What Should I Learn First: Introducing LectureBank for NLP Education and Prerequisite Chain Learning

Irene Li **Alexander R. Fabbri** **Robert R. Tung** **Dragomir R. Radev**
Department of Computer Science, Yale University
{irene.li,alexander.fabbri,robert.tung,dragomir.radev}@yale.edu

Abstract

Recent years have witnessed the rising popularity of Natural Language Processing (NLP) and related fields such as Artificial Intelligence (AI) and Machine Learning (ML). Many online courses and resources are available even for those without a strong background in the field. Often the student is curious about a specific topic but does not quite know where to begin studying. To answer the question of "what should one learn first,"we apply an embedding-based method to learn prerequisite relations for course concepts in the domain of NLP. We introduce LectureBank, a dataset containing 1,352 English lecture files collected from university courses which are each classified according to an existing taxonomy as well as 208 manually-labeled prerequisite relation topics, which is publicly available [1]. The dataset will be useful for educational purposes such as lecture preparation and organization as well as applications such as reading list generation. Additionally, we experiment with neural graph-based networks and non-neural classifiers to learn these prerequisite relations from our dataset.

Introduction

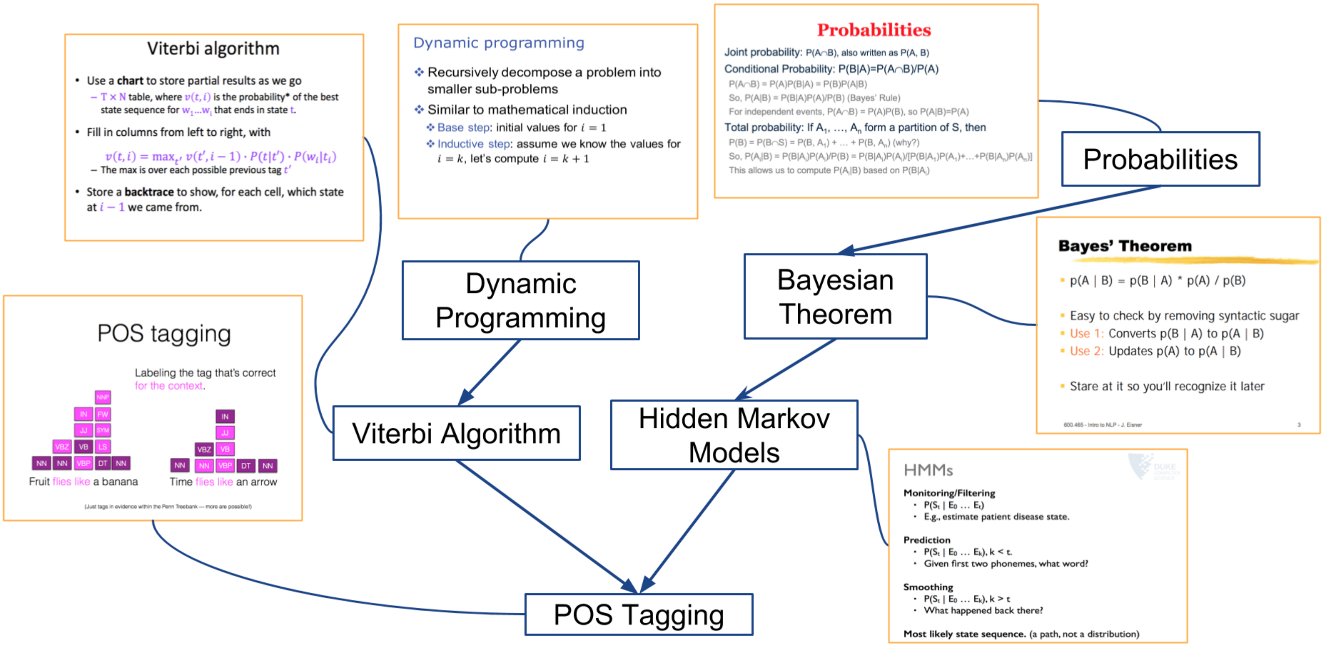
As more and more online courses and resources are becoming accessible to the general public, researching advanced topics has become more feasible. A large amount of educational material is available online, although it is not structured in an organized way. (Margolis and Laurence 1999) suggested that *concepts* are one of the most fundamental constructs and that having an order for learning and organizing them is essential for acquiring new knowledge. To be able to capture the concept organization and dependencies for NLP, we study the problem of building concept prerequisite chains. We treat each concept as a single vertex, and learn the dependencies to ultimately build a concept graph as in (Gordon et al. 2016). We define a *prerequisite* to be the directed dependency between two vertices. Once prerequisite relations among concepts are learned, these relations can be used for downstream tasks and applications such as generating reading lists for readers based on a query as well as curriculum planning (Gordon et al. 2017).

