TUBE: Embedding Behavior Outcomes for Predicting Success

Daheng Wang[†], Tianwen Jiang[‡,†], Nitesh V. Chawla[†], Meng Jiang[†]

[†]Department of Computer Science and Engineering, University of Notre Dame, Notre Dame, IN 46556, USA
[‡]Research Center for Social Computing and Information Retrieval, Harbin Institute of Technology, China
{dwang8,tjiang2,nchawla,mjiang2}@nd.edu

ABSTRACT

Given a project plan and the goal, can we predict the plan's success rate? The key challenge is to learn the feature vectors of billions of the plan's components for effective prediction. However, existing methods did not model the behavior outcomes but component proximities. In this work, we define a measurement of behavior outcomes, which forms a test tube-shaped region to represent "success", in a vector space. We propose a novel representation learning method to learn the embeddings of behavior components (including contexts, plans, and goals) by preserving the behavior outcome information. Experiments on real datasets show that our proposed method significantly improves the performance of goal prediction as well as context recommendation over the state-of-the-art.

KEYWORDS

Behavior modeling, Representation learning, Effectiveness, Recommender systems

ACM Reference Format:
Daheng Wang, Tianwen Jiang, Nitesh V. Chawla, Meng Jiang. 2019. TUBE: Embedding Behavior Outcomes for Predicting Success. *In The 25th ACM SIGKDD Conference on Knowledge Discovery & Data Mining (KDD '19), August 4–8, 2019, Anchorage, AK, USA.* ACM, NY, NY, USA, 9 pages. https://doi.org/10.1145/3292500.3330867

1 INTRODUCTION

Every human being wants success. Every data scientist wants to publish in the KDD conference. This is one of the reasons that we make project plans, write codes, do experiments, and work days and nights. Once the goal is determined, it has been time to predict the success rate of the project plan: if the goal was less likely to be achieved, we would look for ways to adjust the plan towards an increase of the rate. But how do we predict the success?

Suppose you are a reviewer on a project plan. The goal is to write a paper as the project's outcome and publish it in KDD. Some information about the plan will be given to you as follows, and please assess the success rate (i.e., going up or down):

- This project's topic is "social media";

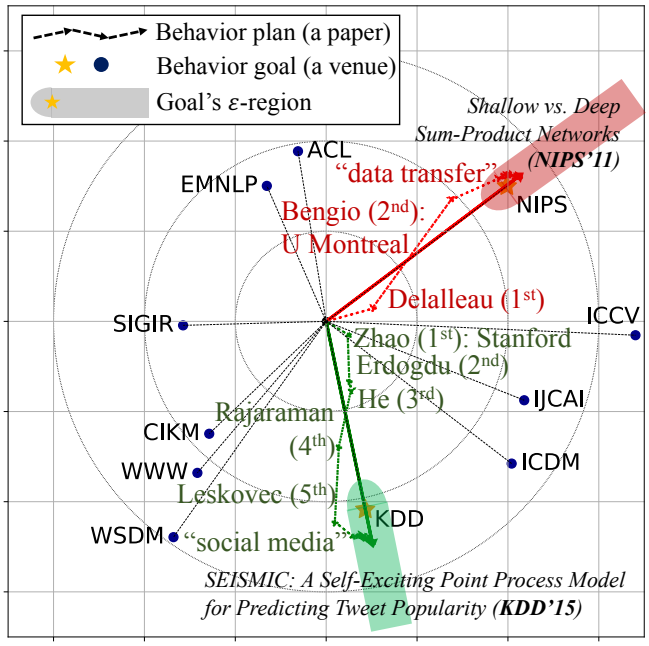


Figure 1: We propose to learn the representation of behavior contexts (e.g., authors, keywords) and goals (e.g., venues) by preserving the outcomes of behavior plan. A behavior plan consists of the contexts as its components, represented as a combination of the contexts in the vector space. If the plan was effective (i.e. accepted to the venue), it would fall in the ε-region of the goal (shaped like a test tube), where ε is the distance from the plan to the goal's direction; otherwise, it would be outside of the region. This is a 2-D visualization of our results, showing two successful behaviors (i.e., accepted papers) in the vector space though losing some dimensions.

- The leading author will be a Stanford student;
- The last author will be Prof. Jure Leskovec (Stanford);
- The research problem is to predict tweet popularity...

We have no idea about your assessment but here is ours. *First*, "social media" is absolutely a popular topic in KDD community but it does not mean it is a strong paper. *Second*, if the project is led by a Stanford's lab, it means something – the leading author is likely to be smart, creative, and hard-working, as being from the highly recognized and reputed institution, but it does not mean the paper will be accepted! When we come to the *third* point, the success rate significantly goes up: It is not only because Prof. Leskovec is of great renown in the KDD community and has published many highly-cited papers but also because he is an expert in "social media"

KDD '19, August 4–8, 2019, Anchorage, AK, USA

ACM ISBN 978-1-4503-6201-6/19/08…$15.00
https://doi.org/10.1145/3292500.3330867

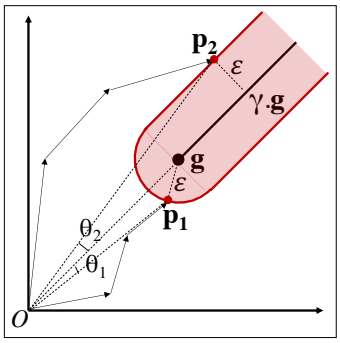


Figure 2: p_1 and p_2 are two behavior plans. g is a behavior goal. ε is the distance from a plan to the goal's direction (ray), forming a test tube-shaped region of effective plans. A smaller ε means higher plan effectiveness and better behavior outcomes.

and he belongs to the same institution with the leading author. The *fourth* point makes our prediction more positive. It turns out to be a paper called *SEISMIC* published in KDD'15.

Open academic datasets (such as AMiner [20] and Microsoft Academic Graph [18]) make hundreds of millions of successful examples (i.e., published papers) available, and the above clues we assess the success are lying beneath the huge data. In this work, we aim at proposing a data mining approach to *predict the success of a plan towards a goal*. This research problem is not limited to the paper-publishing behavior and can be found for many other behavior tasks, e.g., selecting instructional materials for effective K12 education, recruiting players and staff for sports teams, or finding resources and strategies for military purposes.

The key point is to turn billions of plan's components (called *context items* later, such as authors, keywords, and references in a paper) into numerical representations fed into predictive models. Existing approaches formulate the behavior data as an information network and the papers or components or venues as nodes. Then network embedding methods have been widely applied for learning the node's feature vectors by preserving the pair-wise node proximity, neighborhood, or global structure [4, 7, 8, 14, 19, 21]. Unfortunately, the outcomes of behavior, or say, the underlying patterns of the components forming an (un-)successful plan towards the goal, were not preserved in the embeddings, and thus not effective for predicting the success (as shown in the experiments).

To reach the aim, we have three challenges. *First*, the behavior's components, which have different roles (i.e., context items, behavior plans, behavior goals), need to be represented in a low-dimensional vector space. *Second*, the behavior plan's outcome measurement needs to be defined in the vector space. The network-based structural metrics would not longer be applicable. *Third*, failures (i.e., negative examples) may not be observable from the data in many behavior domains. For example, the public academic datasets have tons of accepted papers but no rejected ones.

To address the above challenges, we propose a novel approach, named TUBE, that Embeds Behavior oUTcomes for success prediction. Our idea is to use the vector space in a new way: As shown in Figure 1, TUBE navigates decision makers in the vector space to the point of success, starting from the origin, adding existing context items (*green or red arrows*, e.g., the first author, a term of target problem) into the route, looking for complementary context items (e.g., collaborators, important keywords, necessary references), and approaching the point of the behavior goal (*yellow stars*).

