UPreG: An Unsupervised approach for building the Concept Prerequisite Graph

Varun Sabnis
R V College of Engineering, Bangalore
varunsabnis@gmail.com

Kumar Abhinav
Accenture Labs, Bangalore
k.a.abhinav@accenture.com

Venkatesh Subramanian
Accenture Labs, Bangalore
venkatesh.subramania@accenture.com

Alpana Dubey
Accenture Labs, Bangalore
alpana.a.dubey@accenture.com

Padmaraj Bhat
Accenture Labs, Bangalore
padmaraj.bhat@accenture.com

ABSTRACT

Today, there is a vast amount of online material for learners. However, due to the lack of prerequisite information needed to master them, a lot of time is spent in identifying the right learning content for mastering these concepts. A system that captures underlying prerequisites needed for learning different concepts can help improve the quality of learning and can save time for the learners as well. In this work, we propose an unsupervised approach, UPreG, for automatically inferring prerequisite relationships between different concepts using NLP techniques. Our approach involves extracting the concepts from unstructured texts in MOOC (Massively Open Online Courses) course descriptions, measuring semantic relatedness between the concepts and statistically inferring the prerequisite relationships between related concepts. We conducted both qualitative and quantitative studies to validate the effectiveness of our proposed approach. As there are no ground truth labels for these prerequisite relations, we conducted a user study for the evaluation of the prerequisite relations. We build the concept graph using prerequisite relations. We demonstrate few examples of the learning maps generated from the graph. The learning maps provide prerequisite information and learning paths for different concepts.

Keywords

Prerequisite relation, Text mining, Learning path

1. INTRODUCTION

In today's fast-paced world, skill development and a strong foundation in fundamental concepts are becoming very crucial for career growth. MOOCs, offering a wide variety of courses online are becoming ubiquitous among many learners interested in acquiring knowledge and becoming competent in their field of interest. In this journey, learners need to know the order in which they must learn different concepts to attain a good level of mastery in a specific topic. Knowing the prerequisites when learning a topic improves the learning experience of learners and is influential to the learner's achievements [20]. Prerequisite concepts define the concepts one must know or understand first before attempting to learn or understand something new.

With the increasing amount of educational data available, automatic discovery of concept prerequisite relations has become both an emerging research opportunity and an open challenge. There is a growing interest today in researching different techniques for automatically inferring the prerequisite relations between concepts [17][20]. Various solutions like curriculum planning [23], learning assistant [10], automated reading list generation [9] etc, have been developed based on such techniques.

Prerequisites at the course-level have been manually curated by experts and this helps find prerequisite relations between the concepts covered within the courses. For example, concepts in a course on Optimization are prerequisites to concepts in a course on Deep Learning. An example in this scenario would be the Gradient Descent algorithm being a prerequisite for understanding the Backpropagation algorithm used in Deep Neural Networks. Such relations created manually will not scale in real-world online applications. Modern applications today support learning content from a wide variety of domains and cater to learners from multiple educational backgrounds. Manual processes for creating prerequisite relations in such applications are expensive and time-consuming. Hence, it is necessary to develop solutions that can infer prerequisite relations using automated approaches.

In this work, we propose an unsupervised approach, UPreG, for automatically inferring prerequisite relationships between different concepts using NLP techniques. We built a concepts graph capturing the concepts and the prerequisite relation between them. Concepts here refer to technologies, programming languages, tools, and topics in the Software and Computer Science domain. The concepts graph can be

