# Fisher Information Embedding for Node and Graph Learning

**Dexiong Chen** [* 1 2]  **Paolo Pellizzoni** [* 1 2]  **Karsten Borgwardt** [1 2]

## Abstract

Attention-based graph neural networks (GNNs), such as graph attention networks (GATs), have become popular neural architectures for processing graph-structured data and learning node embeddings. Despite their empirical success, these models rely on labeled data and the theoretical properties of these models have yet to be fully understood. In this work, we propose a novel attention-based node embedding framework for graphs. Our framework builds upon a hierarchical kernel for multisets of subgraphs around nodes (*e.g.,* neighborhoods) and each kernel leverages the geometry of a smooth statistical manifold to compare pairs of multisets, by "projecting" the multisets onto the manifold. By explicitly computing node embeddings with a manifold of Gaussian mixtures, our method leads to a new attention mechanism for neighborhood aggregation. We provide theoretical insights into generalizability and expressivity of our embeddings, contributing to a deeper understanding of attention-based GNNs. We propose both efficient unsupervised and supervised methods for learning the embeddings. Through experiments on several node classification benchmarks, we demonstrate that our proposed method outperforms existing attention-based graph models like GATs. Our code is available at https://github.com/BorgwardtLab/fisher_information_embedding.

## 1. Introduction

Graph-structured data has gained significant attention in various domains in recent years. To analyze this type of data with graph relationships, graph neural networks (GNNs) have been widely used and have been shown to be powerful tools. They have been successfully applied to broad classes of application domains, such as drug discovery (Gaudelet et al., 2021), protein design (Ingraham et al., 2019) and social network analysis (Fan et al., 2019).

GNNs construct multilayer models and iteratively perform neighborhood aggregation on previous layers to generate new node representations (Xu et al., 2018). A popular variant of GNNs, known as graph attention networks (GATs) (Veličković et al., 2018), utilizes an attention mechanism for neighborhood aggregation. The attention mechanism of GATs generalizes traditional average and max pooling of the representations of neighbors by performing a data-dependent weighted average of neighbors. Despite the empirical success of GATs, their theoretical properties have not been fully explored. A deeper understanding of the generalizability and expressivity of attention-based GNNs is crucial for identifying limitations and developing more advanced models. Additionally, GATs rely on labeled data, making them less flexible than unsupervised node embedding methods, *e.g.,* these derived from trainable kernel methods (Chen et al., 2020).

Another important class of methods for dealing with the complexity of graph-structured data is node and graph kernels. Node kernels (Kondor & Vert, 2004; Smola & Kondor, 2003) have demonstrated success in capturing the similarity between two nodes in a graph by leveraging the graph Laplacian. However, this class of kernels are known to possess limitations, including high computational complexity (Smola & Kondor, 2003) and challenges in generalizing across multiple graphs. In contrast, graph kernels (Borgwardt et al., 2020; Kriege et al., 2020) rely on the $\mathcal{R}$-convolution framework, which computes local similarities among substructures and aggregates these similarities. Recent work has also identified limitations of this framework and proposed approaches to overcome them. For example, Togninalli et al. (2019) leverage optimal transport to aggregate node embeddings obtained by the Weisfeiler-Lehman (WL) process, in order to improve their ability to capture complex characteristics of the graph. This kernel was later extended to obtain graph embeddings that can handle large-scale datasets (Kolouri et al., 2020). However, these kernels or embeddings are based on unparameterized node embeddings obtained by averaging the node represen-

---
[*]Equal contribution [1]Department of Biosystems Science and Engineering, ETH Zürich, Switzerland [2]SIB Swiss Institute of Bioinformatics, Switzerland. Correspondence to: Dexiong Chen <dechen@ethz.ch>, Paolo Pellizzoni <ppellizzoni@ethz.ch>.

tations of neighbors, without any non-linearities. This can restrict their ability to accurately predict node properties in certain tasks.

In this work, we propose a novel class of attention-based node embeddings to address the above limitations of existing attention-based graph models. Our method approximates multisets of node features with a class of smooth probability distributions and compares pairs of distributions using a statistical divergence, specifically the Kullback–Leibler (KL) divergence. By leveraging the geometry of the manifold of the smooth distributions, we demonstrate that the divergence can be approximated, locally at an anchor distribution, by an embedding distance. Additionally, we show that the embeddings can be computed explicitly for a manifold of Gaussian mixtures and that the resulting embedding of each node leads to a new attention mechanism, providing theoretical insights into the expressivity and generalizability of this class of node embeddings. We propose efficient unsupervised and supervised methods for learning the node embeddings, using the same architecture. We validate our theoretical findings on synthetic datasets. Our empirical evaluation further shows that our proposed node embeddings can achieve similar or even better performance compared to GATs on several node classification benchmarks. To summarize, our key contributions are:

- We introduce a new framework for node representation learning based on hierarchical kernels for multisets.

- We propose a new general embedding, named Fisher information embedding (FIE), for probability measures and multisets by approximating the KL divergence on a smooth statistical manifold. We further provide theoretical insights into the generalizability and expressivity of our embeddings, contributing to a deeper understanding of attention-based GNNs.

- When restricting the analysis to Gaussian mixtures, we demonstrate that FIE exhibits a closed form and can be learned efficiently. We provide both unsupervised and supervised methods for learning the node embeddings.

- Finally, we validate our theoretical findings on both simulated and real-world data and demonstrate that our methods achieve comparable or better results to existing attention-based GNNs.

We will present related work in Section 2. In Section 3, we introduce a general multilayer kernel embedding for nodes where each kernel operates on multisets of node features (*e.g.,* neighborhoods of nodes). In Section 4, we show how to define such a kernel for multisets, by transforming multisets to probability distributions through maximum likelihood estimation and comparing probability distributions

through a kernel defined on the manifold of parametric distributions. In Section 5, we validate the effectiveness of our node embeddings on both simulated and real-world datasets.

## 2. Related work

This paper focuses on node embedding methods and general set pooling methods. We present and discuss in this section recent work in both fields.

**Node embedding methods** Our method provides an unsupervised node embedding approach, which is a popular and well-established method for graph-structured data. Some well-known unsupervised node embedding methods include DeepWalk (Perozzi et al., 2014) and node2vec (Grover & Leskovec, 2016), which extract node representations by performing random walks on the graph. In Rozemberczki et al. (2021), information from random walks at different scales was pooled to produce the embeddings. Additionally, in Abu-El-Haija et al. (2018), the authors proposed a node embedding method with attention-driven random walks. The method presented in Xu et al. (2019) proposed structural node embeddings that were learned by matching two distinct graphs using the optimal transport framework. In Zhu et al. (2021), the authors presented a node embedding that mapped nodes to their embeddings according to both positional proximity and structural similarity criteria. A simple yet effective node embedding can be extracted using the WL algorithm for graph isomorphism (Shervashidze et al., 2011; Togninalli et al., 2019), which used a message-passing framework to refine the either categorical or continuous node attributes. Chen et al. (2020) proposed a kernel embedding for nodes based on sum aggregations of path features. In contrast, our approach defines a kernel embedding for arbitrary multisets beyond simple sum aggregation, including multisets of path features by Chen et al. (2020).

**Set pooling methods** One of the essential components in message-passing-based graph learning methods is pooling information from node neighborhoods, which often involves pooling a (multi)set of features. One of the first methods to provide an encoding for sets was DeepSets (Zaheer et al., 2017), which transformed the elements of the set with a neural network and pooled them using either their sum or average. Other methods include optimal transport-based pooling (Mialon et al., 2020; Kim, 2021) and trainable set embeddings based on the sliced-Wasserstein distance (Naderializadeh et al., 2021). Set pooling is also used to provide graph-level embeddings, typically by pooling node embeddings. For example, WEGL (Kolouri et al., 2020) obtained a graph embedding by representing the set of node embeddings as a probability distribution and computing an embedding based on the Wasserstein distance to a reference distribution. Additionally, in Cheng & Xu (2022), the au-

thors unified some existing global pooling methods, such as mean, max, and attention pooling, under the framework of optimal transport by finding an optimal transport plan between the samples of the sets and the features of the embeddings.

# 3. A general framework for node learning

In this section, we present our node embedding framework for graphs, discuss its link to previous work and provide insights into its expressivity and generalizability.

## 3.1. Multisets and kernels for multisets

A multiset is a generalized notion of a set that allows multiple instances of its elements. By abuse of notation, we will use the same notation as for a set. Here, we handle multisets of features in $\mathbb{R}^d$ living in

$$\mathcal{X}^d = \Big\{ \mathbf{x} \mid \mathbf{x} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \text{ such that}$$
$$\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d \text{ for some } n \geq 1 \Big\}.$$

The cardinality of a multiset, denoted by $|\cdot|$ is defined as the number of elements by summing the multiplicities. Here, we assume that we have access to a kernel defined on the space of multisets $K_{\mathrm{ms}} : \mathcal{X}^d \times \mathcal{X}^d \to \mathbb{R}$ and its exact or approximate embedding $\psi_{\mathrm{ms}} : \mathcal{X}^d \to \mathbb{R}^p$ such that $K_{\mathrm{ms}}(\mathbf{x}, \mathbf{x}') \approx \langle \psi_{\mathrm{ms}}(\mathbf{x}), \psi_{\mathrm{ms}}(\mathbf{x}') \rangle$ through, *e.g.,* the Nyström approximation (Williams & Seeger, 2000). We will describe in Section 4.2 how to define such a kernel on the space of multisets. It is worth noting that $K_{\mathrm{ms}}$ can be the composition of multiple kernels, and the resulting embedding $\psi_{\mathrm{ms}}$ can be obtained by composing the kernel mappings.