In TUBE, behavior plans (p_1 and p_2) as well as the behavior goal (g) are represented as points in the vector space (as shown in Figure 2). A plan is a combination of its context items. TUBE learns the feature vectors of the context items and goals in the behavior data. Next, we propose the measurement of behavior outcomes:

Definition 1.1 (Plan effectiveness). Given $\mathbf{p}, \mathbf{g} \in \mathbb{R}^d$ as the feature vectors of a plan p and a goal g in a d-dimensional vector space, the goal's *success* is defined as the ray that starts from the point $\mathbf{g}$ and goes off in a particular direction to infinity. The *plan's distance to success* $\varepsilon(p \mid g) \in [0, +\infty)$ is defined as the distance from $\mathbf{p}$ to the success (i.e., ray from $\mathbf{g}$). The *plan effectiveness* is a transformation of $\varepsilon(p \mid g)$, denoted by $e(p \mid g) \in [0, 1]$: when $\varepsilon(p \mid g) = 0$, $e(p \mid g) = 1$. The rigorous mathematical definitions will be given in Section 3.

Note that the *success* is not defined as a point but a ray in the vector space. It gives flexibility of defining a successful plan's *level of achievement* (or *plan achievement*). As shown in Figure 2, the *plan achievement* $\gamma(p \mid g)$ is defined as the norm of the plan point's projection on the ray over the norm of the goal point. This factor of behavior outcomes does exist in real life: we do have highly-cited papers, awarded papers, and ground-breaking papers.

It is worthwhile to highlight that these new definitions of behavior modeling form an ε-region of effective plans for each goal, in which any plan's *distance to success* is not bigger than ε. The ε-region looks like a test tube filled with "effective liquid."

Then, we propose a representation learning method that jointly learns the embeddings of behavior contexts and goals by preserving the behavior outcomes. We assume that the positive examples (i.e., paper records in the datasets) have an effectiveness of 1 (and thus a distance ε of 0). We adopt the negative sampling strategy, which has widely been applied to word embeddings [13], to generate negative examples and assume they have an effectiveness of a small value.

Figure 1 presents a 2-D visualization of ten behavior goals (i.e., conferences) and two successful behavior plans (a KDD'15 paper and a NIPS'11 paper) losing some dimensions in the embedding results. The venue points were located in fields without prior knowledge. The authors are the most important factors of the plans, compared to keywords and references. Senior researchers may obtain long vectors with a direction to their expertise. In the future, we will study more concrete contexts of the papers such as years, datasets, and paper styles (e.g., tables, figures, formulas).

The main contributions of this paper are summarized as follows.

- *A novel problem:* We propose to represent the behavior components (i.e., contexts, plans, goals) in a vector space and learn the embeddings for success prediction.
- *New measurement:* We propose a measurement of behavior outcomes as well as a new concept of (test tube-shaped) ε-region to represent "success" in the vector space.
- *A novel representation learning method:* TUBE preserves the behavior outcome information in the context and goal embeddings. Experimental resutls demonstrate the effectiveness and efficiency of the algorithm.

2 PROBLEM DEFINITION

In this section, we first introduce basic concepts used in this paper, and then formally define the research problem. Table 1 presents the symbols and their descriptions.

Table 1: Symbols used in this paper and their descriptions.

Symbol	Description
g; $\mathcal{G}$	A behavior goal; the set of goals
c; C	A behavior context item; the set of context items
p; $\mathcal{P}$	A behavior plan; the set of plans
$b = (g, p)$; $\mathcal{B}$	A behavior (including a goal and a plan); the set of behaviors
d	Number of dimensions of the vector space
$\mathbf{g}$, $\mathbf{c}$, $\mathbf{p}$	Vector of goal g, context item c, and plan p
$\varepsilon(p \mid g)$	"Distance to success" of plan p given goal g
$\gamma(p \mid g)$	"Achievement" of plan p given goal g
$e(p \mid g)$	Estimated "effectiveness" of plan p given goal g
$\hat{e}(p \mid g)$	Observed "effectiveness" of plan p given goal g

Definition 2.1 (Goal, Plan, Context, and Behavior). A goal g is the objective, or desired result, that a behavior commits to achieve. A plan $p = \{c_1, c_2, ..., c_n\}$ is a set of context items organized together to achieve the goal. A behavior $b = (g, p)$ indicates a process of executing plan p and targeting at goal g.

Taking paper-publishing behavior as an example. The goal is a specific venue. The contexts include authors, keywords, references, and many other items if possibly extracted. The plan is the manuscript that integrates all the contexts with dedicated efforts.

Definition 2.2 (Observed plan effectiveness). Given a behavior $b = (g, p)$, the observed effectiveness of plan p is the probability of the plan p achieving the goal g, denoted by $e(p \mid g) \in [0, 1]$.

As in the above example, if the behavior was either a success (i.e., accepted) or a failure (i.e., rejected) in the data, the observed effectiveness would be either 1 or 0. When the review scores were available, the effectiveness could be set as the mean or median value of the scores. Unfortunately, usually we only have successful cases in real datasets. We will deal with this issue in our approach.

Now we formally define the problem of learning behavior embeddings for preserving the effectiveness information as follows.

Problem. Given behavior data $D = (\mathcal{G}, \mathcal{P}, C, \mathcal{B})$, where $\mathcal{G}$, $\mathcal{P}$, C, and $\mathcal{B}$ are the sets of goals, plans, contexts, and behaviors, **learn** (1) a goal embedding function $f_g(D) : \mathcal{G} \to \mathbb{R}^d$ and (2) a context embedding function $f_c(D) : C \to \mathbb{R}^d$, where $d \ll \min(|\mathcal{G}|, |C|)$ is the number of dimensions, preserving the observed plan effectiveness information $e(p \mid g)$ for any behavior $b = (g, p) \in \mathcal{B}$.

We will evaluate the behavior embeddings on two interesting tasks:

Task 1 (Goal prediction)**. Given** a plan p, **predict** the goals $g \in \mathcal{G}$ that the plan p is likely to achieve (i.e., of good effectiveness).

For example, given a paper plan with authors, keywords, and references, predict the venue this paper is most likely to be accepted to. The performance on this task reflects the plan effectiveness information preserved in the embedding space.

Task 2 (Context recommendation)**.** Given a goal g and a plan p, **recommend** a context item c to be added into p that maximizes the effectiveness of p w.r.t. g.

For example, given a project plan and a target venue, recommend a co-author, keyword, or reference to make it more likely to be accepted. In our experiments, we hide one of the context items and use embeddings to predict the missing item.

3 THE PROPOSED APPROACH

In this section, we introduce the proposed TUBE model for learning behavior embeddings and then present the optimization in detail.

3.1 The TUBE Model for Behavior Embedding

We denote the embedding vector of behavior goal g, context c, and plan p, by $\mathbf{g}, \mathbf{c}, \mathbf{p} \in \mathbb{R}^d$, respectively. For a plan $p \in \mathcal{P}$, we assume its embedding is the sum of the embeddings of its context items:

$$\mathbf{p} = \sum_{c \in p} \mathbf{c}. \tag{1}$$

We define the (potential) *plan achievement* of plan p w.r.t. a given goal g as follows:

$$\gamma(p \mid g) = \frac{\|\mathbf{p}\| \cos\theta}{\|\mathbf{g}\|}, \tag{2}$$

where θ refers to the angle between the plan's vector $\mathbf{p}$ and the goal's vector $\mathbf{g}$:

$$\cos\theta = \frac{\mathbf{p} \cdot \mathbf{g}}{\|\mathbf{p}\|\|\mathbf{g}\|}, \tag{3}$$

and $\gamma \in (-\infty, \infty)$. As shown in Figure 2, when the plan achievement $\gamma \geq 1$, the projection of the plan's vector $\mathbf{p}$ on the the goal's direction locates on the ray starting from $\mathbf{g}$.