Imagine the scenario in Figure 1 in which a student has some basic knowledge of NLP but wants to learn a specific new concept such as *POS tagging*. In order to fully understand this concept, he or she should have an understanding of prerequisite concepts such as *Viterbi Algorithm* and *Markov Models*, as well as the prerequisites for these concepts: *Dynamic Programming Bayes Theorem* and *Probabilities*. Although many search engines can provide relevant documents or learning resources, most of the results are based on relevancy and semantics while few have the ability to provide a reasonable list based on a path to learning a new concept. Additionally, we might want to recommend the corresponding learning materials such as lecture files for each of the concepts. Thus, for educational purposes, we aim to learn the prerequisite relations of each concept pair and eventually apply them in a search engine. Given a query concept word or phrase, the search engine would provide the study materials corresponding to the prerequisites of the query concept.

Recent work has focused on extracting and learning concept dependencies from scientific corpora including the ACL Anthology (Bird et al. 2008) as well as from online courses (Gordon et al. 2016; Pan et al. 2017a; Liu et al. 2016). On the other hand, we are more interested in learning materials such as blogs, surveys and papers, and in this paper, we focus on lecture files, which are usually well-organized and contain a focused topic. We manually collected lecture files by inspection of the file contents and annotated each lecture file according to a taxonomy of 305 topics. We used our LectureBank corpus and a recently published corpus TutorialBank (Fabbri et al. 2018) as training data. We annotated prerequisite relations for each concept pair from a list of 208 concepts provided in (Fabbri et al. 2018). These 208 concepts differ in granularity and scope from the 305 taxonomy topics; they consist of topics for which one could conceivably write a survey and are thus neither too fine-grained or large in scope. Similarly to (Pan et al. 2017a), we focus on learning embedded representations of the concepts. We test the effectiveness of standard classifiers as well as the recently introduced neural link-prediction approaches of Variational(Kipf and Welling 2016b) and vanilla Graph Autoencoders (Schlichtkrull et al. 2018) to discover prerequisite relations.

Our main contributions are the following. First, we introduce our LectureBank dataset with 1,352 English lecture files (51,939 slides) classified according to an existing taxonomy. The dataset can be used directly as study

[1] https://github.com/Yale-LILY/LectureBank



Viterbi algorithm

- Use a **chart** to store partial results as we go
  - $T \times N$ table, where $v(t, i)$ is the probability* of the best state sequence for $w_1 \ldots w_i$ that ends in state t.
- Fill in columns from left to right, with

  $v(t, i) = \max_{t'} v(t', i-1) \cdot P(t|t') \cdot P(w_i|t_i)$
  - The max is over each possible previous tag t'
- Store a **backtrace** to show, for each cell, which state at $i-1$ we came from.

Dynamic programming

- Recursively decompose a problem into smaller sub-problems
- Similar to mathematical induction
  - Base step: initial values for $i = 1$
  - Inductive step: assume we know the values for $i = k$, let's compute $i = k + 1$

Probabilities

Joint probability: P(A∩B), also written as P(A, B)

Conditional Probability: P(B|A)=P(A∩B)/P(A)

P(A∩B) = P(A)P(B|A) = P(B)P(A|B)

So, P(A|B) = P(B|A)P(A)/P(B) (Bayes' Rule)

For independent events, P(A∩B) = P(A)P(B), so P(A|B)=P(A)

Total probability: If A_1, …, A_n form a partition of S, then

P(B) = P(B∩S) = P(B, A_1) + … + P(B, A_n) (why?)

So, P(A_i|B) = P(B|A_i)P(A_i)/P(B) = P(B|A_i)P(A_i)/[P(B|A_1)P(A_1)+…+P(B|A_n)P(A_n)]

This allows us to compute P(A_i|B) based on P(B|A_i)

Bayes' Theorem

- p(A | B) = p(B | A) * p(A) / p(B)
- Easy to check by removing syntactic sugar
- Use 1: Converts p(B | A) to p(A | B)
- Use 2: Updates p(A) to p(A | B)
- Stare at it so you'll recognize it later

600.465 - Intro to NLP - J. Eisner 3

POS tagging

Labeling the tag that's correct for the context.

(Just tags in evidence within the Penn Treebank — more are possible!)

HMMs

Monitoring/Filtering
- $P(S_t \mid E_0 \ldots E_t)$
- E.g., estimate patient disease state.

Prediction
- $P(S_t \mid E_0 \ldots E_k), k < t$.
- Given first two phonemes, what word?

Smoothing
- $P(S_t \mid E_0 \ldots E_k), k > t$
- What happened back there?

Most likely state sequence. (a path, not a distribution)

Figure 1: An example of prerequisite relations from lecture slides depicted as a directed graph. The direction of an edge is from a prerequisite of a concept to the concept itself. For example, *Hidden Markov Models* is a prerequisite of *POS Tagging*. We illustrate each concept with a slide on that topic, selected from our corpus. The references for the slides starting from POS Tagging and moving clockwise are: (Bamman 2017), (Schneider 2018), (Chang 2018), (Radev 2018), (Eisner 2012) and (Konidaris 2016).

material mainly in the fields of NLP or ML, as it covers high-quality university lectures suitable for entry-level researchers or NLP engineers working at the internet or social media companies. The corpus can also be used for topic modeling of scientific topics in addition to the prerequisite chains learning task. Additional details on the dataset can be found in Section 3. Second, we compare novel graph-based deep learning models, which have shown promise in the task of link prediction, with standard classification methods and demonstrate the importance of oversampling in this task.

Related Work

In this section, we briefly describe related work on prerequisite chain learning using concept graphs as well as recent developments in neural graph-based methods.

