One-Class Order Embedding for Dependency Relation Prediction

Meng-Fen Chiang
Living Analytics Research Centre
Singapore Management University
mfchiang@smu.edu.sg

Ee-Peng Lim
Living Analytics Research Centre
Singapore Management University
eplim@smu.edu.sg

Wang-Chien Lee
The Pennsylvania State University
University Park, PA 16802, USA
wlee@cse.psu.edu

Xavier Jayaraj Siddarth ASHOK
Living Analytics Research Centre
Singapore Management University
xaviera@smu.edu.sg

Philips Kokoh PRASETYO
Living Analytics Research Centre
Singapore Management University
pprasetyo@smu.edu.sg

ABSTRACT

Learning the *dependency relations* among entities and the hierarchy formed by these relations by mapping entities into some order embedding space can effectively enable several important applications, including knowledge base completion and prerequisite relations prediction. Nevertheless, it is very challenging to learn a good order embedding due to the existence of partial ordering and missing relations in the observed data. Moreover, most application scenarios do not provide non-trivial negative dependency relation instances. We therefore propose a framework that performs dependency relation prediction by exploring both rich semantic and hierarchical structure information in the data. In particular, we propose several negative sampling strategies based on graph-specific centrality properties, which supplement the positive dependency relations with appropriate negative samples to effectively learn order embeddings. This research not only addresses the needs of automatically recovering missing dependency relations, but also unravels dependencies among entities using several real-world datasets, such as course dependency hierarchy involving course prerequisite relations, job hierarchy in organizations, and paper citation hierarchy. Extensive experiments are conducted on both synthetic and real-world datasets to demonstrate the prediction accuracy as well as to gain insights using the learned order embedding.

CCS CONCEPTS

• **Computing methodologies** → *Machine learning*; *Learning to rank*;

ACM Reference Format:
Meng-Fen Chiang, Ee-Peng Lim, Wang-Chien Lee, Xavier Jayaraj Siddarth ASHOK, and Philips Kokoh PRASETYO. 2019. One-Class Order Embedding for Dependency Relation Prediction. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '19), July 21–25, 2019,Paris, France.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3331184.3331249

SIGIR '19, July 21–25,2019, Paris, France

ACM ISBN 978-1-4503-6172-9/19/07…$15.00
https://doi.org/10.1145/3331184.3331249

1 INTRODUCTION

Motivation. Graph is a widely adopted data representation in diverse real-world scenarios. Learning representation of nodes and edges in graph structures has been well studied for many graph analytics applications [15][27][4]. In form of low-dimensional vectors, embedding representation typically preserves the degree of network proximity [23][22], as various applications, such as node classification and link prediction, requires the learned representations to capture properties of the graph structure. Nevertheless, little attention has been paid to exploring *order embeddings* that preserve relative order among a given set of nodes in an ordering hierarchy. Such order embeddings are expected to encode rich information of the hierarchical structure, and hence can be used to predict missing dependency links.

Indicative representations of text and partial ordering relations amongst entities, such as hierarchical structure, can capture semantic and structural information embedded among them. Entities with positive or negative ordering relations tend to be semantically related to each other, and thus are comparable. However, not all pairs of entities have an ordering relationship. Specifically, semantically unrelated pairs of entities are not comparable and thus should not have any ordering relations between them. For example, a course "machine learning" depends on several prerequisite courses, including "data structure", "theory of computation", and "artificial intelligence" as they are background to the machine learning course, as indicated in many computer science curricula. Moreover, intuitively, "probabilistic models" is semantically closer to "fundamentals of probability" than it is to "geographical information systems spatial databases". Thus, to establish the relatedness between these courses, we may explore the semantic proximity of title and textual content of these courses. Indeed, most of today's representation learning techniques aim to learn entity representations that are **semantics-preserving**, i.e., semantically similar entities are mapped into a nearby area in the semantic embedding space. In this paper, we argue that it is also essential for representation of entities to be **order-preserving**, i.e., antisymmetric relations between two entities are captured in the embedding space. There are relatively very little research on order-preserving embedding. One such order-preserving embedding technique, developed by Vilnis

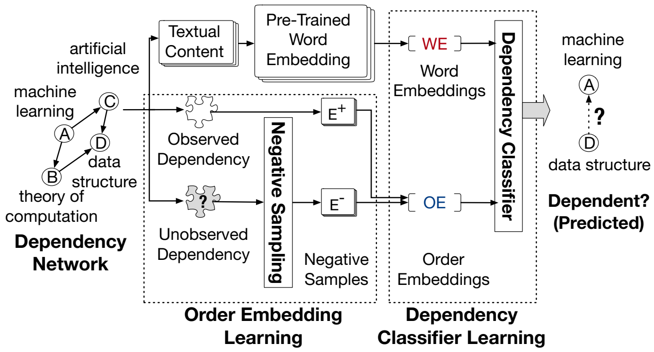

Figure 1: Overview of Proposed Framework.

and McCallum, assigns each entity a location in the embedding space such that the relative positions of entities to the origin of the representation space (i.e., zero coordinates) determine the partial ordering among them [26].

*** wlee: In addition to motivating the technical aspect of embedding techniques, I think it's important to motivate the application aspect, i.e., the needs and application of dependency relationsh.

In this paper, we explore a compositional approach to incorporate both semantics- and order-preserving embedding to predict dependency relations of entity pairs. Past research on order embeddings [24] [2] [26] have shown to be effective for word hypernym classification and textual entailment tasks where only entities themselves and their relations are represented. The textual and content features of these entities are largely neglected. To the best of our knowledge, no prior research has attempted to combine both semantic- and order-preserving embeddings for prediction of dependency relations.

Objectives. This research focuses on two objectives: (1) designing a new framework to accurately recover missing dependency relations, which is essential in automatic completion of knowledge bases; (2) applying the proposed framework on real-world datasets, such as course dependency hierarchy, job hierarchy, and paper citation hierarchy, using the automatically learned *order embeddings* to gain insights on hierarchical ordering. Both objectives are to be achieved in solving the following fundamental research problem:

Problem: (Dependency Relation Prediction). *Given a set of observed entity pairs with dependency relations, determine whether an unseen entity pair have a positive dependency relation or not.*

Overview of Our Proposed Approach. We address the dependency relation prediction problem as a classification task and propose a two-step framework as shown in Figure 1. The Step 1 of the framework learns the order embedding for entity pairs. Here, the order embedding is a low-dimensional vector that can be used to differentiate ordering relations among entities according to their positions in embedding space. Specifically, we learn the *ordering hierarchy* (consists of order embeddings) for the entire set of entities. However, not every entity pair is comparable. To determine ordering relations for *comparable* entities, the Step 2 of the framework combines order embedding and word embedding of entities. Examples of the latter include GloVe [21] and Word2Vec [13]. Next, the framework adopts a compositional approach to combine features from word embeddings and order embeddings for a pair of entities, to train the dependency relation classifiers, which allows us to leverage on both textual content and order embedding information.

*** The paragraph above is a bit confusing to me! Do you mean to learn order embedding as a vector associated with an edge and also learning embeddings for nodes? Or you are actual;ly learning node embeddings only but those nodes should preserve ordering relation? Could you clarify it.

There are two major research challenges in the dependency relation prediction problem and the proposed framework. Firstly, ordering hierarchies are often incomplete as not all ordering relations can be observed in many real applications. Under the open world assumption [14], these non-existing relations can be either negative or unknown [15][27][4]. With only partial ordering relations observed and no knowledge of negative relations, the dependency relation prediction is therefore a *one-class problem*. To determine whether a dependency relation between two entities holds, we thus need good negative sampling strategies to collect diverse and highly likely negative relations between entities. Hence, the first research challenge lies in selecting a subset of "*good*" negative samples from a massive set of unobserved relation candidates

The second research challenge is to demonstrate the advantages of fusing both order-preserving and semantics-preserving representations of entities for dependency relation prediction. Our research seeks to look into a detailed evaluation of our proposed compositional approach in comparison with two baseline approaches: (i) One focuses on learning semantic features from textual content, and uses the semantic features to determine dependency relations; (ii) Another approach is to use order embedding as features to determine dependency relations. To the best of our knowledge, such a study has not been conducted before despite its importance to knowledge base completion.