Next, we define the *distance to success* $\varepsilon(p \mid g)$ of plan p w.r.t. goal g as follows:

$$\varepsilon(p \mid g) = \begin{cases} \|\mathbf{p}\| \sin\theta, & \text{for } \gamma(p \mid g) \geq 1, \\ \|\mathbf{p} - \mathbf{g}\|, & \text{for } \gamma(p \mid g) < 1, \end{cases} \tag{4}$$

where $\varepsilon \in [0, +\infty)$. As shown in Figure 2, the distance from a plan to its goal depends on the plan achievement $\gamma(p \mid g)$: (1) for a plan of low plan achievement (e.g., p_1 in the figure), we have $\gamma(p \mid g) < 1$, so the distance is exactly the Euclidean distance between the plan's point $\mathbf{p}$ and the goal's point $\mathbf{g}$; (2) for a plan of high plan achievement (e.g., p_2 in the figure), we have $\gamma(p \mid g) \geq 1$, so the value of ε is defined as the distance from the plan's point $\mathbf{p}$ to the ray starting from the goal's point $\mathbf{g}$.

Each goal g has a region in which any point has a not-larger-than-ε distance to success, which is called the ε-region of the goal g. The shape of ε-region looks like a test tube, as shown in Figure 2.

Then, we define the *plan effectiveness* of p w.r.t goal g as:

$$e(p \mid g) = \tanh\left(\frac{1}{2\,\varepsilon^2(p \mid g)}\right) \in (0, 1]. \tag{5}$$

It transforms the *distance to success* $\varepsilon(p \mid g)$ into a $(0, 1]$-space for learning so that when $\varepsilon(p \mid g) = 0$, $e(p \mid g) = 1$; when $\varepsilon(p \mid g) = +\infty$, $e(p \mid g) = 0$. This is an estimation on the observed plan effectiveness $\hat{e}(p \mid g)$. To preserve the behavior outcomes, the embedding learning is to optimize the following objective function:

$$O = d(\hat{e}(\cdot \mid \cdot), e(\cdot \mid \cdot)), \tag{6}$$

where $d(\cdot, \cdot)$ is the distance between two distributions. We choose to minimize the KL-divergence of the observed and estimated effectiveness distributions. By replacing $d(\cdot, \cdot)$ with the KL-divergence, we have:

$$O = -\sum_{p \in \mathcal{P}, g \in \mathcal{G}} \hat{e}(p \mid g) \log e(p \mid g). \tag{7}$$

Finally, by substituting Eq. (1), (2), (4) and (5) into Eq. (7), we can rewrite the objective function as:

$$O = \begin{cases} -\sum_{p\in\mathcal{P},g\in\mathcal{G}} \hat{e}(p\,|\,g)\,\log\tanh\left(\frac{1}{2\,\|\sum_{c\in p}\mathbf{c}\|^2\,\sin^2\theta}\right), \\ \qquad\qquad \text{for } \|\sum_{c\in p}\mathbf{c}\|\cos\theta - \|\mathbf{g}\| \geq 0, \\ -\sum_{p\in\mathcal{P},g\in\mathcal{G}} \hat{e}(p\,|\,g)\,\log\tanh\left(\frac{1}{2\,\|\sum_{c\in p}\mathbf{c}-\mathbf{g}\|^2}\right), \\ \qquad\qquad \text{for } \|\sum_{c\in p}\mathbf{c}\|\cos\theta - \|\mathbf{g}\| < 0. \end{cases} \tag{8}$$

3.2 Optimization

We adopt the asynchronous stochastic gradient algorithm (ASGD) [15] for optimizing Eq. (8). In each step, the ASGD algorithm samples one positive behavior and t negative behaviors and updates the model parameters, where t is called the rate of *negative sampling* ($t \geq 1$). We present the objective functions for both positive examples O^+ and negative examples O^- where $O' = O^+ + t \times O^-$. Then, we derive the the gradients of O^+ and O^-, w.r.t. context items $\mathbf{c}$ and goal $\mathbf{g}$, respectively. To make it clear, the final optimization objective is to minimize the function O'.

Learning with positive examples: Given a positive example (i.e., observed behavior) b with goal g and plan p, the objective function can be specified as:

$$\begin{aligned} O^+ &= -\hat{e}(p\,|\,g)\,\log\tanh\left(\frac{1}{2\,\varepsilon^2(p\,|\,g)}\right) \\ &= \begin{cases} -\hat{e}(p\,|\,g) \times \log\tanh\left(\frac{1}{2\,\|\sum_{c\in p}\mathbf{c}\|^2\,\sin^2\theta}\right), \\ \qquad\qquad \text{for } \|\sum_{c\in p}\mathbf{c}\|\cos\theta - \|\mathbf{g}\| \geq 0, \\ -\hat{e}(p\,|\,g) \times \log\tanh\left(\frac{1}{2\,\|\sum_{c\in p}\mathbf{c}-\mathbf{g}\|^2}\right), \\ \qquad\qquad \text{for } \|\sum_{c\in p}\mathbf{c}\|\cos\theta - \|\mathbf{g}\| < 0, \end{cases} \end{aligned} \tag{9}$$

where $\cos\theta$ can be re-written from Eq. (1) and (3):

$$\cos\theta = \frac{\sum_{c\in p}\mathbf{c}\cdot\mathbf{g}}{\|\sum_{c\in p}\mathbf{c}\|\|\mathbf{g}\|}. \tag{10}$$

The gradient w.r.t. the embedding vector $\mathbf{c}$ of a context item c in plan p can be derived as follows:

$$\frac{\partial O^+}{\partial \mathbf{c}} = \begin{cases} \frac{-2\,\hat{e}(p\,|\,g)}{\sinh\left(\frac{\|\mathbf{g}\|^2}{\|\mathbf{p}\|^2\,\|\mathbf{g}\|^2-(\mathbf{p}\cdot\mathbf{g})^2}\right)} \times \frac{(\mathbf{p}\cdot\mathbf{g})\|\mathbf{g}\|^2\,\mathbf{g}-\|\mathbf{g}\|^4\,\mathbf{p}}{\left(\|\mathbf{p}\|^2\,\|\mathbf{g}\|^2-(\mathbf{p}\cdot\mathbf{g})^2\right)^2}, \\ \qquad\qquad \text{for } \|\sum_{c\in p}\mathbf{c}\|\cos\theta - \|\mathbf{g}\| \geq 0, \\ \frac{-2\,\hat{e}(p\,|\,g)}{\sinh\left(\frac{1}{\|\mathbf{p}\|^2+\|\mathbf{g}\|^2-2\,\mathbf{p}\cdot\mathbf{g}}\right)} \times \frac{\mathbf{g}-\mathbf{p}}{(\|\mathbf{p}\|^2+\|\mathbf{g}\|^2-2\,\mathbf{p}\cdot\mathbf{g})^2}, \\ \qquad\qquad \text{for } \|\sum_{c\in p}\mathbf{c}\|\cos\theta - \|\mathbf{g}\| < 0, \end{cases} \tag{11}$$

and, the gradient w.r.t. the goal's embedding vector $\mathbf{g}$ can be derived as follows:

$$\frac{\partial O^+}{\partial \mathbf{g}} = \begin{cases} \frac{-2\,\hat{e}(p\,|\,g)}{\sinh\left(\frac{\|\mathbf{g}\|^2}{\|\mathbf{p}\|^2\,\|\mathbf{g}\|^2-(\mathbf{p}\cdot\mathbf{g})^2}\right)} \times \frac{(\mathbf{p}\cdot\mathbf{g})\|\mathbf{g}\|^2\,\mathbf{p}-(\mathbf{p}\cdot\mathbf{g})^2\,\mathbf{g}}{\left(\|\mathbf{p}\|^2\,\|\mathbf{g}\|^2-(\mathbf{p}\cdot\mathbf{g})^2\right)^2}, \\ \qquad\qquad \text{for } \|\sum_{c\in p}\mathbf{c}\|\cos\theta - \|\mathbf{g}\| \geq 0, \\ \frac{-2\,\hat{e}(p\,|\,g)}{\sinh\left(\frac{1}{\|\mathbf{p}\|^2+\|\mathbf{g}\|^2-2\,\mathbf{p}\cdot\mathbf{g}}\right)} \times \frac{\mathbf{p}-\mathbf{g}}{(\|\mathbf{p}\|^2+\|\mathbf{g}\|^2-2\,\mathbf{p}\cdot\mathbf{g})^2}, \\ \qquad\qquad \text{for } \|\sum_{c\in p}\mathbf{c}\|\cos\theta - \|\mathbf{g}\| < 0. \end{cases} \tag{12}$$

Learning with negative examples: During negative sampling, the goal of negative behavior b' is sampled from the set of all goals $\mathcal{G}$ just like the goal of positive behavior, but the plan for negative behavior b' does not necessarily come from the set of observed plans $\mathcal{P}$. The *plan space for negative behaviors* is a combination (with replacement) of context items in C of arbitrary size.