Concept Graphs

Previous work collected online courses including Computer Science, Statistics and Mathematics from Massachusetts Institute of Technology , California Institute of Technology, Princeton University and Carnegie Mellon University and proposed an approach for inference within and across course-level and concept-level directed graphs (Liu et al. 2016). We focused on detailed concept-level mainly in the NLP domain. (Gordon et al. 2016) introduced two methods for discovering concept dependency relations automatically from a text corpus: a cross-entropy approach and an information-flow approach. They tested the methods on the ACL Anthology corpus (Bird et al. 2008). Concepts were determined using LDA topic modeling (Blei, Ng, and Jordan 2003), and prerequisite relations are notably found in an unsupervised setting. (Pan et al. 2017b) proposed a representation learning-based method to learn the concepts and a set of novel features to identify prerequisite relations. However, these methods benefit from the help of Wikipedia to perform entity extraction and improve entity representations. However, a Wikipedia page may not always be available for a concept, and we focus on obtaining concept representations solely from our corpus.

Other work has focused on generating prerequisite relations among concepts on a Massive Open Online Courses (MOOCs) corpus (Pan et al. 2017a). They proposed seven types of features covering course concept semantics, course video context and course structure. They cast the prerequisite chain problem as a binary classification problem by evaluating the prerequisite relationship between each concept pair and applying four different binary classifiers. While they constructed three datasets to evaluate their methods, the coverage of the datasets is relatively small, as only up to 244 topics and three domains (Machine Learning, Data Structure and Algorithms and Calculus) are considered. In this paper, we expand the range of domains and the number of courses.

Additionally, a manually-collected and categorized corpus of about 6,300 resources on NLP and the related fields of ML, AI, and IR was recently introduced by (Fabbri et al. 2018). This corpus was released with a list of 208 topics for which prerequisite relationships were annotated, making it a complementary dataset to ours. However, only a single annotator annotated each relation and prerequisite annotations were ternary versus our binary classification. As there have not been any experiments on learning prerequisite chains using this corpus, we choose to take advantage of both TutorialBank and LectureBank to learn prerequisite relations. Besides the 208 topics for prerequisite relationships, they also proposed a university-level NLP course syllabus-like taxonomy of 305 topics, which the surveys, tutorials and resources other than scientific papers are classified into. These 305 topics can be coarse-grained, such as *Natural Language Processing*, *Artificial Intelligence*. Also, there are redundant topics such as *Classification and kNN 1* and *Classification and kNN 2*. In comparison, the 208 topics are more fine-grained and suitable for prerequisite chain learning, although there are some overlap topics with the 305 topics. Of note, as in the TutorialBank dataset, deep learning topics are a major focus as both datasets consist of work from the last few years, and an abundance of tutorials and resources on deep learning have been published in this time, thus explaining this bias.

Graph Convolutional Networks

Using neural networks on structured data structures such as graphs is a difficult problem. Graph Convolutional Networks (GCNs) aim to tackle this task by drawing inspiration from spectral graph theory. (Defferrard, Bresson, and Vandergheynst 2016) design fast localized convolutional filters on graphs for an image-processing task while (Kipf and Welling 2016a) apply GCNs to a number of semi-supervised graph-based classification tasks, reporting faster training times and better predictive accuracy. More recently GCNs have been applied to problems such as machine translation (Bastings et al. 2017) and summarization (Yasunaga et al. 2017). GCNs have also been applied to the task of link prediction or predicting relations among entities. For this task, some, but not all, of the relations among entities are given during training, and the goal is to predict additional unobserved relations during testing. Finding prerequisite relations can be viewed as link prediction, where the vertices are concepts, and the edges are the prerequisite relations, or lack thereof.

The work from (Kipf and Welling 2016b) introduced a non-probabilistic Graph Autoencoder (GAE) as well as the Variational Graph Autoencoder (VGAE). These models build upon work on autoencoders and variational autoencoders for unsupervised learning on graph-structured data for tasks such as predicting links in a citation network. These models combine a GCN encoder with inner product decoders and latent variables in the case of VGAE. (Schlichtkrull et al. 2018) extend GCNs and variational graph autoencoders to model large-scale relational data for the tasks of link prediction and entity classification, and thus we examine the applicability of these models for the prerequisite chains learning task.

LectureBank Dataset

In this section, we introduce the LectureBank dataset, analysis, statistics, annotations, and then compare it with other similar datasets.

Data Collection and Presentation

We collected online lecture files from 60 courses covering 5 different domains, including NLP, ML, AI, deep learning (DL) and information retrieval (IR). For copyright reasons, we are releasing only the links to the individual lectures.

In order to make the course slides more accessible to the user, we have indexed all slide lectures and created a search engine which allows the user to browse courses and slides according to queries. As related material on a subject can come from a variety of courses, we believe gathering them into a single search engine dramatically reduces the amount of time a student will spend searching for relevant material. The search engine will be made publicly available.

Dataset Analysis and Statistics

Our LectureBank dataset focuses on courses from 5 domains, and detailed statistics can be found in Table 1. The table reports the number of courses, lecture files, pages tokens, the average tokens per lecture file and the average tokens per page. For preprocessing, we used the PDFMiner[2] python package to extract the texts from the PDF files, and python-pptx[3] to extract Powerpoint (PPT) presentations. If a course provided both PDF and PPT versions, we kept the PDF files and removed the PPT files.

