

# Representation Learning with Weighted Inner Product for Universal Approximation of General Similarities

Geewook Kim, Akifumi Okuno, Kazuki Fukui and Hidetoshi Shimodaira

Graduate School of Informatics, Kyoto University  
RIKEN Center for Advanced Intelligence Project

{geewook, okuno, k.fukui}@sys.i.kyoto-u.ac.jp, shimo@i.kyoto-u.ac.jp

## Abstract

We propose *weighted inner product similarity* (WIPS) for neural network-based graph embedding. In addition to the parameters of neural networks, we optimize the weights of the inner product by allowing positive and negative values. Despite its simplicity, WIPS can approximate arbitrary general similarities including positive definite, conditionally positive definite, and indefinite kernels. WIPS is free from similarity model selection, since it can learn any similarity models such as cosine similarity, negative Poincaré distance and negative Wasserstein distance. Our experiments show that the proposed method can learn high-quality distributed representations of nodes from real datasets, leading to an accurate approximation of similarities as well as high performance in inductive tasks.

## 1 Introduction

Representation learning of graphs, also known as graph embedding, computes vector representations of nodes in graph-structured data. The learned representations called *feature vectors* are widely-used in a range of applications, e.g., community detection or link prediction on social networks [Perozzi et al., 2014; Tang et al., 2015; Hamilton et al., 2017]. Words in a text corpus also constitute a co-occurrence graph, and the learned feature vectors of words are successfully applied in many natural language processing tasks [Mikolov et al., 2013; Pennington et al., 2014; Goldberg, 2016].

The feature vector is a model parameter or computed from the node’s attributes called *data vector*. Many of state-of-the-art graph embedding methods train feature vectors in a low dimensional Euclidean space so that their inner product expresses the similarities representing the strengths of association among nodes. We call this common framework as *inner product similarity (IPS)* in this paper. IPS equipped with a sufficiently large neural network is highly expressive and it has been proved to approximate arbitrary Positive Definite (PD) similarities [Okuno et al., 2018], e.g., cosine similarity. However, IPS cannot express non-PD similarities.

To express some non-PD similarities, similarity functions based on specific kernels other than inner product have also been considered. For instance, Nickel et al. [2017] applied

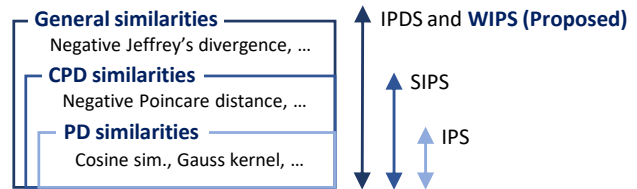


Figure 1: Proposed WIPS is capable of approximating any general similarities, including PD, CPD and a variety of other similarities.

Poincaré distance to learn a hierarchical structure of words in hyperbolic space. Whereas practitioners are required to specify a latent similarity model for representation learning on a given data, finding the best model is not a simple problem, leading to a tedious trial and error process. To address the issue, we need a highly expressive similarity model.

Recently, a simple similarity model named *shifted inner product similarity (SIPS)* has been proposed by introducing bias terms in IPS model [Okuno et al., 2019]. SIPS can approximate arbitrary conditionally PD (CPD) similarities, which includes any PD similarities and some interesting similarities, e.g., negative Poincaré distance [Nickel and Kiela, 2017] and negative Wasserstein distance [Kolouri et al., 2016]. However, the approximation capability of SIPS is still limited to CPD similarities.

To further improve the approximation capability, *inner product difference similarity (IPDS)* has also been proposed by taking the difference between two IPS models [Okuno et al., 2019, See Supplement E]. Any general similarities including indefinite kernels can be expressed as the difference between two PD similarities [Ong et al., 2004], thus they can be approximated by IPDS arbitrarily well. However, only the concept of IPDS has been shown in the literature [Okuno et al., 2019] without any experiments.

In this paper, we first examine IPDS on a range of applications to investigate its effectiveness and weakness. There are, in fact, several practical concerns in IPDS such as a specification of dimensionalities of the two IPS models. Then, in order to remedy the weakness of IPDS, we propose a new model named *weighted inner product similarity (WIPS)*. The core idea of WIPS is to incorporate element-wise weights to IPS so that it generalizes all the above-explained similarity models. WIPS reduces to IPS when all the weights are +1,

and it also reduces to IPDS when they are +1 or -1. The weight values are optimized continuously by allowing positive and negative values, and thus WIPS is free from model selection, except for the choice of dimensionality. Through our extensive experiments, we verify the efficacy of WIPS in real-world applications.

The main contributions of this work are: (i) A new model WIPS is proposed for a universal approximation of general similarities without model selection (see Figure 1). (ii) Experiments of WIPS, as well as IPDS, are conducted for showing their high approximation capabilities and effectiveness on a range of applications. (iii) Our code is publicly available<sup>1</sup>.

## 2 Related Works

In this section, we overview some existing methods concerned with similarity learning.

**Metric learning** is a kind of similarity learning where similarity between two data vectors  $\mathbf{x}_i, \mathbf{x}_j$  is captured by some metric function  $d(\mathbf{x}_i, \mathbf{x}_j)$  [Bellet *et al.*, 2013; Kulis, 2013]. A vast amount of previous works focused on linear metric learning, where Mahalanobis distance  $d_M(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^\top M (\mathbf{x}_i - \mathbf{x}_j)}$  is trained with respect to positive-semidefinite (PSD) matrix  $M$ . Finding such a PSD matrix  $M$  in this context is interpreted as finding a linear projection  $L$  that transforms each data vector by  $L^\top \mathbf{x}$  [Goldberger *et al.*, 2005], since the PSD matrix  $M$  can be decomposed as  $M = LL^\top$ . Therefore, it can also be seen as learning linear embedding so that similar feature vectors get closer. It is extended to learning non-linear transformation by replacing  $L$  with neural networks, which has been known as Siamese architecture [Bromley *et al.*, 1994]. When it comes to just capturing the similarities, especially with bilinear form of  $\mathbf{x}_i^\top M \mathbf{x}_j$ ,  $M$  no longer needs to be PSD [Chechik *et al.*, 2009]; these non-metric learning methods are closely related to our non-PD similarities, but they do not learn embedding representations of data vectors.

**Kernel methods** represent similarity between two data vectors by a positive-definite (PD) kernel. Although the similarity can be shown as the inner product associated with Reproducing Kernel Hilbert Space (RKHS), explicit embedding representations of data vectors are avoided via the kernel trick. Kernel methods have been extended to conditionally PD (CPD) kernels [Schölkopf, 2001] for representing negative distances, and also to general kernels including indefinite kernels [Ong *et al.*, 2004; Oglic and Gaertner, 2018] for representing a variety of non-PD kernels in real-world situations [Schleif and Tino, 2015]. The general kernels are theoretically expressed as inner products associated with Reproducing Kernel Kreĭn Space (RKKS), but such representations in infinite dimensions are avoided in practice.