In practice, we find two useful strategies: (1) keeping the plan fixed as the plan for positive behavior and randomly sample a different goal; and, (2) keeping the goal fixed as the goal for positive behavior and randomly sample a subset of positive behavior's plan. Basically, the first strategy means, in the positive behavior, the plan should be tailored for the specific goal; and, the second strategy means, in the positive behavior, any context item is indispensable for the plan to achieve its goal.

If a sampled pair of p and g, constituting a behavior $b' = (g, p)$, does not exist in the observed behavior data, the objective function can be specified as follows:

$$\begin{aligned} O^- &= -\log\tanh\left(\frac{\varepsilon^2(p\,|\,g)}{2}\right), \\ &= \begin{cases} -\log\tanh\left(\frac{\|\sum_{c\in p}\mathbf{c}\|^2\sin^2\theta}{2}\right), \\ \qquad\qquad \text{for } \|\sum_{c\in p}\mathbf{c}\|\cos\theta - \|\mathbf{g}\| \geq 0, \\ -\log\tanh\left(\frac{\|\sum_{c\in p}\mathbf{c}-\mathbf{g}\|^2}{2}\right), \\ \qquad\qquad \text{for } \|\sum_{c\in p}\mathbf{c}\|\cos\theta - \|\mathbf{g}\| < 0. \end{cases} \end{aligned} \tag{13}$$

The gradient w.r.t. the embedding vector $\mathbf{c}$ of a context item c in plan p of behavior b' can be derived as follows:

$$\frac{\partial O^-}{\partial \mathbf{c}} = \begin{cases} \frac{-2}{\sinh\left(\frac{\|\mathbf{p}\|^2\,\|\mathbf{g}\|^2-(\mathbf{p}\cdot\mathbf{g})^2}{\|\mathbf{g}\|^2}\right)} \times \frac{\|\mathbf{g}\|^2\,\mathbf{p}-(\mathbf{p}\cdot\mathbf{g})\,\mathbf{g}}{\|\mathbf{g}\|^2}, \\ \qquad\qquad \text{for } \|\sum_{c\in p}\mathbf{c}\|\cos\theta - \|\mathbf{g}\| \geq 0, \\ \frac{-2}{\sinh(\|\mathbf{p}\|^2+\|\mathbf{g}\|^2-2\,\mathbf{p}\cdot\mathbf{g})} \times (\mathbf{p}-\mathbf{g}), \\ \qquad\qquad \text{for } \|\sum_{c\in p}\mathbf{c}\|\cos\theta - \|\mathbf{g}\| < 0, \end{cases} \tag{14}$$

and, the gradient w.r.t. the goal embedding vector $\mathbf{g}$ of behavior b' can be derived as follows:

$$\frac{\partial O^-}{\partial \mathbf{g}} = \begin{cases} \frac{-2}{\sinh\left(\frac{\|\mathbf{p}\|^2\,\|\mathbf{g}\|^2-(\mathbf{p}\cdot\mathbf{g})^2}{\|\mathbf{g}\|^2}\right)} \times \frac{(\mathbf{p}\cdot\mathbf{g})^2\,\mathbf{g}-(\mathbf{p}\cdot\mathbf{g})\|\mathbf{g}\|^2\,\mathbf{p}}{\|\mathbf{g}\|^4}, \\ \qquad\qquad \text{for } \|\sum_{c\in p}\mathbf{c}\|\cos\theta - \|\mathbf{g}\| \geq 0, \\ \frac{-2}{\sinh(\|\mathbf{p}\|^2+\|\mathbf{g}\|^2-2\,\mathbf{p}\cdot\mathbf{g})} \times (\mathbf{g}-\mathbf{p}), \\ \qquad\qquad \text{for } \|\sum_{c\in p}\mathbf{c}\|\cos\theta - \|\mathbf{g}\| < 0. \end{cases} \tag{15}$$

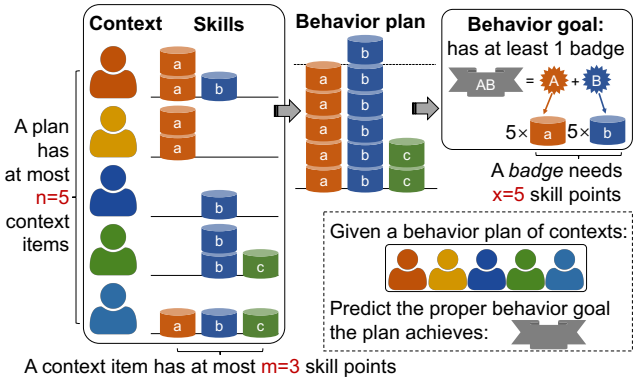


Figure 3: Synthetic $\bar{D}_3$ generation: (1) we generated skill point distributions for each context; (2) the plan combined skill points from all its contexts; (3) a proper goal, of badges, was assigned to the plan. Predicting the goal for a plan is non-trivial when the skill or badge information is hidden.

Table 2: Statistics of synthetic datasets

Dataset	#Skill $k = \lvert\bar{\mathcal{S}}_k\rvert$	#Goal, #Context $\lvert\bar{\mathcal{G}}_k\rvert = \lvert\bar{\mathcal{C}}_k\rvert$	#Plan $\lvert\bar{\mathcal{P}}_k\rvert$	#Valid behavior $\lvert\bar{\mathcal{B}}_k\rvert$
$\bar{D}_3$	3	19	4.25×10^4	3,082
$\bar{D}_4$	4	34	5.76×10^5	11,416
$\bar{D}_5$	5	55	5.46×10^6	30,830
$\bar{D}_6$	6	83	3.92×10^7	68,480
$\bar{D}_7$	7	119	2.25×10^8	133,322
$\bar{D}_8$	8	164	1.08×10^9	236,112
$\bar{D}_9$	9	219	4.49×10^9	389,406

4 EXPERIMENTS

In this section, we set up an interesting scenario (including synthetic data and task definition) to evaluate the proposed idea. Then we do the prediction and recommendation tasks on real datasets. The empirical analysis covers effectiveness, robustness, and sensitivity. Our open-source code package and datasets are available on Github: https://github.com/dmsquare/tube.

4.1 Results on Synthetic Datasets

This section includes (1) dataset generation and task definition, (2) results on method performance, and (3) embedding visualization.

4.1.1 Dataset generation. We simulate a "game" scenario of making plans to achieve goals. We will explain the following concepts used in the data generation: skill, context, plan, badge, and goal. Figure 3 illustrates how a synthetic dataset $\bar{D}_k$ was generated through the game setting (where $k = 3$ in the figure):

(1) Suppose there are k *skills*. If $k = 3$, we name the three skills "a", "b", and "c". You may consider them as "Strength", "Dexterity", "Vitality", the character attributes in Diablo (developed by Blizzard).