Additional Annotation

Prerequisite Annotation In addition to using our own dataset for prerequisite chain learning, we make use of a recently introduced corpus of resources on topics related to NLP. (Fabbri et al. 2018) introduced a set of 208 topics on NLP and related fields and annotated all of the concept pairs as one of the following: not a prerequisite, somewhat a prerequisite or a true prerequisite. While the annotations emphasize the breadth of pairs annotated, we think certain dependency relations might not be clear to annotate, such as ”long dependencies” (if A is a prerequisite of B and B is a prerequisite of C, should we count A as a prerequisite of C or not). We aim to improve classification agreement by making the criteria more precise and having additional annotators annotate the topics. Each of our annotators annotated each prerequisite relation, and we report high inter-annotator agreement. Our annotators consist of two PhD students working on NLP. We obtained a Cohen’s kappa (Cohen 1960) of 0.7 which according to (Landis and Koch 1977) is considered substantial agreement. We asked the annotators the following question for each topic pair (A, B): is A a prerequisite of B; i.e., do you think learning the concept A will help one to learn the concept B? Even if the distance between two concepts in a potential concept graph is large, if one topic is typically learned before the other in a university

[2] https://pypi.python.org/pypi/pdfminer3k/

[3] https://python-pptx.readthedocs.io/en/latest/#

Domain	#courses	#lectures	#slides	#tokens	#tokens/lecture	#tokens/slide
NLP	35	764	29,661	1,570,578	2,055.73	52.95
ML	12	260	10,720	866,728	3,333.57	80.85
AI	5	101	4,911	265,460	2,628.32	54.05
DL	4	148	3,270	582,502	3,935.82	178.14
IR	4	79	3,377	157,808	1,997.57	46.73
Overall	60	1352	51,939	3,443,076	2,546.65	66.29

Table 1: LectureBank Dataset Statistics: within each domain, we have a certain number of courses; each course consists of lectures files; each lecture file has multiple individual slides. The contrasting number of tokens per slide for DL is a result of the small sample size and the courses chosen.

course and is helpful in building up knowledge for learning another concept, we considered this earlier concept a prerequisite of the other. Thus, while the criteria may be subjective, we direct the annotators to refer to standard university curricula for unclear prerequisite pairs. Only binary *yes* (positive) or *no* (negative) answers are possible. This is in contrast to the ternary classification of (Fabbri et al. 2018), who report a kappa score of .3 on the same prerequisite pairs. We chose binary annotation over the ternary annotation of (Fabbri et al. 2018) as this is the same setup as in related work such as (Pan et al. 2017a). Additionally, the choice of binary as opposed to ternary classification pertains to the precision and recall trade-off which we discuss in the results section below. We decided that concepts which are ”somewhat prerequisites” as in (Fabbri et al. 2018) should be labelled as prerequisites so that they are not missed as potential missing areas of knowledge.

We took the intersection of the two annotators' annotations, which resulted in a labeled directed concept graph with 208 concept vertices and 921 edges. If concept A is a prerequisite of concept B, the edge direction goes from concept vertex A to concept vertex B. We also observed some cycles between a pair of vertices within the concept graph. We found 12 such pairs in our labeled concept graph. These pairs consist of very closely related topics such as *Domain Adaptation* and *Transfer Learning* and *LDA* and *Topic Modeling*, suggesting that in the future we may combine these pairs into a single concept. There are 7 independent topics which have no prerequisite relationships with the rest of the topics. They are: *Morphological Disambiguation, Weakly-supervised learning, Multi-task Learning, ImageNet, Human-robot interaction, Game playing in AI, data structures and algorithms*. The topics were proposed by (Fabbri et al. 2018), and so we chose to keep all the topics in our experiments.

We also list the concept vertices that have the largest in-degree and out-degree in Table 2. In-degree illustrates that the concept vertex has many prerequisite concepts; out-degree illustrates that the concept vertex is a prerequisite to many other concepts. The concepts with large in-degree are advanced concepts which require much background knowledge in order to be learned well, while the list of concepts with large out-degree are more fundamental concepts. We also observed the longest path in the constructed concept graph, which consists of 14 concepts in the path: *Matrix Multiplication, Differential Calculus, Backpropagation, Backpropagation Through Time, Artificial Neural Network, Word Embeddings, Word2Vec, Seq2Seq, Neural Machine Translation, BLEU, IBM Translation Models, ROUGE, Automatic Summarization, Scientific Article Summarization*.

Classification We used the TutorialBank taxonomy from (Fabbri et al. 2018), which contains 305 topics of varying granularity. Based on a university-level NLP course syllabus, the TutorialBank taxonomy was then expanded to other topics from other courses from IR, AI, ML and DL. The 305 taxonomy topics cover a wide range of topics in NLP area, and we only use these topics during our manual labeling of our lecture dataset as an additional annotation work. We manually classified all 1,352 LectureBank lecture files into the TutorialBank taxonomy. In Table 3 we show the top 10 more frequent TutorialBank taxonomy labels for our corpus classification.