## 3.2. A multilayer kernel for nodes and graphs

Let us denote by $\mathcal{G} = (\mathcal{V}, \mathcal{E}, a)$ a graph with vertices $\mathcal{V}$ and edges $\mathcal{E}$ associated with node attributes $a : \mathcal{V} \to \mathbb{R}^{d_0}$. For any node $v \in \mathcal{V}$, we denote by $\mathcal{S}_{\mathcal{G}}(v)$ a certain multiset of subgraphs in $\mathcal{S}$ rooted at $v$ in $\mathcal{G}$ and assume that there exists an *injective* mapping $h$ from $\mathcal{S}$ to $\mathbb{R}^{d_{1/2}}$ that represents any subgraph in $\mathcal{S}$ as a $d_{1/2}$-dimensional vector where $d_{1/2}$ denotes some hidden dimension. We denote by $h(\mathcal{S}_{\mathcal{G}}(v))$ the multisets in $\mathcal{X}^{d_{1/2}}$ consisting of the images of any subgraph in $\mathcal{S}_{\mathcal{G}}(v)$ under $h$. Then, for any two nodes $v$ and $v'$ in two graphs $\mathcal{G}$ and $\mathcal{G}'$, we can define a class of kernels based on the previously defined multiset kernel:

$$K^{(1)}(v, v') := K_{\mathrm{ms}}^{(1)} \left( h(\mathcal{S}_{\mathcal{G}}(v)), h(\mathcal{S}_{\mathcal{G}'}(v')) \right). \quad (1)$$

An (approximate) embedding for the above kernel is given by $\psi_1(v) := \psi_{\mathrm{ms}}^{(1)}(h(\mathcal{S}_{\mathcal{G}}(v))) \in \mathbb{R}^{d_1}$. This results in a new graph with the same set of vertices and edges but a different

attribute function $\mathcal{G}_1 = (\mathcal{V}, \mathcal{E}, \psi_1 : \mathcal{V} \to \mathbb{R}^{d_1})$. Repeating this process $T$ times results in a sequence of graphs $\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_T$, where each graph $\mathcal{G}_t$ carries a new node embedding $\psi_t : \mathcal{V} \to \mathbb{R}^{d_t}$. A final graph-level embedding can also be obtained using a new multiset kernel, by viewing a graph as a multiset of node embeddings. Finally, a multilayer kernel between two nodes can be defined as:

$$K(v, v') := \sum_{t=0}^{T} \langle \psi_t(v), \psi_t(v') \rangle_{\mathbb{R}^{d_t}}, \quad (2)$$

where $\psi_0 = a$. This class of kernels includes several previous graph kernels such as Chen et al. (2020) and Morris et al. (2022).

### 3.2.1. EXAMPLES OF $\mathcal{S}_{\mathcal{G}}$ AND $h$

**Neighborhoods** Similar to GNNs, a straightforward choice of $\mathcal{S}_{\mathcal{G}}(v)$ is the neighborhood of $v$, *i.e.,* $\mathcal{S}_{\mathcal{G}}(v) := \mathcal{N}_{\mathcal{G}}(v)$ or the neighborhood including the root node $\mathcal{S}_{\mathcal{G}}(v) := \mathcal{N}_{\mathcal{G}}(v) \cup \{v\}$. $h_t$ at layer $t \geq 1$ can be simply chosen as the node embedding from the previous layer $h_t := \psi_{t-1}$. Alternatively, one could also consider the 2-tuples consisting of $v$ and its neighbors, namely $\mathcal{S}_{\mathcal{G}}(v) := \{(v, u) : u \in \mathcal{N}_{\mathcal{G}}(v)\}$, and $h_t(v, u) := \frac{1}{2}(\psi_{t-1}(v) + \psi_{t-1}(u))$ as used in Togninalli et al. (2019) and Kolouri et al. (2020).

**Paths and higher order multisets** Similar to Chen et al. (2020), one could also use paths to define the set of subgraphs. Specifically, let us denote by $\mathcal{P}_{\mathcal{G}}^k(v)$ the paths from $v$ of fixed length $k$. Then, we define $\mathcal{S}_{\mathcal{G}}(v) := \mathcal{P}_{\mathcal{G}}^k(v)$ and $h_t(p) := \psi_{t-1}(p)$ for any path $p$ as the concatenation of the node features in this path. For higher order multisets defined by the $k$-dimensional WL algorithms (Morris et al., 2022), one could define $\mathcal{S}_{\mathcal{G}}$ and $h$ in a similar way as in previous work.

In this work, we only focus on the above example of neighborhoods which has a tight link with GATs. We illustrate this step via the arrow (a) of Figure 1.

### 3.2.2. THEORETICAL PROPERTIES

Here, we discuss the necessary conditions for defining a good kernel embedding on multisets and will present in Section 4 an embedding satisfying these conditions. A good kernel embedding $\psi_{\mathrm{ms}}$ should be injective to guarantee the expressive capability of the node embeddings. This injectivity assumption was also widely adopted previously (Xu et al., 2018) to prove the expressivity of GNNs. In particular when we consider the neighborhoods example for $\mathcal{S}_{\mathcal{G}}$ and $h$, we can show the following link between our embedding and WL test by using similar arguments to Xu et al. (2018, Theorem 3)

**Lemma 3.1.** *If $\psi_{\mathrm{ms}}^{(t)}$ is injective for all $t$ in $1, \ldots, T$, then*

$\psi_t(v) \neq \psi_t(v')$ *for any $v$ and $v'$ that the WL isomorphism test decides as non-isomorphic.*

In addition to expressivity, a good kernel should also guarantee the stability and invariance of the predictions, which is controlled by the induced reproducing kernel Hilbert space (RKHS) norm. The RKHS norm also provides a natural way to control the model complexity, leading to generalization bounds through, *e.g.,* Rademacher complexity and margin bounds (Boucheron et al., 2005; Shalev-Shwartz & Ben-David, 2014). Specifically, the generalization bound is given by the following classical result:

**Lemma 3.2** (Boucheron et al. (2005))**.** *Consider a binary task with labels in $\mathcal{Y} = \{-1, 1\}$ and nodes on a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, a)$. Let us denote by $\mathcal{D}$ the data distribution of samples from $\mathcal{D}$ and $\mathcal{Y}$. We define $L(f) := \mathbb{P}_{(v,y) \sim \mathcal{D}}(yf(v) < 0)$ as the expected error of a function $f : \mathcal{V} \to \mathbb{R}$. For a training dataset $(v_1, y_1), \ldots, (v_N, y_N)$, we define the training error with confidence margin $\gamma > 0$ as $L_N^\gamma(f) := 1/N \sum_{i=1}^N \mathbf{1}_{y_i f(x_i) < \gamma}$. Then, with probability $1 - \delta$, we have, for any $\gamma > 0$ and $f \in \mathcal{H}_K$ the RKHS of $K$ in* (2)

$$L(f) \leq L_N^\gamma(f) + O\left(\frac{\bar{B} \|f\|_{\mathcal{H}_K}}{\gamma N} + \sqrt{\frac{\log(1/\delta)}{N}}\right), \quad (3)$$

*where $\bar{B} = \sqrt{1/N \sum_{i=1}^N K(v_i, v_i)}$.*

However, the RKHS norm of $f$ is generally unknown and depends on the regularity of $\psi_t$ which is hard to characterize. In section 4.3, we will show weak results that implicitly control the RKHS norm of $f$. A more comprehensive theoretical analysis of the generalization bounds will be the subject of future work.

# 4. Fisher information embedding for multisets

In this section, we present a kernel for probability measures and multisets. We prove that the associated kernel embedding is a local approximation of the KL divergence, providing theoretical insights into the generalizability and expressivity of the embeddings. For a manifold of Gaussian mixtures, we show that the induced kernel embedding can be computed explicitly and results in a new class of attention-based node embeddings.

## 4.1. A tangent-based kernel for probability measures

Let us consider a smooth statistical manifold $\mathcal{M}$ defined as a subset of probability measures. We assume that $\mathcal{M}$ is a Riemannian manifold and denote by $T_\mu \mathcal{M}$ its tangent space at every point $\mu \in \mathcal{M}$ endowed with a positive definite inner product $g_\mu : T_\mu \mathcal{M} \times T_\mu \mathcal{M} \to \mathbb{R}$, denoted by $\langle , \rangle_{g_\mu}$. We also assume that there exists a retraction mapping $R_\mu$ :

$T_\mu \mathcal{M} \to \mathcal{M}$ on each point $\mu \in \mathcal{M}$ satisfying $R_\mu(0) = \mu$ and $\mathrm{D}R_\mu(0) = \mathrm{id}_{T_\mu \mathcal{M}}$. By the inverse function theorem, $R_\mu$ is a local diffeomorphism and we denote by $R_\mu^{-1}$ its local inverse. Now at every point $\mu$, we can define a positive definite kernel on the proximity of $\mu$ in $\mathcal{M}$:

$$K_\mu(u, v) = \langle R_\mu^{-1}(u), R_\mu^{-1}(v) \rangle_{g_\mu}. \quad (4)$$

By using the mathematical tools from information geometry, we show that this kernel approximates well any divergence associated to the Riemannian metric $g$, which is given by the following theorem adapted from Amari & Nagaoka (2000, Theorem 3.20):

**Theorem 4.1.** *Let $D$ be any divergence. We have*

$$D(u\|\mu) + D(\mu\|v) - D(u\|v) = K_\mu(u, v) + O(\Delta^2), \quad (5)$$

*where $\Delta := \max\{\|\xi(u) - \xi(\mu)\|, \|\xi(v) - \xi(\mu)\|\}$ for an arbitrary coordinate system $\xi$.*

The above kernel is a general tangent-based kernel, generalizing several embeddings proposed in the context of optimal transport, such as Mialon et al. (2020) and Kolouri et al. (2020). In the following section, we will focus on an example of this kernel that exhibits a closed form, which can be easily used in practice.