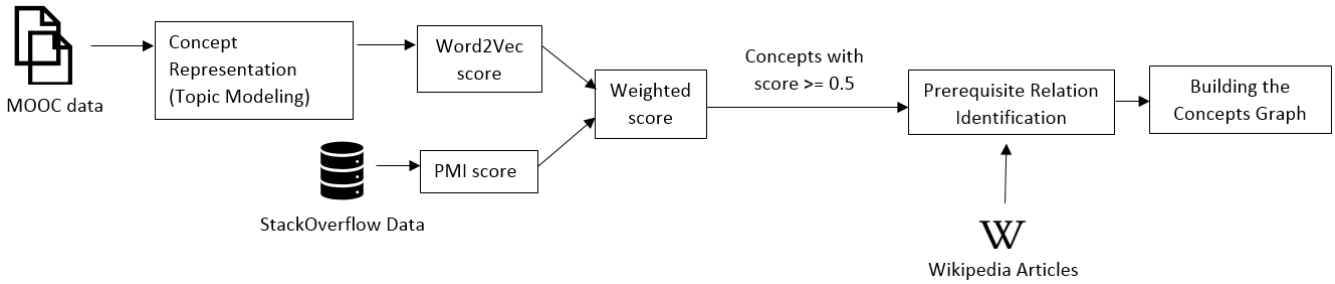

Figure 1: Flow Diagram

leveraged to find the right content for the learner, including the prerequisite content. We conducted both qualitative and quantitative studies to validate the effectiveness of our proposed approach. As there are no ground truth labels for these prerequisite relations, we conducted a user study for the evaluation of the prerequisite relations. We observed that our approach is effectively able to infer the prerequisite relations between concepts. The approach can be extended to other domains as well.

This paper is structured as follows. We present the related work in Section 2. In Section 3, we describe our approach for concept graph generation followed by the evaluation methodology and results in Section 4. In section 5, we discuss the challenges we encountered while building the concepts graph. Finally, Section 6 concludes with future work.

2. RELATED WORK

Pan et al. [17] propose a learning-based method for latent representations of course concepts. They defined various features and trained a classifier that can identify prerequisite relations among concepts. Roy et al. [20] proposed PREREQ, a supervised learning method for inferring concept prerequisite relations. The approach uses latent representations of concepts obtained from the Pairwise Latent Dirichlet Allocation model, and a neural network based architecture. They assumed that concept prerequisites are available to train supervised model. Yu et al. [24] present an improved version PREREQ-S by introducing students' video watch order to enhance the video dependency network. They sorted the watched videos of each student by time and utilize these sequences for replacing the video sequences. They apply two simple DNN models, which first encode the embeddings of the concept pairs and then train an MLP to classify the prerequisite ones. Alzetta et al. [3] applied a deep learning-based approach for prerequisite relation extraction between educational concepts of a textbook. Lu et al. [13] proposed an iterative prerequisite relation learning framework, iPRL, which combines a learning based model and recovery based model to leverage both concept pair features and dependencies among learning materials. Liang et al. [12] addressed the problem of recovering concept prerequisite relations from university course dependencies. They [11] further applied active learning to the concept of prerequisite learning problem. Pal et al. [16] proposed an approach to find the order of concepts from textbooks using the rule-based method. Prior work assumes the prerequisite relationship pairs available as ground truth and apply supervised learning approach. However, acquiring labeled prerequisite pairs is time-consuming and expensive. Currently, the major drawback of supervised learning is that it doesn't perform well over cross-domains [16]. To the best of our knowledge, we are the first to apply unsupervised approach to extract the prerequisite relationship for software domain.

3. APPROACH

In this section, we discuss our approach to build the concepts graph. It is a directional graph where nodes represent the concepts and the edges between nodes represent the prerequisite relationship between them. Our approach in building the concepts graph involves concept representation, measuring semantic similarity between the concepts and identification of the prerequisite relationship between them.

3.1 Concept Representation

The descriptions of the courses in MOOCs contain rich information about the concepts that will be taught to the learners. Many courses do not have annotated course tags to represent the concepts taught in the course. It is very expensive and time-consuming to manually create course tags from the course content [13]. Hence, the concepts must be extracted from the course content using text mining approaches. We collected course metadata from different MOOCs (Udemy and edX) and our internal Learning Management System. We apply Latent Dirichlet Allocation (LDA) [4], a topic modeling algorithm on each course description to extract the concepts. The algorithm generates a topical distribution for each course description. To determine the most relevant topic that represents the concepts a course covers, the topic with highest probability from the distribution is selected. After performing several iterations, we found that setting k=5 (number of topics to be extracted) gave the best results. We extract a total of 9750 unique concepts.