**Graph embedding** can be interpreted as learning representations of nodes that preserve similarities between nodes defined from their links or neighbourhood structure. The feature vector of a node is computed from its data vector using linear transformation [Yan *et al.*, 2007] or non-linear transformation [Wang *et al.*, 2016], whereas feature vectors are

treated as model parameters when data vectors are not observed [Tang *et al.*, 2015]. Word embedding with Skip-gram model [Mikolov *et al.*, 2013] in natural language processing can also be interpreted as graph embedding without data vectors. They are often 2-view graphs, where vector representations of words and those of contexts are parameterized separately. Some graph embedding methods implement such 2-view settings [Tang *et al.*, 2015], and its generalization to a multi-view setting is straightforward [Okuno *et al.*, 2018]. Interestingly, most of these graph embedding models employ the inner product for computing similarity, thus they can be covered by our argument, although we work only on a 1-view case for simplicity. Note that graph convolution networks [Hamilton *et al.*, 2017] compute the feature vector of a node from the data vectors of its neighbour nodes, so they cannot be covered by our simple setting. However, these methods also use the inner product for computing similarity, hence, they may be improved by using our similarity models.

## 3 Representation Learning for Graph Embedding

Let us consider an undirected graph consisting of  $n (\in \mathbb{N})$  nodes  $\{v_i\}_{i=1}^n$  with a weighted adjacency matrix  $(w_{ij}) \in \mathbb{R}_{\geq 0}^{n \times n}$ . The symmetric weight  $w_{ij} = w_{ji} \geq 0$  is called *link weight*, that represents the observed strength of association between a node pair  $(v_i, v_j)$  for all  $1 \leq i < j \leq n$ . We may also consider a *data vector*  $\mathbf{x}_i \in \mathcal{X}$  at  $v_i$ , that takes a value in a set  $\mathcal{X} \subset \mathbb{R}^p$  ( $p \in \mathbb{N}$ ). The data vector represents attributes or side-information of the node. If we do not have any such attributes, we use *1-hot vector* instead, i.e.,  $\mathbf{x}_i$  is  $n$ -dimensional vector whose  $i$ -th entry is 1 and 0 otherwise, for  $i = 1, \dots, n$ . Our data consists of  $\{w_{ij}\}_{i,j=1}^n$  and  $\{\mathbf{x}_i\}_{i=1}^n$ .

Following the setting of graph embedding described in [Okuno *et al.*, 2018; Okuno *et al.*, 2019], we consider a *feature vector*  $\mathbf{y}_i \in \mathcal{Y}$  at  $v_i$ , that takes a value in a set  $\mathcal{Y} \subset \mathbb{R}^K$  for some dimensionality  $K \in \mathbb{N}$ . Feature vectors are computed by a continuous transformation  $\mathbf{f} : \mathcal{X} \rightarrow \mathcal{Y}$  as

$$\mathbf{y}_i := \mathbf{f}(\mathbf{x}_i) \quad (i = 1, 2, \dots, n).$$

For two data vectors  $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$ , the strength of association between  $\mathbf{x}$  and  $\mathbf{x}'$  is modeled by a *similarity* function

$$h(\mathbf{x}, \mathbf{x}') := g(\mathbf{f}(\mathbf{x}), \mathbf{f}(\mathbf{x}')),$$

where  $g : \mathcal{Y}^2 \rightarrow \mathbb{R}$  is a symmetric continuous function, e.g., the inner product  $g(\mathbf{y}, \mathbf{y}') = \langle \mathbf{y}, \mathbf{y}' \rangle = \sum_{k=1}^K y_k y'_k$ . We employ a simple random graph model by specifying the conditional expectation of  $w_{ij}$  given  $\mathbf{x}_i$  and  $\mathbf{x}_j$  as

$$E(w_{ij} | \mathbf{x}_i, \mathbf{x}_j) = \nu(h(\mathbf{x}, \mathbf{x}')),$$

where  $\nu : \mathbb{R} \rightarrow \mathbb{R}$  is a non-linear function; e.g.,  $\nu(x) = \sigma(x) := (1 + \exp(-x))^{-1}$  is a sigmoid function when  $w_{ij} \in \{0, 1\}$  follows Bernoulli distribution, or  $\nu(x) = \exp(x)$  when  $w_{ij} \in \{0, 1, \dots\}$  follows Poisson distribution.

For learning distributed representations of nodes from the observed data, we consider a parametric model  $\mathbf{f}_\theta : \mathcal{X} \rightarrow \mathcal{Y}$  with unknown parameter vector  $\theta \in \Theta$ . Typically,  $\mathbf{f}_\theta$  is a non-linear transformation implemented as a vector valued

<sup>1</sup><https://github.com/kdrl/WIPS>

neural network (NN), or a linear transformation. The similarity function  $h$  is now modelled as

$$h(\mathbf{x}, \mathbf{x}'; \boldsymbol{\theta}, \boldsymbol{\lambda}) := g_{\boldsymbol{\lambda}}(\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}), \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}')), \quad (1)$$

where  $g_{\boldsymbol{\lambda}}(\mathbf{y}, \mathbf{y}')$  has also a parameter vector  $\boldsymbol{\lambda}$  in general. This architecture for similarity learning is called Siamese network [Bromley *et al.*, 1994]. The parameter  $\boldsymbol{\theta} \in \Theta$  as well as  $\boldsymbol{\lambda}$  may be estimated by maximizing the log-likelihood function. For the case of Bernoulli distribution (i.e., the binary  $w_{ij}$ ), the objective function is

$$\begin{aligned} & \sum_{1 \leq i < j \leq n} w_{ij} \log \nu(h(\mathbf{x}_i, \mathbf{x}_j; \boldsymbol{\theta}, \boldsymbol{\lambda})) \\ & + \sum_{1 \leq i < j \leq n} (1 - w_{ij}) \log(1 - \nu(h(\mathbf{x}_i, \mathbf{x}_j; \boldsymbol{\theta}, \boldsymbol{\lambda}))), \quad (2) \end{aligned}$$

which is optimized efficiently by mini-batch SGD with negative sampling [Okuno *et al.*, 2018]. Once the optimal parameters  $\hat{\boldsymbol{\theta}}$  and  $\hat{\boldsymbol{\lambda}}$  are obtained, we compute feature vectors as  $\mathbf{y}_i = \mathbf{f}_{\hat{\boldsymbol{\theta}}}(\mathbf{x}_i)$ ,  $i = 1, \dots, n$ . For inductive tasks with newly observed data vectors  $\mathbf{x}_i$  ( $i = n + 1, n + 2, \dots$ ), we also compute feature vectors by  $\mathbf{y}_i = \mathbf{f}_{\hat{\boldsymbol{\theta}}}(\mathbf{x}_i)$  unless  $\mathbf{x}_i$  are 1-hot vectors, and then we can compute the similarity function  $g_{\hat{\boldsymbol{\lambda}}}(\mathbf{y}_i, \mathbf{y}_j)$  for any pair of feature vectors.