## 4.2. Fisher information embedding for parametric probability distributions and multisets

Here, we consider a parametric family of probability distributions $\mathcal{M} = \{p_\theta(x)\}_{\theta \in \Theta}$ with $\Theta \subset \mathbb{R}^m$ and $\theta$ denoting a global coordinate system of $\mathcal{M}$. A Riemannian metric defined by the Fisher information metric at every point $p_{\theta_0}$ is given by:

$$\langle \theta, \theta' \rangle_{g_{\theta_0}} := \theta^\top I(\theta_0) \theta' \text{ for any } \theta, \theta' \in T_{\theta_0} \mathcal{M} = \Theta, \quad (6)$$

where $I(\theta_0)$ denotes the Fisher information matrix given by

$$I(\theta_0) := \mathbb{E}[\nabla_\theta \log p_\theta(x)|_{\theta=\theta_0} \nabla_\theta \log p_\theta(x)|_{\theta=\theta_0}^\top] \in \mathbb{R}^{m \times m}. \quad (7)$$

This symmetric matrix is positive definite and is the Hessian matrix of the KL divergence:

**Lemma 4.2.** *Let us denote by $D(\theta_0, \theta) := \mathrm{KL}(p_{\theta_0} \| p_\theta) = \int p_{\theta_0}(x) \log p_{\theta_0}(x)/p_\theta(x) \mathrm{d}x$. If $\theta$ is close to $\theta_0$, we have*

$$D(\theta_0, \theta) = \frac{1}{2}(\theta - \theta_0)^\top I(\theta_0)(\theta - \theta_0) + o((\theta - \theta_0)^2). \quad (8)$$

The retraction mapping at $\theta_0$ is simply defined via the parametric coordinates: $R_{\theta_0} : \theta \mapsto p_{\theta + \theta_0}(x) \in \mathcal{M}$. Since $R_{\theta_0}$ is invertible on the entire $\Theta$, the tangent-based kernel in Equation (4) for this parametric family is defined on $\mathcal{M} \times \mathcal{M}$ and is given by:

$$K_{\theta_0}(p_\theta, p_{\theta'}) = (\theta - \theta_0)^\top I(\theta_0)(\theta' - \theta_0). \quad (9)$$
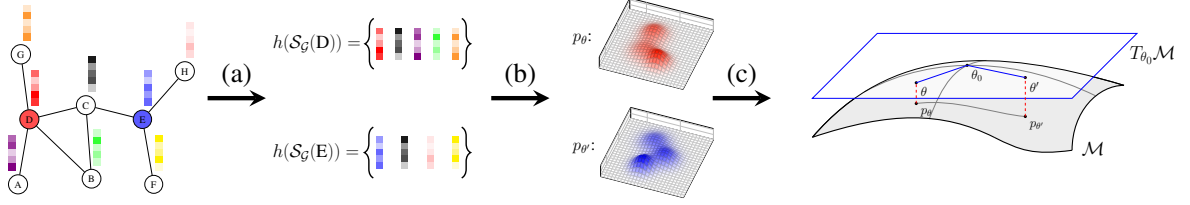
*Figure 1.* An illustration of the Fisher Information Embedding for nodes. (a) Multisets $h(\mathcal{S}_\mathcal{G}(\cdot))$ of node features are obtained from the neighborhoods of each node. (b) Multisets are transformed to parametric distributions, *e.g.,* $p_\theta$ and $p_{\theta'}$, via maximum likelihood estimation. (c) The node embeddings are obtained by estimating the parameter of each distribution using the EM algorithm at an anchor distribution $p_{\theta_0}$ as the starting point. The last panel shows a representation of the parametric distribution manifold $\mathcal{M}$ and its tangent space $T_{\theta_0}\mathcal{M}$ at the anchor point $\theta_0$. The points $p_\theta$ and $p_{\theta'}$ represent probability distributions on $\mathcal{M}$ and the gray dashed line between them their geodesic distance. The red dashed lines represent the retraction mapping $R_{\theta_0}^{-1}$.

And the *Fisher information embedding (FIE) for probability distributions* is defined as:

$$\varphi_{\theta_0}(p_\theta) := I(\theta_0)^{1/2}(\theta - \theta_0). \tag{10}$$

However, when comparing two multisets, $\theta$ and $\theta'$ generally are not known but only samples $\mathbf{x}$ and $\mathbf{x}'$ respectively from the corresponding distributions are known. Since the Fisher information metric locally behaves similarly to the KL divergence one could "project" the distribution of $\mathbf{x}$ onto $\mathcal{M}$ by finding a probability distribution in $\mathcal{M}$ that minimizes its divergence with $p_\mathbf{x}$, where $p_\mathbf{x}$ denotes the true density of $\mathbf{x}$. This amounts to computing the maximum log-likelihood (ML) estimator:

$$\begin{aligned} \theta_{\mathrm{ML}}(\mathbf{x}) &:= \underset{\theta \in \mathcal{M}}{\arg\min}\, \mathrm{KL}(p_\mathbf{x} \| p_\theta) \\ &= \underset{\theta \in \mathcal{M}}{\arg\max}\, \mathbb{E}_{x \sim p_\mathbf{x}(x)}[\log p_\theta(x)]. \end{aligned} \tag{11}$$

For any multisets $\mathbf{x}$ and $\mathbf{x}'$ in $\mathcal{X}^d$, we assume that their ML estimators are accessible and denoted by $\theta_{\mathrm{ML}}(\mathbf{x})$ and $\theta_{\mathrm{ML}}(\mathbf{x}')$ respectively. Then, similar to kernel in Eq. (9), we can define a kernel for multisets:

$$K_{\theta_0}(\mathbf{x}, \mathbf{x}') = (\theta_{\mathrm{ML}}(\mathbf{x}) - \theta_0)^\top I(\theta_0)(\theta_{\mathrm{ML}}(\mathbf{x}') - \theta_0), \tag{12}$$

associated with the kernel mapping

$$\varphi_{\theta_0}(\mathbf{x}) := I(\theta_0)^{1/2}(\theta_{\mathrm{ML}}(\mathbf{x}) - \theta_0), \tag{13}$$

which we call the *Fisher information embedding for multisets*. We illustrate both ML estimation and comparing probability distributions through step (b) and (c) in Figure 1.

However, ML estimators do not necessarily have a closed-form solution in general and could still be hard to compute in practice. Fortunately, if the probability family can be described more simply with an additional latent variable $z$ as $p_\theta(x, z)$, one can instead consider the following simpler estimator dependent of $\theta_0$:

$$\theta_{\mathrm{ML}_{\theta_0}}(\mathbf{x}) := \underset{\theta \in \mathcal{M}}{\arg\max}\, \mathbb{E}_{z|x,\theta_0}[\log p_\theta(x, z)]. \tag{14}$$

This estimator can be seen as a good approximation of the ML estimator (their relationship is given in the Appendix) and can be computed by performing a single iteration of the expectation maximization (EM) algorithm using the initial parameter $\theta_0$. More EM iterations could be performed to approximate better the ML estimator. In Section 4.4, we will show how to explicitly compute this estimator when restricted to a manifold of Gaussian mixtures.

### 4.3. Theoretical analysis of the embedding

We characterize the Fisher information embedding by linking it to the KL divergence. For any anchor parameter $\theta_0 \in \Theta$ and any pair of parameters $\theta, \theta' \in \Theta$, we show that the $\ell_2$-distance between the Fisher information embeddings of $p_\theta$ and $\theta'$ approximates the KL divergence locally. Specifically,

**Theorem 4.3.** *We have*

$$D(p_\theta \| p_{\theta'}) = \frac{\left\| \varphi_{\theta_0}(p_\theta) - \varphi_{\theta_0}(p_{\theta'}) \right\|^2}{2} + O(\Delta^2), \tag{15}$$

*where* $\Delta = \max\{\|\theta - \theta_0\|, \|\theta' - \theta_0\|\}$.

We also show that if the ML estimators in (11) satisfy some conditions, the FIE for multisets is injective:

**Theorem 4.4.** $\varphi_{\theta_0}$ *in* (10) *is injective in* $\Theta$. *If* $\theta_{\mathrm{ML}}$ *defined in* (11) *is injective, then* $\varphi_{\theta_0}$ *in* (13) *is also injective in* $\mathcal{X}^d$.

This lemma combined with Lemma 3.1 shows that the FIE can be as expressive as the WL isomorphism test if the ML estimator is good enough. We can also show that the FIE for probability distributions is Lipschitz:

**Theorem 4.5.** $\varphi_{\theta_0}$ *in* (10) *is* $\left\| I(\theta_0)^{1/2} \right\|_2$-*Lipschitz on* $\Theta$ *such that*

$$\|\varphi_{\theta_0}(p_\theta) - \varphi_{\theta_0}(p_{\theta'})\|_2 \le \left\| I(\theta_0)^{1/2} \right\|_2 \|\theta - \theta'\|_2.$$

The Lipschitz constant plays an important role in deriving the generalization bounds of both deep networks and deep

kernels, as studied in Bartlett et al. (2017); Neyshabur et al. (2018) and the above theorem offers some insights into the generalization properties of FIE. A more complete study of the generalization bounds of FIE will be left for future work.