3.2 Semantic similarity between concepts

The Semantic similarity measure between concepts gives a measure of the semantic relatedness between them. Concepts that appear in the same context or appear together very often have higher semantic similarity scores. Semantic Similarity computation eliminates noise present in the results of the topic modeling algorithm and reduces the possibilities of weak relations in the concepts graph. It is also useful in prerequisite relation identification as it is likely that concepts appearing in similar contexts will have better chances of being identified with prerequisite relation. This improves the selection of candidates in the concepts graph.

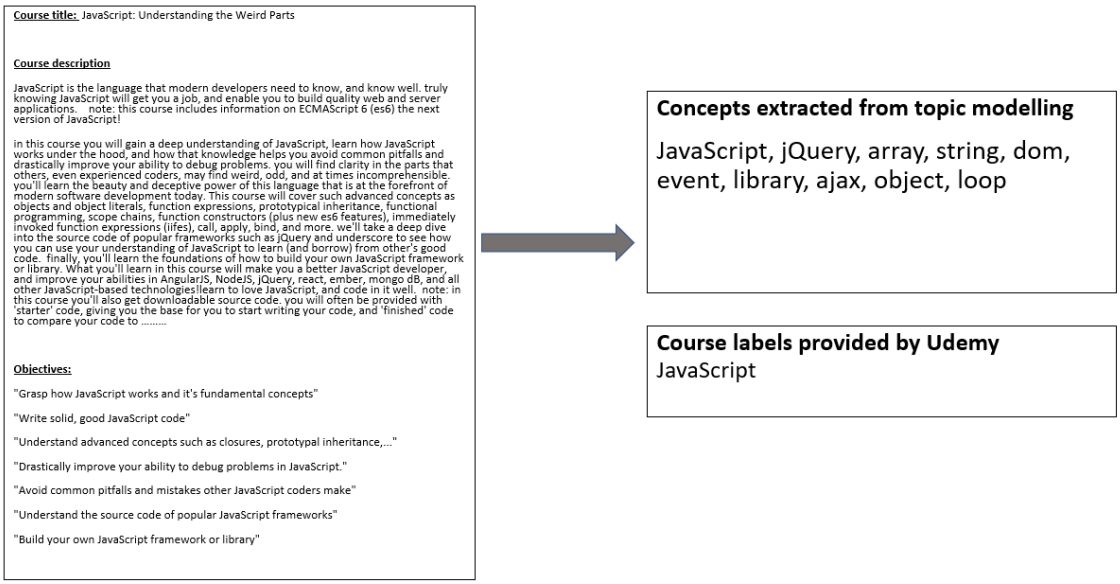

Figure 2: Concepts generated for a JavaScript course

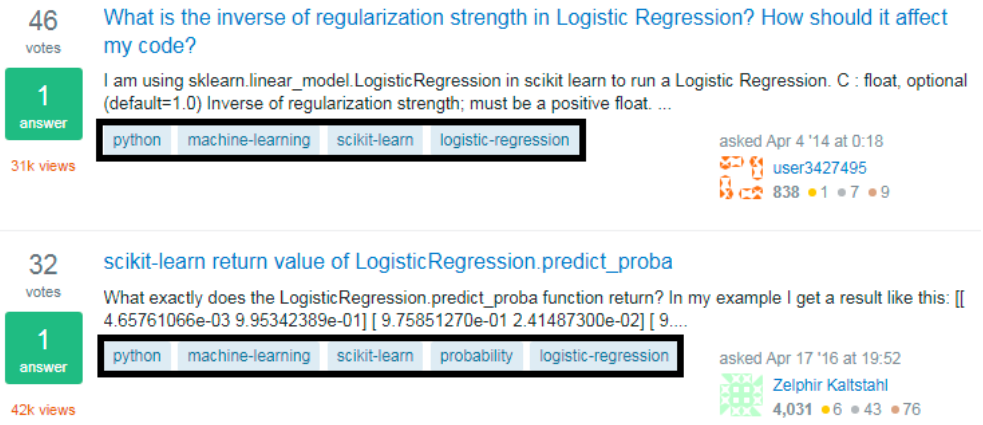

Figure 3: Stack Overflow questions and tags

To measure semantic similarity between the concepts we compute Pointwise Mutual Information (PMI) and Word2Vec cosine similarity scores. The Semantic similarity scores between the concepts are computed as the weighted average of the two scores.