Contributions. The contributions of our work are summarized as follows:

- With vector order embedding model [24] used for learning ordering hierarchy among entities, we propose several negative sampling strategies based on graph centralities to effectively address the one-class problem, by using a training set of positive dependency relations and sufficient negative samples for order embedding learning.
- We propose the dependency relation classification framework that fuses both semantic and order embeddings of entities.
- We conduct extensive experiments to demonstrate the effectiveness of proposed framework using both synthetic ordering hierarchies and several real-world datasets.

2 RELATED WORK

Relational Learning. The goal of relational machine learning is to infer relationships between objects and answer queries [15][27][4]. Given a database of partially annotated relations, relational machine learning aims to learn rules or models to answer questions on unseen relations. This task arises in many settings such as biological pathways, analysis of question answering. A variety of techniques from Statistical relational models (SRL) community are proposed to tackle such tasks with partially annotated relations. SRL can be fundamentally divided into two categories: latent feature models and observable models. Markov Logic Nets (MRF) learns a set of

rules and weights to infer all unobserved relations jointly by maximizing probability. Embedding approaches, e.g., RESCAL, learn embeddings to predict relations. For example, the neural embedding models predict scores based on subjects, objects, and predicate embeddings[6][3][17][11][16]. Graph mining approaches on the other hand explore graph features, common ancestor or random walks, to predict relations. Ultimately, Google knowledge vault projects, automatic knowledge graph construction from the Web, fusion latent and observable models to improve the modeling power.
Prerequisite Relation Learning. Recent work has studied prerequisite relations learning among of educational concept map [12][10][19][20][5][1][9]. Prerequisite chains play an important role in curriculum planning and reading list generation [8][7]. Liu et al. focus on learning prerequisite relation for relation prediction among a set of concepts in university courses [12], while Liang et al. focus on recovery of prerequisite relations [10]. Liu et al. address CGL (concept graph learning) for two-level prerequisite learning, course level and concept level. Given observed course-level prerequisite rations, CGL.Class and CGL.Rank explicitly learn full prerequisite relations between concepts from course pair relations. Pan et al. [19] propose to automatically identify all course concepts from online MOOCs video clips [20]. Chen et al. propose a structural EM to learn an optimal Bayesian Network structure, representing the dependency relations among skills, which best explains the distribution of student performance data [5]. However, the scalability and efficiency of the learning mechanism remain unclear as the data consists of only a handful of students, examine items and skill variables. Liang et al. explore applicability of active learning to address the issue of limited training data for prerequisite classification on pairs of concepts [9]. They adopt pool-based active learning scenario to train a classifier with a substantially small dataset by incorporating four types of "valuable" unlabeled instances in particular prioritized way at each iteration. To the best of our knowledge, there is no existing work on prerequisite relation learning which tackles relation recovery via representation learning from partial orders of structural data.
Order Embedding Learning. Order embeddings are shown to be effective for word hypernym classification, image-caption ranking and textual entailment [24][26][2][25]. Vendrov et al. [24] propose to learn asymmetric relationships with deterministic vector order embeddings (VOE) of non-negative coordinates with partial order structure from incomplete data. Due to its limitation of expressiveness of deterministic vector order embeddings, recent work incorporate uncertainty in learning order representation to enrich expressivness and enable prediction with uncertainty, such as probabilistic extensions of order embeddings [26][2] and box lattice representation of order embeddings [25]. Athiwaratkun et al. introduce density order embedding (DOE) to model hierarchical data via encapsulation of probability densities [2]. In particular, they propose a new loss function, graph-based negative sample selections, and a penalty relaxation to induce soft partial orders.

3 ORDER EMBEDDING LEARNING AND NEGATIVE SAMPLING STRATEGIES

3.1 Vector-Based Order Embedding

There are a few order embedding methods proposed in the literature [2, 24, 26]. They all seek to assign a geometric representation to each entity such that the partially ordered entities are mapped into a latent space where certain geometric relationship between them are preserved. In this paper, we follow [24] to formalize the notion of *partially ordered entities* as follows.

Definition 3.1. (Partially Ordered entities) A partially ordered set of entities V has a binary order relation $\preceq$ such that for $v_i, v_j, v_k \in V$, the following properties hold: (1) $v_i \preceq v_i$ (reflexivity), (2) if $v_i \preceq v_j$ and $v_i \neq v_j$, then $v_j \npreceq v_i$ (antisymmetry), and (3) if $v_i \preceq v_j$ and $v_j \preceq v_k$, then $v_i \preceq v_k$ (transitivity).

Motivated by Vendrov and others [24], we want to learn a mapping f from V into a **Vector-based Order Embedding Space** Y where each entity is represented as a geometric point and the ordering relation between two entities is represented as a geometric relationship between the points representing the two entities. For entities with unknown order relations, we can then use their point representations in the embedding space Y to infer their order relations.

A crucial property of the mapping function f, preserving the order relations in the vector-based order embedding space, is defined as follows:

Definition 3.2. (Order preserving) $f : (V, \preceq) \rightarrow (Y, \preceq)$ is order-preserving if $\forall v_i, v_j \in V$, $v_i \preceq v_j$ if and only if $f(v_i) \preceq f(v_j)$.

Objective Function. To learn a good order-preserving mapping f, Vendrov et al. define the geometric relation between two entities v_i and v_j in the embedding space based on the conjunction of total order on each dimension d of embedding space Y. That is, $f(v_i) \preceq f(v_j)$ if and only if $v_{i,d} > v_{j,d}, \forall 1 \leq d \leq N_Y$ where N_Y is the dimension size of Y.

The above hard constraints can be implemented using a loss function that penalizes order violations in $Y \in \mathbb{R}^{N_Y}$ for the given set of ordered pairs $v_i \preceq v_j$:

$$d(v_i, v_j) = ||\max(0, f(v_j) - f(v_i))||^2 \quad (1)$$

where $d(v_i, v_j) = 0$ if $f(v_i) \preceq f(v_j)$ according to the conjunction of total orders; and $d(v_i, v_j) > 0$ if there is an order violation.

Given a set of positively ordered relations E^+ and a set of negatively ordered relations E^-, the objective function to learn an order-embedding mapping f is defined as a max-margin loss that encourages positively ordered relations to have zero penalty, and negatively ordered relations to have penalty greater than a margin:

$$O = \sum_{(v_i, v_j) \in E^+} d(f(v_i), f(v_j)) + \sum_{(v'_i, v'_j) \in E^-} \max\{0, \alpha - d(f(v'_i), f(v'_j))\} \quad (2)$$

3.2 Negative Sampling Strategies

Problem Analysis. Let $E^+ = \{(v_i, v_j)\}$ be the set of observed positive dependency relations, and E^- be the set of negative dependency relations to be determined. We seek to derive $P_{(v_i,v_j)}$, the likelihood of $(v_i, v_j) \notin E^+$ being a negative dependency relation. If $P_{(v_i,v_j)} = 1$ or 0, it suggests $v_i \preceq v_j$ or $v_i \npreceq v_j$ with full certainty. Hence, in negative sampling, two questions should be answered: (1) *what is the size of negative samples* $|E^-|$? and (2) *what is the sampling distribution* $P_{(v_i,v_j)}$?

For the first question, there are essentially the *full approach* and *subsampled approach* [28]. The full approach determines E^- to be the set of all unobserved relations, i.e., $E^- = \{(v_i, v_j) | (v_i, v_j) \notin E^+\}$. The full approach can be computationally expensive especially when the observed dependency relations are extremely sparse, i.e., the size of E^- can be exponentially greater than the size of E^+[28]. Moreover, it may misjudge some unobserved positive relations as negative. The subsampled approach, on the other hand, aims to select a reasonable subset of the entire unobserved relations as negative samples. The subsampled approach is thus a necessary approximation of the full approach. The size of subsampled negative samples is advised to be a constant multiple of the positive samples, i.e., $|E^-| = O(|E^+|)$ [28].

Simple Sampling Strategies (S1, S2): For the second question, one can assume a uniform distribution for $P_{(v_i, v_j)}$ for $v_i \preceq v_j$ not observed in E^+. In other words, for each positive dependency relation $(v_i, v_j) \in E^+$, we generate c negative pairs of either $(v_i, v_k) \notin E^+$ or $(v_k, v_j) \notin E^+$ where v_k's are randomly selected from V. This is also referred to **Simple Strategy 1 (S1)**. Another simple negative sampling strategy is to reverse each positive dependency relation (v_i, v_j) and use the reverse relation as negative sample. We call this the **Simple Strategy 2 (S2)**. Moreover, we explore three aspects of graph structure to derive negative samples: (1) *local neighborhood*, (2) *global neighborhood*, and (3) *descendant structure*. Table 1 gives an overview of the proposed sampling strategies.