Another taxonomy which was considered for our classification was the 2012 ACM Computing Classification System (CCS) [4], a poly-hierarchical ontology that can be utilized in semantic web applications. The system is hierarchically structured into four levels; *Machine Translation*, for example, can be found in the following branch: *Computing Methodologies* → *Artificial Intelligence* → *Natural Language Processing* → *Machine Translation*. The *Artificial Intelligence* directory has 8 subcategories including *Natural Language processing*, and 8 subcategories under the *Natural Language Processing* directory. However, rather than focusing on the larger scope of computing or even AI in general, our main focus in NLP. Compared with CCS, the TutorialBank taxonomy covers detailed categorization, focusing on NLP and related fields insofar as they related to NLP, making it very suitable for our desired classification.

Vocabulary Alongside our lecture files we also provide a vocabulary list containing 1,221 terms with the help of the LectureBank Corpus, the vocabulary can be used as an in-domain reference while it mainly focuses on NLP and related areas. Rather than using Wikipedia to create a corpus vocabulary, we took the union of three topic sets: the TutorialBank taxonomy topics, the 208 topics labeled for prerequisite chains and the topics extracted from LectureBank. The first two parts are provided by (Fabbri et al. 2018), and we contributed more fine-grained topic words from our LectureBank. Different from (Fabbri et al. 2018) who manually propose topic terms, we propose topic terms in an automatical way: we found keywords from LectureBank by taking

[4] https://www.acm.org/publications/class-2012

Most in-degree Concept Vertices	Count	Most out-degree Concept Vertices	Count
Neural Machine Translation	19	Data Structures and Algorithms	106
Variational Autoencoders	15	Probabilities	105
Stack LSTM	13	Linear Algebra	98
Seq2seq Models	13	Matrix Multiplication	72
Highway Networks	12	Bayes Theorem	59
DQN	12	Conditional Probability	58
Bidirectional Recurrent Neural Networks	11	Differential Calculus	21
Convolutional Neural Networks	11	Activation Functions	20
Multilingual Word Embeddings	11	Loss Function	19
Capsule Networks	11	Entropy	17
Topic Modeling	10	Data Preprocessing	17
Neural Turing Machine	10	Backpropagation	17
Recursive Neural Networks	10	Artificial Neural Networks	16
Attention Models	10	Backpropagation Through Time	14
Generative Adversarial Networks	10	Information Theory	13

Table 2: Concept vertices from our annotated concept graph with the largest in-degree and out-degree

Topic	Count
Introduction to Neural Networks	92
Machine Learning Resources	56
Information Retrieval	31
Classification	30
Probabilistic Reasoning	29
Word Embeddings	25
Hidden Markov Models	20
NLP Resources	20
Machine Translation Basics	19
Monte Carlo Methods	19

Table 3: Counts of the most frequent taxonomy topic labels of the LectureBank files

the header section of each individual lecture slide and post-processing and filtering that list. This method can be extended to other online resources such as blog posts and papers to enlarge the vocabulary. In the future, the vocabulary can be potentially used as additional concepts for creating a concept graph in an automated manner. This vocabulary can also serve an educational purpose for preparing lecture topics.

Comparison with Similar Datasets

MOOCs In a recent research, (Pan et al. 2017a) evaluated on three corpora extracted from Massive Open Online Courses (MOOCs), containing three domains (Machine Learning, Data Structures and Algorithms and Calculus). They are the first to study prerequisite relations in a MOOC corpus. The corpora contain video subtitles and speech scripts, and the number of topics ranges from 128 to 244. They used Wikipedia to help obtain entity representations.

Coursera and XuetangX Similarly, (Pan et al. 2017b) constructed four course corpora in two domains (Computer Science and Economics) and in two languages (English and Chinese) from video captions. In each corpus, the number of courses varies from 5 to 18. Also, they have a significant number of candidate concepts for each corpus which ranges from 27,571 to 79,009. Similarly, they also benefit from the help of Wikipedia[5] and the Baidu encyclopedia[6] when learning prerequisite relations.

ACL Anthology Scientific Corpus A corpus for prerequisite relations among topics was presented by(Gordon et al. 2016). They used topic modeling to generate a list of 305 topics from the ACL Anthology (Bird et al. 2008). While the focus of this corpus is NLP, the resources come from academic papers, while we focus on tutorials from TutorialBank and our corpus of lectures. Additionally, they only annotate a subset of the topics for prerequisite annotations while we annotate two orders of magnitude larger in terms of prerequisite edges and show strong inter-annotator agreement.

Prerequisite Chain Learning

In this section, we introduce our two-step framework: concept feature learning and a recently proposed neural graph-based method in addition to traditional classification methods.

Concept Feature Learning

The first step is to extract concept vectors from various documents or lectures. We trained a Doc2Vec model (Le and Mikolov 2014), an unsupervised model which can train dense vectors as representations for variable-length texts such as sentences or documents. Our Doc2Vec model was trained using Gensim[7]. We set the dimension of the document vector to be 300, and the model was trained in the manner of Distributed Memory Model of Paragraph Vectors (PV-DM) (Le and Mikolov 2014). We obtained document representations from our LectureBank data and TutorialBank data from (Fabbri et al. 2018) separately as well as after combining the corpora. We then took the trained Doc2Vec model to infer the embedded vector of a given concept. This dense vector topic representation is then used as input to our models.