3.2.1 *Pointwise Mutual Information*

PMI gives a measure of concept association used in information theory [6]. It gives a measure of how likely two concepts would occur together when compared to their independent occurrences in the data. For computing the PMI of concept pairs, tags of Stack Overflow questions obtained from Stack Overflow data dumps were used. The author posting a question on Stack Overflow is asked to provide tags associated with the posted question (as shown in Figure 3). Tags that appear often together across all the questions are likely to be strongly related. Higher the score between the two concepts, the more similar they are. We assume that the concepts occurring together have some correlation over a large set of pairs. To compute the PMI scores, we leverage the Stack Overflow dump consisting of 1,000,000 Stack Overflow questions along with their tags [21]. PMI score between any two concepts c_1 and c_2 is defined as:

$$PMI(c_1, c_2) = max\left(0, \frac{\log\left[p(c_1) \cdot p(c_2)\right]}{\log p(c_1, c_2)} - 1\right) \quad (1)$$

Here $p(c_1, c_2)$ is the probability of co-occurrence of concepts c_1 and c_2. It is fraction of Stack Overflow questions in which concepts c_1 and c_2 co-occur as tags. $p(c_1)$ and $p(c_2)$ is the probability of the independent occurrence of concepts c_1 and c_2 as tags across all Stack Overflow questions. The score obtained is a normalized score that takes values between 0 and 1. This ensures PMI and Word2Vec similarity scores have the same scale when taking their weighted average.

3.2.2 *Word2Vec Embeddings*

Raw word frequency is not a great measure of association between words. One problem is that raw frequency is very skewed and not very discriminative. It also does not capture

the kinds of contexts shared between the words, which word embedding techniques capture [2]. We apply Word2Vec approach to learn semantic relatedness between concepts. The Word2Vec model is based on the intuition that words which are similar in context appear closer in the word embedding space. Word2Vec algorithm uses a neural network model to learn word associations from a large corpus of text. We use skip-gram model [15] to learn word embeddings which are low dimensional vector representations of the extracted concepts. The neural network is trained using a text corpus of course descriptions. We train the skip-gram model for generating 300-dimensional word embeddings. Word2Vec neural network is trained using the text corpus of the course description and objectives. We train Word2Vec model on a corpus of 64,150 courses using the Python library gensim [18] with default parameters. Some of the Word2Vec similarity scores between concepts are captured in Table 2. Word2Vec(W2V) similarity score between the concepts is computed as the cosine similarity between these word embeddings.

$$W2V(\mathbf{c_1}, \mathbf{c_2}) = \frac{\mathbf{c_1} \cdot \mathbf{c_2}}{\|\mathbf{c_1}\|\|\mathbf{c_2}\|} \quad (2)$$

Here $\mathbf{c_1}$ and $\mathbf{c_2}$ represent 300 dimensional embedding vectors of concept c_1 and c_2.

Finally, we compute the similarity score as a weighted average of the above two scores. For simplicity, we set the weights to 0.5.

$$Sim(c_1, c_2) = w_1 \cdot W2V(\mathbf{c_1}, \mathbf{c_2}) + w_2 \cdot PMI(c_1, c_2) \quad (3)$$

We observed that extracted concepts can appear with different representations in the Stack Overflow question tags. Examples include synonymous pairs such as node.js and nodejs, javascript and js, mvc and model view controller, etc. To identify such instances, we use the Stack Overflow synonym tag api [22] and identify the matching or synonymous concepts in the Stack Overflow tags. We also filter out irrelevant concepts having no occurrence or synonyms in the Stack Overflow tags. After this process, we end up with 5200 concepts. During the computation of probabilities for PMI scores, we also consider the occurrence count of the synonyms. For example, when computing PMI between javascript and any other concept, we compute the independent and co-occurrence probabilities by counting occurrences of both javascript and js tags in the Stack Overflow questions.

