)$$
where \( u_i \) is the feature vector for the lecture, and \( tf(w_i) \) is the term-frequency of the keyword \( w_i \) in the lecture.

- **Weighted BOW**: Each term-frequency \( tf(w_i) \) in (6) is weighted by \( s(w_i) \) mentioned in Subsection 4.1.2:
\[
u = \frac{1}{N_w} \sum_{i=1}^{n} s(w_i) tf(w_i) v_{w_i}, \tag{7}
\]

Since we propose two different weights \( s(w_i) \) in Subsection 4.1.2, there are two different weighted BOW representation schemes here.

- **Word embedding representation**: We train the domain-specific word embedding model \( \text{glove} \) [23,24] with corpora collected from Wikipedia. We first extract key terms \( \text{topia.termextract} \) [13] from the course transcriptions, and use the obtained key terms as queries to retrieve all related articles in Wikipedia. All these articles including others linked to these articles in Wikipedia are taken as the training corpora for training the word embedding model for the course. With this model, we represent each keyword \( w_i \) as a word vector \( v_i \). The new keyword set is then, \( V = \{ v_1, v_2, ..., v_n \} \), where \( v_i \in \mathbb{R}^d \), \( d \) is the dimension of the word embedding model. Finally, we accumulate and average the word vectors for the keywords ever mentioned in the given lecture, similarity weighted, to be taken as the feature vector,
\[
u = \frac{1}{N_w} \sum_{i=1}^{n} s(w_i) tf(w_i) v_{w_i}, \tag{8}
\]
where \( N_w \) is the total count of all keywords in the set \( W \) appearing in the given lecture. This feature vector \( u \) in (8) is in fact very similar to the weighted BOW representation mentioned above in (7), except for weighted BOW each dimension of \( u \) is for a keyword, while here each dimension of \( u \) is a dimension for the word embedding model.

4.2. Support Vector Machine (SVM) Classification

Below \( u_i \) and \( u_j \) are the feature vectors of the lectures being considered as mentioned above, \( \alpha_{ij} \) is the corresponding prerequisite relationship, i.e., \( \alpha_{ij} > 0 \) if \( u_i \) is the prerequisite of \( u_j \) and \( \alpha_{ij} < 0 \) otherwise. \( M \) is a weight matrix and also the set of parameters to be learned. With \( M \) trained, we can then predict the prerequisite relations between lectures \( i \) and \( j \). Two different schemes are considered:

- **Directional Matrix**: \( M \in \mathbb{R}^{D} \), where \( D \) is the dimensionality of \( u_i \), either \( n \) in (8) or \( d \) in (9):
\[
\tilde{a}_{ij} = M \cdot (u_i - u_j), \tag{9}
\]

- **Transformation Matrix**: \( M \in \mathbb{R}^{d \times d} \)
\[
\tilde{a}_{ij} = u_i^T M u_j, \tag{10}
\]

Which assigns a weight to each component of \( u_i \) multiplied by each component of \( u_j \). The criterion for optimizing matrix \( M \) is then defined as:
\[
\min_{M} \left\{ \sum_{i,j} \left[ 1 - \alpha_{ij}(\tilde{a}_{ij} + b) \right]_+ + \lambda \|M\|^2 \right\}, \tag{11}
\]
where \( \| \cdot \|^2 \) is the matrix Frobenius norm, \( (1 - v)_{+} = \max(0, 1 - v) \) denotes the hinge function, \( b \) is the bias, \( (11) \) is actually the formula for SVM. Given a set of training lectures and their labeled prerequisite relationships \( (\alpha_{ij} = +1 \text{ or } -1) \), \( (11) \) can be trained with SVM algorithms [25,26].

5. Experimental Results

5.1. Course Material Description

We chose to focus on the courses on two areas, Natural Language Processing (NLP) and General Chemistry, each with two courses. The two NLP courses are offered by Stanford University [28] and Columbia University [29] lectures, having average lengths of 465.8 and 243.2 seconds each respectively consisting of 121 and 101 lectures, while the two Chemistry courses were from University of Kentucky [30] and Rice University [31,32] respectively containing 132 and 72 lectures, having average length of 463.9 and 877.8 seconds each. These courses are on the Coursera platform. The manual transcriptions for the audio and the title of each lecture were available and used in the experiments.

5.2. Linking Lectures with Very Similar Content

Three experts (two researchers and one graduate student) with NLP background were recruited to label whether two lectures (each in one of the two NLP courses) have very similar content. The average kappa values among the three experts were 82%. The ground truth was obtained by expert voting. Precision, recall and F1-measure were used as the evaluation measures. Precision is the percentage of lecture pairs linked by the computer to have similar content which were consistent with the ground truth, while recall is the percentage of lecture pairs considered to have very similar content in the ground truth which were similarly identified by the system. Two-fold cross validation were performed in the tests. The results are reported in Table 1. The upper part of Table 1 (labeled “Individual”) reports the results of using the individual pair similarity in Section 3.1. The rows (a) to (e) are the results of the different similarity measures mentioned in Section 3.1 with rows (a) to (c) based on the manual transcripts of the audio in the lectures, while (d) and (e) on the titles. The key terms in (b) were extracted by the key term extraction toolkit, \( \text{topia.termextract} \) which picked the nouns and noun phrases as key terms by POS tagging and some heuristic rules. LDA in (c) was implemented with \( \text{MALLET} \) [27] with 128 topics. We see the key terms yielded better results than considering all words (rows (b) vs (a)). Also, the latent topic based similarity yielded the best results among all the similarity measures based on the audio transcripts (rows (c) vs (a), (b)), obviously because the latent topics can handle the synonym problems to some extent.