[5] https://dumps.wikimedia.org/enwiki/20170120/

[6] https://baike.baidu.com/

[7] https://radimrehurek.com/gensim/

Prerequisite Chain Learning as Linking Prediction

Concept prerequisite chain learning can be viewed as a type of link prediction problem with prerequisite relations as links among concepts. The goal of this model is to learn a directed graph $\mathcal{G} = (V, E)$, where the vertices V correspond to concepts and the edges E correspond to the prerequisite relationships (if any) among the concepts. The model takes as input an adjacency matrix A and a feature matrix $\mathbf{X}$ of size $n \times d$, where n is the number of vertices/concepts and d is the number of input features per vertex.

Graph Autoencoders In the case of the non-probabilistic GAE, a $n \times f$ embeddings matrix $\mathbf{A}$, where f is the size of the embeddings, is parameterized by a two-layer GCN:

$$\mathbf{Z} = GCN(\mathbf{X}, \mathbf{A})$$

and the reconstructed adjacency matrix is:

$$\hat{\mathbf{A}} = \sigma(\mathbf{Z}\mathbf{Z}^T)$$

Variational Graph Autoencoders As an extension to the above model, the stochastic latent variables z_i, summarized in $\mathbf{Z}$, are introduced and $\mathbf{Z}$ is modeled by a Gaussian prior distribution $\prod_i \mathcal{N}(\mathbf{z}_i, \mathbf{0}, \mathbf{I})$. As in (Kipf and Welling 2016b), we use the following inference model parameterized by a two-layer GCN:

$$q(\mathbf{Z}|\mathbf{X}, \mathbf{A}) = \prod_{i=1}^{N} q(\mathbf{z}_i|\mathbf{X}, \mathbf{A})$$

where

$$q(\mathbf{z}_i|\mathbf{X}, \mathbf{A}) = \mathcal{N}(\mathbf{z}_i|\boldsymbol{\mu}_i, diag(\sigma_i^2)),$$

the matrix of mean vectors is $\boldsymbol{\mu} = GCN_{\boldsymbol{\mu}}(\mathbf{X}, \mathbf{A})$ and $\log \boldsymbol{\sigma} = GCN_{\boldsymbol{\sigma}}(\mathbf{X}, \mathbf{A})$. The two-layer GCN is defined as:

$$GCN(\mathbf{X}, \mathbf{A}) = \widetilde{\mathbf{A}} ReLU(\widetilde{\mathbf{A}}\mathbf{X}\mathbf{W}_0)\mathbf{W}_1,$$

where $\mathbf{W}_i$ represents the weight matrix at level i and $\widetilde{\mathbf{A}} = \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$ is the symmetrically normalized adjacency matrix. The following generative model results:

$$P(\mathbf{A}|\mathbf{Z}) = \prod_{i=1}^{N}\prod_{j=1}^{N} p(\mathbf{A}_{ij}|\mathbf{z}_i, \mathbf{z}_j),$$

where

$$p(A_{ij} = 1|\mathbf{z}_i, \mathbf{z}_j) = \sigma(\mathbf{z}_i^T \mathbf{z}_j)$$

The variational lower bound $\mathcal{L}$ is optimized *w.r.t.* the variational parameters $\mathbf{W}_i$:

$$\mathcal{L} = E_{q(\mathbf{Z}|\mathbf{X},\mathbf{A})}[\log p(\mathbf{A}|\mathbf{Z})] - KL[q(\mathbf{Z}|\mathbf{X}, \mathbf{A})||p(\mathbf{Z})]$$

where $KL[q(\cdot)||p(\cdot)]$ is the Kullback-Leibler divergence between q and p.

Experiments

We treat the predictions of our prerequisite chain learning problem as a binary classification result among pairs of concepts and report precision, recall and F1 scores. We report results on 5 fold cross validation where the test set contains 10% of the positive prerequisite labels, following (Kipf and Welling 2016b). For the task of learning prerequisite chains, positive samples are usually rare, leading to imbalanced datasets. More specifically, we first divided training and testing where it was guaranteed that 10% positive samples were selected in the testing set, then added the same number of negative samples into the testing set and took the rest samples as the training set. Finally, during each run, we oversampled on the training set, and report an average score of the 5 runs. We have 921 positive concept pairs and 41,829 negative concept pairs in total before oversampling.

Method	Precision	Recall	F1
TutorialBank			
NB	0.761	0.453	0.567
SVM	0.832	0.703	**0.761**
LR	0.819	0.604	0.693
RF	**0.871**	0.459	0.599
GAE	0.634	0.884	0.725
VGAE	0.599	**0.895**	0.717
LectureBank			
NB	**0.853**	0.611	0.710
SVM	0.835	0.668	**0.740**
LR	0.840	0.640	0.724
RF	0.831	0.624	0.712
GAE	0.577	0.905	0.705
VGAE	0.545	**0.921**	0.684
TutorialBank + LectureBank			
NB	0.614	0.670	0.641
SVM	**0.824**	0.688	**0.748**
LR	0.794	0.613	0.690
RF	0.787	0.519	0.625
GAE	0.594	0.899	0.715
VGAE	0.578	**0.916**	0.708

Table 4: Experiment results using oversampling with Doc2Vec concept representations trained on three settings: TutorialBank, LectureBank and TutorialBank combined with LectureBank.