3.3 Identifying Concept Relation

In this section, we explain the process of identifying the prerequisite relationship between different concepts. We only consider the concept pairs with high semantic similarity scores. It is very likely that concept pairs that have very low semantic similarity scores are not related at all and we can ignore such pairs. For example, it is not useful to learn the relationship between Neural Network and PHP which are not related and occur in different domains (deep learning and web development respectively). However, it would be interesting to study the concept pairs Gradient Descent and Backpropagation which are algorithms used in machine learning and share high semantic similarity scores. Inferring the relation that Gradient Descent is a prerequisite of Backpropagation and not vice-versa would be useful. To infer such relations, we make use of Wikipedia articles. For each pair of concepts with high semantic similarity (threshold of 0.5), we compute the concept relevancy scores. For concepts c_1 and c_2, we measure how often the concept c_1 is referred in the Wikipedia article of concept c_2 and vice-versa. Based on the concept relevancy scores, we can infer the prerequisite relation. For example, we know that Java is a prerequisite of Spring Boot. So, it is quite possible that in an explanation for Spring Boot (a Java Web framework), the concept Java would be mentioned more often when compared to the concept Spring Boot being mentioned in an explanation about Java. Algorithm 1 captures the steps to identify the prerequisite relation between concepts.

Algorithm 1 Prerequisite relation inference between concepts

Input: Pair of concepts c_i and c_j which are strongly related, and Wikipedia Knowledge articles.

Output: Relationship between concept pairs (prerequisite relationship) i.e. c_1 is prerequisite of c_2 or vice-versa

1: Tokenize the knowledge articles for all the concepts (C_n), where C_n is set of concepts
2: **for** ordered pair concepts (c_i, c_j) **do**
3: Compute Concept Relevancy scores (CRS) for ordered pairs (c_i, c_j) as

$$CRS(c_i, c_j) = \frac{TF(c_i \in D_j)}{V(D_i, D_j)}$$

$$CRS(c_j, c_i) = \frac{TF(c_j \in D_i)}{V(D_i, D_j)}$$

where c_i and c_j are the concepts for which CRS is computed, D_i and D_j are the wikipedia articles for concepts c_i and c_j respectively, $TF(c_i \in D_j)$ captures the term frequency for concept c_i in wikipedia article D_j, $TF(c_j \in D_i$ captures the term frequency for concept c_j in wikipedia article D_i, and $V(D_i, D_j)$ is the normalization term that captures the total vocabulary in articles D_i and D_j.
4: If $CRS(c_i, c_j) > CRS(c_j, c_i)$, then c_i is prerequisite of c_j and vice-versa
5: **end for**

Table 1: Data collected from online learning platforms

Platform	# Courses	Categories
Udemy	13601	Software development, and design
Edx	1072	Software development
Internal LMS	49202	Software development, and design

3.4 Learning Maps

The identified prerequisite relation pairs were used to build the concept graph. The concept graph has 1325 concepts and 1868 edges. We use networkx [8] python library to build the concept graph. We pass the adjacency list created from the identified concept-prerequisite pairs as an input to the library. The edges in the graph have directions from the concept node to the prerequisite node. The learning maps are built for each concept in the graph using the Depth-first search (DFS) algorithm. They are represented as DFS trees generated by the algorithm. To visualize the learning maps