(2) Each *context item* (like a *character*) has at most m skill points. A context may assign multiple points to the same skill type. For example, the first context in the figure has 3 skill points: two on skill "a" and one on skill "b". We denote this context by c_{aab}. So the other four context items are denoted by c_{aa}, c_b, c_{bbc}, c_{abc}, respectively.

Table 3: With a small number of dimensions d (4 or 5), our proposed TUBE model has been able to achieve higher-than-0.9 accuracy on predicting the goal for synthetic plans. Note that for dataset $\bar{D}_9$, the task is rather challenging, that is to classify a plan into one category among as many as 219!

Dataset	Number of dimensions d						
	3	4	5	6	7	8	9
$\bar{D}_3$	0.8857	0.8931	0.9109	0.9117	0.9221	0.9097	0.9149
$\bar{D}_4$	0.8755	0.9167	0.9273	0.9240	0.9336	0.9315	0.9336
$\bar{D}_5$	0.8572	0.9136	0.9461	0.9515	0.9568	0.9562	0.9557
$\bar{D}_6$	0.8386	0.9133	0.9315	0.9618	0.9621	0.9693	0.9686
$\bar{D}_7$	0.8384	0.9062	0.9300	0.9600	0.9709	0.9635	0.9638
$\bar{D}_8$	0.7769	0.9081	0.9354	0.9612	0.9660	0.9672	0.9680
$\bar{D}_9$	0.7249	0.9012	0.9339	0.9621	0.9660	0.9697	0.9714

Intuitively, the point number of a specific skill given to a context item represents the level of the character's expertise .

(3) Each *plan* (like a *team*) has at most n context items. (n is like the maximum team size.) In the figure, the plan has 5 contexts. *This plan, or this team, is going to achieve a specific goal in the game.*

(4) To achieve a *goal* (or say, to pass a game stage), the plan has to earn a specific set of badge. In the figure, the goal is named "AB" and it consists of two badges, "A" and "B".

(5) Each *badge* corresponds to a skill type, and it takes x points of the skill to earn one badge. A plan may earn multiple badges of the same skill type. For example, if the number of points on skill "a" is between 5 and 9, the plan earns one "A" badge; if the number is between 10 and 14, the plan earns two "A" badges. We name the *goal* with its badges: if the goal requires two "A" badges and one "B" badge, the goal is "AAB".

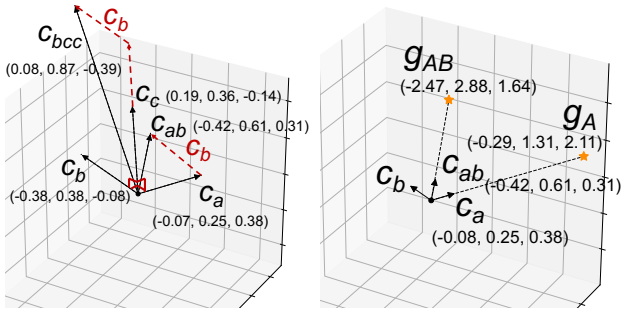
So, when (i) the distributions of contexts' skill points and (ii) the plan's composition are determined, we can assign a specific, proper goal to the plan. As shown in Figure 3, the plan has 5 points on skill "a", 6 points on "b", and 2 points on "c", so the goal is "AB".

When the number of skill types k is determined, or say, the set of skills $\bar{\mathcal{S}}_k$ is finalized, we can generate the set of context items $\bar{\mathcal{C}}_k$, the set of plans $\bar{\mathcal{P}}_k$, and the set of goals $\bar{\mathcal{G}}_k$; and finally, make a synthetic dataset $\bar{D}_k = (\bar{\mathcal{G}}_k, \bar{\mathcal{P}}_k, \bar{\mathcal{C}}_k, \bar{\mathcal{B}}_k)$. We vary k from 3 to 9. Table 2 presents the statistics of the synthetic datasets.

We set $m = 3$ and $n = x = 5$. The number of plans $\lvert\bar{\mathcal{P}}_k\rvert$ increases exponentially from 42,503 ($k = 3$) to almost 4.5 billion ($k = 9$). We build a set of valid behaviors $\bar{\mathcal{B}}_k$ by pairing each plan with a goal and filtering out a plan when (i) the goal is not valid, say, has no badge; or (ii) the plan contains more than m extra skill points that could not turn into badges.

4.1.2 Results on goal prediction. For each synthetic dataset, the task is to predict a plan's goal. Note that (i) *the skills, badges, or skill/badge point distributions are hidden*, and (ii) *the contexts and goals are anonymized*. So the algorithms would just be aware of the plan's composition (as context ids) and goal (as identifier as well) in the training data. This is extremely challenging: it is a $\lvert\bar{\mathcal{G}}_k\rvert$-class classification task (when $k = 9$, the number of classes is 219)!

We use the TUBE method to learn the context and goal embeddings and simply calculate and rank the effectiveness of test plans



(a) Context embeddings (b) Goal embeddings

Figure 4: The context and goal embeddings were learned from synthetic data w/o prior knowledge of skills or badges. The relations between these embeddings in the vector space are identical to their names as expected.

w.r.t the goals for prediction. We use 10-fold cross validation. Table 3 presents the classification *accuracy* and our observations are as follows: (i) The accuracy increases as the embedding size d does because higher dimensionality encodes more information. It meets a plateau when d becomes equal to, or slightly bigger than the number of skills k (say, the model complexity is equal to or higher than the data complexity). (ii) When d is bigger than 4, the accuracy is always higher than 0.9. Surprisingly that it is even higher when k is bigger (and thus the number of classes is bigger). This demonstrates TUBE's model effectiveness and capability.

4.1.3 Visualizing context and goal embeddings. Figure 4 presents the context embeddings (as black arrows) and goal embeddings (as yellow stars) that were learned from the synthetic dataset $\bar{D}_3$ when $d = 3$. Note that the learning process did not use any prior knowledge about the skills, skill points, or badges. None of the embedding was calculated based others. When looking at the relations between these embeddings, we have observations as follows.

First, the contexts that have one point of a skill type (i.e., c_a, c_b, c_c) tend to be orthogonal with each other in the vector space: $\mathbf{c}_a \cdot \mathbf{c}_b = 0.08$, $\mathbf{c}_a \cdot \mathbf{c}_c = 0.02$ and $\mathbf{c}_b \cdot \mathbf{c}_c = 0.07$. And they have similar lengths: $\|\mathbf{c}_a\| = 0.46$, $\|\mathbf{c}_b\| = 0.54$, and $\|\mathbf{c}_c\| = 0.43$.

Second, the embeddings of the contexts of multiple skill points are similar with a composition of the embeddings of each skill point. For example, we find the distance between $\mathbf{c}_{ab}$ and $\mathbf{c}_a + \mathbf{c}_b$ is tiny:

$$\|\mathbf{c}_{ab} - (\mathbf{c}_a + \mathbf{c}_b)\| \quad = \quad 0.04. \tag{16}$$

Third, the relation between a goal embedding and the corresponding context embedding matches our setting on the number of skill points a badge requires ($x = 5$):

$$\|\mathbf{g}_A - 5 \times \mathbf{c}_a\| \,/\, \|\mathbf{g}_A\| \quad = \quad 0.09, \tag{17}$$

$$\|\mathbf{g}_{AB} - 5 \times \mathbf{c}_{ab}\| \,/\, \|\mathbf{g}_{AB}\| \quad = \quad 0.08. \tag{18}$$

We conclude that the proposed TUBE method can discover the underlying features (i.e., the skills) hidden in the behavior data and encode into the context and goal embeddings.

4.2 Results on Real Datasets

Here we first give data description and experimental settings. Then, we present results on the prediction and recommendation tasks.

Table 4: Statistics of paper-publishing behavior datasets: the dataset was named based on the number of goals (venues).