## 4 Existing Similarity Models and Their Approximation Capabilities

The quality of representation learning relies on the model (1) of similarity function. Although arbitrary similarity functions can be expressed if  $g_{\boldsymbol{\lambda}}$  is a large neural network with many units, our  $g_{\boldsymbol{\lambda}}$  are confined to simple extensions of inner product in this paper. All the models reviewed in this section are very simple and they do not have the parameter  $\boldsymbol{\lambda}$ . You will see very subtle extensions to the ordinary inner product greatly enhance the approximation capability.

### 4.1 Inner Product Similarity (IPS)

Many of conventional representation learning methods use IPS as its similarity model. IPS is defined by

$$h_{\text{IPS}}(\mathbf{x}, \mathbf{x}'; \boldsymbol{\theta}) = \langle \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}), \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}') \rangle,$$

where  $\langle \mathbf{y}, \mathbf{y}' \rangle = \sum_{k=1}^K y_k y'_k$  represents the inner product in Euclidean space  $\mathbb{R}^K$ . Note that,  $h_{\text{IPS}}(\mathbf{x}, \mathbf{x}'; \boldsymbol{\theta})$  reduces to Mahalanobis inner product  $\mathbf{x}^{\top} \mathbf{L}_{\boldsymbol{\theta}} \mathbf{L}_{\boldsymbol{\theta}}^{\top} \mathbf{x}'$  [Kung, 2014], by specifying  $\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x})$  as a linear-transformation  $\mathbf{L}_{\boldsymbol{\theta}}^{\top} \mathbf{x}$  with a matrix  $\mathbf{L}_{\boldsymbol{\theta}} \in \mathbb{R}^{p \times K}$  parameterized by  $\boldsymbol{\theta} \in \Theta$ .

For describing the approximation capability of IPS, we need the following definition. The similarity function  $h$  is also said as a kernel function in our mathematical argument.

**Definition 1 (Positive definite kernel).** A symmetric function  $h : \mathcal{X}^2 \rightarrow \mathcal{R}$  is said to be *positive-definite (PD)* if  $\sum_{i=1}^n \sum_{j=1}^n c_i c_j h(\mathbf{x}_i, \mathbf{x}_j) \geq 0$  for any  $\{\mathbf{x}_i\}_{i=1}^n \subset \mathcal{X}$  and  $\{c_i\}_{i=1}^n \subset \mathbb{R}$ . This definition of PD includes positive semi-definite. Note that  $h$  is called negative definite when  $-h$  is positive definite.

For instance,  $h(\mathbf{x}, \mathbf{x}') = g(\mathbf{f}(\mathbf{x}), \mathbf{f}(\mathbf{x}'))$  is PD when  $g(\mathbf{y}, \mathbf{y}') = \langle \mathbf{y}, \mathbf{y}' \rangle$  (IPS), and  $g(\mathbf{y}, \mathbf{y}') = \langle \mathbf{y}, \mathbf{y}' \rangle / \|\mathbf{y}\|_2 \|\mathbf{y}'\|_2$  (cosine similarity).

IPS using a sufficiently large NN has been proved to approximate *arbitrary* PD similarities [Okuno *et al.*, 2018, Theorem 3.2], in the sense that, for any PD-similarity  $h_*$ ,  $h_{\text{IPS}}(\mathbf{x}, \mathbf{x}'; \hat{\boldsymbol{\theta}})$  equipped with a NN converges to  $h_*(\mathbf{x}, \mathbf{x}')$  as the number of units in the NN and the dimension  $K$  increase. The theorem is proved by combining the universal approximation theorem of NN [Funahashi, 1989; Cybenko, 1989; Yarotsky, 2017; Telgarsky, 2017], and the Mercer's theorem [Minh *et al.*, 2006]. Therefore, IPS, despite its simplicity, covers a very large class of similarities. However, IPS cannot approximate non-PD similarities [Okuno *et al.*, 2019], such as negative Poincaré distance [Nickel and Kiela, 2017] or negative-squared distance  $h(\mathbf{x}, \mathbf{x}') = -\|\mathbf{f}(\mathbf{x}) - \mathbf{f}(\mathbf{x}')\|_2^2$ .

### 4.2 Shifted Inner Product Similarity (SIPS)

For enhancing the IPS's approximation capability, Okuno *et al.* [2019] has proposed *shifted inner product similarity (SIPS)* defined by

$$h_{\text{SIPS}}(\mathbf{x}, \mathbf{x}'; \boldsymbol{\theta}) = \langle \tilde{\mathbf{f}}_{\boldsymbol{\theta}}(\mathbf{x}), \tilde{\mathbf{f}}_{\boldsymbol{\theta}}(\mathbf{x}') \rangle + u_{\boldsymbol{\theta}}(\mathbf{x}) + u_{\boldsymbol{\theta}}(\mathbf{x}'),$$

where  $\tilde{\mathbf{f}}_{\boldsymbol{\theta}} : \mathcal{X} \rightarrow \mathbb{R}^{K-1}$  and  $u_{\boldsymbol{\theta}} : \mathcal{X} \rightarrow \mathbb{R}$  are continuous functions parameterized by  $\boldsymbol{\theta}$ . Thus  $\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}) = (\tilde{\mathbf{y}}, z) := (\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}), u_{\boldsymbol{\theta}}(\mathbf{x})) \in \mathbb{R}^K$  represents the feature vector for  $\mathbf{x}$ . By introducing the bias terms  $u_{\boldsymbol{\theta}}(\mathbf{x}) + u_{\boldsymbol{\theta}}(\mathbf{x}')$ , SIPS extends the approximation capability of IPS.

For describing the approximation capability of SIPS, we need the following definition.

**Definition 2 (Conditionally positive definite kernel).** A symmetric function  $h : \mathcal{X}^2 \rightarrow \mathcal{R}$  is said to be *conditionally PD (CPD)* if  $\sum_{i=1}^n \sum_{j=1}^n c_i c_j h(\mathbf{x}_i, \mathbf{x}_j) \geq 0$  for any  $\{\mathbf{x}_i\}_{i=1}^n \subset \mathcal{X}$  and  $\{c_i\}_{i=1}^n \subset \mathbb{R}$  satisfying  $\sum_{i=1}^n c_i = 0$ .

CPD includes many examples, such as any of PD similarities, negative squared distance, negative Poincaré distance used in Poincaré embedding [Nickel and Kiela, 2017] and some cases of negative Wasserstein distance [Xu *et al.*, 2018].

SIPS using a sufficiently large NN has been proved to approximate *arbitrary* CPD similarities [Okuno *et al.*, 2018, Theorem 4.1]. Therefore, SIPS covers even larger class of similarities than IPS does. However, there still remain non-CPD similarities [Okuno *et al.*, 2018, Supplement E.3] such as negative Jeffrey's divergence and Epanechnikov kernel.

