Unsupervised Approach for Modeling Content Structures of MOOCs

Fareedah ALSaad[*]
University of Illinois at Urbana-Champaign
alsaad2@illinois.edu

Abdussalam Alawini
University of Illinois at Urbana-Champaign
alawini@illinois.edu

ABSTRACT

With the increased number of MOOC offerings, it is unclear how these courses are related. Previous work has focused on capturing the prerequisite relationships between courses, lectures, and concepts. However, it is also essential to model the content structure of MOOC courses. Constructing a precedence graph that models the similarities and variations of learning paths followed by similar MOOCs would help both students and instructors. Students can personalize their learning by choosing the desired learning path and lectures across several courses guided by the precedence graph. Similarly, by examining the precedence graph, instructors can 1) identify knowledge gaps in their MOOC offerings, and 2) find alternative course plans. In this paper, we propose an unsupervised approach to build the precedence graph of similar MOOCs, where nodes are clusters of lectures with similar content, and edges depict alternative precedence relationships. Our approach to cluster similar lectures based on PCK-Means clustering algorithm that incorporates pairwise constraints: Must-Link and Cannot-Link with the standard K-Means algorithm. To build the precedence graph, we link the clusters according to the precedence relations mined from current MOOCs. Experiments over real-world MOOC data show that PCK-Means with our proposed pairwise constraints outperform the K-Means algorithm in both Adjusted Mutual Information (AMI) and Fowlkes-Mallows scores (FMI).

Keywords

Precedence Graph, Clustering, Pairwise Constraints, Precedence Relations, Alternative Learning Paths, Common Learning Path.

1. INTRODUCTION

According to Class Central [19], by the end of 2019, over 13 thousand MOOCs have been announced or launched by more than 900 universities worldwide. With such an increase in online courses, it becomes increasingly hard for learners to understand similarities and differences among courses that cover similar topics. For instance, Coursera[1], one of the leading MOOC platforms, offers several *"Machine Learning"* courses, such as *"Machine Learning"* form Stanford University, *"Machine Learning with Python"* offered by IBM, *"Machine Learning for All"* from University of London, etc. Understanding the content structure across such similar courses can be very challenging. Consequently, MOOCs users may waste time choosing a course among a broad set of similar MOOC offerings.

Previous work studied ways for capturing prerequisite relationships between courses [23, 11], between lectures within (or among) courses [5, 6], or between concepts discussed within (or across) courses [2, 10, 15, 23, 11]. While modeling prerequisite relationships is crucial for understanding the content and knowledge structure of a specific domain, prerequisites do not reveal content overlap in similar courses. Further, modeling MOOC content in terms of prerequisite relations cannot detect the variations in the learning path between similar MOOCs.

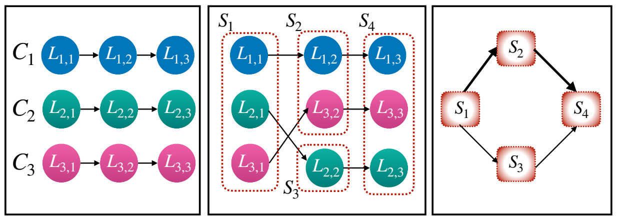
In this paper, we propose to model the content structure of similar MOOC offerings as a precedence graph. This graph can be useful for both learners and instructors. Learners can use the graph to build a customized learning plan as well as to explore how various courses explain the same topic. As for instructors, the graph can be used to identify any missing knowledge in their MOOCs offering, hence help them improve their courses. Section 3.2 elaborates on other possible applications of our proposed MOOCs precedence graph.

More precisely, we introduce an unsupervised approach to model the content structures of MOOCs. Figure 1 demonstrates the proposed idea. Given a set of courses that have some overlap in their content, we first cluster lectures based on their content similarity into clusters; each cluster represents a node in the precedence graph (see Figure 1 (b)). Then, the clusters are linked according to their lectures precedence relations mined from current MOOCs as depicted in Figure 1 (c). Linking clusters of similar content based on the precedence relations can reveal the various possible paths followed by similar courses and also capture which path is considered more common in these courses.

To cluster lectures based on their content similarity, we utilize a constraint-based clustering algorithm called Pairwise Constrained K-Means (PCK-Means). PCK-Means guides the clustering process by using two constraints: Must-Link and Cannot-Link. The idea is to guide the clustering process, by using the constraints, to focus on clustering lectures across courses instead of within courses to capture the similarity between courses. To measure the content similarity between lectures, we exploit both lecture titles and transcripts as they both encode enough information about the content of lectures. By using cosine similarity, we measure the similarity between lectures

[*]King AbdulAziz University, Jeddah, Saudi Arabia.
[1]https://www.coursera.org

Fareedah Alsaad and Abdussalam Alawini "Unsupervised Approach for Modeling Content Structures of MOOCs" In: *Proceedings of The 13th International Conference on Educational Data Mining (EDM 2020),* Anna N. Rafferty, Jacob Whitehill, Violetta Cavalli-Sforza, and Cristobal Romero (eds.) 2020, pp. 18 - 28



(a) Lecture Sequences in Different Courses. (b) Clustering Similar Lectures. (c) Linking Clusters.

Figure 1: *The basic idea of modeling the content of MOOCs to construct the precedence graph. Given similar Courses with some overlaps in content represented as sequences of lectures, the precedence graph is constructed by clustering lectures based on content similarity and link the clusters using the precedence relations between lectures.*

and construct the constraint examples to guide the clustering process. Our experiment on real MOOC dataset shows that PCK-Means with our proposed constraints outperforms standard K-Means algorithm in both Adjusted Mutual Information (AMI) and Fowlkes-Mallows scores (FMI).

After clustering similar lectures, we construct the precedence graph by linking clusters based on the precedence relations and label clusters using salient and key terms in each cluster. The generated precedence graph reveals the popular learning path and some alternative paths in our MOOCs dataset.

The rest of the paper is organized as follows. Section 2 presents related work. In section 3, we demonstrate the idea of modeling the content structure of MOOCs by an illustrative example and also present some applications of the precedence graph before we formally define our problem in Section 4. Section 5 describes how we represent the content of MOOCs using word count and embedding representations. In section 6, we explain PCK-Means algorithm and present our method of generating the lists of pairwise constraints. In section 7, we demonstrate the process of linking and labeling clusters to construct the precedence graph. Section 8 elaborates on our approach for the evaluation and presents some learning path examples extracted from the generated precedence graph. Finally, we conclude our work in section 9.

2. RELATED WORK

There has been recently a growing body of work that addresses the problem of modeling the content of MOOCs. Most of this work has focused on capturing the prerequisite relationships between courses [23, 11], between lectures or segments of lectures [5, 6], or between concepts discussed within or across courses [2, 10, 15, 23, 11]. These studies have developed supervised and unsupervised approaches to model only the prerequisite relations in MOOCs. In this paper, we go further and develop an unsupervised approach to capture the similarities and variations of learning paths between MOOCs in the same domain. Our work models the precedence relations (i.e., the implicit prerequisite relationships) between concepts by clustering similar lectures among different courses. Therefore, our model can revel popular learning paths shared by several courses along with alternative possible paths to learn the topic covered by these similar courses.