Local Neighborhood-based Sampling Strategies (L1-L6): The key assumption behind local neighborhood-based negative sampling is that entities with large out-degree should be ones that many other entities may depend on (or precede many other entities), while entities with large in-degree should be ones that depend on many other entities (or be preceded by many other entities). Hence, the likelihood of an unobserved relation (v'_i, v'_j) being negative can be estimated by the out-degree and in-degree of v'_i and v'_j, respectively. We therefore propose L1-L6 strategies as summarized in Table 1. Given a positive dependency relation (v_i, v_j), the strategies select negative samples as follows:

- L1 gives v'_i with high in-degree (e.g., a senior-level job or course) a high probability for $(v_{i'}, v_j) \in (V \times V) - E^+$ to be used as a negative sample.
- L2 gives v'_j with high out-degree (e.g., an entry-level job, or an advanced course) a high probability for $(v_i, v_{j'}) \in (V \times V) - E^+$ to be used as a negative sample.
- L3 considers $(v_{i'}, v_{j'})$ a high probable negative sample if $v_{i'}$ has high in-degree and $v_{j'}$ has high out-degree.
- L4-L6 follow the same intuition of L1-L3 respectively except that high relative out- and in-degrees are used instead.

In the negative sampling process, we follow the *sampling with replacement* scheme, where the selected negative order relations are not removed from the population $V \times V - E^+$. In other words, each relation candidate is independently drawn from the full set of unobserved negative relations according to the sampling distributions controlled by one of L1-L6 until the expected sample size $|E^-_{tr}| = c \times |E^+_{tr}|$ (where c is a constant) is reached.

Global Neighborhood Sampling Strategies (G1-G6): Instead of deriving the probability of sampling alternatives of v_i and v_j to form negative samples using their local neighborhood properties (i.e., in-degree and out-degree), the global neighborhood sampling

Table 1: Specification of the Subsampled Variants.

Strategy	Distribution	Population
S1	uniform	$(v_i, v_j) \in E^-$
S2	NA	$(v_j, v_i) \in E^+$
L1	$P_{(v_i,v_j)} \propto indg(v_i)$	
L2	$P_{(v_i,v_j)} \propto outdg(v_j)$	
L3	$P_{(v_i,v_j)} \propto indg(v_i) + outdg(v_j)$	$(v_i, v_j) \in E^-$
L4	$P_{(v_i,v_j)} \propto \frac{indg(v_i)}{1+outdg(v_i)}$	
L5	$P_{(v_i,v_j)} \propto \frac{outdg(v_j)}{1+indg(v_j)}$	
L6	$P_{(v_i,v_j)} \propto \frac{indg(v_i)}{1+outdg(v_i)} + \frac{outdg(v_j)}{1+indg(v_j)}$	
G1	$P_{(v_i,v_j)} \propto ancs(v_i)$	
G2	$P_{(v_i,v_j)} \propto ancs(v_j)$	
G3	$P_{(v_i,v_j)} \propto ancs(v_i) + desc(v_j)$	$(v_i, v_j) \in E^-$
G4	$P_{(v_i,v_j)} \propto \frac{ancs(v_i)}{1+desc(v)}$	
G5	$P_{(v_i,v_j)} \propto \frac{desc(v_j)}{1+ancs(v_j)}$	
G6	$P_{(v_i,v_j)} \propto \frac{ancs(v_i)}{1+desc(v)} + \frac{desc(v_j)}{1+ancs(v_j)}$	
D0	uniform	$(v_i, v_j) \in E^- \wedge$
D1-D6	same as L1-L6 except different sample population	$v_i \in T(w) \wedge$
DG1-DG6	same as G1-G6 except different sample population	$v_j \in T(w) - T(v_i)$

considers global neighborhood properties (i.e., ancestors and descendants). We define the descendants (or ancestors) to be entities that can be reached transitively by following the $\preceq$ relation (or inverse of $\preceq$ relation). These strategies assume that: (i) a node $v_{i'}$ with many ancestors suggests that $v_{i'}$ is unlikely to be ordered before others. Hence, $(v_{i'}, v_j)$ is likely to be a negative order relation; (ii) a node $v_{j'}$ with many descendants suggests that it is unlikely to be ordered after others. Hence, $(v_i, v_{j'})$ is likely to be a negative order relation. Based on the above assumptions, we derive the sample strategies G1-G6 in Table 1.

Descendants-Constrained Sampling Strategy (D0-DG6): Other than measuring the likelihood for a relation to occur, another idea is to explore the relevance between a pair of entities in the sampling process. Athiwaratkun et al. introduced a structural constraint on relations by their relative positions in a network [2]. Specifically, for a node w with at least two descendants, they proposed to randomly select $v_i \in T(w)$, where $T(w)$ denotes all descendants of w, and then randomly select $v_j \in T(w) - T(v_i)$. (v_i, v_j) is taken as a negative sample if $(v_i, v_j) \notin E^+$. In this manner, w is intuitively the common ancestor of v_i and v_j subject to v_j is not a descendant of v_i. In essence, this structural constraint suggests a *relevant* but *not close* relations between v_i and v_j. We refer to such structural constraint as *descendants constraint*. We incorporate such descendants constraint into previously defined sampling strategies. Each strategy subject to descendants constraint thus has a limited sample population. Table 1 depicts the specification of D0-D6. D0 is the original negative selection strategy proposed in [2]. Note that the set of unobserved relations subject to descendants constraint is a subset of the full unobserved relations. To sample negative relations by any of descendants-constrained strategy, we go through every node w with at least two descendants. For each node w, we sample one unobserved relation (v_i, v_j) according to $P_{(v_i, v_j)}$ among all possible unobserved relations (v_i, v_j) that follow the descendants constraint. We repeatedly scan through every node w until the expected sample size $|E^-_{tr}|$ is reached. Likewise, the sampling strategies combined with the descendants constraint and global neighborhood (DG1-DG6) are summarized in Table 1.

Table 2: Data Statistics.

Synthetic Dataset				Real-World Dataset			
Type	$\|V\|$	$\|E\|$	Density	**Type**	$\|V\|$	$\|E\|$	Density
Sparse	20	49-50	∼0.13	**Course**	654	1,675	$\sim 3.92\times10^{-3}$
Moderate	20	83-99	∼0.26	**OrgJob**	3,297	755,493	$\sim 6.95\times10^{-2}$
Dense	20	129-134	∼0.34	**Citation**	5,859	535,608	$\sim 1.56\times10^{-2}$

Ensemble Strategies. Each proposed sampling strategy has its merits and deficiencies. For instance, local and global neighborhood-based strategies can provide a huge quantity of negative samples. Descendants-constrained strategies can provide negative samples of high relevance. We therefore propose to create an ensemble of diverse sampling strategies which may provide a more reliable solution because the blended collection of negative samples from multiple strategies could enrich the sample diversity and quantity. For the ensemble approach, we highlight the top performing strategy in each category. We explore uniform aggregation of them to generate the ensemble scheme $A = \cup_{i=1}^{k} E^-_{s_i}$, where $E^-_{s_i}$ is the set of negative samples by strategy s_i.

4 EXPERIMENTS ON SYNTHETIC DATASETS

In this section, we conduct two experiments on synthetic graphs in order to achieve the following goals: (1) to understand how negative sampling strategies impact the quality of order embedding, and (2) to explore the usefulness of order embeddings for the entity ranking task and dependency relation prediction task.

4.1 Ground-Truth Dependency Generation

Consider a synthetic graph G=(V,E) of graph size $|V|$. We first generate the ordering attributes in a $|D|$-dimensional ordering space for each entity $v \in V$ to entail the order relations. Then, we generate the semantic attributes in another $|D|$-dimensional semantic space for each entity $v \in V$ to entail the semantic relations. Finally, we generate the dependency relations as ground-truth by aligning both order and semantic relations entailed in their respective spaces.