Models

Binary classifiers We compared the neural graph-based methods with the following classifiers: Naïve Bayes classifier (*NB*), SVM with linear kernel (*SVM*), Logistic Regression (*LR*) and Random Forest classifier (*RF*). After obtaining the concept representations for each possible concept pair as described above, we concatenated the source concept and target concept embeddings together, used the corresponding prerequisite chain label as the class label and fit the classifiers.

GAE We used the same concept embeddings described above as vertex features for the GAE model. We followed the same parameters from (Kipf and Welling 2016b). We trained using gradient descent on batches the size of the entire training dataset for 200 epochs using the Adam optimizer (Kingma and Ba 2014) and a learning rate of 0.01. The two-layer GCN encoding contains 32 hidden units in the first layer and 16 hidden units in the second layer.

Results and Analysis

As shown in Table 4, we report precision, recall and F1 scores of our embedding-based concept representation with

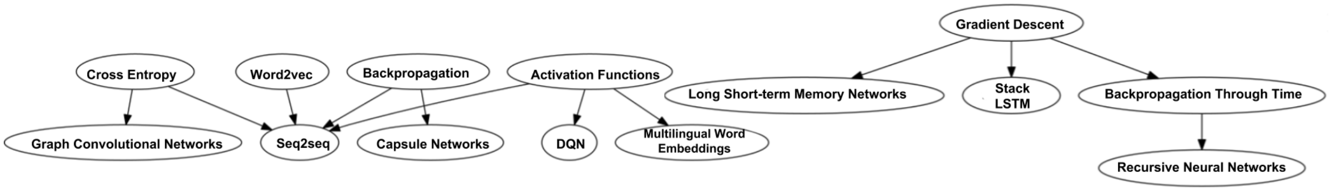

Gradient Descent
Cross Entropy
Word2vec
Backpropagation
Activation Functions
Long Short-term Memory Networks
Stack LSTM
Backpropagation Through Time
Graph Convolutional Networks
Seq2seq
Capsule Networks
DQN
Multilingual Word Embeddings
Recursive Neural Networks

Figure 2: A subset of the recovered concept graph. The diagram is based on predictions on our test data. Some dependencies may be missing if the corresponding concept pairs were not in the test data or were predicted to have a negative label, e.g., the potential edge between *Backpropagation Through Time* and *Backpropagation*.

the Naïve Bayes classifier, SVM with linear kernel, Logistic Regression and Random Forest classifier, along with the vanilla graph autoencoder (*GAE*) and variational graph autoencoder (*VGAE*). The concept representation was trained using Doc2Vec under three different settings: only using TutorialBank, only using LectureBank, and on the combination of TutorialBank + LectureBank.

For the four binary classifiers, we observe a high precision and a low recall in all three different Doc2vec model settings. On the other hand, the GAE and VGAE show high recall and low precision. In general, SVM beats the other methods with high F1 scores among all corpora. The SVM and other basic classifiers benefit greatly from oversampling; we found that initial experiments with imbalanced datasets yielded very poor results. We modified the adjacency input to allow for parallel edges in a multi-graph to account for oversampled inputs in the GAE and VGAE, but the changes in performance were minimum. Intuitively, these models explicitly model the desired concept graph structure and are able to represent features in the vertices and propagate them through the graph. However, these specific networks were applied to the case of citation networks in which case parallel edges are non-existent. Thus we found that the SVM performed better in the general binary classification setting when using oversampling.

In terms of precision and recall for downstream tasks such as a search engine, for two classifiers with the same F1 score, we prefer the one with a higher recall. This coincides with our desired interface; we want the user to be able to mark subjects which are already known and be presented with potential prerequisites which are not known. We would rather give the student more concepts which they need to know rather than fewer in which case they may miss important fundamental knowledge.

Although the highest F1 score was achieved when topic embeddings were trained on TutorialBank via SVM classifier, the variance between the three datasets is not significant. A potential explanation might be the that TutorialBank and LectureBank contain similar content and coverage. TutorialBank has notably four times as many documents and some notably longer resources such as topic surveys than LectureBank, which may improve the performance of our Doc2Vec document representation. However, we can see that the F1 score is only slightly higher than that of LectureBank. Additionally, the list of concepts was provided by (Fabbri et al. 2018) for the TutorialBank dataset, so we expected these topics to be broadly included in the TutoriaBank dataset.

Concept Graph Recovery

According to the F1 score, the highest-performing model is SVM with Doc2Vec trained on TutorialBank. To demonstrate the effectiveness of our model, we took a single training and testing fold from our 5-fold cross-validation experiments randomly and tried to recover the concept graph on the corresponding testing data by predicting the labels. Figure 2 shows a subset of the recovered concept graph containing 14 vertices and 12 edges. We observe some reasonable paths, for example: *Gradient Descent*, *Backpropagation Through Time*, *Recursive Neural Networks*. Another reasonable dependency relation shows that *Seq2seq* is dependent upon multiple prerequisites such as *Word2vec*, *Backpropagation* and *Activation Functions*.

