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

Sheng-syun Shen[1], *Hung-yi Lee*[1], *Shang-wen Li*[2], *Victor Zue*[2], *Lin-shan Lee*[1]

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

r03942071@ntu.edu.tw, hungyilee@ntu.edu.tw, swli@mit.edu, zue@mit.edu, lslee@gate.sinica.edu.tw

10.21437/Interspeech.2015-49

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.

1. Introduction

The increasing popularity of Massive Open Online Courses (MOOCs) [1] has resulted in huge number of courses available over the Internet under various MOOCs 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 $\mathcal{X}$, $\mathcal{Y}$ and $\mathcal{Z}$ 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 $\mathcal{X}$ (x_2) has content very similar to that of lecture 3 of $\mathcal{Y}$ (y_3), while z_1 is the prerequisite of y_2 and y_4 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), $\{z_1 \to y_2 \to (y_3, x_2) \to x_3 \to y_4 \to z_5\}$, 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 3. 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.

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 $\mathcal{X}$ and $\mathcal{Y}$ as in Fig. 1 based on their audio transcriptions or lecture titles, and then choose the pairs with similarity exceeding a threshold.

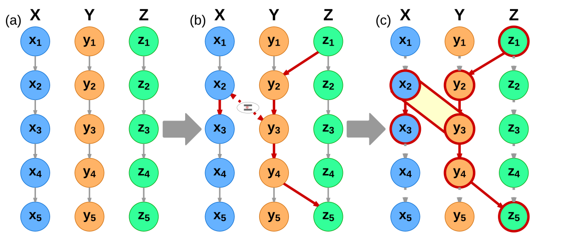

Figure 1: Structuring lectures in multiple courses with a map : (a) retrieved lectures in courses $\mathcal{X}$, $\mathcal{Y}$ and $\mathcal{Z}$, (b) linking lectures with similar content and predicting the prerequisite relationship, (c) personalized learning path chosen by a learner.

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 [14] from the course transcriptions, rather than on all words in the lexicon. 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, $S(x_i, y_j) = \sum_{k=1}^{T} P(t_k|x_i)P(t_k|y_j)$. 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 [15] (each dimension is a rule, and it is 1 if used and 0 otherwise), based on which cosine similarity can be evaluated. If two titles have similar parsing trees and similar words, the lectures usually have similar content.

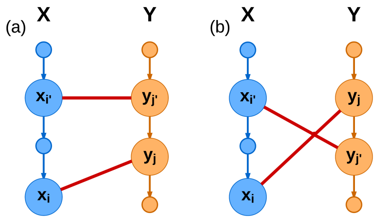

Figure 2: (a) Reasonable links and (b) crossover links.

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, for (x_i, y_j) and $(x_{i'}, y_{j'})$ are two pairs of linked lectures in two courses $\mathcal{X}$ and $\mathcal{Y}$ and $i > i'$ and $j > j'$ as in Fig. 2(a). These two links seem reasonable because x_i follows $x_{i'}$ in $\mathcal{X}$ and y_j follows $y_{j'}$ in $\mathcal{Y}$. 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 lecture pairs for $\mathcal{X}$ and $\mathcal{Y}$, $\mathcal{L} = \{(x_{i_1}, y_{j_1}), ..., (x_{i_k}, y_{j_k}), ..., (x_{i_{|\mathcal{L}|}}, y_{j_{|\mathcal{L}|}})\}$, where (x_{i_k}, y_{j_k}) represents the k-th pair of similar lectures with x_{i_k} in $\mathcal{X}$ and y_{j_k} in $\mathcal{Y}$. We define an objective function $F(\mathcal{L})$ to be maximized,

$$F(\mathcal{L}) = \sum_{(x_i,y_j)\in\mathcal{L}} S(x_i, y_j) - \lambda_1 C(\mathcal{L}) - \lambda_2|\mathcal{L}|, \qquad (1)$$

$$C(\mathcal{L}) = \sum_{(x_i,y_j),(x_{i'},y_{j'})\in\mathcal{L}} c((x_i, y_j), (x_{i'}, y_{j'})), \qquad (2)$$

$$c((x_i, y_j), (x_{i'}, y_{j'})) = \begin{cases} |i - i'|+|j - j'|, & \text{crossover,} \\ 0, & \text{otherwise.} \end{cases} \qquad (3)$$

where λ_1 and λ_2 are parameters to be determined by a development set. The first term on the right hand side of (1) is to accumulate the similarity values obtained in Section 3.1 for all linked pairs in $\mathcal{L}$. From (2)(3), the second term on the right hand side of (1) 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 (2). The last term in (1), $-\lambda_2|\mathcal{L}|$, prevents from including too many links, since any linked pair (x_i, y_j) may contribute to the first term in (1) if $S(x_i, y_j)$ is positive.