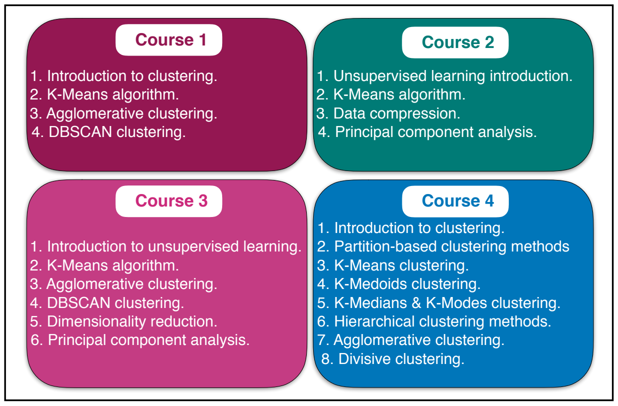
To model the prerequisite relationships, some studies have used external knowledge such as Wikipedia to support identifying educational concepts [10] or to represent concepts using Wikipedia articles or categories [15, 23, 11]. Using Wikipedia to identify concepts has some weaknesses: (1) some concepts are not included in Wikipedia [15] and thus can affect the performance of the model, (2) the mapping between course concepts to Wikipedia is not always accurate, which can affect the quality of the extracted concepts [10], and (3) using Wikipedia categories affects concept granularity by preferring more general concepts [2]. Instead of using Wikipedia, the work by ALSaad et al. [2] has exploited pre-trained part-of-speech-guided phrasal segmentation to extract phrases from course content and then manually group synonym phrases to represent concepts. In our work, instead of relying on external knowledge or manually improve the concepts, we represent the precedence graph nodes by salient terms using simple TF-IDF and bag-of-words representations. Our method represent each cluster with key terms by accumulating lecture representation vectors of each cluster and exploiting the top ranked words to represent clusters. Accumulating the vector representations of similar lectures helps in extracting representative terms that express the content of each cluster clearly.

Another related line of work is the use of prerequisite relations between concepts to organize learning units and predict the precedence relationships between them [1, 13]. The studies [1, 13] have proposed supervised approaches that rely on features extracted from external knowledge such as Wikipedia [1] and DBpedia [13] to infer the prerequisite relations between concepts. While the work [1] assumed that concepts are given, the study [13] manually extracted concepts by annotators. Our work is different as instead of inferring the prerequisite relations between concepts and then organizing them according to the precedence relations, we leverage the precedence relations between lectures in existing MOOCs to detect the precedence relations between the nodes in the precedence graph. Each node in the precedence graph is labeled automatically with key concepts that clearly express the content of each node without the use of external knowledge.

The work by Shah et al. [20] is the most relevant work to ours. The study has proposed a method for linking similar courses to construct a map of lectures connected by two types of relations: similar and prerequisite. The goal of the map is to help students find the desired learning path that fits their interests and backgrounds. Our work is very similar as we construct the precedence graph that depicts the different possible learning paths. However, instead of linking lectures by similar and prerequisite relations, we cluster lectures based on content similarity and connect clusters according to the precedence relations. Our approach reveals the similarities and variations of learning paths between different courses by capturing popular learning paths shared by many courses in the domain, hence emphasizes the importance of the common, comprehensive and alternative learning paths.

3. MODELING MOOCS CONTENT

In this section, we explain the idea of modeling the content of MOOCs as a precedence graph by using an illustrative example. We also discuss possible applications of the mined precedence graph.


Course 1
1. Introduction to clustering.
2. K-Means algorithm.
3. Agglomerative clustering.
4. DBSCAN clustering.

Course 2
1. Unsupervised learning introduction.
2. K-Means algorithm.
3. Data compression.
4. Principal component analysis.

Course 3
1. Introduction to unsupervised learning.
2. K-Means algorithm.
3. Agglomerative clustering.
4. DBSCAN clustering.
5. Dimensionality reduction.
6. Principal component analysis.

Course 4
1. Introduction to clustering.
2. Partition-based clustering methods
3. K-Means clustering.
4. K-Medoids clustering.
5. K-Medians & K-Modes clustering.
6. Hierarchical clustering methods.
7. Agglomerative clustering.
8. Divisive clustering.

Figure 2: *The sequence of lectures in four different courses that explain the topic Unsupervised Learning.*

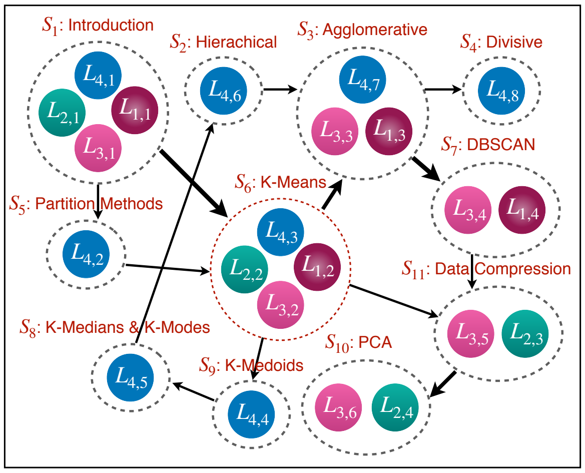


Figure 3: *The mined precedence Graph from courses in Figure 2. Lectures are grouped into clusters to construct the nodes of the graph. Edges depict the precedence relationships between clusters where thick edges represent the edges with high weights and thus indicate how common are the relations between the nodes.*

3.1 An Illustrative Example

For illustration purposes, let us assume that a MOOC platform offers four courses about *Unsupervised Learning* topic in machine learning as shown in Figure 2. Each course explains the topic using a sequence of lectures. As can be seen in Figure 2, there are some overlaps between these four courses as they all teach the same topic, but there are also some variations. The variation in each course is based on instructors' perspectives and background about the topic, instructors' teaching styles, and also the learning objective of each course. Some courses are abstract as they focus on the theory behind the topic while other courses are more concrete as they demonstrate the topic by illustrating real-world examples. Courses also vary in the coverage of topics as some courses are concise while other courses cover topics in more details. For example, **Course 1** and **Course 2** in Figure 2 are examples of concise courses that focus only on teaching the main concepts in the topic. In contrast, **Course 3** and **Course 4** are examples of courses that elaborate more in the topic by providing more detailed concepts.

Given the similarities and variations between these courses that explain the same subject, we investigate the following questions. *how these courses are related? What are the common concepts taught by the majority of these courses? Is there a common learning path shared by most of these courses? what are the alternative paths to study the topic?* Modeling the content structure of these courses as a precedence graph is a crucial step to help learners and educators with answering these questions.

The first step in building the precedence graph is to cluster lectures based on their content similarity and then construct a node in the graph for each cluster. Figure 3 shows the cluster assignment of each course lecture of Figure 2. As illustrated in Figure 3, all the introductory lectures, the first lecture of each course, are grouped into one cluster (cluster S_1) as all these lectures introduce the topic of *Unsupervised Learning*. Similarly, all the lectures about the concepts "*K-Means Algorithm*", "*Agglomerative Clustering*", and "*DBSCAN*" are clustered into three different clusters: S_6, S_3, and S_7 respectively. Furthermore, lectures about "*Data Compression*" are clustered into cluster S_{11} while lectures taught "*Principal Component Analysis*" concept are clustered into cluster S_{10}.

After clustering similar lectures and finding the nodes of the precedence graph, the next step in building the graph is to link these scattered clusters to reveal the precedence relations between clusters. To that end, we use the precedence relations between adjacent lectures of the same course to construct the edges between nodes (clusters) in the precedence graph. For instance, we add a directed edge from cluster S_5 to cluster S_6 to determine the precedence relation between these two nodes according to the sequence of lectures 2 and 3 in course 4. To reflect the strength of each precedence relations between two nodes (i.e., how common are the relations between the nodes), we attach each edge in the precedence graph with different weights. Edge weights are calculated by accumulating the frequency of lecture sequences in various courses. For example, as shown in Figure 3, the strength weight of edge $S_1 \to S_6$ should be higher than the strength weight of edge $S_1 \to S_5$ as three out of the four courses (1,2, and 3) have the sequence $S_1 \to S_6$ while only one course (4) shows the sequence $S_1 \to S_5$.