### 4.4. Fisher information embedding induced by the manifold of Gaussian mixtures

In this section, we specialize our general framework to a particular statistical manifold. Let us consider a simple Gaussian mixture as the parametric family, given by

$$p_\theta(x) := \frac{1}{p} \sum_{j=1}^{p} \mathcal{N}(x; \mu_j, I), \qquad (16)$$

where we only assume $\mu_j$ to be parameters and $\theta := \{\mu_1, \ldots, \mu_p\}$. We also assume that $\theta_0 = \{w_0, \ldots, w_p\}$. Let $\mathbf{z} = \{z_1, \ldots, z_n\}$ be the latent variables that determine the component from which the observation originates such that $p_\theta(x_i) = \sum_{j=1}^{p} p_\theta(x_i, z_i = j)$. Then, for any multiset $\mathbf{x}$, the corresponding log-likelihood can be lower bounded by the Jensen inequality:

$$\frac{1}{n} \sum_{i=1}^{n} \log p_\theta(x_i) = \sum_{i=1}^{n} \log \left( \sum_{j=1}^{p} \alpha_{ij} \frac{p_\theta(x_i, z_i = j)}{\alpha_{ij}} \right)$$
$$\geq \sum_{i=1}^{n} \sum_{j=1}^{p} \alpha_{ij} \log \frac{p_\theta(x_i, z_i = j)}{\alpha_{ij}},$$

for any $\boldsymbol{\alpha} \in \Pi := \{\alpha_{ij} \geq 0, \sum_j \alpha_{ij} = 1\}$. The EM algorithm consists of the following two steps through a maximization-maximization process:

- **E-step:** This step consists of maximizing the right-handed term with respect to $\alpha$, when fixing $\theta = \theta_0$ leading to:

$$\min_{\boldsymbol{\alpha} \in \Pi} - \log p_{\theta_0}(x_i, z_i = j) - H(\boldsymbol{\alpha}),$$

where $H(\boldsymbol{\alpha}) := -\sum_j \alpha_{ij} \log \alpha_{ij}$ denotes the entropy. Without further constraints on $\boldsymbol{\alpha}$, one can show that the optimal $a_{ij}$'s are

$$\alpha_{ij} = p_\theta(z_i = j | x_i) = \frac{\mathcal{N}(x_i, w_j, \Sigma_j)}{\sum_{l=1}^{p} \mathcal{N}(x_i, w_l, \Sigma_l)}.$$

As argued in the Appendix, one can add additional constraints on $\boldsymbol{\alpha}$, *e.g.,* to avoid collapsed solutions. For particular constraints, the optimal $a_{ij}$'s can be found via solving an optimal transport problem.

- **M-step:** This step consists of maximizing the right-handed term with respect to $\theta$. In fact, $\theta$ takes the same

form as a weighted MLE for a normal distribution, thus:

$$\mu_j = \frac{\sum_i \alpha_{ij} x_i}{\sum_i \alpha_{ij}}. \qquad (17)$$

Under the assumption that the optimal parameter $\theta_{\mathrm{ML}}$ is close to $\theta_0$, the procedure converges in one or few iterations.

As shown in Sánchez et al. (2013), under the condition that the components of the Gaussian mixture are well separated, the Fisher information matrix for the parametric family we consider (Eq. (16)) can be approximated by $I(\theta_0) = \frac{1}{p} I$. An approximate form for the FIE on the manifold of Gaussian mixtures is then given by:

$$\varphi_{\theta_0}(\mathbf{x}) = \frac{1}{\sqrt{p}} \left( \left\| \sum_{j=1}^{p} \frac{\sum_i \alpha_{ij} x_i}{\sum_i \alpha_{ij}} - \theta_0 \right). \qquad (18)$$

It is worth noting that the sum of weights $\alpha_{ij} / \sum_i \alpha_{ij}$ equals 1, resulting in a new attention-like mechanism. The parameter of this embedding is the parameter of the anchor distribution $\theta_0$, which can be learned in either unsupervised or supervised way, as described in Section 4.5. While a similar embedding was explored in Kim (2021), our embeddings have a geometric interpretation. Additionally, the $p$ components can be interpreted as "heads", but unlike in GATs, they are not independent. This new form of attention mechanism, as demonstrated in our experimental evaluation in Section 5, yields comparable classification accuracy to GATs.

We discuss here briefly the complexity: each iteration of the EM algorithm involves computing the $np$ entries of the matrix $\boldsymbol{\alpha}$, where $n$ is the number of elements in $\mathbf{x}$ and $p$ is the number of components in the mixture model. Each entry $\alpha_{ij}$ can be computed in constant time after some pre-processing, which leads to a complexity of $O(np)$ per iteration.

As discussed in Section 3.1, the kernel mapping $\psi_{\mathrm{multiset}}$ can be obtained via the composition of multiple kernel mappings. In such a way, one can add addtional non-linearities to the FIE for multisets, making it reminiscent of the attention mechanisms used in Veličković et al. (2018); Brody et al. (2022). This can be achieved by defining $\psi_{\mathrm{multiset}} = \psi \circ \varphi_{\theta_0}$, with $\psi$ a nonlinear function. For example, one can use $\psi(x) = \mathrm{ReLU}(W^\top x)$, or the exponential dot-product kernel used in (Chen et al., 2020) to obtain an unsupervised embedding.

### 4.5. Learning algorithms

We provide both unsupervised and supervised approaches to learn the main parameters $\theta_0$ in the FIE in (18).

**Unsupervised learning** Following the definition, $p_{\theta_0}$ must be close to any data densities $p_\mathbf{x}$ and thus well characterize the the context distribution. As a consequence, we
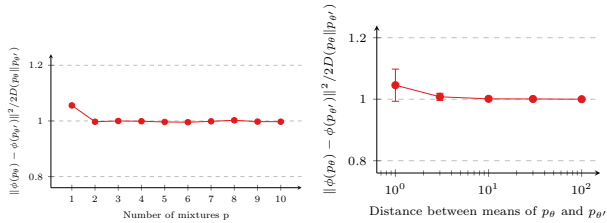
*Figure 2.* Simulation study on the ratio between the KL divergence of two distributions and the squared distance between their embeddings, varying some parameters of the underlying distributions.
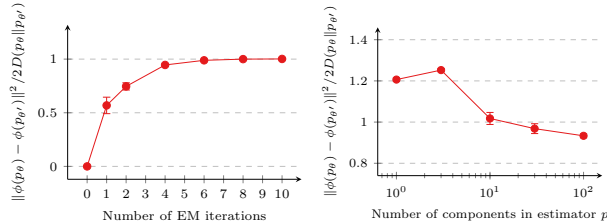


*Figure 3.* Simulation study on the ratio between the KL divergence between two distributions and the squared distance between their embeddings, varying some parameters of embedding method.

can fit the statistical model $p_{\theta_0}$ on the union of the multisets from the training dataset. Particularly in the case of neighborhood multisets in Section 3.2.1, $\theta_0$ can be simply learned by fitting the Gaussian mixture model on a subset of node features sampled from the training dataset. The same process can be used to learn $\theta_0$ in layer $t > 1$ after computing the node embeddings at layer $t - 1$. Furthermore, we remark that learning the embeddings can be carried out in *a single forward pass*, and is therefore much more efficient than end-to-end supervised learning approaches by GNNs. In practice, we observe that k-means, known as a special case of EM algorithms (Lücke & Forster, 2019), performs comparably while being less computationally costly and thus we use k-means to learn $\theta_0$ for all the experiments.

**Supervised learning** Similar to previous studies (Chen et al., 2020; Kim, 2021), $\theta_0$ can also be viewed as model parameters and can be learned end-to-end by minimizing an objective function for a downstream task. In this case, $\theta_0$ (from all FIE layers) will be learned with all the other parameters with back-propagation, including the parameters in the non-linear function and those in the last linear layer.

## 5. Experiments

In this section, we first validate our theoretical findings on synthetic datasets. Then, we compare FIE to existing GNNs on real-world datasets and show that FIE can achieve comparable performance to GATs. Additionally, we show that

unsupervised FIE embedding combined with a state-of-the-art boosting method can achieve significant improvement compared to existing unsupervised node embedding methods and similar performance to its supervised counterpart.

### 5.1. Evaluation on synthetic datasets

We provide empirical evidence that substantiates the results of Theorem 4.3. In the case of Gaussian mixtures, we show that the $\ell_2$-distance in the FIE space approximates well the KL divergence between the underlying distributions. We study the approximation quality under various configurations of the distributions and embedding hyperparameters.

**Experimental setup** We conduct a simulation study on synthetic data. We generate samples from two Gaussian mixture distributions, $p_\theta$ and $p_{\theta'}$, and use the Goldberg approximation (Goldberger et al., 2003) to compute an approximation of the KL divergence between the two Gaussian mixtures as the ground truth. Using these generated samples, we compute the FIEs for the two distributions, as described in Eq. (18). We then analyze the relationship between the $\ell_2$-distance of the two embeddings and $2\mathrm{KL}(p_\theta|p_{\theta'})$ as we vary the parameters of the distributions or the embedding hyperparameters.

**Sensitivity to underlying distributions** Here, we keep the hyperparameters of FIE fixed and only vary the underlying distributions to understand under which conditions the FIE accurately approximates the KL divergence. We generate distributions $p_\theta$ and $p_{\theta'}$ as mixtures of three Gaussians with identity covariance. We vary respectively (i) the number of components, (ii) the distance between the mean of $p_\theta$ and of $p_{\theta'}$, and (iii) the distance among the components in $p_{\theta'}$. As shown in Figure 2, the squared embedding distances closely match the KL divergence $2\mathrm{KL}(p_\theta|p_{\theta'})$, as predicted by Theorem 4.3.

**Sensitivity to embedding hyperparameters** In this set of experiments, we fix $p_\theta$ and $p_{\theta'}$ and vary the parameters of our embedding method. Specifically, we alter the number of components, $p$, in the Gaussian mixtures between 1 and 100, and the number of EM iterations, $M$, in the maximum likelihood estimation between 0 and 10. The results in Figure 3 demonstrate that the squared distance between the embeddings approaches the KL divergence as the number of EM iterations increases, as the estimation error decreases. Additionally, we observe that the quality of the approximation improves as the number of components in the Gaussian mixtures increases.