A greedy algorithm can be used to solve the problem of maximizing (1). In each iteration, the system links the pair of lectures that increases (1) 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

We select the top-n most frequently used words (with stop words deleted first) as the keywords [14]. This gives a set of n keywords for a course u, $\mathcal{W} = \{w_1, w_2, ..., w_n\}$, based on which the feature vectors can be built.

4.1.2. Semantic Weights for Keywords

We wish to develop some weights for the keywords, $\{s(w_1), s(w_2), ..., s(w_n)\}$, indicating their importance in semantics, where $s(w_I)$ is the weight for w_i.

- ***Semantic depth in WordNet:*** A "hypernym" is a word whose semantic field includes another word. For every keyword w_i found above in $\mathcal{W}$, we find it in the hypernymy tree from WordNet and traverse back to the root [16–20]. Number of steps needed to arrive at the root can be taken as $s(w_i)$ since it represents the semantic depth of w_i. Larger $s(w_i)$ implies w_i is more specific.
- ***Late occurrence ratio:*** For a course of m lectures, if a keyword w_i in $\mathcal{W}$ appears the first time in the l-th lecture, we have $s(w_i) = \frac{l}{m}$, so $s(w_i)$ is between [0,1]. Larger values of $\frac{l}{m}$ imply w_i first appears later in the course, or it is more specific in the respective domain.

4.1.3. Feature Vector Representation Schemes

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

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

$$u = [tf(w_1) \quad tf(w_2) \quad ... \quad tf(w_n)], \tag{4}$$

where u 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 (4) is weighted by $s(w_i)$ mentioned in Subsection 4.1.2,

$$u = [s(w_1)tf(w_1) \quad s(w_2)tf(w_2) \quad ... \quad s(w_n)tf(w_n)], \tag{5}$$

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 [21–24] with corpora collected from Wikipedia. We first extract key terms [14] from the course transcriptions, and use the obtained key terms as queries to retrieve all related articles in Wiki. 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, $\mathcal{V} = \{v_1, v_2, ..., v_n\}$, 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 to be taken as the feature vector,

$$u = \frac{1}{N_w}\sum\nolimits_{i=1}^{n} s(w_i)tf(w_i)v_i, \tag{6}$$

where N_w is the total count of all keywords in the set $\mathcal{W}$ appearing in the given lecture. This feature vector u in (6) is in fact very similar to the weighted BOW representation mentioned above in (5), 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, $\hat{a}_{ij}$ is the corresponding prerequisite relationship, i.e., $\hat{a}_{ij} > 0$ if u_i is the prerequisite of u_j and $\hat{a}_{ij} < 0$ 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 weight schemes are considered:

Directional Matrix: $M \in \mathbb{R}^D, \hat{a}_{ij} = M \cdot (u_i - u_j),$ (7)

Transformation Matrix: $M \in \mathbb{R}^{D\times D}, \hat{a}_{ij} = u_i^\intercal M u_j,$ (8)

Equation (8) assigns a weight to each component of u_i multiplied by each component of u_j, and D is the dimensionality of u, either n in (4)(5) or d in (6). The criterion for optimizing matrix M is then defined as:

$$\min_M \sum\nolimits_{i,j} [1 - a_{ij}(\hat{a}_{ij} + b)]_+ + \lambda\|M\|^2, \tag{9}$$

where $\|\cdot\|^2$ is the matrix Forbenius norm,$(1 - v)_+ = \max(0, 1 - v)$ denotes the hinge function, b is the bias, (9) is actually the formula for SVM. Given a set of training lectures and their labeled prerequisite relationships ($a_{ij} = +1 \; or \; -1$), (9) can be trained with SVM algorithms [25–27].

5. Experimental Results

5.1. Course Material Description

We chose to focus on the courses on two areas, Natural Language Processing 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 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 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. 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)(e) on the titles. The key terms in (b) were extracted by the key term extraction toolkit, *topia.termextract* [33]. LDA in (c) was implemented with *MALLET* [34] 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 [35] 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 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)).

The last row (g) of Table 1 (labeled "**Global**") is for the approach considering global structure as in Section 3.2. The similarity measure $S(x_i, y_j)$ in (1) 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.

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 results considering global structure in Section 3.2.