As mentioned earlier, the mined precedence graph can help us in revealing some hidden structures in similar MOOCs. For instance, it is clear from Figure 3 that the path $\{S_1 \to S_6 \to S_3 \to S_7\}$ is more common than other paths. The reason is that three courses (1,2, and 3) explain the concepts "*K-Means Algorithm*" after introducing the topic and two of them (courses 1 and 3) present the concepts "*Agglomerative Clustering*" and "*DBSCAN*" after that. In addition to indicating the common path, the mined precedence graph can also reveal other possible paths to learn the topic such as the path $\{S_1 \to S_5 \to S_6 \to S_3 \to S_4\}$, or the path $\{S_1 \to S_5 \to S_6 \to S_{11} \to S_{10}\}$. All these paths are valid and, off course, choosing a path depends on students' learning objectives.

In general, the mined precedence graph helps in capturing the similarities and variations of the learning paths of similar courses in our illustrative examples. In section 8.3, we present some learning path examples from the precedence graph generated by our approach.

3.2 Precedence Graph Applications

Our mined precedence graph can be used to support several applications for improving the learning and teaching process. However, before discussing these applications, we first want to clarify that (in this paper), we define a student as a person who uses MOOCs as modularized resources to learn topics of their choice (as opposed to taking a full course as part of a certificate program.) According to Zheng et al. [24], one of the motivations for a student to register for a MOOC is to learn some desired concepts on-demand. Once they achieve their learning goals, this type of student usually stops participating in the course.

Our precedence graph can support the following applications.

Personalized (customized) course plans. Our precedence graph can help students develop custom learning plans. Students can examine the graph to identify possible alternative paths for learning a topic and then choose the path that best fits their needs. For instance, a student might choose to follow one of the following two paths: $\{S_6 \rightarrow S_9 \rightarrow S_8\}$ or $\{S_6 \rightarrow S_{11} \rightarrow S_{10}\}$ shown in Figure 3. The former path helps the student explore and learn about various clustering algorithms:"*K-Means Algorithm*", "*K-Medoids Algorithm*", and "*K-Medians and K-Modes Algorithms*", while the latter path helps the student learn about the concepts of "*Data Compression*", and "*PCA*" with "*K-Means*" clustering algorithm.

An overview/summary of a topic. There are two ways in which the precedence graph can be used to help students obtain a quick overview of a particular topic of interest. First, students can use the graph to follow the most common path that is shared among several courses (i.e., the path with the highest edge weights.) For instance, students can follow the path: $\{S_1 \rightarrow S_6 \rightarrow S_3 \rightarrow S_7\}$ as this is the path with the highest edge weights in the graph shown in Figure 3. This path introduces the topic of *Unsupervised Learning* first before presenting three important and well-known clustering algorithms: "K-Means Algorithm", "Agglomerative Clustering Algorithm", and "DBSCAN Algorithm". Second, using summarization algorithms, we can generate a summary of the lectures in each node (cluster) of the most common path in the precedence graph. Such a succinct representation of clusters would provide students with a concise summary of the topic they want to learn.

Acquiring expert knowledge. Our precedence graph can also be used by students who are interested in becoming experts in a particular domain. The graph allows students to easily determine how the knowledge of a domain is structured. It also allows them to choose the path that exposes them to a varity of concepts related to the topic they want to learn. For example, to learn the most about unsupervised learning, a student can follow the longest path in the precedence graph shown in Figure 3: $\{S_1 \rightarrow S_5 \rightarrow S_6 \rightarrow S_9 \rightarrow S_8 \rightarrow S_2 \rightarrow S_3 \rightarrow S_4 \rightarrow S_7 \rightarrow S_{11} \rightarrow S_{10}\}$. Clustering similar lectures from various courses into the same clusters can also help this type of students as they can explore how different courses explain the same concept.

Helping instructors improve their courses. In addition to helping students with their learning process, the mined precedence graph can also aid instructors in understanding the structure of their MOOC offerings. By examining the precedence graph, instructors can identify potential knowledge gaps (missing topics) or a better ordering of the topics, and hence incorporate the new knowledge in their next course offerings.

4. PROBLEM DEFINITION

The design of a MOOC mimics that of a typical on-campus course in which the fundamental structure is a sequence of lectures. By leveraging the sequences of lectures and the content similarities between lectures from similar courses, we can model the knowledge structure of similar (i.e., courses that cover the same topic) MOOCs as a precedence graph. The nodes of this graph are groups of similar lectures, labeled by dominant and salient terms in these lectures. The edges of the graph represent the alternative precedence relations between nodes. Each edge can be assigned different weights that reflect the strength of the relation.

We formally define the problem as follows. Given a set of courses $X = \{C_1, C_2, C_3, \ldots, C_n\}$, where n is the total number of courses. We assume that all courses in X have the same difficulty level, and there are some content overlaps between courses. Each course C_i is an ordered list of lectures $C_i = [L_{i1}, L_{i2}, \ldots, L_{i|C_i|}]$, where $|C_i|$ is the total number of lectures in the course C_i. Each lecture L_{ij} is represented using the title t_{ij} and the lecture transcript d_{ij}. The goal is to model the content structure of similar MOOCs by constructing the precedence graph as a directed graph $G = (V, E)$ where V is the set of nodes, $V = \{S_1, S_2, S_3, \ldots, S_{|V|}\}$ (the number of nodes $|V|$ is given), and $E = \{e_1, e_2, e_3, \ldots, e_{|E|}\}$ is the set of edges between nodes. Edges in the graph $\mathbf{G}$ are directed edges to indicate the precedence relations between nodes. Each node in the precedence graph is a cluster or a group of lectures that have similar content. For example, $S_v = \{L_{i1}, L_{i2}, L_{j5}\}$ is a cluster that has the first two lectures from course C_i and the fifth lecture from course C_j. We represent the precedence graph G as an edge weight matrix $\mathbf{G} \in \mathbb{R}^{|V| \times |V|}$ where each entry of matrix $\mathbf{G}$ contains the edge weight. For instance, the edge weight of the entry g_{ij} reflects the strength of the precedence relationship from cluster S_i to cluster S_j.

To construct the precedence graph, we need first to find the set of nodes V of the precedence graph by grouping similar lectures using both lecture titles t_{ij} and lecture transcripts d_{ij}. Then, we compute the edge weight between pairs of nodes by leveraging the sequence of lecture in each course C_i. Sections 6 and 7 explain our proposed approach to build the precedence graph.

5. MOOC CONTENT REPRESENTATION

In this section, we demonstrate how we represent MOOC lectures by exploiting two representations: 1) *sparse representation* that is based on word count, and 2) *dense representation* to capture the semantic similarity between text. The purpose of using these two representations is to compare how each of them affects the performance of clustering.

To represent lectures, we use the sparse representation, a robust and straightforward representation based on the count of words. We represent lecture titles as vectors of word count using Bag-Of-Words (BOW) representation. Since lecture titles are short and concise, the frequency of each word in the BOW vector is usually one. The bag-of-words representation can be thought of as a bit vector where a bit is set to 1 when the word occurs in the title and set to 0, otherwise.