The results for the lexical similarity and syntactic parsing tree similarity of the titles are respectively shown in rows (d) and (e). It is clear that syntactic parsing tree similarity outperformed the lexical similarity (rows (e) vs (d)). The titles were usually brief without redundant words, as lectures having similar titles usually have very similar content. This is why the

<table>
<thead>
<tr>
<th>Audio Transcripts</th>
<th>Precision (%)</th>
<th>Recall (%)</th>
<th>F1 (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a) all terms</td>
<td>13.8</td>
<td>24.6</td>
<td>17.3</td>
</tr>
<tr>
<td>(b) key terms</td>
<td>33.8</td>
<td>26.5</td>
<td>28.8</td>
</tr>
<tr>
<td>(c) topics</td>
<td>48.9</td>
<td>30.2</td>
<td>37.2</td>
</tr>
<tr>
<td>(d) all terms</td>
<td>52.7</td>
<td>20.7</td>
<td>29.7</td>
</tr>
<tr>
<td>(e) syntax</td>
<td>56.5</td>
<td>18.9</td>
<td>27.9</td>
</tr>
<tr>
<td>(f): (a)+(b)+(c)+(d)+(e)</td>
<td>42.9</td>
<td>52.7</td>
<td>47.2</td>
</tr>
</tbody>
</table>

Table 1: The results of linking lectures with similar content. The upper part (rows (a) to (f) in “Individual”) are those using different individual pair similarity measures in Section 3.1 while the lower part (row (g) in “Global”) is the result considering global structure in Section 3.2.

[2] The parsing trees were generated by Stanford CoreNLP toolkit.
results based on the titles had higher precision than those based on the audio transcriptions (rows (d), (e) vs (a), (b), (c) for “Precision”). On the other hand, because the titles may be too brief to cover all concepts mentioned in the lectures, sometimes lectures with very similar content were considered different if their common concepts were not implied by the titles, which led to lower recall (rows (d), (e) vs (a), (b), (c) for “Recall”). Row (f) are the results using the average of the five similarity measures in rows (a) to (e), giving a precision not as high as the best ones (rows (f) vs (c) (d), (e) for “Precision”), but the highest recall and F1-measure among all (rows (f) vs (a) to (e) for “Recall” and “F1”).

The last row (g) of Table 1 labeled “Global” is for the approach considering global structure as in Section 4.2. The similarity measure $S(x_i, y_j)$ in (5) is the average of the five measures used in row (f). Compared to row (f) with the same $S(x_i, y_j)$, considering the global structure in addition was significantly better in both precision and recall (rows (g) vs (f)). Clearly, the global structure is very helpful. Also, in row (g) all the precision, recall and F1 were significantly higher than 50%.

5.3. Prerequisite prediction

The two sets of parallel courses in NLP and Chemistry were both used. We simply assume that for two lectures in the same chapter of the same course, the one given earlier is the prerequisite of the other one given later. So for the two courses in each field, we took the lecture sequence order for one course as training data to learn the model to predict the prerequisite relationships of the lectures in the other course in the same field, while the sequence order for the latter was taken as the ground truth in testing. Although here we only used the model to predict the prerequisite relationships for lectures in courses in the same field, it is believed that the model learned should be equally applied on lectures in courses in other fields as well.

The accuracy of the prerequisite prediction was used as the measurement. The results are in Table 2. The two columns labeled with NLP and Chemistry in Table 2 are respectively for the results on the NLP and Chemistry courses. The upper part of Table 2 labeled as “Direct” is for the Directional Matrix in (9), with rows (a) to (e) for different feature vector representations and weights. The lower part “Transf” is for the Transformation matrix in (10) instead.

<table>
<thead>
<tr>
<th></th>
<th>NLP</th>
<th>Chemistry</th>
</tr>
</thead>
<tbody>
<tr>
<td>Direct</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(a) Bag-of-word (BOW)</td>
<td>68.1</td>
<td>61.4</td>
</tr>
<tr>
<td>Weighted BOW (b) Late-occurrence</td>
<td>65.5</td>
<td>62.8</td>
</tr>
<tr>
<td>Word Embedding (c) WordNet</td>
<td>70.0</td>
<td>63.3</td>
</tr>
<tr>
<td>Transf</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Word Embedding (d) Late-occurrence</td>
<td>69.5</td>
<td>64.8</td>
</tr>
<tr>
<td>Embedding (e) WordNet</td>
<td>73.3</td>
<td>65.2</td>
</tr>
</tbody>
</table>

Table 2: The performance of predicting prerequisite relationships. The upper part “Direct” is for the Directional matrix in (9), with rows (a) to (e) for different feature vector representations and weights. The lower part “Transf” is for the Transformation matrix in (10) instead.

6. Conclusions

We propose to structure related lectures in different courses retrieved from a query into a learning map by linking lectures of very similar content and predicting the prerequisites. Learners can then define their personalized learning path on the map and learn the desired information efficiently. In addition to objective evaluations, the user study show that 97.4% of users agree that such a map is helpful in learning.
7. References


[29] https://www.coursera.org/course/nlanpp.


[31] https://class.coursera.org/genchem1-001.