Dataset	Context item			Goal	Behavior
	#Author	#Keyword	#Reference	#Venue	#Paper
D_{2K}	103,460	9,962	254,402	2,000	806,211
D_{5K}	222,991	20,785	293,771	5,000	1,003,614
D_{10K}	409,504	28,670	463,959	10,000	1,133,443

4.2.1 Data description. We collected 1.3M papers published in 13,081 venues from the Microsoft Academic Graph. Based on this raw collection, we built 3 behavior datasets D_{2K}, D_{5K}, and D_{10K} by limiting the number of venues (ranked by their relevance with "data mining") included to 2, 000, 5, 000 and 10, 000, respectively. For each paper, its authors, keywords, and references are considered as context items forming the paper plan; and, its venue is considered as the goal. The statistics of the three datasets are given in Table 4.

4.2.2 Baseline methods. We compare our TUBE method against the state-of-the-art network embedding methods, and a very recent method for success prediction:

(1) LINE [19]: It preserves both local and global structure of the network by conducting edge sampling. We use the advanced version that concatenates the 1st order and the 2nd order representations to get the final representations of nodes.
(2) NODE2VEC [8]: This method uses biased random walks to capture the homophily and structural equivalence properties of network. We also considered the DEEPWALK [14] model here. Since DEEPWALK can be seen as a special case of NODE2VEC that uses truncated uniform random walks, we only report the better performance among them in experiments.
(3) VERSE [21]: It is able to preserves the distributions of a selected vertex-to-vertex similarity measure in homogeneous network such as Personalized PageRank, SimRank and etc.
(4) BINE [7]: This method aims at learning the representations of vertices in a bipartite network. It conducts biased random walks to preserve the long-tail distribution of vertices.
(5) METAPATH2VEC [4]: It is the state-of-the-art method for heterogeneous network embedding. It samples meta path-based random walks for learning. We use the advanced version METAPATH2VEC++ which also conducts heterogeneous negative sampling in network.
(6) LEARNSUC [24]: It is a recent work that formulates behavior as a multi-type itemset instead of a node in networks, and learns item embeddings collectively for success prediction.

4.2.3 Parameter settings. The embedding size d is set as 128 for all methods. To make the comparisons fair, we fix the sampling size $s = 500M$ for all methods: (1) for random walk based methods, $s = r \cdot l \cdot |\mathcal{V}|$, where r is repetition of walks per node, l is walk length, and $\mathcal{V}$ is the set of vertices (contexts, goals, behaviors); (2) for LEARNSUC and TUBE, $s = \sum_{b \in \mathcal{R}} |b|$, where $\mathcal{R}$ is the set of all training samples. The rate of negative sampling t is 5. We conducted reasonable amount of tuning on critical parameters using grid search: for NODE2VEC, p, q are found in $\{0.25, 0.50, 1, 2, 4\}$ as suggested by authors; for BINE, p, β, γ are found in $\{0.01, 0.05, 0.1, 0.5, 1\}$; for VERSE, we use the recommended Personalized PageRank as the network similarity measure; for METAPATH2VEC, meta-path scheme

Table 5: Our proposed TUBE outperforms the state-of-the-art models on predicting the goal (i.e., venue), given the behavior plan (i.e., paper's components), in terms of *recall scores* (higher is better) and *harmonic mean of ranks (HMR)* (smaller is better) on the three datasets (2000/5000/10000-class classification).

Method	D_{2K}					D_{5K}					D_{10K}				
	R@1	R@5	R@10	R@20	HMR	R@1	R@5	R@10	R@20	HMR	R@1	R@5	R@10	R@20	HMR
LINE [19]	0.49%	2.22%	3.88%	6.70%	49.61	0.37%	1.58%	2.67%	4.57%	71.80	0.30%	1.28%	2.18%	3.89%	84.06
NODE2VEC [8]	0.55%	2.06%	3.09%	4.30%	66.64	0.50%	1.90%	2.77%	3.85%	73.75	0.40%	1.91%	3.05%	4.04%	78.99
VERSE [21]	0.82%	2.41%	4.57%	6.16%	52.66	0.62%	2.20%	4.19%	4.91%	68.26	0.51%	1.84%	3.87%	4.64%	74.85
BINE [7]	1.19%	2.78%	5.13%	6.83%	44.23	1.10%	2.61%	4.78%	6.53%	59.19	0.98%	2.09%	4.34%	5.60%	68.50
METAPATH2VEC [4]	1.54%	2.90%	5.62%	6.89%	41.74	1.26%	2.62%	4.86%	6.42%	58.61	0.92%	**2.15%**	4.21%	5.69%	69.38
LEARNSUC [24]	0.25%	3.75%	10.06%	**20.08%**	27.87	0.12%	2.22%	7.22%	14.37%	38.84	0.05%	1.26%	**5.41%**	10.20%	58.28
TUBE	**5.10%**	**11.82%**	**14.93%**	18.32%	**11.60**	**3.43%**	**9.62%**	**12.43%**	**15.17%**	**15.32**	**1.02%**	1.66%	4.17%	**10.68%**	**51.57**

"KAPVPAK" is used to guide random walks; for LEARNSUC, item type weight schemes are picked from {(3, 1, 1), (1, 3, 1), (1, 1, 3)}. All other parameters are set to typical values used in previous studies.

4.2.4 Goal prediction. In this task, we feed the behavior/network embeddings into a logistic regression model to predict the probability of each goal a paper plan achieves, or the probability of the goal being assigned to a paper node as a label. We use 10-fold cross validation and two evaluation metrics:

• Recall@i (i=1, 5, 10, 20): A paper has only one venue in the ground truth though it may be qualified to be accepted to some other venue. This metric is to check whether the top-i predictions can find the true venue. A *higher* recall means better performance.

• Harmonic Mean of Ranks (HMR) [24]: This metric is to see whether the method ranks the true venue at the top. A *smaller* value of HMR indicates better performance.

Table 5 presents the results of baseline methods and our TUBE model on goal prediction. Our observations are as follows.

Overall performance. The best-performed network embedding method is METAPATH2VEC, achieving HMRs of 41.74, 58.61, and 69.38 on D_{2K}, D_{5K} and D_{10K}, respectively. But the best baseline method is the itemset embedding-based LEARNSUC. Despite that its R@1 values are a bit lower than METAPATH2VEC, it achieves HMRs of 27.87, 38.84 and 58.28 on the three datasets, which relatively decreasing the rank over METAPATH2VEC by **33.2%**, **33.7%**, and **16.0%**, respectively. So, the network representation was not able to preserve the set structure of behavior plan. We observe that our TUBE model performs the best over all baseline methods: it achieves HMRs of 11.60, 15.32, and 51.57 (**58.4%**, **60.6%**, and **11.6%** lower than LEARNSUC). It is not easy: it means that for a classification task of as many as 2,000 labels, we can find the truth within the top 12; for 10,000 labels, we can find the truth within the top 52. And it makes R@1 scores (%) of 5.10, 3.43, and 1.02 (relatively **231.2%**, **172.2%**, and **11.0%** higher than METAPATH2VEC). The improvemenets are consistent across three datasets, although it is more salient on D_{2K} because of the relative high data density.

Comparing with network embedding methods. The best homogeneous network embedding method, VERSE, that reserves Personalized PageRank property, can have HMRs of 52.66, 68.26, and 74.85 on the three datasets. It performs better than NODE2VEC and LINE indicating the choice of similarity measure affects the performance. Since bipartite network is a special case of heterogeneous network, the BINE model benefits from incorporating partial heterogeneity information and thus performs better than homogeneous network embedding methods. It scores HMRs of 44.23, 59.19, and 68.50, respectively. The heterogeneous network embedding method METAPATH2VEC performs the best with R@1s (%) of 1.54, 1.26, 0.92, and with HMRs of 41.74, 58.61, and 69.38. Compared with METAPATH2VEC, our TUBE model further improves R@1s to 5.10, 3.43, and 1.02, and has lower HMRs by **72.2%**, **43.29%**, and **25.70%**.