### 4.3 Inner Product Difference Similarity (IPDS)

For further enhancing the SIPS's approximation capability, Okuno *et al.* [2019] has also proposed *inner-product difference similarity (IPDS)*. By taking the difference between two IPSs, we define IPDS by

$$\begin{aligned} h_{\text{IPDS}}(\mathbf{x}, \mathbf{x}'; \boldsymbol{\theta}) &= \langle \mathbf{f}_{\boldsymbol{\theta}}^+(\mathbf{x}), \mathbf{f}_{\boldsymbol{\theta}}^+(\mathbf{x}') \rangle \\ &\quad - \langle \mathbf{f}_{\boldsymbol{\theta}}^-(\mathbf{x}), \mathbf{f}_{\boldsymbol{\theta}}^-(\mathbf{x}') \rangle, \end{aligned}$$

where  $\mathbf{f}_{\boldsymbol{\theta}}^+ : \mathcal{X} \rightarrow \mathbb{R}^{K-q}$  and  $\mathbf{f}_{\boldsymbol{\theta}}^- : \mathcal{X} \rightarrow \mathbb{R}^q$  ( $0 \leq q \leq K$ ) are continuous functions parameterized by  $\boldsymbol{\theta} \in \Theta$ . Thus  $\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}) = (\mathbf{y}^+, \mathbf{y}^-) := (\mathbf{f}_{\boldsymbol{\theta}}^+(\mathbf{x}), \mathbf{f}_{\boldsymbol{\theta}}^-(\mathbf{x})) \in \mathbb{R}^K$  represents the feature vector for  $\mathbf{x}$ . SIPS with  $K$  dimensions

is expressed as IPDS with  $K + 2$  dimensions by specifying  $\mathbf{f}^+(\mathbf{x}) = (\tilde{\mathbf{f}}_\theta(\mathbf{x}), u_\theta(\mathbf{x}), 1) \in \mathbb{R}^{K+1}$  and  $\mathbf{f}^-(\mathbf{x}) = u_\theta(\mathbf{x}) - 1 \in \mathbb{R}$ . Therefore, the class of similarity functions of IPDS includes that of SIPS. Also note that IPDS with  $q = 1$  and a constraint  $h_{\text{IPDS}}(\mathbf{x}, \mathbf{x}; \theta) = -1$  gives the bilinear form in the hyperbolic space, which is used as another representation for Poincaré embedding [Nickel and Kiela, 2018; Leimeister and Wilson, 2018]. For vectors  $\mathbf{y}_i = (\mathbf{y}_i^+, \mathbf{y}_i^-)$ , the bilinear-form  $\langle \mathbf{y}_i^+, \mathbf{y}_j^+ \rangle - \langle \mathbf{y}_i^-, \mathbf{y}_j^- \rangle$  is known as an inner product associated with a pseudo-Euclidean space [Greub, 1975], which is also said to be an indefinite inner product space, and such inner products have been used in applications such as pattern recognition [Goldfarb, 1984].

For describing the approximation capability of IPDS, we need the following definition.

**Definition 3 (Indefinite kernel).** A symmetric function  $h : \mathcal{X}^2 \rightarrow \mathbb{R}$  is said to be *indefinite* if neither of  $h$  nor  $-h$  is positive definite. We only consider  $h$  which satisfies the condition

$$h_1 = h_2 + h \text{ is PD for some PD kernel } h_2,$$

so that  $h$  can be decomposed as  $h = h_1 - h_2$  with two PD kernels  $h_1$  and  $h_2$  [Ong *et al.*, 2004, Proposition 7].

We call  $h$  is *general similarity* function when  $h$  is positive definite, negative definite, or indefinite.

IPDS using a sufficiently large NN (with many units, large  $K - q$  and  $q$ ) has been proved to approximate *arbitrary* general similarities [Okuno *et al.*, 2018, Theorem E.1], thus IPDS should cover an ultimately large class of similarities.

#### 4.4 Practical Issues of IPDS

Although IPDS can universally approximate any general similarities with its high approximation capability, only the theory of IPDS without any experiments nor applications has been presented in previous research [Okuno *et al.*, 2019]. There are several practical concerns when IPDS is implemented. The optimization of  $\mathbf{f}_\theta$  for IPDS can be harder than IPS or SIPS without proper regularization, because the expression of IPDS is redundant; for arbitrary  $\mathbf{f}_\theta^0$ , replacing  $\mathbf{f}_\theta^\pm(\mathbf{x})$  with  $(\mathbf{f}_\theta^\pm(\mathbf{x}), \mathbf{f}_\theta^0(\mathbf{x}))$ , respectively, results in exactly the same similarity model but with redundant dimensions. Choice of the two dimensionalities, i.e.,  $K - q$  and  $q$ , should be important. This leads to our development of new similarity model presented in the next section.

### 5 Proposed Similarity Model

We have seen several similarity models which can be used as a model of (1) for the representation learning in graph embedding. In particular, IPDS has been mathematically proved to approximate any general similarities, but the choice of the two dimensionalities can be a practical difficulty. In order to improve the optimization of IPDS, we propose a new similarity model by modifying IPDS slightly. Our new model has a tunable parameter vector  $\lambda$  in  $g_\lambda$ , whereas all the models reviewed in the previous section do not have the parameter  $\lambda$ . Everything is simple and looks trivial, yet it works very well in practice.

WIPS	$\lambda$	$\mathbf{f}_\theta$
IPS	$\mathbf{1}_K$	$\mathbf{f}_\theta$
SIPS	$(\mathbf{1}_{K+1}, -1)$	$(\tilde{\mathbf{f}}_\theta(\mathbf{x}), u_\theta(\mathbf{x}), 1, u_\theta(\mathbf{x}) - 1)$
IPDS	$(\mathbf{1}_{K-q}, -\mathbf{1}_q)$	$(\mathbf{f}_\theta^+(\mathbf{x}), \mathbf{f}_\theta^-(\mathbf{x}))$

Table 1: WIPS expresses the other models by specifying  $\lambda$  and  $\mathbf{f}_\theta$ .

#### 5.1 Weighted Inner Product Similarity (WIPS)

We first introduce the inner product with tunable weights.

**Definition 4 (Weighted inner product).** For two vectors  $\mathbf{y} = (y_1, y_2, \dots, y_K)$ ,  $\mathbf{y}' = (y'_1, y'_2, \dots, y'_K) \in \mathbb{R}^K$ , *weighted inner product* (WIP) equipped with the weight vector  $\lambda = (\lambda_1, \lambda_2, \dots, \lambda_K) \in \mathbb{R}^K$  is defined as

$$\langle \mathbf{y}, \mathbf{y}' \rangle_\lambda := \sum_{k=1}^K \lambda_k y_k y'_k.$$

The weights  $\{\lambda_k\}_{k=1}^K$  may take both positive and negative values in our setting; thus, WIP is an indefinite inner product [Böttcher and Lancaster, 1996].