Table 2: Semantic similarity scores from Word2Vec Embeddings and PMI

c_1	c_2	PMI Scores	W2V Scores
Hadoop	Hive	0.67	0.43
MongoDB	NoSQL	0.64	0.72
JavaScript	jQuery	0.44	0.68
JavaScript	NodeJS	0.57	0.61
Neural Network	Backpropagation	0.74	0.61
Blockchain	Cryptocurrency	0.34	0.73
Inheritance	Polymorphism	0.54	0.62
ASP.NET	Java	0.17	0.08
NodeJS	Promise	0.54	0.21
ASP.NET	C#	0.20	0.42
Hadoop	Java	0.1	0.23
SVM	Classification	0.62	0.43
RDBMS	SQL	0.19	0.37
Machine Learning	Linear Algebra	0.18	0.49

we use d3.js force layout [5]. In visualizing the learning maps we reverse the edge direction, i.e, from prerequisite node to concept node. This is done for the purpose of meaningful and easy identification of prerequisites in the learning maps. The learning maps for the concepts Blockchain and Java Spring framework are shown in Figure 4. The root node colored in blue represents the main concept and all nodes below the root node colored in orange represent the concepts that are prerequisites for the main concept. The child nodes represent the prerequisite concepts for its parent node concept.

Table 3: Extracted prerequisite relation between concepts

c_1	c_2
Distributed systems	Mapreduce
Probability	Logistic Regression
Encryption	Cryptography
Smart Contract	Ethereum
Backpropagation	Neural Networks
Regression	Neural Networks
JavaScript	NodeJS

4. EVALUATION AND RESULTS

4.1 Datasets

We collected metadata about various courses from MOOC platforms and our internal Learning Management System (LMS) using REST APIs. We fetched data from categories relevant to Software Development and Design. The distribution of the number of courses fetched from different platforms is shown in Table 1. There are 13,600 courses from Udemy, 1,050 courses from edX, and 49,500 courses from our LMS in the Software Development and Design category. The output from the REST APIs was in JSON format and each had a different schema. Hence, we selected MongoDB, a NoSQL database to store the retrieved data.

We apply text pre-processing on course metadata. Specifically, the course descriptions from Udemy contain HTML tags. We parse the HTML tags in course descriptions using Beautiful Soup [19]. We remove stopwords and apply Lemmatization and Stemming to reduce words to their base forms. We also create custom stopwords manually by analyzing the topic modeling output. We stored pre-processed data in MongoDB for further processing and evaluation.

4.2 Evaluating extracted concepts

We apply Latent Dirichlet Allocation (LDA), a topic modeling algorithm to infer topics from the course descriptions. We extract five topics from each course description. Each topic is a vector representation that not only indicates the words belonging to the topic but also the probability of the words belonging to the topic. From the topical distribution for the course description, the words from the topic with maximum probability were considered and stored against each course metadata as tags in the database. Figure 2 shows the description and the tags obtained for a Javascript course in Udemy.

To evaluate the concepts extracted from the course description, we apply the Overlap Coefficient to measure the similarity between the concepts extracted from the course description and concepts tagged by Udemy. The overlap coefficient, or Szymkiewicz–Simpson coefficient, is a similarity measure that measures the overlap between two finite sets [1]. It is related to the Jaccard index and is defined as the size of the intersection divided by the smaller of the size of the two sets. Mathematically, we define the Concept overlap coefficient as

$$concept\_overlap(X, Y) = \frac{|X \cap Y|}{min(|X|, |Y|)} \quad (4)$$

where concept_overlap(X, Y) captures the average concept overlap between two sets X and Y, X is the concepts extracted from topic modeling, Y is the concepts tagged in course descriptions of Udemy dataset, and N is the number of course descriptions in the dataset. We observed the average concept overlap coefficient to be 0.97. This shows that the concepts extracted from the topic modeling algorithm quite well capture the relevant concepts covered in the course. Udemy's course description contains a maximum of two concepts tagged. We further analyzed how well our approach is able to identify the other concepts from course descriptions, not captured in Udemy's concepts tag. We performed a quantitative analysis with 20 Subject Matter