			Precision (%)	Recall (%)	F1 (%)
Individual	Audio Transcripts	(a) all terms	13.8	24.6	17.3
		(b) key terms	33.8	26.5	28.8
		(c) topics	48.9	30.2	37.2
	Title	(d) all terms	52.7	20.7	29.7
		(e) syntax	56.5	18.9	27.9
	(f): (a)+(b)+(c)+(d)+(e)		42.9	52.7	47.2
Global (g)			53.6	54.6	**54.1**

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 it believed that the model learned should be equally applied on lectures in courses in other fields as well.

Table 2: The performance of predicting prerequisite relationships. The upper part (**Direct**) is for the Directional matrix in (7), with rows (a) to (e) for different feature vector representations and weights. The lower part (**Transf**) is for the Transformation matrix in (8) instead.

Accuracy (%)			**NLP**	**Chemistry**
Direct	(a) Bag-of-word (BOW)		68.1	61.4
	Weighted BOW	(b) Late-occur	65.5	62.8
		(c) WordNet	70.0	63.3
	Word Embedding	(d) Late-occur	69.5	64.8
		(e) WordNet	73.3	65.2
Transf	Word Embedding	(f) Late-occur	69.4	66.6
		(g) WordNet	**76.1**	**67.0**

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 "**Directional**") is for the Directional Matrix in (7), with rows (a) to (e) for different feature vector representation schemes (Section 4.1.3) and two semantic weights (Section 4.1.2). Row (a) is for Bag-of-word vector in (4), and rows (b) and (c) for weighted BOW in (5) using late occurrence ratio (row (b)) and WordNet (row (c)) respectively. We noted that the late occurrence ratio was not helpful for NLP courses (rows (b) *vs* (a) for NLP), probably because different lecturers have different habits in using words, so the late occurrence ratios doesn't necessarily carry prerequisite relation information across courses. On the other hand, semantic depth from WordNet improved the performance on both NLP and Chemistry (rows (c) *vs* (a)(b)), or the hypernym tree structure is helpful in predicting the prerequisite relationships.

The results of semantic weighs for distributed word embedding representations are in rows (d) and (e). We found that word-embedding representation outperformed weighted BOW in all cases (rows (d)(e) *vs* (b)(c)). Although late occurrence ratio messed up the results while using weighted BOW schemes, it actually performed better with word embedding representation (rows (d) *vs* (a)). Moreover, semantic depth from WordNet with word embedding representations outperformed all the previous methods (rows (e) *vs* (a)(b)(c)(d)), probably because with using word embedding representations, lectures using very different words to describe very similar concepts can be properly represented in the continuous space.

We then used the Transformation matrix in (8) with the top-2 results obtained above (word embedding with weights in rows (d) and (e)). The results are in the lower part of Table 2 (rows (f)(g) under). We see more parameters learned with (8) yielded better results with the same features (rows (f)(g) *vs* (d)(e)), or considering the interaction between different feature dimensions is helpful.

5.4. User study

We conducted user study to compare the MOOCs user interface as proposed in Fig. 1(c) (**Proposed**) with two baselines. The first baseline displayed the originally retrieved lectures in sequences as in Fig. 1(a) (**Original**), while the second one first linked the lectures with similar content using the proposed approach and then ordered them randomly (**Half Random**). We entered three queries in NLP domain and showed the three corresponding interfaces mentioned above to 13 users, all having NLP background, and asked if they agreed that these interfaces helped in learning. There are 5 selections for them to choose, from "strongly disagree"(1 point) to "strongly agree"(5 points). The results are in Fig. 3. The average points for **Original**, **Half Random**, and **Proposed** are respectively 3.31, 1.56, and 4.52. Moreover, around 53.8% of the users strongly agreed that the proposed approaches help in learning, while much less did so for the original interface (Fig. 3(c) *vs* (a)). The much worse situation in Fig. 3(b) implies the importance of lecture sequence order even given similar lectures linked.

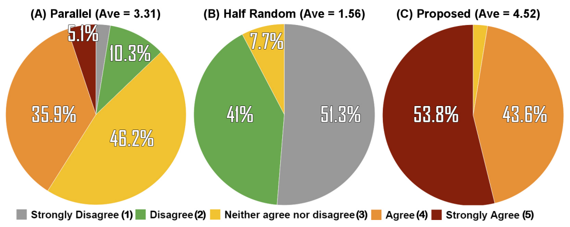

Figure 3: Users were asked if they agreed the interfaces help in learning, (a) Original, (b) Half Random, (c) Proposed.

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

[1] L. Pappano, "The year of the mooc," *The New York Times*, vol. 2, no. 12, p. 2012, 2012.

[2] http://youtu.be/VGKvPShhmiQ.