Order Relation Generation. To generate a $|D|$-dimensional vector $v \in \mathbb{R}^{|D|}$ for each node $v \in V$, the value $v_{i,d}$ in each dimension $1 \le d \le |D|$ is generated by a uniform distribution between an arbitrary value range, say, [-2, 2]. According to the relative position to the origin in the $|D|$-dimensional space, we generate an order relation from entity v_i to entity v_j if the values of v_i in every dimension is smaller than that of v_j. Specifically, the adjacency matrix A can be constructed as follows:

$$A_{i,j} = \begin{cases} 1 & if\ v_{i,d} < v_{j,d}, \forall 1 \le d \le |D| \\ 0 & otherwise \end{cases} \tag{3}$$

where $E = \{(v_i, v_j)|A_{i,j} = 1\}$ is the set of observed order relations and $\bar{E} = \{(v_i, v_j)|A_{i,j} = 0\}$ is the full set of unobserved relations.

Semantics Relation Generation. We arbitrary select two separated areas in the $|D|$-dimensional semantic space, e.g., Box1: $x \in$[0,2], $y \in$[0,2], and Box2: $x \in$[30,32], $y \in$[30,32] in a 2-dimensional space. To generate a $|D|$-dimensional vector $v \in \mathbb{R}^{|D|}$ for each node $v \in V$, we first randomly determine which box v belongs to, then the value $v_{i,d}$ in each dimension $1 \le d \le |D|$ is generated by a uniform distribution within the value range of the specified box. We generate 30 synthetic graphs with three density levels (i.e., sparse, moderate, and dense) as summarized in Table 2.

Train/Test Graph Generation. Given a graph G, we randomly determine a held-out dataset consisting of 20% relations for evaluation. The 20% held-out relations form a testing graph G_{ts} and the rest 80% relations form a training graph G_{tr}.

4.2 Entity Ranking

We evaluate the quality of order embedding by the *entity ranking* task. Given a training graph G_{tr}=(V,E) with observed relations E and unobserved relations $\bar{E}$, we aim to derive a ranked list of entities V according to their order embeddings learned from G_{tr}.

Order Embedding Learning. Given a sample size n, we collect a balanced training set for order embedding learning which consists of n observed dependency relations as positive training set (denoted as E^+_{tr}) and n unobserved dependency relations as negative training set (denoted as E^-_{tr}). We follow the sampling with replacement scheme to collect positive and negative dependency relations. E^+_{tr} is determined by randomly drawn n relations from E in G_{tr}, whereas E^-_{tr} is drawn according to a specified negative sampling strategy from $\bar{E}$ in G_{tr}. For instance, we may obtain E^-_{tr} using S2 strategy by collecting each reversed observed relation in E from $\bar{E}$, i.e., $E^-_{tr} = \{(v_j, v_i)|(v_i, v_j) \in E^+_{tr}\}$. Finally, We learn order embeddings of V from E^+_{tr} and E^-_{tr} given embedding space dimensionality $|D|$. The total number of learnable parameters is O = ($|V| \times |D|$).

Ranking Quality Measurement. We compare relative positions of entities V in the $|D|$-dimensional original space and the $|D|$-dimensional embedding space. To formally measure the ordering quality, we compute *Kendall rank correlation coefficient* $\tau_d \in [-1, 1]$ along dimension d=[1,2]. Given the order relations R^d_{GT} as the ground-truth ranking along dimension d and the learned order embeddings R_{OE}, we compute τ coefficient with adjustment for ties as follows. $\tau_d(R^d_{GT}, R_{OE}) = \frac{n_c - n_d}{\sqrt{n_c+n_d+n_0}\sqrt{n_c+n_d+n_1}}$, where n_c is the number of concordant pairs, n_d is the number of discordant pairs, n_0 is the number of ties only in R^d_{GT}, and n_1 the number of ties only in R_{OE}. Finally, we take the average of rank correlation in both dimensions $\mu(\tau) = \frac{1}{|D|}\sum_{d=1}^{|D|} \tau_d$ as the overall ranking quality.

Results. Table 3 gives a full comparison of order embeddings learned with various negative sampling strategies in terms of average rank correlations against the ground-truth ordering. Each point refers to the average rank correlation of one sampling strategy across ten graphs of the same density level, using 5-fold cross validations, i.e., average of 50 experiments. We compare the rank correlations of the compared order embeddings using graphs of three density levels $\in \{sparse, moderate, dense\}$.For ease of comparison, we discretize the average rank correlation $\mu(\tau)$ into four levels: (i) tier 1 ($0.75 \le \mu(\tau)$), (ii) tier 2 ($0.5 \le \mu(\tau) < 0.75$), (iii) tier 3 ($0.25 \le \mu(\tau) < 0.5$), and lastly (iv) tier 4 ($\mu(\tau) < 0.25$).

We conclude the experimental results with three key observations. First, order embeddings perform better when the graphs are denser. For instance, when n=400 the performance (in terms of $\mu(\tau)$) of most strategies achieve tier 1 quality, compared with the

Table 3: Tier-based Quality of Order Embeddings (n=400).

	Sparse	Moderate	Dense
Tier 1	None	None	S2,L1,L4,G2,G4-G6,D0-D3,D5-D6,DG2,DG4-DG6
Tier 2	S1,G3,G6	27 strategies	S1,L2,L3,L5,L6,G1,G3,D4,DG1,DG3

Table 4: Sample Statistics and Performance Comparison (n=800, 10 Dense Graphs, 5-fold).

Strategy (OE)	OE(d=2) Train (OE) $\|\{E_{tr}^+\}\|$	$\|\{E_{tr}^-\}\|$	Train/Test(LR) $\|\{E_{tr}\}\|$	$\|\{E_{ts}\}\|$	$\mu(\tau)$	WE	OE	OE⊕WE	OE(d=1) Train (OE) $\|\{E_{tr}^+\}\|$	$\|\{E_{tr}^-\}\|$	Train/Test(LR) $\|\{E_{tr}\}\|$	$\|\{E_{ts}\}\|$	$\mu(\tau)$	WE	OE	OE⊕WE
S1	63	291.5	140.6	33	.325	.56	.713	**.719**	76.9	281.4	178	43.9	.473	.54	.677	**.681**
S2	63	63	140.6	33	.712	.56	.715	**.72**	76.9	76.9	178	43.9	.573	.54	.697	**.716**
L1	63	222.2	140.6	33	.401	.56	.709	**.718**	76.9	232.7	178	43.9	.426	.54	.659	**.661**
L2	63	218.4	140.6	33	.412	.56	.689	**.692**	76.9	230.4	178	43.9	.423	.54	.652	**.66**
L3	63	255.3	140.6	33	.362	.56	.7	**.704**	76.9	257.9	178	43.9	.465	.54	.662	**.678**
L4	63	178.1	140.6	33	.378	.56	.697	**.703**	76.9	186.3	178	43.9	.414	.54	.627	**.653**
L5	63	179.7	140.6	33	.412	.56	.691	**.694**	76.9	185.2	178	43.9	.433	.54	.634	**.661**
L6	63	219.3	140.6	33	.399	.56	.701	**.705**	76.9	218.9	178	43.9	.408	.54	.652	**.671**
G1	63	219.9	140.6	33	.383	.56	.698	**.709**	76.9	231.5	178	43.9	.444	.54	.644	**.659**
G2	63	217.7	140.6	33	.376	.56	.693	**.706**	76.9	229.1	178	43.9	.429	.54	.647	**.653**
G3	63	255.4	140.6	33	.373	.56	.702	**.706**	76.9	257.4	178	43.9	.435	.54	.645	**.668**
G4	63	173.8	140.6	33	.391	.56	.705	**.711**	76.9	178.1	178	43.9	.426	.54	.637	**.654**
G5	63	175.4	140.6	33	.383	.56	.691	**.709**	76.9	178.5	178	43.9	.415	.54	.644	**.661**
G6	63	216.1	140.6	33	.411	.56	.702	**.713**	76.9	215.5	178	43.9	.413	.54	.627	**.657**
D0	63	16.1	140.6	33	.568	.56	.5	**.611**	76.9	20	178	43.9	.465	.54	.497	**.578**
D1	63	14	140.6	33	.575	.56	.505	**.61**	76.9	18.9	178	43.9	.451	.54	.485	**.561**
D2	63	13.3	140.6	33	.559	.56	.507	**.614**	76.9	18	178	43.9	.46	.54	.493	**.582**
D3	63	15.5	140.6	33	.549	.56	.491	**.596**	76.9	19.9	178	43.9	.462	.54	.495	**.571**
D4	63	13.6	140.6	33	.566	.56	.5	**.611**	76.9	18.6	178	43.9	.465	.54	.488	**.566**
D5	63	13.2	140.6	33	.569	.56	.515	**.617**	76.9	17.6	178	43.9	.455	.54	.487	**.581**
D6	63	15.3	140.6	33	.551	.56	.505	**.605**	76.9	19.7	178	43.9	.459	.54	.487	**.574**
DG1	63	14	140.6	33	.554	.56	.503	**.61**	76.9	18.9	178	43.9	.466	.54	.5	**.567**
DG2	63	13.3	140.6	33	.569	.56	.507	**.614**	76.9	18.1	178	43.9	.463	.54	.494	**.579**
DG3	63	15.6	140.6	33	.575	.56	.501	**.602**	76.9	19.9	178	43.9	.461	.54	.497	**.571**
DG4	63	13.7	140.6	33	.567	.56	.499	**.609**	76.9	18.4	178	43.9	.458	.54	.489	**.565**
DG5	63	13.1	140.6	33	.576	.56	.507	**.614**	76.9	17.5	178	43.9	.453	.54	.488	**.584**
DG6	63	15	140.6	33	.548	.56	.501	**.613**	76.9	19.7	178	43.9	.456	.54	.498	**.579**
A:S1+S2+L1+G6	63	261	140.6	33	.35	.56	.705	**.716**	76.9	263.3	178	43.9	.44	.54	.654	**.672**
A:S1+S2+L1+D5	63	163.7	140.6	33	.52	.56	.698	**.716**	76.9	237.1	178	43.9	.47	.54	.685	**.68**
A:S1+S2+L1+DG2	63	236.3	140.6	33	.43	.56	.706	**.724**	76.9	237	178	43.9	.46	.54	.657	**.667**
A:S1+S2+G6+D5	63	233.9	140.6	33	.4	.56	.7	**.714**	76.9	238.2	178	43.9	.44	.54	.66	**.683**
A:S1+S2+G6+DG2	63	233.9	140.6	33	.42	.56	.705	**.718**	76.9	238.2	178	43.9	.47	.54	.66	**.67**
A:S1+S2+D5+DG2	63	181.1	140.6	33	.48	.56	.696	**.704**	76.9	182.6	178	43.9	.48	.54	.682	**.686**