Comparing with LEARNSUC [24]. LEARNSUC is the most competitive baseline model. It formulates behavior as an itemset and learns the item embeddings prserving the set structure. Here, LEARNSUC score HMRs of 27.87, 38.84, and 58.28 on the three datasets, lower than those of METAPATH2VEC by **33.2%**, **33.7%**, and **16.0%**, respectively. LEARNSUC makes the highest scores of R@20 on D_{2K} and R@10 on D_{10K}. However, for most of the metrics, our proposed TUBE outperforms LEARNSUC with a big margin (relatively 58.4%, 60.6%, and 11.6% as given in the overall comparison). This is because LEARNSUC was not able to model the conditional success of a behavior plan w.r.t. a goal. LEARNSUC considers the goal of a behavior as a context item in the behavior. There was no distinction from other contexts such as authors, keywords, or references. TUBE explicitly defines and preserves the conditional outcome of a behavior plan w.r.t. its goal, thus making superior performance in predicting the goal. We observe that TUBE yields much higher R@1 and R@5 values compared with LEARNSUC. This is because TUBE learns the goal embeddings that maintain the information of the goal's difficulty level in the context of plans.

4.2.5 Context recommendation for plan effectiveness. This task is to hide one of the context items from a positive plan and see whether the models can predict the hidden one given the goal. The idea is to find the best context from tons of remaining to improve the plan effectiveness. We focus on the dataset D_{2K} and use 10-fold cross validation. We also use the two kinds of evaluation metrics: (1) Recall@i (i=10, 20, 50, 100) and (2) Harmonic Mean of Ranks (HMR). The task is much more challenging than goal prediction, because the size of label set becomes huge: we have 103,460 author candidates, 9,962 keyword candidates, and 254,502 reference candidates.

Table 6 presents the performance of all methods on this task. Our observations are as follows.

Recommending a co-author. TUBE makes an HMR of 59^{th} on finding the true, hidden co-author from 103,460 author candidates.

Table 6: Our proposed TUBE performs the best on predicting a hidden author/keyword/reference when given a plan (paper's components) and goal (venue), in terms of *recall scores* (higher is better) and *harmonic mean of ranks (HMR)* (smaller is better) on the dataset D_{2K} (103460/9962/254502-class classification). It can put the best collaborator at the top 59 among 103K authors.

Method	Author					Keyword					Reference				
	R@10	R@20	R@50	R@100	HMR	R@10	R@20	R@50	R@100	HMR	R@10	R@20	R@50	R@100	HMR
LINE [19]	0.28%	0.39%	1.22%	2.05%	626.51	0.21%	0.49%	1.14%	2.36%	568.57	0.05%	0.05%	0.44%	0.53%	2772.57
NODE2VEC [8]	0.30%	0.48%	0.96%	1.61%	840.49	0.36%	0.63%	1.09%	2.12%	454.66	0.05%	0.10%	0.25%	0.34%	4031.51
VERSE [21]	0.37%	0.44%	0.67%	1.22%	666.68	0.42%	0.66%	1.20%	2.11%	415.35	0.08%	0.09%	0.40%	0.51%	2555.39
BINE [7]	0.72%	0.83%	1.17%	2.30%	418.37	0.81%	1.15%	1.48%	2.20%	218.33	0.12%	0.19%	0.40%	0.55%	1798.33
METAPATH2VEC [4]	0.81%	1.02%	1.15%	2.46%	367.92	1.03%	1.68%	**3.41%**	**5.35%**	162.79	0.13%	0.17%	0.41%	0.54%	1863.20
LEARNSUC [24]	1.33%	1.78%	2.65%	4.07%	231.25	0.80%	1.17%	1.72%	3.24%	199.00	0.15%	0.31%	0.46%	0.82%	1280.42
TUBE	**2.96%**	**3.64%**	**5.57%**	**8.74%**	**59.09**	**1.25%**	**1.90%**	3.34%	5.07%	**146.80**	**0.24%**	**0.54%**	**0.93%**	**1.37%**	**649.54**

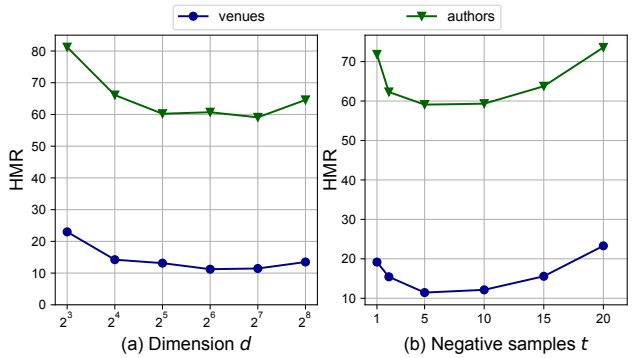


Figure 5: The proposed model is *NOT* sensitive to #Dimension d (left) or the number of #Negative sample t (right).

This is such a challenging task that the best baseline method LEARNSUC can only score an HMR of 231. TUBE model improves relatively by **122.6%** on R@10 and decreases by **74.4%** on HMR, compared to LEARNSUC. It demonstrates that TUBE is effective on recommending collaborators for a paper plan and a target venue.

Recommending a keyword. TUBE makes an R@10 of 1.25% and an HMR of 146.80 given a set of 9,962 keyword candidates. Over the best baseline method LEARNSUC, the R@10 is increased by **56.3%** and the HMR is decreased by **26.3%**. TUBE is more effective on recommending keywords and co-authors for a project plan.

Recommending a reference. This is the most challenging task because we have 254,502 reference candidates. TUBE ranks the real reference at the HMR of 649^{th}, whereas the best baseline LEARNSUC ranks it at the HMR of 1280^{th}. This shows TUBE performs the best but we still have a long way to go for real use of saving the author's time on finding proper references.

4.2.6 Case study and visualization. Figure 1 use a 2-D plane to visualize 10+ venues in the field of computer science and 2 true positive predictions (i.e., predicted to be accepted by certain conferences and were accepted). We can see the venues (goals) form several groups in the vector space w/o prior knowledge: ACL with EMNLP (natural language processing), CIKM with WSDM and WWW (web sciences), ICCV with NIPS and IJCAI (artificial intelligence, machine learning, and computer vision), etc. though some dimensions are not presented.

The two true positives are as follows: one was published in KDD'15 by Zhao *et al.* [28], shown as a green path and vectors; the other paper was published in NIPS'11 by Delalleau *et al.* [2], shown as red vectors. The paper's point starts from the origin, includes the vectors of contexts step by step (some big and some small), and finally, falls into a venue's small *test tube-shaped* success region. The influence of each author on the paper are reflected by the length of his/her embedding vector (though may not have to be the contribution to the paper): Leskovec and Bengio play important roles in the "success" of these papers. The keywords make the paper's point aligns closer to the orientation of the goal's position, while the references contribute some though comparably small amount to the success of these papers.

4.2.7 Parameter sensitivity. We test the sensitivity of TUBE on two parameters: (1) the number of dimensions d and (2) the negative sampling rate t. In Figure 5, we report the HMR of predicting venues (goals) and recommending authors (contexts) on the dataset D_{2K}. We observe that TUBE has stable performance when d is bigger than 2^3 which is too small to model real paper-publishing behaviors. For different negative samples t, TUBE performs well even when t is as small as 2. The best setting is between 5 and 15. We conclude that TUBE is insensitive to d and t.

5 RELATED WORK

In this section, we review methods of related topics to our work.