### 5.2. Evaluation on real-world datasets

We evaluate FIE and compare its variants to state-of-the-art attention-based GNNs and unsupervised node embedding

*Table 1.* Classification accuracy on real world datasets. The reported values are taken from the literature (denoted with an asterisk) or are obtained from our reimplementations. In this latter case, we report the mean and the standard deviation over 10 runs.

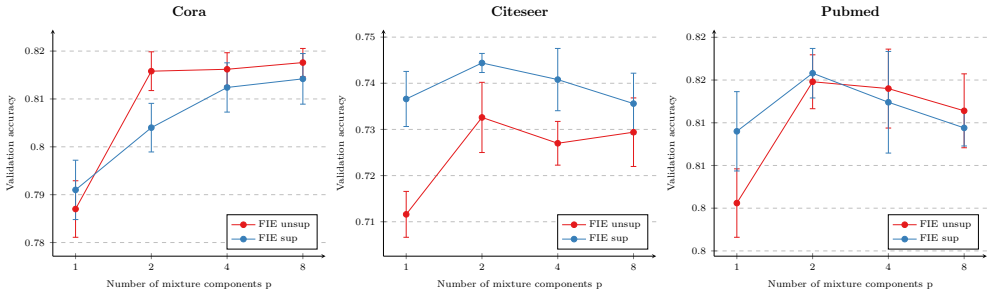| Method | Semi-supervised (transductive) learning tasks | | | Supervised learning tasks | | |
| --- | --- | --- | --- | --- | --- | --- |
| | Cora | Citeseer | Pubmed | Reddit | ogbn-arxiv | ogbn-products |
| WWL | $76.60 \pm 0.00$ | $66.60 \pm 0.00$ | $78.70 \pm 0.00$ | $96.13 \pm 0.04$ | $64.67 \pm 0.08$ | $62.53 \pm 0.11$ |
| Node2vec | $73.55 \pm 0.07$ | $54.01 \pm 1.16$ | $71.22 \pm 1.04$ | $95.58 \pm 0.08$ | $70.67 \pm 0.09$ | $74.77 \pm 0.14$ |
| FIE unsup | $\mathbf{82.36 \pm 0.28}$ | $72.02 \pm 1.34$ | $\mathbf{79.22 \pm 0.17}$ | $\mathbf{96.61 \pm 0.02}$ | $72.17 \pm 0.07$ | $79.24 \pm 0.11$ |
| MLP | $55.1^*$ | $46.5^*$ | $71.4^*$ | - | $55.50 \pm 0.23^*$ | $61.06 \pm 0.08^*$ |
| Label Propagation | $68.0^*$ | $45.3^*$ | $63.0^*$ | - | $68.32 \pm 0.00^*$ | $74.34 \pm 0.00^*$ |
| GCN | $81.4 \pm 0.50^*$ | $70.9 \pm 0.50^*$ | $79.0 \pm 0.3^*$ | - | $71.74 \pm 0.29^*$ | $78.97 \pm 0.33^*$ |
| GraphSAGE | - | - | - | $94.32^*$ | $71.49 \pm 0.27^*$ | $78.70 \pm 0.36^*$ |
| GAT | $81.59 \pm 0.67$ | $70.08 \pm 0.58$ | $79.14 \pm 0.92$ | $96.37 \pm 0.18$ | $71.59 \pm 0.38^*$ | $79.04 \pm 1.54^*$ |
| GATv2 | $82.04 \pm 0.82$ | $69.72 \pm 1.03$ | $75.66 \pm 0.93$ | $96.58 \pm 0.01$ | $71.87 \pm 0.25^*$ | $\mathbf{80.63 \pm 0.70}^*$ |
| FIE sup | $81.79 \pm 0.99$ | $\mathbf{72.51 \pm 0.34}$ | $78.35 \pm 0.26$ | $96.25 \pm 0.06$ | $\mathbf{72.39 \pm 0.21}$ | $79.25 \pm 0.25$ |



*Figure 4.* Effect of number of mixture components $p$ on the validation accuracy for semi-supervised learning datasets.

methods on several real-world datasets for (semi-) supervised node classification.

**Datasets and experimental setup** We assess the performance of our method with six widely used benchmark datasets for node classification, including Cora, Citeseer, Pubmed (Sen et al., 2008) as semi-supervised transductive learning datasets and Reddit (Hamilton et al., 2017), ogbn-arxiv (Hu et al., 2020), ogbn-products (Hu et al., 2020) as medium- or large-scale supervised learning datasets.

We compare our method to both unsupervised and supervised methods for node representation learning. Unsupervised node embedding methods include WWL (Togninalli et al., 2019), which use simple average aggregation of neighborhoods, and Node2vec (Grover & Leskovec, 2016). The supervised comparison partners include MLP (Hu et al., 2020), Label propagation (Zhu & Ghahramani, 2002), GCN (Kipf & Welling, 2017), GraphSAGE (Hamilton et al., 2017), GAT (Veličković et al., 2018), and GATv2 (Brody et al., 2022).

All results for the comparison partners are either taken from the original paper (denoted with an asterisk in the table) or obtained by our reimplementation if not available. All results are computed from 10 runs using different random seeds with the optimal hyperparameters selected on the vali-

dation set. Full details on the datasets, experimental setup and implementation details can be found in the Appendix.

**Unsupervised node embedding** In Section 4.5, we propose an unsupervised method for learning the anchor distribution parameter $\theta_0$ through a layer-by-layer k-means algorithm, similar to Chen et al. (2020). We sample a subset of 300,000 nodes from each layer and perform k-means on the subset to compute $\theta_0$. Once learned, the node embeddings for that layer can be computed. This method requires only *one single forward pass* to obtain the final node embeddings. We then train a classifier using the concatenated node embeddings across all layers, as outlined in Eq. (2). For semi-supervised learning datasets, we use a logistic regression model, which is less prone to overfitting. For large supervised datasets, we use a LightGBM classifier (Ke et al., 2017), the hyperparameters of which are automatically tuned by FLAML (Wang et al., 2021). The same classifiers are used for other comparison partners.

Our results presented in Table 1 demonstrate that our unsupervised embedding (FIE unsup) consistently outperforms existing unsupervised methods by a large margin over all datasets. More remarkably, our unsupervised embedding achieves comparable or better performance compared to supervised methods, including GAT variants and its supervised counterpart, showing the effectiveness of our unsuper-
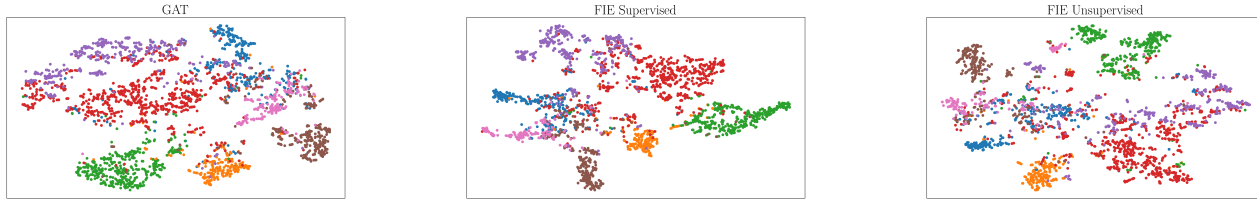
*Figure 5.* Left and center panels show the t-SNE visualization of the node embedding space on the last hidden layer (with hidden dimension 64) for GAT and supervised FIE on the Cora dataset. Right panel shows the t-SNE visualization of the unsupervised FIE (dimension 64) on the Cora dataset. The embedding colors represent the node labels, and are consistent throughout the plots.

vised learning strategy. The results suggest that learning a good classifier on a general-purpose node representation in a decoupled way can lead to strong performance.

**Supervised node embedding** In addition to our unsupervised method for learning the anchor distribution parameter $\theta_0$, we also propose a supervised approach for training the entire model end-to-end, as described in Section 4.5. We utilize a cross entropy loss and the Adam optimizer (Kingma & Ba, 2015) for optimization. Our unsupervised embedding method serves as a natural initialization for $\theta_0$ and we use it for semi-supervised learning datasets. Our results, shown in Table 1, demonstrate that our supervised FIE method achieves comparable or better performance compared to GAT variants, without the needing complex tricks in the model architecture such as using Leaky ReLU.

**Effect of number of mixture components** In this study, we investigate the effect of the number of components $p$ used in the Gaussian mixture model on our FIE approach. Specifically, we aim to demonstrate that the aggregation mechanism in Eq. (18) derived from our approach is superior to simple average pooling. Figure 4 illustrates the impact of varying the number of components on validation accuracy for both supervised and unsupervised variants across several datasets. As the number of components $p$ decreases to 1, our FIE method approaches average pooling of the neighbors. However, as $p$ increases, our FIE behaves similarly to other multi-head attention-based message passing methods, as the concatenation of weighted averages of neighbors. Our results suggest that using $p > 1$ leads to substantial improvements in performance, with the optimal value of $p$ varying depending on the dataset. For example, on Cora, $p = 8$ produces the best results, while the best performance is attained at $p = 2$ for Citeseer and Pubmed. This study illustrates the superiority of our attention mechanism to average pooling and suggests that the optimal value of $p$ should be selected through cross validation.

**t-SNE visualization of embedding spaces** To further demonstrate the effectiveness of our FIE method qualitatively, we present a visualization of the node embedding

space for both unsupervised and supervised training modes. Figure 5 shows the t-SNE visualization of the node embeddings for the Cora dataset using both GAT and FIE. All methods produce well-separated clusters for different classes. The clusters in the FIE embeddings obtained via unsupervised training are less distinct, as expected since the method has no access to class labels. Despite this, the resulting embedding is able to achieve classification performance comparable to or even better than the other methods.