For representing lecture transcripts, we use the Term-Frequency Inverse-Document-Frequency (TF-IDF) representation. TF-IDF weighting takes into consideration the count of words in documents as well as the popularity of words in the corpus, hence gives higher weights to the words that are more frequent in the document and less popular in the corpus.

In our model, each lecture is represented by two vectors: a BOW vector to represent the title and a TF-IDF vector to capture the content of the transcript. The drawbacks of this representation are 1) it generates high dimensional sparse vectors, and 2) it cannot capture the semantic similarity between similar words.

To overcomes the limitations of TF-IDF and BOW representations, we use an alternative (dense) representation to model MOOC content: the unsupervised smoothed inverse frequency (uSIF) [8]. The uSIF is a simple, yet effective method for generating sentence embeddings without any labeled data. It is an improvement of smoothed inverse frequency (SIF) [18], one of the state-of-the-art embedding representation for longer pieces of text. The basic idea of uSIF is to exploit the pretrained word embeddings such as Word2Vec [14] or Glove [16] that capture the semantic meaning between words to learn the embeddings of sentences and paragraphs taking into consideration the frequency of words in the text. For more information about uSIF, please refer to [8].

For the embedding representation of lectures, we use uSIF with Glove pretrained word embeddings [4] to represent both lecture titles and transcripts. The number of feature dimensions in embedding vectors is 100 dimensions.

6. CLUSTERING LECTURES OF MOOCS

To construct the nodes of the precedence graph, lectures are grouped into clusters based on their content similarity. We can use any clustering algorithm such as K-Means to do the clustering of lectures. However, one problem of using K-Means or some other clustering algorithms is that they will cluster similar lectures not just across courses but also within courses. For example, if one course explains the topic "Gradient Descent in Logistic Regression" and then later explains the topic "Gradient Descent in Neural Networks", then there is a high chance that the clustering algorithm would group these two lectures into the same cluster as the course instructor would use almost the same terminology to explain these two topics. However, our goal is to capture the similarity of lectures across courses to reveal common learning paths utilized by many courses as well as other alternative learning paths. Therefore, we need to restrict the clustering process to cluster lectures from different courses rather than within the same course. To do that, we need to guide the clustering algorithm by imposing some constraints; which is infeasible with the standard K-Means algorithm. Therefore, we decided to exploit a constraint-based clustering algorithm called Pairwise Constrained K-Means (PCK-Means) [3] to guide the clustering process.

6.1 PCK-Means Clustering Algorithm

PCK-Means clustering algorithm [3] is a variation of the standard K-Means algorithm that incorporates distance between points as well as pairwise constraints to guide the clustering process. PCK-Means is a semi-supervised approach where users provide some labels or pairwise constraints that the algorithm uses to improve the clustering. Since collecting labels from users is expensive, we propose an unsupervised method by automatically find suitable labels or constraints to guide the clustering process (discussed in section 6.2).

Pairwise constraints can be used to determine the prior knowledge about the domain by specifying which instances (in our case lectures) should or should not be clustered together [21, 3]. There are two types of pairwise constraints: **Must-Link** and **Cannot-Link**. Must-Link constraint specifies pairs of instances (lectures) that need to be grouped into the same cluster, while Cannot-Link constraint determines pairs that should not be in the same cluster. Each type of pairwise constraint applies a penalty function when the constraint is violated. The objective function of PCK-Means is to 1) choose partitions that minimize the penalty cost of each constraint, and 2) minimize the sum of the square distance between the points and the centroids of the clusters they belong to.

More formally, let $\mathcal{M}$ be a list of Must-Link constraint, which includes tuples of lectures (L_i, L_j) that needs to be clustered together. Let $\mathcal{C}$ be a list of Cannot-Link constraint. Each item in $\mathcal{C}$ is a lecture pair of the form (L_i, L_j) where lecture L_i and L_j should not be in same cluster. Each tuple in $\mathcal{M}$ and $\mathcal{C}$ is order-independent. Assume $W = \{w_{i,j}\}$ and $\overline{W} = \{\overline{w}_{i,j}\}$ are the sets of penalty costs of violating the **Must-Link** and **Cannot-Link** constraints respectively. Each lecture L_i is assigned to a cluster $\mathcal{S}_i$, where $\mathcal{S}_i \in \{h\}_{h=1}^{|V|}$, by minimizing both the distance between L_i and the cluster centroid $\mu_{\mathcal{S}_i}$ and the penalty costs of violating the constraints. The objective function of PCK-Means algorithm is as follow:

$$\mathcal{J}_{pckm} = \frac{1}{2}\sum_{L_i \in X} \|L_i - \mu_{\mathcal{S}_i}\|^2 + \sum_{(L_i,L_j)\in\mathcal{M}} w_{ij}\mathbb{1}[\mathcal{S}_i \neq \mathcal{S}_j] + \sum_{(L_i,L_j)\in\mathcal{C}} \overline{w}_{ij}\mathbb{1}[\mathcal{S}_i = \mathcal{S}_j] \quad (1)$$

The first part of the objective function is K-Means objective function while the second and the third parts are the accumulated penalty costs of violating the Must-Link and Cannot-Link constraints respectively. The $\mathbb{1}[.]$ is the indicator function where $\mathbb{1}[\textit{true}] = 1$ and $\mathbb{1}[\textit{false}] = 0$.

In the initialization step of PCK-Means, examples of the pairwise constraints are used to estimate the centroids of clusters. Before initializing the cluster centroids, PCK-Means finds the transitive closure of tuples in Must-Link constraint and appends them to the list of Must-Link constraints. Then the updated list is used to create λ neighborhood sets. For each pair of neighborhoods, P_i and P_j with at least one pair of points that appear in the Cannot-Link list, PCK-means generates Cannot-Link constraint tuples between every pair of points in P_i and P_j and appends these tuples to the Cannot-Link constraints. Then the algorithm gets λ neighborhoods where links of type Must-Link constraint connect points within each neighborhood, and links of type Cannot-Link constraint connect some neighborhoods. If λ is higher than the number of clusters, $\lambda > |V|$, then the algorithm chooses the neighborhood sets with the largest number of instances to initialize the clusters and the centroids of each cluster. In contrast, if λ is less than the number of clusters, $\lambda < |V|$, then PCK-Means initializes the clusters from the λ neighborhoods and looks for a point that has links of type Cannot-Link constraint to all the λ neighborhoods. If so, it initializes a new $\lambda + 1$ cluster from this point. Otherwise, PCK-Means chooses the remaining $|V| - \lambda$ clusters randomly.

In general, the PCK-Means clustering algorithm is an iterative algorithm where it starts by using the pairwise constraints to initialize the clusters. Then, iteratively (1) assign points (or lectures) to clusters that minimize the combined objective function and then (2) re-estimate the centroids of each cluster according to the cluster assignment of each point. These two steps are repeated until the algorithm converges. For more information about the algorithm, please refer to [3].

6.2 Pairwise Constraints

To build the precedence graph, we use Must-Link and Cannot-Link constraints to guide the clustering process. Must-Link constraint includes a list of pairs of lectures that have higher chance to be similar while Cannot-Link constraint contains a list of lecture pairs that have lower chance to be part of the same clusters such as lectures from the same course. Yet, the question is how to find good examples of lecture pairs for the lists of Must-Link and Cannot-Link constraints.