Representation learning. Learning representations of network data, or network embedding, aims at learning the low-dimensional vector representations of nodes in network while preserving the pair-wise proximities [1, 9–11]. LINE [19] first introduced the notion of 1st and 2nd order proximity to preserve both local and global structure of the network by conducting edge sampling. DEEPWALK [14] used truncated uniform random walks to explore the neighborhood of a node and expected nodes with higher proximity yield similar representations. NODE2VEC [8] extended it to use biased random walks to capture the homophily and structural equivalence properties of network. VERSE [21] was designed to preserve the distributions of a selected pair-wise similarity measure in network such as Personalized PageRank or SimRank. Besides methods focusing on homogeneous networks, BINE [7] was able to learn the representations of vertices in a bipartite network by conducting biased random walks to preserve the long-tail distribution of vertices. For heterogeneous network embedding, METAPATH2VEC [4] was based on meta-path-based random walks and the heterogeneous

Skip-gram model. There is another line of methods that utilized deep models [6, 22, 23, 26]. However, none of these existing methods has an effective formulation of behavior and does not preserve the outcome information of behaviors. A recent work LearnSuc [24] formulated behavior as a multi-type itemset structure instead of nodes in network and preserved the behavioral success property. But this model does not have an explicit definition of behavior outcomes. Our TUBE model learns behavior embeddings via formulating a behavior as a goal-plan pair and preserving the well-defined plan effectiveness and achievement.

Quantifying success. There exists a wide line of research on quantifying success in various fields and areas [12, 16, 17, 27]. Wang *et al.* [25] proposed a model for predicting long-term scientific impact by collapsing the citation histories of papers from different journals and disciplines into a single curve to model the citation dynamics of individual papers. To predict the success in art, Fraiberger *et al.* [5] used a Markov model to predict the career trajectory of individual artists and documents the strong path and history dependence of valuation in art. Yucesoy *et al.* [27] proposed a model aiming at predicting which books will become bestsellers. And, Deville *et al.* [3] studied quantifying the career choices such as changing institutions affecting scientific outcomes. However, the definition of success, or outcome, in previous studies varies greatly from paper citations number to art value. Our TUBE model gives out a mathematical definition of the estimated behavior outcomes in a vector space and is capable of working with behavior data from different areas.

6 CONCLUSIONS

Given a project plan and the goal, in this work, we tried to predict the plan's success rate. We defined a measurement of behavior outcomes, which formed a test tube-shaped region to represent "success", in a vector space. We proposed a novel representation learning method to learn the embeddings of behavior components (including contexts, plans, and goals) by preserving the behavior outcome information. Experiments on real datasets demonstrated that our proposed method significantly improved the performance of goal prediction as well as context recommendation over the state-of-the-art.

ACKNOWLEDGMENTS

The authors would like to thank the anonymous referees for their valuable comments and helpful suggestions. This Research was supported in part by NSF Grants IIS-1447795 and CNS-1622914.

REFERENCES

[1] Peng Cui, Xiao Wang, Jian Pei, and Wenwu Zhu. 2018. A survey on network embedding. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* (2018).
[2] Olivier Delalleau and Yoshua Bengio. 2011. Shallow vs. deep sum-product networks. In *Advances in Neural Information Processing Systems (NIPS)*. 666–674.
[3] Pierre Deville, Dashun Wang, Roberta Sinatra, Chaoming Song, Vincent D Blondel, and Albert-László Barabási. 2014. Career on the move: Geography, stratification, and scientific impact. *Scientific reports* 4 (2014), 4770.
[4] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. 2017. metapath2vec: Scalable representation learning for heterogeneous networks. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining (KDD)*. ACM, 135–144.
[5] Samuel P Fraiberger, Roberta Sinatra, Magnus Resch, Christoph Riedl, and Albert-László Barabási. 2018. Quantifying reputation and success in art. *Science* 362, 6416 (2018), 825–829.
[6] Hongyang Gao, Zhengyang Wang, and Shuiwang Ji. 2018. Large-scale learnable graph convolutional networks. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*. ACM, 1416–1424.
[7] Ming Gao, Leihui Chen, Xiangnan He, and Aoying Zhou. 2018. BiNE: Bipartite Network Embedding.. In *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 715–724.
[8] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining (KDD)*. ACM, 855–864.
[9] Huan Gui, Jialu Liu, Fangbo Tao, Meng Jiang, Brandon Norick, and Jiawei Han. 2016. Large-scale embedding learning in heterogeneous event data. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE, 907–912.
[10] Huan Gui, Jialu Liu, Fangbo Tao, Meng Jiang, Brandon Norick, Lance Kaplan, and Jiawei Han. 2017. Embedding learning with events in heterogeneous information networks. *IEEE transactions on knowledge and data engineering* 29, 11 (2017), 2428–2441.
[11] Meng Jiang, Peng Cui, Nicholas Jing Yuan, Xing Xie, and Shiqiang Yang. 2016. Little is much: Bridging cross-platform behaviors through overlapped crowds. In *Thirtieth AAAI Conference on Artificial Intelligence (AAAI)*.
[12] Meng Jiang, Christos Faloutsos, and Jiawei Han. 2016. Catchtartan: Representing and summarizing dynamic multicontextual behaviors. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 945–954.
[13] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems (NIPS)*. 3111–3119.
[14] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD)*. ACM, 701–710.
[15] Benjamin Recht, Christopher Re, Stephen Wright, and Feng Niu. 2011. Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In *Advances in neural information processing systems (NIPS)*. 693–701.
[16] Vedran Sekara, Pierre Deville, Sebastian E Ahnert, Albert-László Barabási, Roberta Sinatra, and Sune Lehmann. 2018. The chaperone effect in scientific publishing. *Proceedings of the National Academy of Sciences (PNAS)* 115, 50 (2018), 12603–12607.
[17] Roberta Sinatra, Dashun Wang, Pierre Deville, Chaoming Song, and Albert-László Barabási. 2016. Quantifying the evolution of individual scientific impact. *Science* 354, 6312 (2016), 5239.
[18] Arnab Sinha, Zhihong Shen, Yang Song, Hao Ma, Darrin Eide, Bo-june Paul Hsu, and Kuansan Wang. 2015. An overview of microsoft academic service (mas) and applications. In *Proceedings of the 24th international conference on world wide web (WWW)*. ACM, 243–246.
[19] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th international conference on world wide web (WWW)*. International World Wide Web Conferences Steering Committee, 1067–1077.
[20] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. 2008. Arnetminer: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD)*. ACM, 990–998.
[21] Anton Tsitsulin, Davide Mottin, Panagiotis Karras, and Emmanuel Müller. 2018. Verse: Versatile graph embeddings from similarity measures. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web (WWW)*. International World Wide Web Conferences Steering Committee, 539–548.
[22] Ke Tu, Peng Cui, Xiao Wang, Fei Wang, and Wenwu Zhu. 2018. Structural deep embedding for hyper-networks. In *Thirty-Second AAAI Conference on Artificial Intelligence (AAAI)*.
[23] Daixin Wang, Peng Cui, and Wenwu Zhu. 2016. Structural deep network embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining (KDD)*. ACM, 1225–1234.
[24] Daheng Wang, Meng Jiang, Qingkai Zeng, Zachary Eberhart, and Nitesh V Chawla. 2018. Multi-type itemset embedding for learning behavior success. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*. ACM, 2397–2406.
[25] Dashun Wang, Chaoming Song, and Albert-László Barabási. 2013. Quantifying long-term scientific impact. *Science* 342, 6154 (2013), 127–132.
[26] Hongwei Wang, Jia Wang, Jialin Wang, Miao Zhao, Weinan Zhang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. Graphgan: Graph representation learning with generative adversarial nets. In *Thirty-Second AAAI Conference on Artificial Intelligence (AAAI)*.
[27] Burcu Yucesoy and Albert-László Barabási. 2016. Untangling performance from success. *EPJ Data Science* 5, 1 (2016), 17.
[28] Qingyuan Zhao, Murat A Erdogdu, Hera Y He, Anand Rajaraman, and Jure Leskovec. 2015. Seismic: A self-exciting point process model for predicting tweet popularity. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 1513–1522.