moderate graphs where all strategies achieve tier 2 quality, and the sparse graphs where only three strategies achieve tier 2 quality. Second, the different sampling approaches showcase their merits and deficiencies through comparisons on graphs of varied densities. For sparse graphs when n=400, some global neighborhood-based sampling approaches achieve the tier 2 quality, whereas all strategies collectively achieve either tier 1 or tier 2 quality. We also obtain better embeddings quality when more samples are provided. We skip the details due to the space limit.

4.3 Dependency Relation Prediction

Given a training graph $G_{tr}=(V,E)$ with partial-observed relations and a testing graph G_{ts} with a set of held-out relations, we aim to learn a dependency classifier from G_{tr} to precisely predict dependency and non-dependency relations in G_{ts}.

Representation Fusion and Dependency Classifiers. We explore the expressive power of three types of representations: (i) semantic attributed representation (WE); (ii) order embedding (OE); and (iii) fused semantic order embeddings (OE⊕WE), where ⊕ is the concatenate operator. Semantic attributed representation is generated during the process of semantics relation generation, which is our simulation of the real-world word embeddings. Order embeddings are learned with respective negative sampling strategies. Given a representation of any type for each entity $v_i \in V$, we form the representation for each relation (v_i,v_j) by taking subtraction v_j from v_i.

Prediction Quality Measurement. To avoid training and testing sampling in favor of some particular representations, we collect the observed relations in G_{tr}, denoted as E_{tr}^+, and randomly select unobserved relations in G_{tr}, denoted as E_{tr}^-, where $|E_{tr}^+| = |E_{tr}^-|$. We conduct 5-fold cross validation to avoid biased data split for each training graph G_{tr} of the same density level. Given a set of relation representations of particular form $\{WE, OE, OE \oplus WE\}$ from E_{tr}^+ and E_{tr}^-, we utilize logistic regression (LR) to learn a binary classifier to differentiate positive and negative dependencies. Specifically, we learn the linear function $p(y|e = (v_i, v_j))$, where y=1 if v_i depends on v_j, and y=−1 otherwise. We measure the prediction quality using accuracy as follows. $\frac{|TP \cup TN|}{|E_{ts}^+ \cup E_{ts}^-|}$, where E_{ts}^+ (E_{ts}^-) are dependency (non-dependency) relations in testing graph G_{ts}, $\{e|e \in E_{ts}^+ \wedge p(y = 1|e) > p(y = -1|e)\}$ is the true positive set, and $\{e|e \in E_{ts}^- \wedge p(y = -1|e) > p(y = 1|e)\}$ is the true negative set.

Dimensionality Study (d). Table 4 shows the sample statistics and prediction accuracy across various negative sampling strategies. A key observation is that the prediction accuracy using OE with d=2 is consistently better than OE with $d = 1$ across multiple strategies. This may suggest that with the right setting of embedding dimensionality (d=2), OE can achieve the best prediction performance.

Representation Study. A key observation in Table 4 is that the prediction accuracy using blended representation OE⊕WE is consistently better than using either OE or WE alone. Notice that any testing relation (v_i,v_j) comes from one of four quadrants: (i) v_i and v_j are ordered and similar in semantics, (ii) ordered but dissimilar in semantics, (iii) non-ordered but similar in semantics, and (iv) non-ordered and dissimilar in semantics. The error cases of OE-driven classifier, which is trained without the semantic knowledge, lie under quadrant ii, whereas the error cases of WE-driven classifier, which is trained without the ordering knowledge, lie under quadrant iii. The OE⊕WE-driven classifier can better differentiate cases lies under quadrants ii and iii because it is trained with both ordering and semantic knowledge, and thus outperforms others.

Sample Ensemble Study. Another key observation in Table 4 is that sample ensembles can effectively lead to better performance. For instance, S2 alone, which takes the reverse of the observed relations as negative samples, is obviously a wining strategy amongst all. Nonetheless, the overall predictive power can be further optimized with the sample ensemble technique, providing .724 accuracy compared to S2 at .72 in Table 4. This suggests that the negative samples contributed by S1, S2, L1, and DG2, not only increase in quantity (236.3 unique negative samples on average) but also provide diverse and useful signals, resulting in accurate order embeddings.

5 EXPERIMENTS ON REAL DATASETS

We conduct experiments on real-world networks for two purposes: (1) to quantitatively and qualitatively study the quality of order embedding, and (2) to demonstrate the usefulness of order embeddings for entity ranking and dependency relation prediction tasks.

5.1 Datasets

We first describe thee real-world datasets for entity ranking task and dependency relation prediction task. The statistics of each network and their transitive closure is summarized in Table 2.

Course Dependency Hierarchy (Course). We use the course dependency dataset[1] from 11 cs-related universities in US. There are 654 unique courses, and 861 dependency relations among courses. For each course, we extract the course title (TT) and course description (DE) from course content. After removing stop words, we obtain a vocabulary of 3,987 words.

Singapore Organization Job Hierarchy (OrgJob). There are 3,297 standardized job titles and 26,550 immediate reporting lines between supervisor and subordinate job titles from Singapore organization job hierarchy. We develop a job title parser to extract job function from each job title [18].

ACM Citation Network (Citation). There are 1.1M articles with at least one citation in the latest ACM citation network [2]. We retrieve those articles in five different research areas: KDD, ICML, NIPS, WWW, and SIGIR, resulting in 5,859 papers and 60,759 citations. For each paper, we extract paper title (TT) as the textual content. After removing stop words, we obtain 6,548 vocabularies.

5.2 Experiment Setup

Representation Baselines. We include several representation baselines for comparison in our experiments.