6.2.1 *Must-Link Constraint*

As our goal is to capture the content similarity between lectures across courses, we want to feed the algorithm with similar lectures from different courses that have higher chance to be part of the same cluster as examples of Must-Link instances. To do that, we can use the cosine similarity measure to calculate the similarity score between lectures from different courses and choose lecture pairs with a similarity score exceeds some predefined threshold.

Besides Similar lectures across courses, some similar lectures within the same course can be good examples of Must-Link instances. Adjacent lectures can have very similar content and hence they should be grouped together in the same cluster. For instance, the two adjacent lectures "K-Means Algorithm" and "Initialization of K-Means Clustering" have similar content as they talked about *K-Means Clustering Algorithm* and thus they need to be grouped together. Therefore, we add adjacent lectures that have a similarity score greater than the predefined threshold.

We propose two approaches to capture the similarity between lectures within courses or among courses. First, we use the cosine similarity between two lectures represented by lecture transcripts. Pairs of lectures are considered similar when they have similar content and hence the cosine similarity score would be high. Second, we use the cosine similarity between two lectures represented by lecture titles. We believe that two lectures are similar when they have very similar titles even when there are some variations in the content. One reason is that instructors sometimes explain the topic from different perspectives. For instance, one instructor might explains the lecture with a title "K-Means Clustering Algorithm" by using examples while another instructor might explains the same lecture by illustrating the theory behind it. Although the content is different, both lectures explain the same topic but from different perspectives. Another reason of using lecture titles is due to the average length of lectures in MOOCs. Lectures in MOOCs are usually shorter in length compared with regular university classes. As a result, some instructors split the topic into two or more lectures. Usually these lectures have very similar titles and should be clustered together even if their content might vary. Therefore, we decided to utilize lecture titles to measure the similarity between lectures in addition to lecture transcripts. However, we use two different predefined thresholds K_1 and K_2 to capture the lectures similarity using titles and transcripts respectively as we have to set a higher threshold for titles to minimize the noise.

In general, the list of Must-Link constraint contains any similar lectures across courses and similar adjacent lectures within courses.

6.2.2 *Cannot-Link Constraint*

Unlike the Must-Link constraint, Cannot-Link constraint is used to indicate lecture pairs that should not be part of the same clusters. Since we want to force the clustering algorithm to capture the similarity between lectures across courses, we add lecture pairs from the same course to the list of Cannot-Link constraint. However, not any pair can be added to the list as some adjacent lectures can have similar content or similar titles and hence need to be grouped into the same cluster. Therefore, to determine lecture pairs that are suitable to be examples of Cannot-Link constraint, we apply the cosine similarity on the transcripts of two adjacent lectures. When the cosine similarity of two adjacent lectures, L_{ij} and $L_{i(j+1)}$ of course C_i, are less than a predefined threshold K_3, then we can say that there is a **topic shift** and hence we can add these two adjacent lectures to the list of Cannot-Link constraint. However, before adding any lecture pairs to the list of Cannot-Link constraint, we need to ensure that the pair is not part of the Must-Link constraint and its transitive closure list. In addition to adding the two adjacent lectures L_{ij} and $L_{i(j+1)}$, we also pair the lecture L_{ij} with all the subsequent lectures of lecture $L_{i(j+1)}$ since there is a shift in the topic. As a result, we add the lectures $(L_{ij}, L_{i(j+z)})$ where $1 < z < |C_i| - j$, to the list of Cannot-Link constraint if they are not part of the Must-Link constraint and its transitive closure list.

In general, the purpose of Cannot-Link constraint is to restrict the clustering algorithm from clustering lectures within courses in order to capture the similarity between different courses. As a result, by using Must-Link and Cannot-Link constraints, the clustering algorithm learns to cluster lectures from across courses and only cluster adjacent lectures within the same course if they are similar.

7. BUILDING PRECEDENCE GRAPH

Building the precedence graph from similar MOOCs has three steps: (1) Cluster similar lectures to construct the node of the graph, (2) Link the nodes by a directed weighted edge to determine the precedence relations between nodes, and (3) Represent each node by dominant and salient terms mined from lectures belong to each nodes. In the previous section, we explain how we cluster similar lectures using PCK-Means algorithm with our proposed Must-Link and Cannot-Link constraints. In this section, we first present our method of linking the precedence graph nodes before illustrating our approach of labeling each node.

7.1 Linking Clusters

After clustering similar lectures, we need to link the scattered clusters to construct the precedence graph. As we mentioned earlier, we utilize lecture sequences in each course. We can think of the sequence of lectures in MOOCs as implicit prerequisite relationships between lectures as these sequences are carefully designed by experts. When instructors design courses, they usually maintain the prerequisite order constraints between lectures by placing prerequisite lectures before the dependent lectures. In addition, according to the *locality of references* property [1], when designing a course plan, a dependent lecture should appear as soon as possible after the prerequisite lecture to reduce students comprehension burden. Therefore, tackling the various sequence orders of lectures from different courses helps in linking clusters of lectures from across courses and thus captures the precedence relations between clusters.

To link the scattered clusters, we use the precedence relations between adjacent lectures to infer the precedence relations between clusters. If two adjacent lectures L_{ij} and $L_{i(j+1)}$ of course C_i appear in two different clusters, then these two clusters need to be linked by an edge with a direction from the cluster that includes lecture L_{ij} to the cluster that has lecture $L_{i(j+1)}$. Sometimes some adjacent lectures appear in the same cluster and hence we ignore the sequence relation of these lectures.

To capture the strength of the precedence relations between clusters, and hence how these relations are common in current MOOCs, we attach each edge with different wights. We accumulate the frequency of courses that have adjacent lectures clustered into two different clusters to determine the weight between these two clusters. The equation to determine the edge weight is as follow:

$$W(\mathcal{S}_i \rightarrow \mathcal{S}_j) = \sum_{\forall C \in X} \sum_{z=1}^{|C|-1} \mathbb{1}[L_z \in \mathcal{S}_i \wedge L_{z+1} \in \mathcal{S}_j] \qquad (2)$$

where $W(\mathcal{S}_i \rightarrow \mathcal{S}_j)$ is the weight of the edge between cluster $\mathcal{S}_i$ and $\mathcal{S}_j$ and $\mathbb{1}[.]$ is an indicator function where $\mathbb{1}[\mathit{true}] = 1$ and $\mathbb{1}[\mathit{false}] = 0$.

Since the edge weights determine the popularity of relations across similar courses, edge weights are not normalized to be between 0 and 1 because normalization will produce misleading weights. For example, if we use normalized edge weights, then the edge that connects two clusters that have adjacent lectures from one course will have the same weight; which is equal to 1, to the edge that connects two clusters that have adjacent lectures from N courses. Therefore, we use unnormalized edge wights to capture the popularity of the precedence relations.

7.2 Labeling Clusters

Each node in the precedence graph is labeled by some key terms to represent the topics or key concepts discussed by the lectures attached to this node. To extract the key terms from lectures, we exploit lecture titles and transcripts represented by bag-of-words and TF-IDF representations respectively. Lecture titles are very concise and usually have the key terms in lectures. On the other hand, lecture transcripts are more elaborative and would help in extracting other important key terms that demonstrate topics or key concepts of each cluster.