## 6. Discussion

We have proposed the Fisher information embedding model, a novel class of attention-based node embeddings that can be learned flexibly in either supervised or unsupervised settings. By leveraging tools from information geometry, our method constructs a new attention mechanism, offering deeper insights into the geometric aspects of attention-based models. Although supervised node embedding methods are widely used in practice, our unsupervised node embeddings demonstrate comparable performance. Our work is related to a recent class of graph models, namely graph transformers (Ying et al., 2021; Mialon et al., 2020; Chen et al., 2022; Rampášek et al., 2022), which also use an attention-based operator (Vaswani et al., 2017), called self-attention, on the full set of nodes. However, rather than pooling the multiset, the self-attention returns another multiset of node features. Our work provides a first step towards understanding self-attention from an information geometry perspective.

A limitation of our approach is its reliance on Gaussian mixtures for parameter estimation using the EM algorithm. However, our framework has potential for incorporating a wider range of distribution families, which could lead to enhanced results. Exploring these extensions, alongside investigation into the generalization bounds of the FIE model, presents a promising direction for future research.

## Acknowledgements

# References

Abu-El-Haija, S., Perozzi, B., Al-Rfou, R., and Alemi, A. A. Watch your step: Learning node embeddings via graph attention. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

Amari, S.-i. and Nagaoka, H. *Methods of information geometry*, volume 191. American Mathematical Soc., 2000.

Bartlett, P. L., Foster, D. J., and Telgarsky, M. J. Spectrally-normalized margin bounds for neural networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Borgwardt, K., Ghisu, E., Llinares-López, F., O'Bray, L., and Rieck, B. Graph kernels: State-of-the-art and future challenges. *Foundations and Trends® in Machine Learning*, 13(5-6):531–712, 2020.

Boucheron, S., Bousquet, O., and Lugosi, G. Theory of classification: A survey of some recent advances. *ESAIM: probability and statistics*, 9:323–375, 2005.

Brody, S., Alon, U., and Yahav, E. How attentive are graph attention networks? In *International Conference on Learning Representations (ICLR)*, 2022.

Chen, D., Jacob, L., and Mairal, J. Convolutional kernel networks for graph-structured data. In *International Conference on Machine Learning (ICML)*, 2020.

Chen, D., O'Bray, L., and Borgwardt, K. Structure-aware transformer for graph representation learning. In *International Conference on Machine Learning (ICML)*, 2022.

Cheng, M. and Xu, H. Revisiting pooling through the lens of optimal transport. *arXiv preprint arXiv:2201.09191*, 2022.

Fan, W., Ma, Y., Li, Q., He, Y., Zhao, E., Tang, J., and Yin, D. Graph neural networks for social recommendation. In *The world wide web conference*, 2019.

Gaudelet, T., Day, B., Jamasb, A. R., Soman, J., Regep, C., Liu, G., Hayter, J. B., Vickers, R., Roberts, C., Tang, J., et al. Utilizing graph machine learning within drug discovery and development. *Briefings in bioinformatics*, 22(6):bbab159, 2021.

Goldberger, J., Gordon, S., Greenspan, H., et al. An efficient image similarity measure based on approximations of kl-divergence between two gaussian mixtures. In *Proceedings of the International Conference on Computer Vision (ICCV)*, volume 3, pp. 487–493, 2003.

Gourieroux, C. and Monfort, A. *Statistics and econometric models*, volume 1. Cambridge University Press, 1995.

Grover, A. and Leskovec, J. node2vec: Scalable feature learning for networks. In *Proceedings of the ACM International Conference on Knowledge Discovery and Data Mining (KDD)*, 2016.

Hamilton, W., Ying, Z., and Leskovec, J. Inductive representation learning on large graphs. *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Hu, W., Fey, M., Zitnik, M., Dong, Y., Ren, H., Liu, B., Catasta, M., and Leskovec, J. Open graph benchmark: Datasets for machine learning on graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Ingraham, J., Garg, V., Barzilay, R., and Jaakkola, T. Generative models for graph-based protein design. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., and Liu, T.-Y. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Kim, M. Differentiable expectation-maximization for set representation learning. In *International Conference on Learning Representations (ICLR)*, 2021.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

Kolouri, S., Naderializadeh, N., Rohde, G. K., and Hoffmann, H. Wasserstein embedding for graph learning. In *International Conference on Learning Representations (ICLR)*, 2020.

Kondor, R. and Vert, J.-P. Diffusion kernels. *kernel methods in computational biology*, pp. 171–192, 2004.

Kriege, N. M., Johansson, F. D., and Morris, C. A survey on graph kernels. *Applied Network Science*, 5(1):1–42, 2020.

Lücke, J. and Forster, D. k-means as a variational em approximation of gaussian mixture models. *Pattern Recognition Letters*, 125:349–356, 2019.

Mialon, G., Chen, D., d'Aspremont, A., and Mairal, J. A trainable optimal transport embedding for feature aggregation and its relationship to attention. In *International Conference on Learning Representations (ICLR)*, 2020.

Morris, C., Rattan, G., Kiefer, S., and Ravanbakhsh, S. Speqnets: Sparsity-aware permutation-equivariant graph networks. In *International Conference on Machine Learning (ICML)*, 2022.

Naderializadeh, N., Comer, J. F., Andrews, R., Hoffmann, H., and Kolouri, S. Pooling by sliced-wasserstein embedding. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Neyshabur, B., Bhojanapalli, S., and Srebro, N. A pac-bayesian approach to spectrally-normalized margin bounds for neural networks. In *International Conference on Learning Representations (ICLR)*, 2018.

Perozzi, B., Al-Rfou, R., and Skiena, S. Deepwalk: Online learning of social representations. In *Proceedings of the ACM International Conference on Knowledge Discovery and Data Mining (KDD)*, 2014.

Rampášek, L., Galkin, M., Dwivedi, V. P., Luu, A. T., Wolf, G., and Beaini, D. Recipe for a general, powerful, scalable graph transformer. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Rozemberczki, B., Allen, C., and Sarkar, R. Multi-scale attributed node embedding. *Journal of Complex Networks*, 2021.

Sánchez, J., Perronnin, F., Mensink, T., and Verbeek, J. Image classification with the fisher vector: Theory and practice. *International journal of computer vision*, 105 (3):222–245, 2013.

Sen, P., Namata, G., Bilgic, M., Getoor, L., Galligher, B., and Eliassi-Rad, T. Collective classification in network data. *AI magazine*, 29(3):93–93, 2008.

Shalev-Shwartz, S. and Ben-David, S. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

Shervashidze, N., Schweitzer, P., Van Leeuwen, E. J., Mehlhorn, K., and Borgwardt, K. M. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research (JMLR)*, 12(9), 2011.

Smola, A. J. and Kondor, R. Kernels and regularization on graphs. In *Learning Theory and Kernel Machines*, pp. 144–158. Springer, 2003.

Togninalli, M., Ghisu, E., Llinares-López, F., Rieck, B., and Borgwardt, K. Wasserstein weisfeiler-lehman graph kernels. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. Graph attention networks. In *International Conference on Learning Representations (ICLR)*, 2018.

Wang, C., Wu, Q., Weimer, M., and Zhu, E. Flaml: A fast and lightweight automl library. *Proceedings of Machine Learning and Systems*, 3:434–447, 2021.

Williams, C. and Seeger, M. Using the nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2000.

Xu, H., Luo, D., Zha, H., and Duke, L. C. Gromov-wasserstein learning for graph matching and node embedding. In *International Conference on Machine Learning (ICML)*. PMLR, 2019.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How powerful are graph neural networks? In *International Conference on Learning Representations (ICLR)*, 2018.

Ying, C., Cai, T., Luo, S., Zheng, S., Ke, G., He, D., Shen, Y., and Liu, T.-Y. Do transformers really perform badly for graph representation? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R. R., and Smola, A. J. Deep sets. *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Zhu, J., Lu, X., Heimann, M., and Koutra, D. Node proximity is all you need: Unified structural and positional node and graph embedding. In *Proceedings of the SIAM International Conference on Data Mining (SDM)*, 2021.

Zhu, X. and Ghahramani, Z. Learning from labeled and unlabeled data with label propagation. In *Technical Report*. Carnegie Mellon University, 2002.

# Appendix

This appendix provides both theoretical and experimental materials and is organized as follows: Section A provides a more detailed background on graph attention networks. Section B presents proofs for all the Lemmas and theorems. Section C provides variations for the E-step of FIE on the manifold of Gaussian mixtures. Section D provides experimental details and additional results.

## A. Background on graph attention networks

GATs (Veličković et al., 2018) are GNNs that have an attention mechanism to weight the contributions of each of the nodes when aggregating features from a neighborhood in the message passing framework. Indeed, the GAT model uses a score function $e(h_i, h_j) = \text{LeakyReLU}(a^\top[Wh_i\|Wh_j])$ to decide the importance of the features $h_j$ of a neighbor $j$ of node $i$. $a$ and $W$ are learnable parameters. The attention score for edge $(i, j)$ is then computed as $\alpha_{ij} = \text{softmax}_j(e(h_i, h_j))$.

Then, GAT aggregates the features from the neighborhood $\mathcal{N}_i$ of $i$ by computing the weighted sum

$$h'_i = \sigma\left(\sum_{j\in\mathcal{N}_i} \alpha_{ij}Wh_j\right)$$

Subsequently, (Brody et al., 2022) proposed a different attention mechanism, dubbed GATv2, that computes the attention scores as $e(i, j) = a^\top\text{LeakyReLU}(W[h_i\|h_j])$, which increases the representative power of the model.