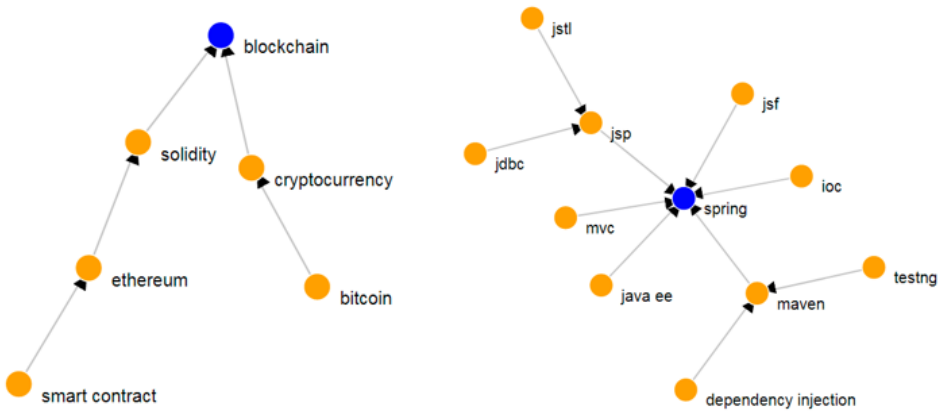

Figure 4: Learning maps for Blockchain and Java Spring framework

Experts (SMEs). The SMEs are having experience ranging from 5-10 years and have worked on different technologies in IT companies. We randomly sampled 100 courses offered on Udemy and provided five courses to each SME along with inferred concepts for each course. The SMEs were asked to provide their response on whether these inferred concepts are relevant for the course or not. We computed the accuracy considering SME's responses as true labels. We observed the accuracy of inferred concepts to be 0.73.

4.3 Evaluating concept Prerequisite Relations

There are no ground truth labels available for inferred prerequisite relationships. To assess the effectiveness of prerequisite pairs generated by our approach, we conducted a quantitative analysis with 25 SMEs to identify if a concept c_1 is a prerequisite for another concept c_2. We created five groups with 5 SMEs in each group. We randomly sampled 250 concept prerequisite pairs. Each group is provided with 50 concept prerequisite pairs. We used the Majority voting approach to aggregate their responses. We computed the accuracy of these pairs considering the SME's response as ground truth labels. We observed the accuracy of concept prerequisite pairs to be 0.82. We also measure inter-rater agreement amongst experts using Fleiss' Kappa [14]. Fleiss' Kappa is a statistical measure for assessing the reliability of agreement between a fixed number of raters when assigning categorical ratings to a number of items or classifying items. If the raters are in complete agreement then $\kappa = 1$. If there is no agreement among the raters (other than what would be expected by chance) then $\kappa \leq 0$. We observed κ coefficient to be 0.74 which indicates a level of strong agreement among the raters. We believe some level of disagreement may be due to the fact that prerequisites can be subjective [12] i.e. it is difficult to get consensus for some pairs of concepts. Different individuals may have different experiences of acquiring knowledge on specific topics, and this may lead to different opinions of the prerequisite requirement for a topic. Some of the extracted prerequisite relationships are shown in Table 3.

5. CHALLENGES

Some of the challenges that we faced while building the concepts graph.

1. For some concepts extracted from the course description we had disambiguation issues when checked in Wikipedia. For example, Java can refer to a programming language or an island in Indonesia. To deal with this issue, we pass the extracted concepts to google search API [7] and fetch the Wikipedia article that is ranked higher in the search results. Due to the popularity of these software concepts, we observe that relevant results were returned by picking the higher ranked Wikipedia article from the search queries.
2. Our inference of prerequisite relationships is based on reference scores computed from Wikipedia articles of the concepts. These scores may not always provide accurate results. It is possible that articles for some of the concepts may have high reference scores for concepts that are derived from it and not vice-versa.

6. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed our approach to infer prerequisite relations between concepts and build the concept graph. The proposed method does not require manually annotated data which was the major drawback of supervised learning approaches. We use relevant data sources in different steps to incorporate relevant and rich semantic information to infer prerequisite relations accurately. To validate our results, we performed both quantitative and qualitative evaluations. The identified concept prerequisite pairs were evaluated by subject matter experts. We observed an accuracy of 0.82 for the inferred prerequisite relations. We built the concept graph from the prerequisite relation pairs and demonstrated few examples of the learning maps generated from the concept graph. Learning maps can be used in many applications ranging from content-based recommendation systems to more sophisticated online tutoring systems etc. As future work, we plan to extend our research by creating a personalized curriculum planner system that captures the concepts learners currently know and what they want to learn. By leveraging this information, the system will create a personalized learning plan for them using their input information and prerequisite relations. Although, our approaches are not limited to the software domain, we plan to carry out further studies and experimentation to measure the system's generalization to other domains.