The basic idea to extract the key terms is to accumulate the vector representations of each lecture that belongs to the same cluster in order to find the key terms of that cluster. In other words, for all lectures that belong to the same cluster we accumulate the bag-of-word representation vectors of their titles and also add the TF-IDF weighting vectors of their transcripts. Then, we use the top k terms from these two different representations to find the salient terms that represent each cluster. The following is the equation used to specify the key words of each cluster:

$$\begin{aligned} \mathit{Label}(\mathcal{S}_i) = (\max_k \sum_{j=1}^{|D|} \sum_{\forall L \in \mathcal{S}_i} TFIDF(w_j | w_j \in d_L)) \\ \bigcup (\max_k \sum_{j=1}^{|T|} \sum_{\forall L \in \mathcal{S}_i} BOW(w_j | w_j \in t_L)) \qquad (3) \end{aligned}$$

where the first part finds the top k terms by using the TF-IDF representation of lecture transcripts d_L where $|D|$ is the total number of vocabularies in the corpus of lecture transcripts. For each word w_j in the vocabulary, we accumulate the TF-IDF wights of word w_j if the word appears in lecture L that belongs to cluster $\mathcal{S}_i$. Similarly, the second part determines the top k terms by exploiting the bag-of-words representation of titles t_L where the total number of vocabularies in lecture titles is $|T|$. We also accumulate the BOW weights of each word belongs to titles of all lectures that are part of cluster $\mathcal{S}_i$. By taking the union of these two sets of top words, we extract salient terms that clearly explain the content of each clusters.

8. EVALUATION

In this section, we evaluate the performance of our approach for clustering similar lectures using PCK-Means algorithm with the proposed pairwise constraints. We first present the dataset and ground truth we used in our evaluation. Then, we compare the performance of the clustering algorithms using both representations: word counts (sparse representation) and embeddings (dense representation). We also present some examples of the learning paths extracted from the precedence graph that was constructed by our approach. Finally, we discuss some limitations of our study.

8.1 Datasets

We used a dataset of six modules related to *Unsupervised Learning* and *Clustering Algorithms* from five real machine learning and data mining courses offered by the Coursera platform[2]. These modules include "Unsupervised Machine Learning", "Partitioning Based Clustering Methods and Hierarchical Clustering Methods", "Unsupervised Learning", "Clustering", "Clustering With K-Means", and "Hierarchical Clustering" (see Table 1.) The total number of lectures in the dataset is 65 lectures. Each lecture is represented by its title and transcript.

To evaluate the performance of the PCK-Means algorithm and the effectiveness of the proposed constraints, we asked experts to construct the ground truth labels of our dataset. Each of our four experts (a Machine Learning professor, an Information Science professor, a Machine Learning graduate student, and a Database and Information Systems graduate student) manually grouped lectures based on topics similarities. None of the experts is participating in this study.

To measure the level of agreement among our experts, we used the Fleiss' kappa measure. Fleiss' Kappa is a statistical measure of inter-rater agreement used to determine the level of agreement between two or more raters. The kappa score of labels collected from experts was $\kappa = 0.65$, which indicates substantial agreements between the annotators.

After receiving the labeled datasets from our experts, we used the majority votes to decide the cluster assignment of each lecture. For lectures that experts disagreed on their clustering assignment, we decided to follow the advice of our experts and created a new cluster for each lecture. The total number of labeled clusters was 21 clusters.

8.2 Clustering Performance

To evaluate the performance of our clustering approach and to study the effect of using the pairwise constraints on clustering performance, we compared the PCK-Means algorithm to the standard K-Means algorithm. In particular, we focused on two measures: (1) Adjusted Mutual Information (AMI), and (2) Fowlkes-Mallows scores (FMI). Adjusted Mutual Information is a variation of the Mutual Information measure that is used for comparing clustering results. According to

[2]https://www.coursera.org
[3]https://www.coursera.org/learn/advanced-machine-learning-signal-processing
[4]https://www.coursera.org/learn/cluster-analysis
[5]https://www.coursera.org/learn/machine-learning
[6]https://www.coursera.org/learn/machine-learning-with-python
[7]https://www.coursera.org/learn/ml-clustering-and-retrieval

Table 1: *The dataset utilized for the evaluation. It has six modules from five courses. The total number of lectures in the dataset is 65 lectures.*

Courses	**Modules**	**# of Lectures**
Advanced Machine Learning and Signal Processing[3]	Unsupervised Machine Learning	13
Cluster Analysis in Data Mining[4]	Partitioning Based Clustering Methods and Hierarchical Clustering Methods	15
Machine Learning[5]	Unsupervised Learning	12
Machine Learning With Python[6]	Clustering	6
Machine Learning Clustering and Retrieval[7]	Clustering With K-Means	13
	Hierarchical Clustering	6

Table 2: *The performance of clustering algorithms. PCK-Means outperforms the standard K-Means in both representations: (1) TF-IDF for lecture transcripts and bag-of-words (BOW) for lecture titles, (2) The embedding representation (uSIF) for both lecture transcripts and titles. The performance of PCK-Means is statistically significant (represented by *) in both representations.*

	AMI		FMI	
Method	Average	Max	Average	Max
TF-IDF\BOW Representation				
K-Means	0.523	0.597	0.412	0.478
PCK-Means	**0.551***	**0.649**	**0.511***	**0.632**
Embedding Representation (uSIF)				
K-Means	0.395	0.491	0.344	0.452
PCK-Means	0.480*	0.536	0.420*	0.548

Romano et al. [17], AMI measure should be used to evaluate the clustering performance when the reference clustering is unbalanced and contain small clusters. Since we have unbalanced clusters (i.e., some clusters have many lectures while others have one or two lectures), we decided to use AMI for the evaluation. The second metric, Fowlkes-Mallows scores, is a geometric mean of precision and recall where precision determines the correctness of the clustering assignments of lectures while recall measures the completeness of the assignments. Similar to AMI, FMI gives a zero score for random clustering assignments.

Before discussing clustering performance, it worth mentioning that for finding the lists of Must-Link and Cannot-Link constraints, we tried various values for each threshold, K_1, K_2, and K_3, and used the values that gave the highest performance. For TF-IDF and BOW representations, the thresholds were $K_1 = 0.85$, $K_2 = 0.3$, and $K_3 = 0.07$ for titles and transcripts in Must-Link list and for transcript in Cannot-Link list respectively. For the uSIF representation, the thresholds were $K_1 = 0.85$, $K_2 = 0.65$, and $K_3 = 0$. Having $K_3 = 0$ in uSIF representation does not mean that we exclude the list of Cannot-Link constraint. The cosine similarity values in the uSIF embedding representation can have negative values as some values in the embedding vectors are negatives.

Because PCK-Means and K-Means algorithms produce different clustering assignments for each run (based on how the centroids are initialized), we ran each clustering algorithm 20 times. Then we recorded the average and the max scores. TF-IDF and bag-of-words representations have a total number of 1650 dimension features. So, we reduced the number of dimensions before clustering the data by applying the T-distributed Stochastic Neighbor Embedding (t-SNE) algorithm[12]. We also applied the t-SNE reduction technique on the uSIF embedding. However, because the performance of the uSIF was degraded due to the dimensions reduction, we decided to use all 200 dimensions for the embedding representation: 100 for titles, and 100 for transcripts. Table 2 summarizes the results.

The average and max scores for each algorithm are presented in Table 2. We can see from the table that PCK-Means outperforms K-Means in both representations. The differences in performance between PCK-Means and K-Means are statistically significant, using Welch's t-test, with p-value score < 0.01 in TF-IDF\BOW and uSIF representations for FMI measure. In contrast, when using AMI for the comparison, the differences between PCK-Means and K-Means are statistically significant with p-value scores < 0.01 with uSIF representation and p-value < 0.05 with TF-IDF\BOW representation. We also compare the performance of PCK-Means using different representations: TF-IDF\BOW and uSIF. It is clear form the table that PCK-Means with TF-IDF\BOW representation outperforms PCK-Means with uSIF embedding representation where the difference is statistically significant with p-value < 0.01 in both AMI and FMI measures. In general, PCK-Means with TF-IDF\BOW representation achieves the highest performance.