Conclusion and Future Work

In this paper, we introduced LectureBank, a collection of 1,352 English lecture slides from 60 university-level courses and their corresponding class label annotations. In addition, we extracted 1,221 concepts automatically from LectureBank which serve as additional references for in-domain vocabulary. We also release annotation of prerequisite relation pairs on 208 concepts. These annotations will be useful as both an educational tool for the NLP community and a corpus to promote research on learning prerequisite relations.

In the future, we plan to expand LectureBank by enriching the coverage of courses, e.g., adding courses from medical information retrieval or linguistics, and we are planning to increase the corpus size to over 100 courses shortly. To take advantage of LectureBank and the prerequisite chains we have learned so far, we are planning to additionally classify each lecture into one of the prerequisite topics, thus making a search engine which can provide learning materials with the help of prerequisite relations. The search engine will first figure out the prerequisites of a query concept, and then it will be able to provide a list of lectures or even specific pages from them for each prerequisite concept, and in the future, it will take user experience as input to provide better resources for each specific user. Another part of future work is to automatically extract such topics. Finally, we plan to apply our prerequisite chain learning model to other applications such as reading list generation and survey generation.

References

Bamman, D. 2017. Sequence Labeling 1. Course 159/259, UC Berkeley, University Lecture.

Bastings, J.; Titov, I.; Aziz, W.; Marcheggiani, D.; and Simaan, K. 2017. Graph convolutional encoders for syntax-aware neural machine translation. In *EMNLP 2017*.

Bird, S.; Dale, R.; Dorr, B. J.; Gibson, B. R.; Joseph, M. T.; Kan, M.; Lee, D.; Powley, B.; Radev, D. R.; and Tan, Y. F. 2008. The ACL anthology reference corpus: A reference dataset for bibliographic research in computational linguistics. In *LREC 26 May - 1 June 2008*.

Blei, D. M.; Ng, A. Y.; and Jordan, M. I. 2003. Latent Dirichlet Allocation. *Journal of machine Learning research* 3(Jan):993–1022.

Chang, K.-W. 2018. Viterbi and Forward Algorithms. Course CS6501, University of California, Los Angeles, University Lecture.

Cohen, J. 1960. A Coefficient of Agreement for Nominal Scales. *Educational and psychological measurement* 20(1):37–46.

Defferrard, M.; Bresson, X.; and Vandergheynst, P. 2016. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. In *NIPS 2016*.

Eisner, J. 2012. Bayes Theorem. Course 601.465/665, Johns Kopkins University, University Lecture.

Fabbri, A. R.; Li, I.; Trairatvorakul, P.; He, Y.; Ting, W. T.; Tung, R.; Westerfield, C.; and Radev, D. R. 2018. Tutorialbank: A manually-collected corpus for prerequisite chains, survey extraction and resource recommendation. *ACL 2018*.

Gordon, J.; Zhu, L.; Galstyan, A.; Natarajan, P.; and Burns, G. 2016. Modeling Concept Dependencies in a Scientific Corpus. In *ACL 2016*.

Gordon, J.; Aguilar, S.; Sheng, E.; and Burns, G. 2017. Structured generation of technical reading lists. In *ACL 2017*, 261–270.

Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization. *CoRR* abs/1412.6980.

Kipf, T. N., and Welling, M. 2016a. Semi-supervised classification with graph convolutional networks. *CoRR* abs/1609.02907.

Kipf, T. N., and Welling, M. 2016b. Variational graph autoencoders. *NIPS Workshop on Bayesian Deep Learning*.

Konidaris, G. 2016. Hidden Markov Models. Course CPS 270, Duke University, University Lecture.

Landis, J. R., and Koch, G. G. 1977. The Measurement of Observer Agreement for Categorical Data. *Biometrics* 159–174.

Le, Q., and Mikolov, T. 2014. Distributed representations of sentences and documents. In *ICML 2014*.

Liu, H.; Ma, W.; Yang, Y.; and Carbonell, J. 2016. Learning Concept Graphs from Online Educational Data. *JAIR* 55:1059–1090.

Margolis, E., and Laurence, S. 1999. *Concepts: Core Readings*. Mit Press.

Pan, L.; Li, C.; Li, J.; and Tang, J. 2017a. Prerequisite relation learning for concepts in moocs. In *ACL 2017*, volume 1, 1447–1456.

Pan, L.; Wang, X.; Li, C.; Li, J.; and Tang, J. 2017b. Course concept extraction in moocs via embedding-based graph propagation. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, 875–884.

Radev, D. 2018. Probabilities. Course CPSC 477/577, Yale University, University Lecture.

Schlichtkrull, M.; Kipf, T. N.; Bloem, P.; van den Berg, R.; Titov, I.; and Welling, M. 2018. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*, 593–607. Springer.

Schneider, N. 2018. Algorithms for Hmms. Course COSC-572/LING-572, Georgetown University, University Lecture.

Yasunaga, M.; Zhang, R.; Meelu, K.; Pareek, A.; Srinivasan, K.; and Radev, D. 2017. Graph-based neural multi-document summarization. In *CoNLL 2017*.