[3] J. R. Glass, T. J. Hazen, D. S. Cyphers, I. Malioutov, D. Huynh, and R. Barzilay, "Recent progress in the mit spoken lecture processing project." in *Interspeech*, 2007, pp. 2553–2556.

[4] I. Szoke, J. Cernocky, M. Fapso, and J. Zizka, "Speech@ fit lecture browser," in *Spoken Language Technology Workshop (SLT), 2010 IEEE*. IEEE, 2010, pp. 169–170.

[5] K. Riedhammer, M. Gropp, and E. Noth, "The fau video lecture browser system," in *Spoken Language Technology Workshop (SLT), 2012 IEEE*. IEEE, 2012, pp. 392–397.

[6] R. Rose, A. Norouzian, A. Reddy, A. Coy, V. Gupta, and M. Karafiat, "Subword-based spoken term detection in audio course lectures," in *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*. IEEE, 2010, pp. 5282–5285.

[7] H.-y. Lee, S.-R. Shiang, C.-f. Yeh, Y.-N. Chen, Y. Huang, S.-Y. Kong, and L.-s. Lee, "Spoken knowledge organization by semantic structuring and a prototype course lecture system for personalized learning," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 22, no. 5, pp. 883–898, 2014.

[8] Y. Fujii, K. Yamamoto, N. Kitaoka, and S. Nakagawa, "Class lecture summarization taking into account consecutiveness of important sentences." in *INTERSPEECH*, 2008, pp. 2438–2441.

[9] J. Kim, P. J. Guo, C. J. Cai, S.-W. D. Li, K. Z. Gajos, and R. C. Miller, "Data-driven interaction techniques for improving navigation of educational videos," in *Proceedings of the 27th annual ACM symposium on User interface software and technology*. ACM, 2014, pp. 563–572.

[10] D. Shahaf, C. Guestrin, and E. Horvitz, "Trains of thought: Generating information maps," in *Proceedings of the 21st international conference on World Wide Web*. ACM, 2012, pp. 899–908.

[11] ——, "Metro maps of science," in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2012, pp. 1122–1130.

[12] R. Scheines, E. Silver, and I. Goldin, "Discovering prerequisite relationships among knowledge components," Pearson Research Report, Tech. Rep.

[13] Y. Yang, H. Liu, J. Carbonell, and W. Ma, "Concept graph learning from educational data."

[14] Y.-N. Chen, Y. Huang, S.-Y. Kong, and L.-S. Lee, "Automatic key term extraction from spoken course lectures using branching entropy and prosodic/semantic features," in *Spoken Language Technology Workshop (SLT), 2010 IEEE*. IEEE, 2010, pp. 265–270.

[15] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun, "Large margin methods for structured and interdependent output variables," in *Journal of Machine Learning Research*, 2005, pp. 1453–1484.

[16] G. A. Miller, "Wordnet: a lexical database for english," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[17] C. Fellbaum, "Wordnet: An electronic lexical database," *Cambridge, MA: MIT Press*.

[18] R. Snow, D. Jurafsky, and A. Y. Ng, "Learning syntactic patterns for automatic hypernym discovery," *Advances in Neural Information Processing Systems 17*, 2004.

[19] S. Scott and S. Matwin, "Text classification using wordnet hypernyms," in *Use of WordNet in natural language processing systems: Proceedings of the conference*, 1998, pp. 38–44.

[20] S. A. Caraballo, "Automatic construction of a hypernym-labeled noun hierarchy from text," in *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*. Association for Computational Linguistics, 1999, pp. 120–126.

[21] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems*, 2013, pp. 3111–3119.

[22] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[23] T. Mikolov, W.-t. Yih, and G. Zweig, "Linguistic regularities in continuous space word representations." in *HLT-NAACL*, 2013, pp. 746–751.

[24] Q. V. Le and T. Mikolov, "Distributed representations of sentences and documents," *arXiv preprint arXiv:1405.4053*, 2014.

[25] C.-C. Chang and C.-J. Lin, "Libsvm: a library for support vector machines," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 3, p. 27, 2011.

[26] N. Cristianini and J. Shawe-Taylor, *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.

[27] C. J. Burges, "A tutorial on support vector machines for pattern recognition," *Data mining and knowledge discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[28] https://www.coursera.org/course/nlp.

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

[30] https://class.coursera.org/chemistry1-001.

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

[32] https://class.coursera.org/genchem2-001.

[33] https://pypi.python.org/pypi/topia.termextract/1.1.0.

[34] http://mallet.cs.umass.edu/.

[35] http://nlp.stanford.edu/software/corenlp.shtml.