Since uSIF embedding representation uses pretrained word embeddings that allow it to capture the semantic similarity between documents, we expected it to have the highest performance. However, it did not perform as expected. We investigate this issue and found that some words from our dataset of lecture transcripts and titles do not exist in the list of words from the Glove pretrained model. The total number of missing words was 31 words from both lecture titles and transcripts. The missing words includes some key terms, such as *agglomerative*, *dendrogram*, *subcluster*, *medoids*, *sparkml*, and *dbscan*.

To study the effect of using the lecture titles and transcripts when generating the Must-Link constraint, we compared the performance of the PCK-Means algorithm using only Must-link constraint from titles to the performance of the same algorithm using only Must-link constraint from transcripts. We use TF-IDF and bag-of-words representation with the same set of thresholds for the comparison as PCK-Means achieves the highest performance with this representation. We show the results of this experiment in Table 3. The results indicate that using both lecture titles and transcripts to produce the Must-Link constraint achieves the highest score. We conclude that title and transcripts representations are important for capturing the similarity between lectures. We also notice that removing Must-Link tuples of lecture transcripts reduces the clustering performance more than removing title tuples. This is expected as lecture transcripts contain more keywords than titles. However, using only titles to generate the Must-Link constraint tuples achieves comparable results, which also indicates the importance of using titles to capture lectures similarities.

Table 3: *The performance of the clustering in PCK-Means, PCK-No-Title, and PCK-No-Trans using TF-IDF \BOW representation. In PCK-No-Title, we remove all the tuples from Must-Link list that are generated by using lecture titles. In PCK-No-Trans, all Must-Link tuples produced by lecture transcript are removed. Combing both titles and transcripts improves the performance of PCK-Means.*

	AMI		FMI	
Method	Average	Max	Average	Max
PCK-Means	**0.551**	**0.649**	**0.511**	**0.632**
PCK-No-Title	0.534	0.561	0.489	0.576
PCK-No-Trans	0.486	0.559	0.403	0.489

8.3 Examples of Learning Paths

After clustering lectures, we create the precedence graph that represents the six modules by linking the clusters and labeling them with salient terms. We utilize the clusters generated by PCK-Means algorithm with TF-IDF and bag-of-words representation as it achieves the highest performance. In this section, we present several learning paths examples extracted from our precedence graph.

Follow the crowd: The first example we extracted from our precedence graph is the learning path that is shared across many modules in our dataset. Students who follow the most common learning path would have a good overview of the topic as they follow the most popular path that is shared by many courses. Figure 4 depicts the common learning path for the *Unsupervised Learning* topic. The figure shows that the learning path starts with an introduction about Unsupervised Learning, followed by k-Means clustering algorithm and how to choose the number of clusters. Then, dimensionality reduction in clustering is discussed next using the Principle Component Analysis algorithm as an example of dimensionality reduction techniques.

The expert learning path: The second example path we extracted from the precedence graph is one of the longest learning paths. Figure 5 shows a path that spans over seven nodes. This path starts with the partitioning-based clustering methods discussing algorithms, such as “K-Means”, “K-Medians”, “K-Medoids”, and “K-Modes”. It then discusses the application of “K-Means Algorithm” in apache sparkml. Next, it shifts to the hierarchical clustering methods by recommending “Divisive Clustering Algorithm” and “Aglomerative Clustering”. Finally, it presents the “DBSCAN”, a density-based clustering algorithms. This long path is more comprehensive than the common path as it explores more clustering algorithms. Students who are interested in gaining comprehensive knowledge about clustering will find this path very rewarding. Note that such a path does not exist in any of the original six modules we have in our dataset; but it was extracted from the precedence graph constructed by our approach.

Give me some options: Figure 6 shows an example of a sub-graph with several alternative learning paths. To learn the “K-Means” concept, a student can either start with introduction to unsupervised learning or learn about partitioning-based clustering methods. After learning “K-Means”, the student can choose one of the four possible paths: (1) Learn how to choose the number of clusters using “Elbow” methods, (2) Learn about different partitioning algorithms such as “K-Medians”, (3) Move to the hierarchical clustering algorithms and learn “Divisive Algorithm”, or (4) Shift to the hierarchical clustering algorithms and learn “Agglomerative clustering”. Each of these

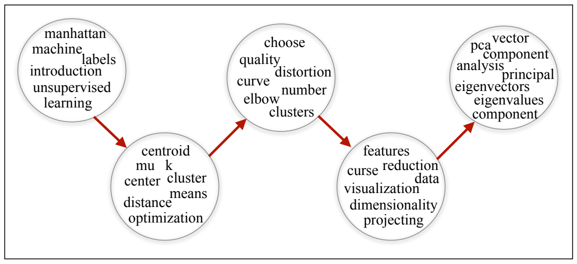

manhattan machine labels introduction unsupervised learning
centroid mu k center cluster means distance optimization
choose quality distortion curve number elbow clusters
features curse reduction data visualization dimensionality projecting
pca vector component analysis principal eigenvectors eigenvalues component

Figure 4: *The common learning path extracted from the Precedence Graph. This path is shared by many modules and includes fundamental concepts in Unsupervised Learning topic.*

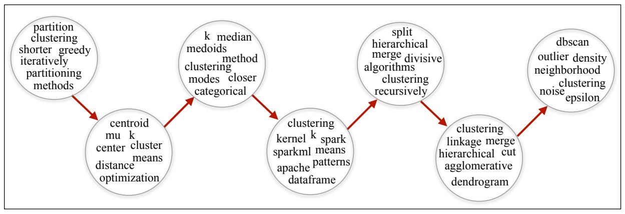

partition clustering shorter greedy iteratively partitioning methods
centroid mu k center cluster means distance optimization
k median medoids method clustering modes closer categorical
clustering kernel k spark sparkml means apache patterns dataframe
split hierarchical merge divisive algorithms clustering recursively
clustering linkage merge hierarchical cut agglomerative dendrogram
dbscan outlier density neighborhood clustering noise epsilon

Figure 5: *An example of a long learning path extracted from the Precedence Graph. This long path can support students who acquiring expert knowledge as it presents many clustering algorithms.*

possible paths are also precedence to other nodes as shown in Figure 6. From the sub-graph, students can choose the learning path that fits their needs. Additionally, the sub-graph shown in Figure 6 gives students a comprehensive overview of how concepts are connected among several courses related to the *Unsupervised Learning* topic.

8.4 Limitations

There are two limitations of our study. First, using the sequence relations among lectures to infer the precedence relations between clusters can cause cycles in the precedence graph. The method proposed in this paper has not addressed the problem of cycles. The naive approach to solve the problem of cycles is to eliminate edges with lower weights that cause cycles in the graph. Further investigation for addressing graph cycles is left as a future work.

Second, in the evaluation we have not examined the performance of our approach in other domains. In the future work, we plan to apply our method on courses from different domain areas and thus generate the precedence graph for each domain.