- Pre-Trained Word Embedding (WE): Glove vectors [21] are global vectors trained for word representations. For fair comparison, we use the vectors of length 50 dimensions trained on 6 billion words from Wikipedia corpus. Another widely used word vectors are the Skip-Gram-based Word2Vec [13]. We have self-trained vectors of 44,449 words and of 50 dimensions on our job posts corpus[3] using the Skip-Gram model of Word2Vec. The job posts corpus is domain-focused and contains ~544,369 unique job posts.
- Vector Order Embedding (OE): We adopt vector order embedding (VOE) [24] to learn order embeddings for entities. The parameter settings of VOE includes learning rate λ=0.01, batch size = 256, and 80 epochs for all experiments.
- Fusion (OE⊕WE): We represent each entity using both word and order embeddings learned on each real-world dataset, based on various negative sampling strategies. The representations of entity $v_i \in V$ is represented as a concatenation of WE⊕OE. The resulting dimension of vector WE⊕OE consists of word embedding (50 by default) and order embeddings ($|D| \in [1, 3, 5]$).

5.3 Entity Ranking

Given a training graph G_{tr}=(V,E) with observed relations E and unobserved relations $\bar{E}$, we aim to derive a ranked list of entities V according to their order embeddings learned from G_{tr}.

Course Dependency Hierarchy. We evaluate the quality of order embedding by the entity ranking task. A valid dependency relation of a course pair (v_i,v_j) represents course v_i depends on course v_j, in which case v_i is the relatively more advanced class and v_j is the fundamental one. A course pair (v_i,v_j) is regarded as positive if the dependency is observed in G_{tr}; otherwise, the course pair is considered as a negative one. Given the sample size n and a sampling strategy, we sample n positive relations E^+_{tr} and n negative relations E^-_{tr} to learn order embeddings for each course. Table 5 shows the top-10 fundamental courses and bottom-10 advanced courses ranked by order embeddings using S2 strategy with the sample size n=5000. The top-10 fundamental courses tend to have higher in-degree and out-degree ratio (42.6 versus 0.1 on average), compared to the top-10 advanced courses (0.0 versus 7.0 on average). This suggests that top fundamental courses are truly prerequisite in the course dependency hierarchy and vice versa.

Organization Job Hierarchy. A valid dependency relation of a job pair (v_i,v_j) represents career progression from job v_i to v_j, in which case v_i is the relatively junior to v_j which is more senior. Given the sample size n and a sampling strategy, we sample n positive relations E^+_{tr} and n negative relations E^-_{tr} to learn order embeddings for each job title. Table 5 shows the top-10 senior jobs and bottom-10 entry-level jobs in Singapore organization job hierarchy by order embeddings using S2 strategy with sample size n=10K. The top-10 senior jobs tend to have higher in-degree and out-degree ratio (1,579.6 versus 225.5 on average), compared to the top-10 entry-level jobs (0.8 versus 161.2). This suggests that top senior jobs are truly high ranking positions in the organization job hierarchy and vice versa.

Citation Network. A valid dependency relation of a paper pair (v_i,v_j) represents paper v_i cites v_j, in which case v_i is the relatively earlier work and v_j is the recent one. Given the sample size n and a sampling strategy, we sample n positive relations E^+_{tr} and n negative relations E^-_{tr} to learn order embeddings for each paper. Table 5 shows the top-10 earliest work and bottom-10 latest work in ACM citation network by order embeddings using S2 strategy with sample size $n=175,000$. The top-10 earliest papers tend to

[1]https://github.com/harrylclc/concept-prerequisite-papers
[2]https://aminer.org/citation
[3]https://www.mycareersfuture.sg

Table 5: Top-10 and Bottom-10 Entities by Order Embeddings (d = 1, $S2$).

Top Fundamental Courses (n = 5K)	**Top Senior Jobs** (n = 100K)	**Top Earliest Papers** (n = 175K)	
Course Title	**Job Title**	**Paper Title**	**Year**
Programming Methodology	Prime Minister	MARS: A Retrieval Tool on the Basis of Morphological Analysis	1984
Intro to Computer Science.	Minister	A Decision Theory Approach to Optimal Automatic Indexing	1982
Introduction to Computer Programming	Trade and Industry Minister	Artificial Intelligence Implications for Information Retrieval	1983
Fundamentals of Programming and Computer Science	Home Affair Minister	A Common Architecture for Different Text Processing Techniques...	1986
Introduction to Programming Techniques	Infrastructure Minister	Evaluation of The 2-Poisson Model as a Basis for Using Term...	1983
Computer Science I: Fundamentals	Defence Minister	An Approach to Natural Language for Document Retrieval	1987
Introduction to EECS I	Permanent Secretary	An Evaluation of Term Dependence Models in Information Retrieval	1982
Programming Abstractions	Manpower Minister	IR, NLP, AI and UFOS: or IR-Relevance, Natural Language Problems...	1986
Object-Oriented Programming I	Finance Minister	The Maximum Entropy Principle in Information Retrieval	1986
Discrete Structures	Education Minister	The Automatic Indexing System AIR/PHYS	1988

Top Advanced Courses (n = 5K)	**Top Entry-Level Jobs** (n = 100K)	**Top Latest Papers** (n = 175K)	
Course Title	**Job Title**	**Paper Title**	**Year**
Collaborative Design (W)	North Mosque Cluster Manager	39 GLAD: Group Anomaly Detection in Social Media Analysis	2014
Advanced Compiler Construction	Deputy Decision Support Service Director	SMVC: Semi-Supervised Multi-View Clustering in Subspace...	2014
Projects in Database Systems	Asset Service & Finance Director	CatchSync : Catching Synchronized Behavior in Large Directed...	2014
Topics in Algorithmic Economics	Air Traffic Control Manager	Exploiting Geographic Dependencies for Real Estate. Appraisal: A...	2014
Projects in Networking	Assistant Policy Director	Gradient Boosted Feature Selection	2014
Advanced Computer Security	Senior Assistant Major Director	Network Mining and Analysis for Social Applications	2014
Advanced Networking	Principal Emergency Preparedness Specialist	Grouping Students in Educational Settings	2014
Advanced Distributed Systems	Agency Service Director	Optimal Recommendations under Attraction, Aversion, and Social...	2014
Advanced Computer Networks	Deputy Pupil Director	Learning Time-Series Shapelets	2014
Array Processing	Assistant Speciality and Service Director	Good-Enough Brain Model: Challenges, Algorithms, and...	2014

Table 6: Performance comparison on course dependency hierarchy (n=5K), where TT (DE) denotes course titles (descriptions).