7. REFERENCES

[1] J. Adler and I. Parmryd. Quantifying colocalization by correlation: the pearson correlation coefficient is superior to the mander's overlap coefficient. *Cytometry Part A*, 77(8):733–742, 2010.

[2] F. Almeida and G. Xexéo. Word embeddings: A survey, 2019.

[3] C. Alzetta, A. Miaschi, G. Adorni, F. Dell'Orletta, F. Koceva, S. Passalacqua, and I. Torre. Prerequisite or not prerequisite? that's the problem! an nlp-based approach for concept prerequisite learning. In *CLiC-it*, 2019.

[4] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, 2003.

[5] M. Bostock, V. Ogievetsky, and J. Heer. D3 data-driven documents. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2301–2309, Dec. 2011.

[6] G. Bouma. Normalized (pointwise) mutual information in collocation extraction. *Proceedings of GSCL*, pages 31–40, 2009.

[7] A. Casagrande. Google Search API. https://github.com/abenassi/Google-Search-API, 2020. [Online; accessed 05-Mar-2021].

[8] A. A. Hagberg, D. A. Schult, and P. J. Swart. Exploring network structure, dynamics, and function using networkx. In G. Varoquaux, T. Vaught, and J. Millman, editors, *Proceedings of the 7th Python in Science Conference*, pages 11 – 15, Pasadena, CA USA, 2008.

[9] J. G. Jardine. Automatically generating reading lists. In *Technical Report UCAM-CL-TR-848*, 02 2014.

[10] L. Jiang, S. Hu, M. Huang, Z. Wang, J. Yang, X. Ye, and W. Zheng. Massistant: A personal knowledge assistant for mooc learners. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations*, pages 133–138, 2019.

[11] C. Liang, J. Ye, S. Wang, B. Pursel, and C. L. Giles. Investigating active learning for concept prerequisite learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[12] C. Liang, J. Ye, Z. Wu, B. Pursel, and C. Giles. Recovering concept prerequisite relations from university course dependencies. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

[13] W. Lu, Y. Zhou, J. Yu, and C. Jia. Concept extraction and prerequisite relation learning from educational data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9678–9685, 2019.

[14] M. L. McHugh. Interrater reliability: the kappa statistic. *Biochemia medica*, 22(3):276–282, 2012.

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

[16] S. Pal, V. Arora, and P. Goyal. Finding prerequisite relations between concepts using textbook. *arXiv preprint arXiv:2011.10337*, 2020.

[17] 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)*, pages 1447–1456, 2017.

[18] R. Řehůřek and P. Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA.

[19] L. Richardson. Beautiful soup documentation. *April*, 2007.

[20] S. Roy, M. Madhyastha, S. Lawrence, and V. Rajan. Inferring concept prerequisite relations from online educational resources. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9589–9594, 2019.

[21] I. Stack Exchange. Stack Exchange Data Dump. https://archive.org/details/stackexchange, 2021. [Online; accessed 05-Mar-2021].

[22] StackExchange. StackExchange. https://api.stackexchange.com/docs/synonyms-by-tags, 2020. [Online; accessed 05-Mar-2021].

[23] Y. Yang, H. Liu, J. Carbonell, and W. Ma. Concept graph learning from educational data. *WSDM 2015 - Proceedings of the 8th ACM International Conference on Web Search and Data Mining*, pages 159–168, 02 2015.

[24] J. Yu, G. Luo, T. Xiao, Q. Zhong, Y. Wang, J. Luo, C. Wang, L. Hou, J. Li, Z. Liu, et al. Mooccube: A large-scale data repository for nlp applications in moocs. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3135–3142, 2020.