9. CONCLUSIONS

In this paper, we developed an approach to build the precedence graph of similar MOOCs that have overlaps in content. Our approach is based on Pairwise Constrained K-Means (PCK-Means) clustering algorithm that incorporates constraints to guide the clustering process to focus on clustering similar lectures across courses. We proposed a method of generating the lists of Must-Link and Cannot-Link constraints. PCK-Means with our generated constraint examples significantly outperforms the standard K-Means algorithm with the TF-IDF and bag-of-words representations achieves the highest performance. Using the clusters of similar lectures as nodes in the precedence graph, we connect each cluster according to the

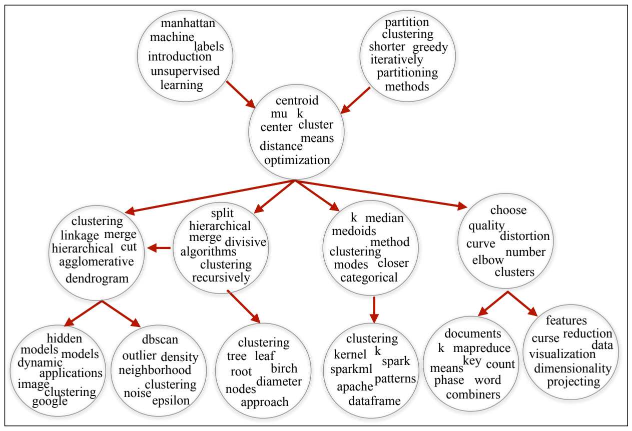

manhattan machine labels introduction unsupervised learning
partition clustering shorter greedy iteratively partitioning methods
centroid mu k center cluster means distance optimization
clustering linkage merge hierarchical cut agglomerative dendrogram
split hierarchical merge divisive algorithms clustering recursively
k median medoids method clustering modes closer categorical
choose quality distortion curve number elbow clusters
hidden models models dynamic applications image clustering google
dbscan outlier density neighborhood clustering noise epsilon
clustering tree leaf root birch nodes diameter approach
clustering kernel k spark sparkml patterns apache dataframe
documents k mapreduce key count means phase word combiners
features curse reduction data visualization dimensionality projecting

Figure 6: *An example of a sub-graph extracted from the Precedence Graph. It depicts some possible paths to learn the topic. It also gives a comprehensive overview about the topic.*

precedence relations between lectures in various courses by directed weighted edges to reflect the strength of the precedence relations between clusters. Finally, we label each node in the precedence graph by key concepts extracted from lectures belonging to each cluster. The generated precedence graph reveals popular learning paths as well as alternative learning paths of learning the topics of MOOCs in our dataset.

The precedence graph constructed by our approach is considered the initial block for building applications that support personalized learning. As an example, we can use the precedence graph to build a tool that visualizes the precedence graph to help learners to choose the desired learning paths that are suitable to their interests and backgrounds. Another application is to build a tool that recommends personalized study plans for students based on their interests and time constraints. As discussed in [9, 7], the main obstacle that faces online learners is not having enough time for the course. Further, according to [24, 22], some learners register for a MOOC with a motivation to learn some concepts and hence they drop the course after they are done with studying the concepts of their interest. Wilkowski et al. [22] found that large groups of learners just wanted to learn some concepts without the purpose of earning certificates. Therefore, it is very important to build an application that recommends study plans based on learners motivation, interests, and time constraints. Our proposed precedence graph would be the initial step for building such applications.

10. ACKNOWLEDGMENTS

We would like to thanks all the four participants in the user study for their time and effort to label the dataset.

11. REFERENCES

[1] R. Agrawal, B. Golshan, and E. Papalexakis. Toward data-driven design of educational courses: A feasibility study. *JEDM-Journal of Educational Data Mining*, 8(1):1–21, 2016.

[2] F. ALSaad, A. Boughoula, C. Geigle, H. Sundaram, and C. Zhai. Mining mooc lecture transcripts to construct concept dependency graphs. In *Proceedings of the 11th International Conference on Educational Data Mining*, pages 467–473. EDM, 2018.

[3] S. Basu, A. Banerjee, and R. J. Mooney. Active semi-supervision for pairwise constrained clustering. In *Proceedings of the 2004 SIAM international conference on data mining*, pages 333–344. SIAM, 2004.

[4] O. Borchers. Fast sentence embeddings. `https://github.com/oborchers/Fast_Sentence_Embeddings`, 2019.

[5] D. Chaplot and K. R. Koedinger. Data-driven automated induction of prerequisite structure graphs. In *Proceedings of the 9th International Conference on Educational Data Mining*, pages 318–323. EDM, 2016.

[6] W. Chen, A. S. Lan, D. Cao, C. Brinton, and M. Chiang. Behavioral analysis at scale: Learning course prerequisite structures from learner clickstreams. In *Proceedings of the 11th International Conference on Educational Data Mining*, pages 66–75. EDM, 2018.

[7] T. Eriksson, T. Adawi, and C. Stöhr. “time is the bottleneck”: a qualitative study exploring why learners drop out of moocs. *Journal of Computing in Higher Education*, 29(1):133–146, 2017.

[8] K. Ethayarajh. Unsupervised random walk sentence embeddings: A strong but simple baseline. In *Proceedings of The Third Workshop on Representation Learning for NLP*, pages 91–100, 2018.

[9] R. F. Kizilcec and S. Halawa. Attrition and achievement gaps in online learning. In *Proceedings of the second (2015) ACM conference on learning@ scale*, pages 57–66. ACM, 2015.

[10] C. Liang, J. Ye, Z. Wu, B. Pursel, and C. L. Giles. Recovering concept prerequisite relations from university course dependencies. 2017.

[11] H. Liu, W. Ma, Y. Yang, and J. Carbonell. Learning concept graphs from online educational data. *Journal of Artificial Intelligence Research*, 55:1059–1090, 2016.

[12] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[13] R. Manrique, J. Sosa, O. Marino, B. P. Nunes, and N. Cardozo. Investigating learning resources precedence relations via concept prerequisite learning. In *2018 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*, pages 198–205. IEEE, 2018.

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

[15] L. Pan, C. Li, J. Li, and J. Tang. Prerequisite relation learning for concepts in moocs. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1447–1456, 2017.

[16] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[17] S. Romano, N. X. Vinh, J. Bailey, and K. Verspoor. Adjusting for chance clustering comparison measures. *The Journal of Machine Learning Research*, 17(1):4635–4666, 2016.

[18] A. Sanjeev, L. Yingyu, and M. Tengyu. A simple but tough-to-beat baseline for sentence embeddings. *Proceedings of ICLR*, 2017.

[19] D. Shah. By the numbers: Moocs in 2019 - class central. `www.classcentral.com/report/mooc-stats-2019/`, 2019.

[20] S.-s. Shen, H.-y. Lee, S.-w. Li, V. Zue, and L.-s. Lee. Structuring lectures in massive open online courses (moocs) for efficient

learning by linking similar sections and predicting prerequisites. In *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[21] K. Wagstaff, C. Cardie, S. Rogers, S. Schrödl, et al. Constrained k-means clustering with background knowledge. In *Icml*, volume 1, pages 577–584, 2001.

[22] J. Wilkowski, A. Deutsch, and D. M. Russell. Student skill and goal achievement in the mapping with google mooc. In *Proceedings of the first ACM conference on Learning@ scale conference*, pages 3–10. ACM, 2014.

[23] Y. Yang, H. Liu, J. Carbonell, and W. Ma. Concept graph learning from educational data. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, pages 159–168. ACM, 2015.

[24] S. Zheng, M. B. Rosson, P. C. Shih, and J. M. Carroll. Understanding student motivation, behaviors and perceptions in moocs. In *Proceedings of the 18th ACM conference on computer supported cooperative work & social computing*, pages 1882–1895. ACM, 2015.