Strategy	**Train (OE)**		**Train/Test(LR)**		**WE**		**OE(d=1)**			**OE(d=3)**			**OE(d=5)**		
(OE)	$\lvert\{E_{tr}^{+}\}\rvert$	$\lvert\{E_{tr}^{-}\}\rvert$	$\lvert E_{tr}\rvert$	$\lvert E_{ts}\rvert$	TT	DE	OE	OE⊕TT	OE⊕DE	OE	OE⊕TT	OE⊕DE	OE	OE⊕TT	OE⊕DE
S1	1675	5000.0	3726.4	847.2	.641	.601	**.834**	.778	.829	**.88**	.831	.886	**.863**	.742	.808
S2	1675	1675.0	3726.4	847.2	.641	.601	**.854**	.759	.805	**.852**	.828	.811	**.85**	.778	.845
L1	1675	4732.0	3726.4	847.2	.641	.601	**.851**	.776	.841	.879	.83	**.888**	**.87**	.795	.785
L2	1675	4951.4	3726.4	847.2	.641	.601	.772	.701	**.778**	**.829**	.768	.82	**.843**	.748	.751
L3	1675	4908.4	3726.4	847.2	.641	.601	.803	.739	**.811**	**.845**	.768	.774	**.825**	.734	.752
L4	1675	4609.6	3726.4	847.2	.641	.601	**.834**	.735	.791	**.886**	.837	.855	**.854**	.749	.785
L5	1675	4949.6	3726.4	847.2	.641	.601	**.819**	.741	.755	**.872**	.767	.817	**.886**	.775	.818
L6	1675	4895.4	3726.4	847.2	.641	.601	**.809**	.745	.794	**.86**	.805	.823	**.853**	.764	.786
G1	1675	4715.8	3726.4	847.2	.641	.601	**.82**	.771	.785	.826	.772	**.852**	**.861**	.77	.778
G2	1675	4950.0	3726.4	847.2	.641	.601	.795	.729	**.831**	.856	.805	**.876**	**.868**	.75	.751
G3	1675	4902.0	3726.4	847.2	.641	.601	.781	.709	**.812**	**.842**	.798	**.842**	**.831**	.748	.766
G4	1675	4580.4	3726.4	847.2	.641	.601	.738	.715	**.758**	**.839**	.746	.772	**.854**	.745	.787
G5	1675	4946.2	3726.4	847.2	.641	.601	.79	.757	**.797**	**.851**	.817	.803	**.853**	.772	.797
G6	1675	4883.2	3726.4	847.2	.641	.601	.789	.74	**.811**	**.821**	.762	.77	**.856**	.775	.773
D0	1675	11.4	3726.4	847.2	.641	.601	**.796**	.617	.606	**.838**	.73	.72	**.799**	.715	.699
D1	1675	11.4	3726.4	847.2	.641	.601	**.69**	.643	.63	**.832**	.731	.72	**.79**	.742	.686
D2	1675	8.2	3726.4	847.2	.641	.601	**.797**	.618	.602	**.827**	.721	.699	**.797**	.727	.718
D3	1675	11.4	3726.4	847.2	.641	.601	**.781**	.635	.637	**.832**	.751	.774	**.811**	.712	.716
D4	1675	11.4	3726.4	847.2	.641	.601	**.773**	.628	.615	**.815**	.7	.722	**.844**	.717	.7
D5	1675	8.2	3726.4	847.2	.641	.601	**.751**	.612	.61	**.837**	.719	.72	**.772**	.707	.71
D6	1675	11.4	3726.4	847.2	.641	.601	**.767**	.63	.635	**.843**	.698	.751	**.82**	.722	.73
DG1	1675	11.4	3726.4	847.2	.641	.601	**.78**	.622	.617	**.803**	.725	.736	**.812**	.734	.714
DG2	1675	8.2	3726.4	847.2	.641	.601	**.803**	.631	.587	**.807**	.714	.72	**.786**	.679	.72
DG3	1675	11.4	3726.4	847.2	.641	.601	**.825**	.634	.612	**.85**	.736	.784	**.858**	.73	.711
DG4	1675	11.4	3726.4	847.2	.641	.601	**.761**	.645	.6	**.849**	.761	.717	**.778**	.723	.713
DG5	1675	8.2	3726.4	847.2	.641	.601	**.781**	.63	.593	**.835**	.737	.747	**.847**	.682	.717
DG6	1675	11.4	3726.4	847.2	.641	.601	**.767**	.638	.609	**.845**	.735	.735	**.802**	.731	.707
A:S1+S2+L4+G5	1666.6	4834.8	3726.4	847.2	.641	.601	**.873**	.765	.806	**.878**	.816	.829	**.895**	.827	.863
A:S1+S2+L4+D6	1669.6	4685.2	3726.4	847.2	.641	.601	**.845**	.754	.788	**.886**	.788	.838	**.901**	.819	.785
A:S1+S2+L4+DG3	1666.8	3618.0	3726.4	847.2	.641	.601	**.803**	.743	.769	**.81**	.769	.803	**.834**	.772	.774
A:S1+S2+G5+D6	1668.0	3736.4	3726.4	847.2	.641	.601	**.883**	.798	.823	**.856**	.794	.823	**.893**	.818	.877
A:S1+S2+G5+DG3	1667.4	3736.8	3726.4	847.2	.641	.601	**.87**	.763	.835	**.884**	.817	.876	**.899**	.841	.857
A:S1+S2+D6+DG3	1667.0	2502.8	3726.4	847.2	.641	.601	**.827**	.759	.738	**.827**	.764	.792	.805	.786	**.814**

have more general topics and higher in-degree and out-degree ratio (471.9 versus 173.2 on average) in the ACM citation network, compared to the top-10 (0.8 versus 0.5).

5.4 Dependency Relation Prediction

We follow Section 4.3 to explore the expressive power of the following three types of representations: semantic attributed representation (WE), order embedding (OE), and fused semantic order

Table 7: Performance comparison, where FN denotes job functions and TT denotes paper titles.

	Organization Job Hierarchy (n=25K, d=1)							Citation Network (n=43.8K, d=1)						
Strategy	**Train (OE)**		**Train/Test(LR)**		**WE**	**OE**	**OE⊕WE**	**Train (OE)**		**Train/Test(LR)**		**WE**	**OE**	**OE⊕WE**
(OE)	$\lvert\{E^+_{tr}\}\rvert$	$\lvert\{E^-_{tr}\}\rvert$	$\lvert E_{tr}\rvert$	$\lvert E_{test}\rvert$	FN	OE	OE⊕FN	$\lvert\{E^+_{tr}\}\rvert$	$\lvert\{E^-_{tr}\}\rvert$	$\lvert E_{tr}\rvert$	$\lvert E_{ts}\rvert$	TT	OE	OE⊕TT
S1	25K	25K	~1.8M	~1.8M	.6	**.783**	.78	43.8K	43.8K	~1.3M	~805.8K	.664	.823	.825
S2	25K	25K	~1.8M	~1.8M	.6	**.951**	**.951**	43.8K	43.8K	~1.3M	~805.8K	.664	**.881**	.879
L1	25K	~23.8K	~1.8M	~1.8M	.6	**.928**	.924	43.8K	~42K	~1.3M	~805.8K	.664	**.858**	.85
L2	25K	~24.8K	~1.8M	~1.8M	.6	**.872**	.87	43.8K	~43K	~1.3M	~805.8K	.664	**.88**	.879
L3	25K	~24.6K	~1.8M	~1.8M	.6	**.867**	.865	43.8K	~43K	~1.3M	~805.8K	.664	**.891**	.886
L4	25K	~18.7K	~1.8M	~1.8M	.6	.795	**.801**	43.8K	~41K	~1.3M	~805.8K	.664	**.854**	.849
L5	25K	~24.8K	~1.8M	~1.8M	.6	**.869**	.867	43.8K	~43K	~1.3M	~805.8K	.664	**.88**	.88
L6	25K	~24.7K	~1.8M	~1.8M	.6	**.856**	.849	43.8K	~43K	~1.3M	~805.8K	.664	**.886**	.884
G1	25K	~23.8K	~1.8M	~1.8M	.6	**.932**	**.932**	43.8K	~42K	~1.3M	~805.8K	.664	**.857**	.85
G2	25K	~24.8K	~1.8M	~1.8M	.6	**.87**	.868	43.8K	~43K	~1.3M	~805.8K	.664	**.881**	.88
G3	25K	~24.6K	~1.8M	~1.8M	.6	**.867**	.866	43.8K	~43K	~1.3M	~805.8K	.664	**.886**	.885
G4	25K	~18.4K	~1.8M	~1.8M	.6	.793	**.801**	43.8K	~41K	~1.3M	~805.8K	.664	**.85**	.842
G5	25K	~24.8K	~1.8M	~1.8M	.6	**.873**	.87	43.8K	~43K	~1.3M	~805.8K	.664	**.879**	.879
G6	25K	~24.7K	~1.8M	~1.8M	.6	.85	**.863**	43.8K	~43K	~1.3M	~805.8K	.664	**.886**	.882
D0	25K	~1.1K	~1.8M	~1.8M	.6	.662	**.699**	43.8K	~23K	~1.3M	~805.8K	.664	.684	**.696**
D1	25K	629	~1.8M	~1.8M	.6	**.731**	.681	43.8K	~14K	~1.3M	~805.8K	.664	.664	**.69**
D2	25K	~1.1K	~1.8M	~1.8M	.6	.658	**.667**	43.8K	~18K	~1.3M	~805.8K	.664	.744	**.769**
D3	25K	~1.1K	~1.8M	~1.8M	.6	.661	**.72**	43.8K	~18K	~1.3M	~805.8K	.664	.666	**.698**
D4	25K	566.8	~1.8M	~1.8M	.6	**.74**	.693	43.8K	~12K	~1.3M	~805.8K	.664	.663	**.75**
D5	25K	760.6	~1.8M	~1.8M	.6	**.657**	.651	43.8K	~12K	~1.3M	~805.8K	.664	.781	**.784**
D6	25K	984.2	~1.8M	~1.8M	.6	**.643**	.611	43.8K	~15K	~1.3M	~805.8K	.664	.663	**.737**
DG1	25K	629	~1.8M	~1.8M	.6	**.696**	.683	43.8K	~14K	~1.3M	~805.8K	.664	.661	**.695**
DG2	25K	~1K	~1.8M	~1.8M	.6	.635	**.663**	43.8K	~18K	~1.3M	~805.8K	.664	.747	**.762**
DG3	25K	~1.2K	~1.8M	~1.8M	.6	.631	**.664**	43.8K	~18K	~1.3M	~805.8K	.664	.663	**.723**
DG4	25K	568.6	~1.8M	~1.8M	.6	.653	**.673**	43.8K	~11K	~1.3M	~805.8K	.664	.666	**.667**
DG5	25K	759	~1.8M	~1.8M	.6	**.621**	.577	43.8K	~12.1K	~1.3M	~805.8K	.664	.77	**.782**
DG6	25K	989.6	~1.8M	~1.8M	.6	.62	**.687**	43.8K	~15K	~1.3M	~805.8K	.664	.668	**.722**

embeddings (OE⊕WE). To predict whether an unseen antisymmetric relation (v_i,v_j) holds or not, we train a dependency classifier using logistic regression which takes representations of entity pairs and the label of entity pairs as input. We report the classification accuracy using 5-fold cross validation.