## B. Proofs

### B.1. Proof of Lemma 3.1

*Proof.* We follow the arguments by Xu et al. (2018, Theorem 3). The WL test applies a predetermined injective hash function $g$ to update the WL node labels $a^{(t)}(v)$:

$$a_t(v) = g\left(\{(a_{t-1}(v), a_{t-1}(u)) : u \in \mathcal{N}(v)\}\right).$$

And our kernel embedding at iteration $t$ in the example of neighborhood multisets is given by

$$\psi_t(v) = \psi_{\text{multiset}}^{(t)}(\{(\psi_{t-1}(v), \psi_{t-1}(u)) : u \in \mathcal{N}(v)\}).$$

We show by induction that, for any iterations $t$, there always exists an injective function $\varphi$ such that $\psi_t(v) = \varphi(a_t(v))$. This apparently holds for $t = 0$ as $\psi_0 = a_0 = a$. Now let us assume that this condition holds for $t - 1$, we will show it also holds for $t$. Substituting $\psi_{t-1}(v)$ with $\varphi(a_{t-1}(v))$ gives us:

$$\psi_t(v) = \psi_{\text{multiset}}^{(t)}(\{(\varphi(a_{t-1}(v)), \varphi(a_{t-1}(u))) : u \in \mathcal{N}(v)\}).$$

Since the composition of injective functions is injective, we have $\psi = \psi_{\text{multiset}}^{(t)} \circ \varphi$ is injective, where by abuse of notation, $\varphi$ is applied element-wise to the multiset. Then, we have

$$\psi_t(v) = \psi \circ g^{-1}(a_t(v)),$$

such that $\psi \circ g^{-1}$ is injective, as being the composition of injective functions. Therefore, we conclude the lemma. □

### B.2. Proof of Lemma 3.2

This is a classical result, and its proof can be found in, *e.g.,* Boucheron et al. (2005).

### B.3. Proof of Theorem 4.1

*Proof.* The proof is adapted from Amari & Nagaoka (2000, Theorem 3.20). Let $(g, \nabla, \nabla^*)$ be a dualistic structure induced by the divergence $D$ as shown in (Amari & Nagaoka, 2000). We denote by $\exp_\mu$ and $\exp_\mu^*$ respectively the exponential maps for $\nabla$ and $\nabla^*$. Theorem 3.20 showed that

$$D(u\|\mu) + D(\mu\|v) - D(u\|v) = \langle \exp_\mu^{-1}(u), \exp_\mu^{*-1}(v)\rangle_g + o(\Delta^3). \tag{19}$$

We also have $\|R_\mu^{-1}(u) - \exp_\mu^{-1}(u)\|_g = O(\Delta^2)$ and $\|R_\mu^{-1}(v) - \exp_\mu^{*-1}(v)\|_g = O(\Delta^2)$ since $R_\mu^{-1}(u) = u - \mu$. Finally, we have

$$\begin{aligned}
&\left|\langle R_\mu^{-1}(u), R_\mu^{-1}(v)\rangle_g - \langle \exp_\mu^{-1}(u), \exp_\mu^{*-1}(v)\rangle_g\right| \\
=&\left|\langle R_\mu^{-1}(u) - \exp_\mu^{-1}(u), R_\mu^{-1}(v)\rangle_g - \langle \exp_\mu^{-1}(u), \exp_\mu^{*-1}(v) - R_\mu^{-1}(v)\rangle_g\right| \\
\leq&\left|\langle R_\mu^{-1}(u) - \exp_\mu^{-1}(u), R_\mu^{-1}(v)\rangle_g\right| + \left|\langle \exp_\mu^{-1}(u), \exp_\mu^{*-1}(v) - R_\mu^{-1}(v)\rangle_g\right| \\
\leq&\|R_\mu^{-1}(v)\|_g\|R_\mu^{-1}(u) - \exp_\mu^{-1}(u)\|_g + \|\exp_\mu^{-1}(u)\|_g\|R_\mu^{-1}(v) - \exp_\mu^{*-1}(v)\|_g = O(\Delta^2),
\end{aligned}$$

where the first inequality uses triangle inequality and the second inequality uses the Cauchy-Schwarz inequality. □

### B.4. Proof of Lemma 4.2

Fisher information is the second derivative of KL divergence, which is a commonly known result in information theory. A proof can be found in *e.g.,* Gourieroux & Monfort (1995, Page 87).

### B.5. Proof of Theorem 4.3

*Proof.* Following the definition in Eq. (10), we have

$$\begin{aligned}
\frac{\|\varphi_{\theta_0}(p_\theta) - \varphi_{\theta_0}(p_{\theta'})\|^2}{2} &= \frac{\left\|I(\theta_0)^{1/2}(\theta - \theta')\right\|^2}{2} \\
&= \frac{1}{2}(\theta - \theta')^\top I(\theta_0)(\theta - \theta').
\end{aligned}$$

When both $\theta$ and $\theta'$ approach $\theta_0$, Lemma 4.2 suggests that

$$\mathrm{KL}(p_\theta\|p_{\theta_0}) = \frac{1}{2}(\theta - \theta_0)^\top I(\theta)(\theta - \theta_0) + o\|\theta - \theta_0\|^2,$$

and

$$\mathrm{KL}(p_{\theta_0}\|p_{\theta'}) = \frac{1}{2}(\theta' - \theta_0)^\top I(\theta_0)(\theta' - \theta_0) + o\|\theta' - \theta_0\|^2.$$

We also have

$$\left|\frac{1}{2}(\theta - \theta_0)^\top I(\theta)(\theta - \theta_0) - \frac{1}{2}(\theta - \theta_0)^\top I(\theta_0)(\theta - \theta_0)\right| = \frac{1}{2}\|I(\theta) - I(\theta_0)\|_2\|\theta - \theta_0\|_2^2,$$

thus

$$\mathrm{KL}(p_\theta\|p_{\theta_0}) = \frac{1}{2}(\theta - \theta_0)^\top I(\theta_0)(\theta - \theta_0) + O\|\theta - \theta_0\|^2,$$

By substituting $\mu$ by $p_{\theta_0}$, $u$ by $p_\theta$ and $v$ by $\theta'$ in Theorem 4.1, we have

$$\mathrm{KL}(p_\theta\|p_{\theta_0}) + \mathrm{KL}(p_{\theta_0}\|p_{\theta'}) - \mathrm{KL}(p_\theta\|p_{\theta'}) = (\theta - \theta_0)^\top I(\theta_0)(\theta' - \theta_0) + O(\Delta^2),$$

where $\Delta = \max\{\|\theta - \theta_0\|, \|\theta' - \theta_0\|\}$. Finally, by substituting the KL divergence terms, we obtain the expected expansion.

□

### B.6. Proof of Theorem 4.4 and Theorem 4.5

The proof of two theorems are straightforward from the definition of $\varphi_{\theta_0}$ in Eq. (10).

**B.7. Relationship between $\theta_{\mathrm{ML}}$ and $\theta_{\mathrm{ML}_{\theta_0}}$**

Here, we show that the estimator $\theta_{\mathrm{ML}_{\theta_0}}$ is a good proxy of the ML estimator.

**Lemma B.1.** *Let*

$$\theta_{\mathrm{ML}_{\theta_0}}(\mathbf{x}) := \arg\max_{\theta \in \mathcal{M}} \mathbb{E}_{z|x,\theta_0}[\log p_\theta(x,z)], \tag{20}$$

*where $p_{\mathbf{x}}$ denotes the true density of $\mathbf{x}$. This is a good proxy for the maximum likelihood estimation of $\theta$.*

*Proof.* We have that

$$\arg\max_{\theta \in \mathcal{M}} \mathbb{E}_{z|x,\theta_0}[\log p_\theta(x,z)] = \arg\max_{\theta \in \mathcal{M}} \mathbb{E}_{z|x,\theta_0}[\log p_\theta(x,z) - \log p_{\theta_0}(z|x)], \tag{21}$$

as the term $\log p_{\theta_0}(z|x)$ does not depend on $\theta$. Thanks to Jensen's inequality we obtain that

$$\mathbb{E}_{z|x,\theta_0}\left[\log \frac{p_\theta(x,z)}{p_{\theta_0}(z|x)}\right] \leq \log\left(\mathbb{E}_{z|x,\theta_0}\left[\frac{p_\theta(x,z)}{p_{\theta_0}(z|x)}\right]\right) = \log p_\theta(x). \tag{22}$$

Moreover, we have that the difference is bounded by

$$\begin{aligned}
\log p_\theta(x) - \mathbb{E}_{z|x,\theta_0}\left[\log \frac{p_\theta(x,z)}{p_{\theta_0}(z|x)}\right] &= \int \left(\log p_\theta(x) - \log \frac{p_\theta(x,z)}{p_{\theta_0}(z|x)}\right) p_{\theta_0}(z|x)\,dz \\
&= \int \left(\log \frac{p_{\theta_0}(z|x)}{p_\theta(z|x)}\right) p_{\theta_0}(z|x)\,dz \\
&= KL\left(p_{\theta_0}(z|x) \,\|\, p_\theta(z|x)\right)
\end{aligned} \tag{23}$$

$\square$

**B.8. Particular case for the manifold of a single Gaussian**

**Lemma B.2.** *Consider as the family of distributions the family of Gaussians $\mathcal{N}(\mu, I)$, parametrized by $\theta = \mu$. Then, we have $D(p_\theta \| p_{\theta'}) = \frac{\|\psi(\theta) - \psi(\theta')\|^2}{2}$.*

*Proof.* We have $D(p_\theta \| p'_\theta) = \frac{\|\mu - \mu'\|^2}{2}$. Moreover, we have that the Fisher information matrix is the identity, i.e. $I(\theta) = I$. Then $\psi(\theta) = \mu$, and $\|\psi(\theta) - \psi(\theta')\|^2 = \|\mu - \mu'\|^2$, so $D(p_\theta \| p'_\theta) = \|\psi(\theta) - \psi(\theta')\|^2$. $\square$

# C. Variations on the E step for Fisher information embedding

In particular, the E-step can be solved in several ways, both analytically and not, depending on the problem at hand:

- Without further constraints on $\boldsymbol{\alpha}$, one can show that the optimal $a_{ij}$'s are

$$\alpha_{ij} = p_\theta(z_i = j|x_i) = \frac{\mathcal{N}(x_i, w_j, \Sigma_j)}{\sum_{l=1}^p \mathcal{N}(x_i, w_l, \Sigma_l)}.$$

- If one adds an hard constraint of the form $\sum_i a_{ij} = n/p$, e.g. to avoid collapsed solutions, then the minimization problem is an entropy-regularized optimal transport problem, which can be solved efficiently by the Sinkhorn–Knopp algorithm. Indeed, the problem can be formulated as

$$\min_{\boldsymbol{\alpha} \in \Pi'} \sum_{i=1}^n \sum_{j=1}^p \alpha_{ij} C_{ij} - H(\boldsymbol{\alpha}),$$

with $C_{ij} = w_i \log p_\theta(x_i, z_i = j)$ and $\Pi' := \{\alpha_{ij} \geq 0, \sum_j \alpha_{ij} = 1, \sum_i a_{ij} = n/p\}$.
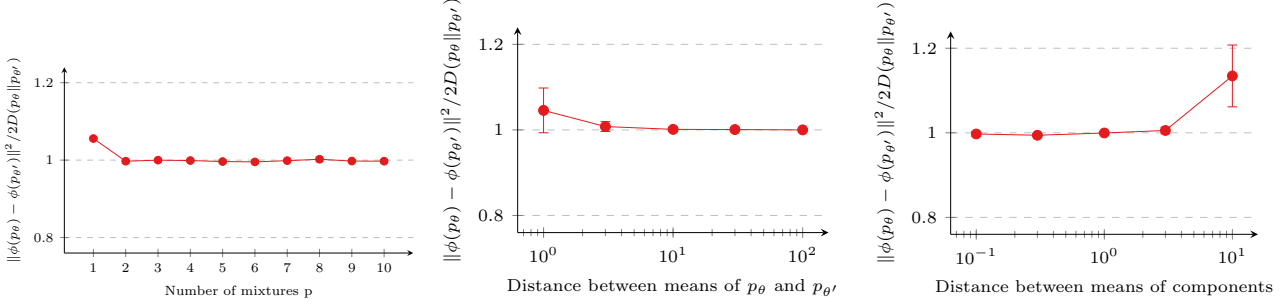
*Figure 6.* Simulation study on the ratio between the KL divergence between two distributions and the squared distance between their embeddings, varying some parameters of the underlying distributions.

- In many situations, adding a hard constraint on the rows of $\boldsymbol{\alpha}$ is too strong of a requirement. Instead, one can add a regularization term in the objective function to skew the optimal solution towards the desired marginal. The E-step can be then formulated as an unbalanced optimal transport (UOT) problem () as follows:

$$\min_{\boldsymbol{\alpha}\in\mathbb{R}_+^{n\times p}}\sum_{i=1}^{n}\sum_{j=1}^{p}\alpha_{ij}C_{ij}+H(\boldsymbol{\alpha})+\tau_1\mathrm{KL}(\boldsymbol{\alpha}\mathbf{1}_n\|\mathbf{1}_p)+\tau_2\mathrm{KL}(\boldsymbol{\alpha}^\top\mathbf{1}_p\|n/p\mathbf{1}_n),$$

with $\tau_1\to\infty$ to enforce that $\sum_j a_{ij}=1$. Similarly to OT, this problem can be solved efficiently with a variant of the Sinkhorn algorithm.

## D. Experimental details and additional results

Here, we provide experimental details and additional experimental results.

### D.1. Evaluation on synthetic datasets

In the first set of experiments, we keep the parameters of the embedding method fixed and we vary the underlying distributions, to understand in which cases the fisher embedding approximates well the KL divergence. In particular, we use as the family of distributions a family of Gaussian mixtures with $k=10$ components, and with each component having identity covariance. Moreover, we fix the number of EM iterations to $T=10$.

We then vary the underlying data-generating distributions as follows.

- In the first simulation we have $p_1$ as a mixture of three Gaussians $\mathcal{N}(\mu_{1,j},I)$, with $\mu_{1,1}=(-5,-2)$, $\mu_{1,1}=(-5,0)$, $\mu_{1,1}=(-5,2)$. Then we vary $p_2$ as a mixture of $\kappa$ Gaussians $\mathcal{N}(\mu_{2,j},I)$, with means evenly spaced between $(5,-2)$ and $(5,2)$, for $\kappa\in[1,10]$.

- In the second simulation, we have that both the distributions are mixtures of three Gaussians $\mathcal{N}(\mu_{i,j},I)$, and we vary the distance between the means of the two distributions, keeping the distance between components of the same distribution fixed. In particular, we let $\mu_{1,1}=(-5,-2)$, $\mu_{1,1}=(-5,0)$, $\mu_{1,1}=(-5,2)$ and $\mu_{2,1}=(-5+d,-2)$, $\mu_{2,1}=(-5+d,0)$, $\mu_{2,1}=(-5+d,2)$, for $d\in[1,100]$.

- In the third simulation, we again have that both the distributions are mixtures of three Gaussians $\mathcal{N}(\mu_{i,j},I)$, but this time we vary the distance between the components of each distribution, keeping the distance between the means of the two distributions fixed. In paricular, we let $\mu_{1,1}=(-5,-d)$, $\mu_{1,1}=(-5,0)$, $\mu_{1,1}=(-5,d)$ and $\mu_{2,1}=(5,-d)$, $\mu_{2,1}=(5,0)$, $\mu_{2,1}=(5,d)$, for $d\in[0.1,10]$.

In the second set of experiments we keep the underlying distributions fixed and the vary the parameters of our embedding method. In particular, we let both distributions be a mixture of three Gaussians $\mathcal{N}(\mu_{1,j},I)$ with $\mu_{1,1}=(-5,-2)$, $\mu_{1,1}=(-5,0)$, $\mu_{1,1}=(-5,2)$ and $\mu_{2,1}=(5,-2)$, $\mu_{2,1}=(5,0)$, $\mu_{2,1}=(5,2)$.

We then vary the number $k$ of components in the parametric family of Gaussian mixtures for the maximum likelihood estimation in $[1,100]$, and the number $T$ of EM iterations in $[0,10]$.

## D.2. Evaluation on real-world datasets

### D.2.1. COMPUTATION DETAILS

All experiments were performed on a shared GPU and CPU cluster equipped with GTX1080 and TITAN RTX. About 20 of these GPUs were used simultaneously, and the total computational cost of this research project was about 500 GPU hours.

### D.2.2. DATASETS

The datasets that we use for classification are classical ones. The statistics for each dataset is summarized in Table 2

| Dataset | Number of nodes | Number of edges | Number of classes |
|---|---|---|---|
| Cora | 2708 | 5429 | 7 |
| Citeseer | 3327 | 4732 | 6 |
| Pubmed | 19717 | 44338 | 3 |
| Reddit | 232965 | 114615892 | 41 |
| ogbn-arxiv | 169343 | 1166243 | 40 |
| ogbn-products | 2449029 | 61859140 | 47 |

*Table 2.* Summary of considered datasets.

### D.2.3. BASELINE RESULTS

We report the results from the following papers. For MLP, for the Cora, Citeseer and Pubmed datasets we report the results from (Veličković et al., 2018), for the ogbn-arxiv and ogbn-products datasets we report the results from (Hu et al., 2020). For Label Propagation, for the Cora, Citeseer and Pubmed datasets we report the results from (Veličković et al., 2018), for the ogbn-arxiv and ogbn-products datasets we report the results from (Hu et al., 2020). For CGN, for the Cora, Citeseer and Pubmed datasets we report the results from (Veličković et al., 2018) and for the ogbn-arxiv and ogbn-products datasets we report the results from (Brody et al., 2022). For GraphSAGE, for the Reddit dataset we report the results from (Hamilton et al., 2017) and for the ogbn-arxiv and ogbn-products datasets we report the results from (Brody et al., 2022). For GAT and GATv2, for the ogbn-arxiv and ogbn-products datasets we report the results from (Brody et al., 2022) and for other datasets we report the results from our reimplementation.

### D.2.4. HYPERPARAMETER CHOICES AND REPRODUCIBILITY

**Hyperparameter choice.** In general, we perform a very limited hyperparameter search to produce the results in Table 1. The hyperparameters for training FIE models on different datasets are summarized in Table 3 and Table 4, respectively for unsupervised and supervised modes of FIE. For supervised learning tasks, a dropout with rate equal to 0.5 is used for training supervised embeddings of FIE. For large supervised datasets (Reddit, ogbn-arxiv, and ogbn-products), we use a LightGBM classifier (Ke et al., 2017), the hyperparameters of which are automatically tuned by FLAML (Wang et al., 2021).

**Optimization.** All our models are trained with the Adam optimizer (Kingma & Ba, 2015) with a fixed learning rate equal to 0.001.

| Hyperparameter | {Cora, Citeseer, Pubmed} | {Reddit, ogbn-arxiv, ogbn-products} |
|---|---|---|
| Number of layers | [2,3,4] | [3, 4, 5] |
| Hidden dimensions | [128, 256, 512] | [128, 256, 512] |
| Number of mixture components | [1, 2, 4, 8] | [1, 2, 4, 8] |

*Table 3.* Hyperparameters for unsupervised mode of FIE.

| Hyperparameter | {Cora, Citeseer, Pubmed} | {Reddit, ogbn-arxiv, ogbn-products} |
|---|---|---|
| Number of layers | [2,3,4] | [3, 4, 5] |
| Hidden dimensions | [16, 32, 64] | [128, 256] |
| Number of mixture components | [1, 2, 4, 8] | [1, 2, 4, 8] |

*Table 4.* Hyperparameters for supervised mode of FIE.