WIP reduces to the ordinary inner product  $\langle \mathbf{y}, \mathbf{y}' \rangle$  when  $\lambda = \mathbf{1}_K := (1, 1, \dots, 1) \in \mathbb{R}^K$ . WIP also reduces to the indefinite inner product  $\langle \mathbf{y}_i^+, \mathbf{y}_j^+ \rangle - \langle \mathbf{y}_i^-, \mathbf{y}_j^- \rangle$  when  $\lambda = (1, \dots, 1, -1, \dots, -1) = (\mathbf{1}_{K-q}, -\mathbf{1}_q) \in \mathbb{R}^K$ .

For alleviating the problem to choose an appropriate value of  $q$  in IPDS, we specify WIP  $g_\lambda(\mathbf{y}, \mathbf{y}') = \langle \mathbf{y}, \mathbf{y}' \rangle_\lambda$  in the model (1). We then propose *weighted inner product similarity* (WIPS) defined by

$$h_{\text{WIPS}}(\mathbf{x}, \mathbf{x}'; \theta, \lambda) = \langle \mathbf{f}_\theta(\mathbf{x}), \mathbf{f}_\theta(\mathbf{x}') \rangle_\lambda,$$

where  $\mathbf{f}_\theta : \mathbb{R}^p \rightarrow \mathbb{R}^K$  is a non-linear transformation parameterized by  $\theta \in \Theta$ , and  $\lambda \in \mathbb{R}^K$  is also a parameter vector to be estimated. The parameter vectors  $\theta$  and  $\lambda$  are jointly optimized in the objective function (2). WIPS obviously includes IPS, SIPS, and IPDS as special cases by specifying  $\lambda$  and  $\mathbf{f}_\theta$  (Table 1). WIPS inherits the universal approximation capability of general similarities from IPDS, but WIPS does *not* extend IPDS for a larger class of similarity functions; WIPS is expressed as IPDS by redefining  $\mathbf{f}_\theta$  so that  $k$ -th element is rescaled by  $\sqrt{\lambda_k}$  for  $\lambda_k > 0$  and  $\sqrt{-\lambda_k}$  for  $\lambda_k < 0$ . However, the discrete optimization of  $0 \leq q \leq K$  in IPDS is mitigated by the continuous optimization of  $\lambda \in \mathbb{R}^K$  in WIPS.

#### 5.2 Interpreting WIPS as a Matrix Decomposition

Let  $h_*$  be a general similarity function that is not limited to PD or CPD similarities. WIPS can be interpreted as the eigen-decomposition of the similarity matrix  $\mathbf{H}_*$  whose  $(i, j)$ -th entry is  $h_*(\mathbf{x}_i, \mathbf{x}_j)$ . Since  $\mathbf{H}_*$  is a symmetric real-valued matrix, it can be expressed as

$$\mathbf{H}_* = \mathbf{U} \mathbf{\Lambda} \mathbf{U}^\top,$$

where  $\mathbf{U} = (\mathbf{u}_1, \dots, \mathbf{u}_n) \in \mathbb{R}^{n \times n}$  is the orthogonal matrix of unit eigen-vectors and  $\mathbf{\Lambda} = \text{diag}(\lambda_1, \dots, \lambda_n)$  is the diagonal matrix of real eigenvalues  $\lambda_k \in \mathbb{R}$ . There will be negative  $\lambda_k$ 's when  $h_*$  is a non-PD similarity. Let us reorder the eigenvalues so that  $|\lambda_1| \geq |\lambda_2| \geq \dots \geq |\lambda_n|$ , and

take the first  $K$  eigen-vectors as  $U_K = (\mathbf{u}_1, \dots, \mathbf{u}_K) \in \mathbb{R}^{n \times K}$  and also  $K$  eigenvalues as  $\Lambda_K = \text{diag}(\boldsymbol{\lambda})$  with  $\boldsymbol{\lambda} = (\lambda_1, \dots, \lambda_K)$ . Now the similarity matrix is approximated as

$$H_* \approx U_K \Lambda_K U_K^\top.$$

We may define feature vectors  $\mathbf{y}_i \in \mathbb{R}^K$  for  $\mathbf{x}_i, i = 1, \dots, n$  as  $U_K^\top = (\mathbf{y}_1, \dots, \mathbf{y}_n)$ . This results in

$$h_*(\mathbf{x}_i, \mathbf{x}_j) \approx \langle \mathbf{y}_i, \mathbf{y}_j \rangle_{\boldsymbol{\lambda}} \quad (3)$$

for all  $1 \leq i < j \leq n$ , and the accuracy improves in (3) as  $K$  approaches  $n$ .

The representation learning with WIPS can be interpreted as a relaxed version of the matrix decomposition without any constraints such as the orthogonality of vectors. Instead, we substitute  $\mathbf{y}_i = \mathbf{f}_\theta(\mathbf{x}_i)$  in (3) and optimize the parameter vectors  $\theta$  and  $\boldsymbol{\lambda}$  with an efficient algorithm such as SGD. The above interpretation gives an insight into how WIPS deals with general similarities; allowing negative  $\lambda_k$  in WIPS is essential for non-PD similarities.

## 6 Experiments

### 6.1 Preliminary

**Datasets.** We use the following networks and a text corpus.

- WebKB Hypertext Network<sup>2</sup> has 877 nodes and 1,480 links. Each node represents a hypertext. A link between nodes represents their hyperlink ignoring the direction. Each node has a data vector of 1,703 dimensional bag-of-words. Each node also has semantic class label which is one of {Student, Faculty, Staff, Course, Project}, and university label which is one of {Cornell, Texas, Washington, Wisconsin}.
- DBLP Co-authorship Network [Prado *et al.*, 2013] has 41,328 nodes and 210,320 links. Each node represents an author. A link between nodes represents any collaborations. Each node has a 33 dimensional data vector that represents the number of publications in each of 29 conferences and journals, and 4 microscopic topological properties describing the neighborhood of the node.
- WordNet Taxonomy Tree [Nickel and Kiela, 2017] has 37,623 nodes and 312,885 links. Each node represents a word. A link between nodes represents a hyponymy-hypernymy relation. Each node has a 300 dimensional data vector prepared from Google’s pre-trained word embeddings<sup>3</sup>.
- Text9 Wikipedia Corpus<sup>4</sup> consists of English Wikipedia articles containing about 123M tokens.

**Similarity models.** We compare WIPS with baseline similarity models IPS, SIPS and IPDS [Okuno *et al.*, 2018; Okuno *et al.*, 2019] in addition to the negative Poincaré distance [Nickel and Kiela, 2017]. Because most feature vectors obtained by Poincaré distance are close to the rim of Poincaré ball, we also consider a natural representation in hyperbolic

space by transformation  $y_i \rightarrow 2y_i/(1 - \|y\|_2^2)$  for downstream tasks (denoted as Hyperbolic in Results). We focus only on comparing the similarity models, because some experiments [Okuno *et al.*, 2018] showed that even the simplest IPS model with neural networks can outperform many existing feature learning methods, such as DeepWalk [Perozzi *et al.*, 2014] or GraphSAGE [Hamilton *et al.*, 2017].