Representations Study. Table 6 compares the performance of predictions made by various representations. An observation made from Table 6 is that the prediction accuracy using OE representation is better than using either WE or OE⊕WE. This suggests that OE is effective and reliable, especially when the textual content between a relevant entity pairs is very different. For example, "machine learning" course transitively depends on "data structure", but nonetheless the dependencies do not necessarily reflect in the semantic similarity by course title (TT) or course description (DE). The expressive power of OE is also observed on OrgJob and Citation networks in Table 7, respectively.

Dimensionality Study (*d*). Another observation in Table 6 is that the common sense of "the higher dimensionality the greater performance" does not necessarily hold. OE (d = 3) does give a better performance than OE (d = 1), but OE (d = 3) is still superior to OE (d = 5) in most cases as reported in Table 6. The order embedding method seeks to map each entity in a given hierarchy into a low dimensional vector with a fixed dimension d. Unfortunately, we have no knowledge of the intrinsic dimensionality to describe a particular hierarchical structure. To unravel the true dimensionality, we therefore empirically extend our experiment to multiple configurations. Table 6 suggests that d = 3 should suffice to preserve course dependency hierarchy.

Subsampled and Ensemble Study. A key observation throughout the synthetic and real datasets is that S1 consistently outperforms other strategies despite their simplicity and low overheads as reported in Table 6 and 7. In particular, the overall predictive power can be further optimized with the sample ensemble technique, providing .901 (d=5) accuracy with 4.4% improvement compared to S2 at .863 (d=5) in Table 6. This suggests that the negative samples contributed by S1, S2, L4, and D6 provides diverse and useful signals (4,685.2 unique negative samples on average), resulting in accurately learning order embeddings.

6 CONCLUSION AND FUTURE WORK

In this paper, we integrate the notions of order embedding and semantic proximity to model dependency relations. In practice, negative dependency relations/samples often are missing from the dependency relation graphs.To address this *one-class problem*, we explore multiple negative sampling strategies based on graph-specific centralities to collect diverse and precise negative information. We learn order embeddings from positive and negative training dependency relations by minimizing the max-margin loss on order-violation penalties. We conduct extensive experiments, using both synthetic and several real datasets, to demonstrate the usefulness of order embeddings and to gain better understanding of order embedding through the tasks of entity ranking and dependency relation prediction. Our studies show that: (i) the proposed negative sampling strategies enable effective and efficient order embedding learning, (ii) ensembles of diverse negative samples lead to robust and mostly better performance, (iii) while order embedding can give strong features, blended feature representations can contribute to better prediction accuracy when dependent entities share similar semantic. As for the future work, one direction is to extend the work to other order embedding models. Another interesting direction is to learn order embeddings for words to directly infer dependency relations.

ACKNOWLEDGMENT

This research is supported by the National Research Foundation, Prime Minister’s Office, Singapore under its International Research Centres in Singapore Funding Initiative. This work is supported in part by the National Science Foundation under Grant No. IIS-1717084.

REFERENCES

[1] Fareedah ALSaad, Assma Boughoula, Chase Geigle, Hari Sundaram, and ChengXiang Zhai. 2018. Mining MOOC Lecture Transcripts to Construct Concept Dependency Graphs. In *EDM*.
[2] Ben Athiwaratkun and Andrew Gordon Wilson. 2018. Hierarchical Density Order Embeddings. In *ICLR*.
[3] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *NIPS*.
[4] Hongyun Cai, Vincent W Zheng, and Kevin Chang. 2018. A comprehensive survey of graph embedding: problems, techniques and applications. *TKDE* (2018).
[5] Yetian Chen, José P González-Brenes, and Jin Tian. 2016. Joint Discovery of Skill Prerequisite Graphs and Student Models.. In *EDM*.
[6] Xin Luna Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Kevin Murphy, Shaohua Sun, and Wei Zhang. 2014. From data fusion to knowledge fusion. *Proceedings of the VLDB Endowment* 7, 10 (2014), 881–892.
[7] Alexander R Fabbri, Irene Li, Prawat Trairatvorakul, Yijiao He, Wei Tai Ting, Robert Tung, Caitlin Westerfield, and Dragomir R Radev. 2018. TutorialBank: A Manually-Collected Corpus for Prerequisite Chains, Survey Extraction and Resource Recommendation. In *ACL*.
[8] Jonathan Gordon, Linhong Zhu, Aram Galstyan, Prem Natarajan, and Gully Burns. 2016. Modeling concept dependencies in a scientific corpus. In *ACL*.
[9] Chen Liang, Jianbo Ye, Shuting Wang, Bart Pursel, and C Lee Giles. 2018. Investigating active learning for concept prerequisite learning. *EAAI* (2018).
[10] Chen Liang, Jianbo Ye, Zhaohui Wu, Bart Pursel, and C Lee Giles. 2017. Recovering Concept Prerequisite Relations from University Course Dependencies.. In *AAAI*.
[11] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion.. In *AAAI*.
[12] Hanxiao Liu, Wanli Ma, Yiming Yang, and Jaime Carbonell. 2016. Learning concept graphs from online educational data. *JAIR* 55 (2016).
[13] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*.
[14] Jack Minker. 1982. On indefinite databases and the closed world assumption. In *International Conference on Automated Deduction*.
[15] Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. 2016. A review of relational machine learning for knowledge graphs. *Proc. IEEE* 104, 1 (2016), 11–33.
[16] Maximilian Nickel, Lorenzo Rosasco, Tomaso A Poggio, and others. 2016. Holographic Embeddings of Knowledge Graphs.. In *AAAI*.
[17] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A Three-Way Model for Collective Learning on Multi-Relational Data.. In *ICML*.
[18] Richard Jayadi Oentaryo, Ee-Peng Lim, Xavier Jayaraj Siddarth Ashok, Philips Kokoh Prasetyo, Koon Han Ong, and Zi Quan Lau. 2018. Talent Flow Analytics in Online Professional Network. *Data Science and Engineering* 3 (2018).
[19] Liangming Pan, Chengjiang Li, Juanzi Li, and Jie Tang. 2017. Prerequisite relation learning for concepts in moocs. In *ACL*.
[20] Liangming Pan, Xiaochen Wang, Chengjiang Li, Juanzi Li, and Jie Tang. 2017. Course Concept Extraction in MOOCs via Embedding-Based Graph Propagation. In *IJCNLP*.
[21] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *EMNLP*.
[22] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: Online Learning of Social Representations. In *SIGKDD*.
[23] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. LINE: Large-scale Information Network Embedding.. In *WWW*.
[24] Ivan Vendrov, Ryan Kiros, Sanja Fidler, and Raquel Urtasun. 2016. Order-embeddings of images and language. In *ICLR*.
[25] Luke Vilnis, Xiang Li, Shikhar Murty, and Andrew McCallum. 2018. Probabilistic Embedding of Knowledge Graphs with Box Lattice Measures. In *ACL*.
[26] Luke Vilnis and Andrew McCallum. 2015. Word representations via gaussian embedding. In *ICLR*.
[27] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge Graph Embedding: A Survey of Approaches and Applications. *TKDE* 29, 12 (2017), 2724–2743.
[28] Hsiang-Fu Yu, Mikhail Bilenko, and Chih-Jen Lin. 2017. Selection of negative samples for one-class matrix factorization. In *SDM*.