**NN architecture.** For computing feature vectors, we use a 2-hidden layer fully-connected neural network  $\mathbf{f}_\theta : \mathbb{R}^p \rightarrow \mathbb{R}^K$  where each hidden layer consists of 2,000 hidden units with ReLU activation function.

**Model training.** The neural network models are trained by an Adam optimizer with negative sampling approach [Mikolov *et al.*, 2013; Tang *et al.*, 2015] whose batch size is 64 and the number of negative sample is 5. The initial learning rate is grid searched over {2e-4, 1e-3}. The dimensionality ratio  $q/K$  of IPDS is grid searched over {0.01, 0.25, 0.5, 0.75, 0.99}. Instead,  $\boldsymbol{\lambda}$  in WIPS is initialized randomly from the uniform distribution of range (0, 1/ $K$ ).

**Implementations.** To make fair “apples-to-apples” comparisons, all similarity models are implemented with PyTorch framework upon the SIPS library<sup>5</sup>. We also made C implementation upon the word2vec program<sup>6</sup>. All our implementations are publicly available at GitHub<sup>7</sup>.

### 6.2 Evaluation Tasks

To assess both the approximation ability of the proposed similarity model as well as the effectiveness of the learned feature vectors, we conduct the following tasks.

**Reconstruction (Task 1).** Most settings are the same as those in Okuno *et al.* [2019]. To evaluate approximation capacity, we embed all nodes and predict all links, i.e., reconstruct the graph structure from the learned feature vectors. The maximum number of iterations for training is set to 100K. ROC-AUC of reconstruction error is evaluated at every 5K iterations and the best score is reported for each model.

**Link prediction (Task 2).** To evaluate generalization error, we randomly split the nodes into train (64%), validation (16%) and test (20%) sets with their associated links. Models are trained on the sub-graph of train set and evaluated occasionally on validation set in iterations; the best evaluated model is selected. The maximum number of iterations and validation interval are 5K/100 for Hypertext Network, and they are 40K/2K for Co-authorship Network and Taxonomy Tree. Then, the trained models are evaluated by predicting links from unseen nodes of test set and ROC-AUC of the prediction error is computed. This is repeated for 10 times and the average score is reported for each model.

**Hypertext classification (Task 3).** To see the effectiveness of learned feature vectors, we conduct hypertext classification with inductive setting. First, feature vectors of all nodes are computed by the models trained for Hypertext Network in Task 2, where  $K$  is also optimized from {10, 50, 100}. Then,

<sup>2</sup><https://linqs.soe.ucsc.edu/data>

<sup>3</sup><https://code.google.com/archive/p/word2vec>

<sup>4</sup>Extension of Text8 corpus, <http://matmahoney.net/dc/textdata>

<sup>5</sup><https://github.com/kdrl/SIPS>

<sup>6</sup><https://github.com/tmikolov/word2vec>

<sup>7</sup><https://github.com/kdrl/WIPS>

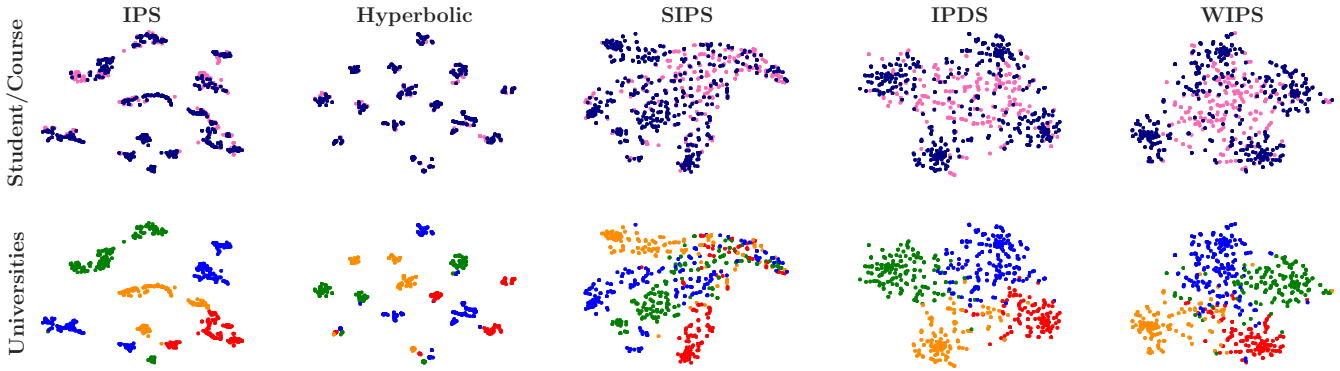


Figure 2: Visualization of Hypertext Network. Two major semantic classes, Student (415 nodes) and Course (218 nodes), are plotted. 10 dimensional feature vectors are mapped to 2d space with T-SNE. The nodes are colored by its semantic labels (upper) for Student (navy) and Course (pink), and also university labels (lower) for Cornell (red), Texas (orange), Washington (green) and Wisconsin (blue). Both class labels are clearly identified with IPDS and WIPS, whereas they become obscure in the other embeddings.

		Dimensionality					
		Reconstruction			Link prediction		
		10	50	100	10	50	100
<b>Hypertext</b>	IPS	91.99	94.23	94.24	77.73	77.62	77.16
	Poincaré	94.09	94.13	94.11	<u>82.21</u>	79.64	79.48
	SIPS	<u>95.11</u>	<u>95.12</u>	<u>95.12</u>	82.01	<u>81.84</u>	<u>81.13</u>
	IPDS	<b>95.12</b>	<b>95.12</b>	<b>95.12</b>	<b>82.59</b>	<b>82.75</b>	<u>82.19</u>
	WIPS	<u>95.11</u>	<b>95.12</b>	<u>95.12</u>	<u>82.38</u>	<u>82.68</u>	<b>82.93</b>
	<b>Co-author</b>	IPS	85.01	86.02	85.80	83.83	84.41
Poincaré		86.84	86.69	86.72	85.82	85.92	85.93
SIPS		90.01	91.35	91.06	<u>88.24</u>	<u>88.69</u>	<u>88.67</u>
IPDS		90.13	91.68	91.59	<b>88.42</b>	<u>88.97</u>	<u>88.85</u>
WIPS		<b>90.50</b>	<b>92.44</b>	<b>92.95</b>	<u>88.16</u>	<b>89.43</b>	<b>89.40</b>
<b>Taxonomy</b>		IPS	79.95	75.80	74.97	67.25	65.71
	Poincaré	91.69	89.10	88.97	83.04	79.52	78.97
	SIPS	98.78	<u>99.75</u>	<u>99.77</u>	90.42	<u>92.12</u>	<u>92.09</u>
	IPDS	<b>99.65</b>	<b>99.89</b>	<b>99.90</b>	<b>95.99</b>	<b>96.37</b>	<u>96.41</u>
	WIPS	99.64	<u>99.85</u>	<u>99.87</u>	<u>95.07</u>	<u>96.36</u>	<b>96.51</b>

Table 2: ROC-AUC for the reconstruction from fully observed data (Task 1) and the inductive link prediction of unseen nodes (Task 2). Boldface is the best, and underlines are 2nd and 3rd scores.

	IPS	Poincaré	Hyperbolic	SIPS	IPDS	WIPS
A	56.08	46.19	47.22	<u>69.09</u>	<u>71.70</u>	<b>73.35</b>
B	91.59	30.17	93.12	<u>93.81</u>	<u>93.81</u>	<b>96.31</b>

Table 3: Classification accuracies on predicting (A) the semantic class label and (B) the university label (Task 3).

a logistic regression classifier is trained on the feature vectors of train set for predicting class labels. The classification accuracy is measured on test set. This is repeated for 10 times and the average score is reported for each model.

**Word similarity (Task 4).** The similarity models are applied to Skip-gram model [Mikolov *et al.*, 2013] to learn word embeddings. To see the quality of the learned word embeddings from Text9 corpus, word-to-word similarity scores induced by the word embeddings are compared with four

	SimLex		YP		WS <sub>SIM</sub>		WS <sub>REL</sub>	
	10	100	10	100	10	100	10	100
IPS	13.6	23.6	17.5	37.3	46.0	73.8	42.3	69.8
SIPS	17.1	31.1	24.9	<u>48.0</u>	55.9	<u>77.0</u>	<u>49.8</u>	<u>71.2</u>
IPDS	16.9	<u>31.3</u>	<u>25.7</u>	<u>48.9</u>	<u>56.2</u>	<u>76.8</u>	<b>49.9</b>	<u>71.4</u>
WIPS	<u>19.2</u>	<b>31.4</b>	<u>27.2</u>	<b>49.0</b>	<b>57.0</b>	<b>78.0</b>	<u>48.7</u>	<b>71.5</b>
SG(K/2)	15.6	27.5	9.90	23.8	20.7	69.1	28.9	67.0
SG*(K/2)	17.0	27.8	18.2	36.4	43.3	75.7	27.1	65.2
SG	<u>18.6</u>	30.9	14.1	31.0	46.1	71.5	46.4	68.7
SG*	<b>20.9</b>	<u>31.3</u>	<b>27.3</b>	39.3	<u>56.3</u>	75.4	39.7	67.1
HSG	19.3	25.8	23.5	39.6	52.9	68.2	36.1	58.2

Table 4: Spearman’s  $\rho$  on the word similarity task (Task 4).

human annotated similarity scores: SimLex-999 [Hill *et al.*, 2015], YP-130 [Yang and Powers, 2005], WS-Sim and WS-Rel [Agirre *et al.*, 2009]. The quality is evaluated by Spearman’s rank correlation. The models are trained with the SGD optimizer used in Mikolov *et al.* [2013]. We set the number of negative samples to 5 and the number of epochs to 5. The initial learning rate is searched over {0.01, 0.025}. As additional baselines, we use the original Skip-gram (SG) [Mikolov *et al.*, 2013] and Hyperbolic Skip-gram (HSG) [Leimeister and Wilson, 2018]. Since SG uses twice as many parameters, we also compare SG with  $K/2$  dimensional word vectors for fair comparisons. To compute similarities with SG, both inner product and cosine similarity (denoted as SG and SG\*) are examined. Each model is trained 3 times and the average score is reported.

### 6.3 Results

Looking at the reconstruction error in Table 2, the approximation capabilities of IPDS and WIPS are very high, followed by SIPS, Poincaré and IPS as expected from the theory. This tendency is also seen in the visualization of Fig. 2. Looking at the link prediction error in Table 2, the difference of models is less clear for generalization performance, but IPDS and WIPS still outperform the others. Table 3 shows the effectiveness of the feature vectors, where IPDS and WIPS have again the best or almost the best scores. As expected, the hyperbolic

representation is much better than Poincaré ball model (Note: this does not apply to Table 2, because these two representations are in fact the same similarity model). Table 4 shows the quality of word embeddings. The new similarity models SIPS, IPDS and WIPS work well with similar performance to SG\* (with twice the number of parameters), whereas IPS works poorly for most cases.

As expected, we can see that IPDS and WIPS are very similar in all the results. Although IPDS and WIPS have the same level of approximation capability, WIPS is much easier to train. For example, the total training time for WIPS was much shorter than IPDS in the experiments. In Task 1 with Hypertext network and  $K = 100$ , IPDS took about 110 min to achieve 95.12 AUC with the five searching points for tuning the dimensionality ratio. But, WIPS only took about 20 min and achieved 95.12. While WIPS stably optimizes the dimensionality ratio of the two IPS models, IPDS is sensitive to the value of dimensionality ratio. For instance, in Task 2 with Hypertext network and  $K = 100$ , we observed that AUC performance of IPDS varies from the worst 64.31 to the best 82.58; WIPS was 82.93.

One of the main reason to use expressive similarity model is to skip the model selection phase, which is often very time-consuming. However, if the training of the model is difficult, the benefit has no meaning and the practitioner needs to take much time and effort. Through the experiments, we can see that WIPS is useful in that it is not only as expressive as IPDS but also easy to train which is very important in practice.

## 7 Conclusion

We proposed a new similarity model WIPS by introducing tunable element-wise weights in inner product. Allowing positive and negative weight values, it has been shown theoretically and experimentally that WIPS has the same approximation capability as IPDS, which has been proved to universally approximate general similarities. As future work, we may use WIPS as a building block of larger neural networks, or extend WIPS to tensor such as  $\sum_{k=1}^K w_k y_k y'_k y''_k$  to represent relations of three (or more) feature vectors  $\mathbf{y}, \mathbf{y}', \mathbf{y}''$ . This kind of consideration is obviously not new, but our work gives theoretical and practical insight to neural network architecture.

## Acknowledgements

We would like to thank anonymous reviewers for their helpful advice. This work was partially supported by JSPS KAKENHI grant 16H02789 to HS, 17J03623 to AO and 18J15053 to KF.

## References

[Agirre *et al.*, 2009] Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalova, Marius Pasca, and Aitor Soroa. A Study on Similarity and Relatedness Using Distributional and WordNet-based Approaches. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, pages 19–27. Association for Computational Linguistics, 2009.

[Bellet *et al.*, 2013] Aurélien Bellet, Amaury Habrard, and Marc Sebban. A survey on metric learning for feature vectors and structured data. *arXiv:1306.6709*, 2013.

[Böttcher and Lancaster, 1996] A. Böttcher and P. Lancaster. *Lectures on Operator Theory and Its Applications*. Fields Institute monographs. American Mathematical Society, 1996.

[Bromley *et al.*, 1994] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a “siamese” time delay neural network. In *Advances in Neural Information Processing Systems 6*, pages 737–744. Morgan-Kaufmann, 1994.

[Chechik *et al.*, 2009] Gal Chechik, Uri Shalit, Varun Sharma, and Samy Bengio. An Online Algorithm for Large Scale Image Similarity Learning. In *Advances in Neural Information Processing Systems 22*, pages 306–314. Curran Associates, Inc., 2009.

[Cybenko, 1989] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems (MCSS)*, 2(4):303–314, 1989.

[Funahashi, 1989] Ken-Ichi Funahashi. On the approximate realization of continuous mappings by neural networks. *Neural Networks*, 2(3):183–192, 1989.

[Goldberg, 2016] Yoav Goldberg. A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research*, 57(1):345–420, 2016.

[Goldberger *et al.*, 2005] Jacob Goldberger, Geoffrey E Hinton, Sam T. Roweis, and Ruslan R Salakhutdinov. Neighbourhood Components Analysis. In *Advances in Neural Information Processing Systems 17*, pages 513–520. MIT Press, 2005.

[Goldfarb, 1984] Lev Goldfarb. A unified approach to pattern recognition. *Pattern Recognition*, 17(5):575–582, 1984.

[Greub, 1975] Werner Greub. *Linear Algebra*. Graduate texts in mathematics. Springer New York, NY, 1975.

[Hamilton *et al.*, 2017] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive Representation Learning on Large Graphs. In *Advances in Neural Information Processing Systems 30*, pages 1024–1034. Curran Associates, Inc., 2017.

[Hill *et al.*, 2015] Felix Hill, Roi Reichart, and Anna Korhonen. Simlex-999: Evaluating Semantic Models with Genuine Similarity Estimation. *Computational Linguistics*, 41(4):665–695, 2015.

[Kolouri *et al.*, 2016] S. Kolouri, Y. Zou, and G. K. Rohde. Sliced Wasserstein Kernels for Probability Distributions. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5258–5267. IEEE, 2016.

[Kulis, 2013] Brian Kulis. Metric Learning: A Survey. *Foundations and Trends in Machine Learning*, 5(4):287–364, 2013.

[Kung, 2014] Sun Yuan Kung. *Kernel Methods and Machine Learning*. Cambridge University Press, 2014.

- [Leimeister and Wilson, 2018] Matthias Leimeister and Benjamin J. Wilson. Skip-gram word embeddings in hyperbolic space. *arXiv:1809.01498*, 2018.
- [Mikolov *et al.*, 2013] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed Representations of Words and Phrases and Their Compositionality. In *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates Inc., 2013.
- [Minh *et al.*, 2006] Ha Quang Minh, Partha Niyogi, and Yuan Yao. Mercer’s Theorem, Feature Maps, and Smoothing. In *Proceedings of the 19th Annual Conference on Learning Theory*, pages 154–168. Springer, 2006.
- [Nickel and Kiela, 2017] Maximillian Nickel and Douwe Kiela. Poincaré Embeddings for Learning Hierarchical Representations. In *Advances in Neural Information Processing Systems 30*, pages 6338–6347. Curran Associates, Inc., 2017.
- [Nickel and Kiela, 2018] Maximillian Nickel and Douwe Kiela. Learning Continuous Hierarchies in the Lorentz Model of Hyperbolic Geometry. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pages 3779–3788. PMLR, 2018.
- [Oglic and Gaertner, 2018] Dino Oglic and Thomas Gaertner. Learning in Reproducing Kernel Krein Spaces. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pages 3859–3867. PMLR, 2018.
- [Okuno *et al.*, 2018] Akifumi Okuno, Tetsuya Hada, and Hidetoshi Shimodaira. A probabilistic framework for multi-view feature learning with many-to-many associations via neural networks. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pages 3885–3894. PMLR, 2018.
- [Okuno *et al.*, 2019] Akifumi Okuno, Geewook Kim, and Hidetoshi Shimodaira. Graph Embedding with Shifted Inner Product Similarity and Its Improved Approximation Capability. In *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 89 of *Proceedings of Machine Learning Research*, pages 644–653. PMLR, 2019.
- [Ong *et al.*, 2004] CS. Ong, X. Mary, S. Canu, and AJ. Smola. Learning with Non-Positive Kernels. In *Proceedings of the 21st International Conference on Machine Learning (ICML)*, pages 81–81. ACM Press, 2004.
- [Pennington *et al.*, 2014] Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543. Association for Computational Linguistics, 2014.
- [Perozzi *et al.*, 2014] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710. ACM, 2014.
- [Prado *et al.*, 2013] Adriana Prado, Marc Plantevit, Céline Robardet, and Jean-Francois Boulicaut. Mining graph topological patterns: Finding covariations among vertex descriptors. *IEEE Transactions on Knowledge and Data Engineering*, 25(9):2090–2104, 2013.
- [Schleif and Tino, 2015] Frank-Michael Schleif and Peter Tino. Indefinite Proximity Learning: A Review. *Neural Computation*, 27(10):2039–2096, 2015.
- [Schölkopf, 2001] Bernhard Schölkopf. The kernel trick for distances. In *Advances in Neural Information Processing Systems 13*, pages 301–307. MIT Press, 2001.
- [Tang *et al.*, 2015] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. LINE: Large-scale Information Network Embedding. In *Proceedings of the 24th International Conference on World Wide Web (WWW)*, pages 1067–1077. International World Wide Web Conferences Steering Committee, 2015.
- [Telgarsky, 2017] Matus Telgarsky. Neural Networks and Rational Functions. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 3387–3393. PMLR, 2017.
- [Wang *et al.*, 2016] Daixin Wang, Peng Cui, and Wenwu Zhu. Structural Deep Network Embedding. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1225–1234. ACM, 2016.
- [Xu *et al.*, 2018] Hongteng Xu, Wenlin Wang, Wei Liu, and Lawrence Carin. Distilled Wasserstein Learning for Word Embedding and Topic Modeling. In *Advances in Neural Information Processing Systems 31*, pages 1723–1732. Curran Associates Inc., 2018.
- [Yan *et al.*, 2007] Shuicheng Yan, Dong Xu, Benyu Zhang, Hong-Jiang Zhang, Qiang Yang, and Stephen Lin. Graph embedding and extensions: A general framework for dimensionality reduction. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(1):40–51, 2007.
- [Yang and Powers, 2005] Dongqiang Yang and David M. W. Powers. Measuring Semantic Similarity in the Taxonomy of WordNet. In *Proceedings of the 28th Australasian Conference on Computer Science - Volume 38*, pages 315–322. Australian Computer Society, Inc., 2005.
- [Yarotsky, 2017] Dmitry Yarotsky. Error bounds for approximations with deep ReLU networks. *Neural Networks*, 94:103–114, 2017.