To estimate the location-scale parameters of a bell-shaped density on attributed graphs, we consider radial densities as approximations. The problem of estimating the parameters of radial densities on graphs is equivalent to the problem of estimating the parameters of truncated Gaussians in a Euclidean space. Based on this result, we adopt the maximum likelihood method for truncated Gaussians. From the estimated probabilities we inferred the conditional probabilities for a Bayes classifier. Experiments on random graphs and four benchmark data sets of the IAM graph database repository and on random weighted graphs are presented and discussed.

1. Introduction

In statistical pattern recognition, generative models are used for modeling probability densities of the underlying patterns. For vectorial patterns, the Gaussian distribution is often used as a first approximation for vector-valued random variables that cluster around a single mean. In addition, it forms the basic building block for Gaussian mixture models. Gaussian mixture models in conjunction with the maximum likelihood method and the EM algorithm form a popular technique for smooth approximation of arbitrarily shaped densities defined on feature vectors.

For patterns represented by graphs, however, theoretically well-established reference models such as the Gaussian distribution and mixtures of Gaussians are still missing, perhaps due to the lack of analytical structure in the graph domain. Current techniques of probabilistic models on graphs aim at inferring statistical variations of structural models from their individual primitives, i.e. from vertices, edges, and/or their attributes [2, 8, 11, 12, 26, 27, 29, 30, 34]. In [33], the primitives from which statistical variation are
inferred are part-based representations. Often, the proposed probabilistic models in the
literature assume independency of their primitives in one way or another. Such assump-
tions are appropriate for the sake of analytical convenience and more or less justified by
empirical findings. Nevertheless, this assumptions are contrary to the nature of graph-
based representations that aim at modeling dependencies between entities.

In this contribution, we suggest an alternative approach following the opposite route.
Instead of inferring the probability model from primitives or parts, we regard attributed
graphs as atomic events of some probability space. In doing so, we need not to impose
independency assumptions on the vertices, edges, and their attributes. We assume that the
graphs obey a bell-shaped distribution. First, we approximate the bell-shaped distribution
by substituting it by a radial density on graphs having two parameters to be estimated: a
center graph at the location of the peak of the bell and a parameter that controls the width
of the bell. We show that estimating the parameters of the radial density on graphs is
equivalent to estimating the parameters of a truncated Gaussian in some Euclidean space.

For truncated Gaussians, we can adopt the maximum likelihood method to estimate the
parameters. We used the estimated probability densities to infer conditional probability
densities with Bayes rule for classifying graph patterns. The simplicity of the proposed
approach and its lower complexity compared to state-of-the-art classifiers sets the stage for
extensions to parameter estimation of mixture models of bell-shaped densities on graphs.

This paper is structured as follows: Section 2 provides a problem statement and presents
the basic idea of the proposed approach. In Section 3, we describe the orbifold framework.
Section 4 presents radial functions on graphs as extensions of Gaussian distributions on
vectors. The maximum likelihood method for estimating the parameters of radial densities
on graphs is proposed in Section 5. We present and discuss experiments in Section 6. The
conclusion in Section 6 summarizes the main findings and presents an outlook for further
research. Proofs have been delegated to the appendix.

2. Problem Statement

This section formalizes the problem of estimating bell-shaped distributions on structured
data.

2.1. Estimating the parameters of bell-shaped distributions

Suppose that \((X_G, d)\) is a metric space of attributed graphs. A bell-shaped distribution on
\(X_G\) with center graph \(C \in X_G\) and width \(\sigma > 0\) is a density function of the form

\[
p : X_G \rightarrow \mathbb{R}, \quad X \mapsto p(X|C, \sigma) = h \left( \frac{d(X, C)}{\sigma} \right),
\]

where \(h : \mathbb{R} \rightarrow \mathbb{R}\) is a monotonously decreasing function such that

\[
C = \arg \max_{X \in X_G} p(X|C, \sigma).
\]
Thus, the center graph $C$ is the location of the peak of the bell and $\sigma$ controls the width of the bell. Note that in general neither $C$ is the expectation nor $\sigma$ is the variance of $p$.

The problem we want to solve is as follows: estimate the unknown parameters $C$ and $\sigma$ of a bell-shaped distribution $p$, given a training set of $N$ data points $X_1, \ldots, X_N \in \mathcal{X}_G$ generated by $p$.

2.2. Approach

To solve the problem of estimating the center $C$ and width $\sigma$ of a bell-shaped distribution $p$, we assume that $p$ is locally well approximated by a radial function of the form

$$h_a : \mathcal{X}_G \to \mathbb{R}, \quad X \mapsto a \cdot \exp \left( -\frac{d(X, C)^2}{2\sigma^2} \right).$$

The parameter $a$ defined by

$$a = \left( \int_{\mathcal{X}_G} h_a(X|C, \sigma)\lambda_G(dX) \right)^{-1}$$

scales the radial function to a density. This approximation reduces the problem of estimating the parameters of a bell-shaped functions to the one of estimating the parameters of a radial density.

We estimate the parameters of a radial density by adopting the maximum likelihood method. In doing so, we face the following difficulties:

1. We can exactly evaluate the improper integral of standard radial functions defined on $\mathbb{R}^D$ giving

$$a = \frac{1}{(2\pi\sigma^2)^{D/2}}.$$

In contrast, a closed-form solution for improper integrals of radial functions on graphs is unknown, in general. Knowing $a$, however, is essential for Bayesian inference.

2. The maximum likelihood method for standard Gaussian distributions exploits vector space addition and local gradient information. Both concepts are unknown in the graph domain.

To overcome these difficulties, an appropriate approach to represent graphs is crucial. The approach we suggest models the graph domain as quotient spaces of some Euclidean space by a finite group action. We call such quotient spaces graph orbifolds. Using the orbifold framework, we can adopt the analytical and geometrical concepts of the underlying Euclidean space for estimating the parameters of radial densities using the maximum likelihood method.

3. Graph Orbifolds

In this section, we introduce attributed graphs and represent them as point of some quotient space, called orbifold. Most of this presentation including proofs of statements and claims is based on the structure space formalism proposed by [13].
3.1. Representation of Graphs

Let $E$ be a Euclidean space. An attributed graph $X = (V, E, \alpha)$ consists of a set $V$ of vertices, a set $E \subseteq V \times V$ of edges, and an attribute function $\alpha : V \times V \to E$, such that $\alpha(i, j) \neq 0$ for each edge and $\alpha(i, j) = 0$ for each non-edge. Attributes $\alpha(i, i)$ of vertices $i$ may take any value from $E$.

For simplifying the mathematical treatment, we assume that all graphs are of order $n$, where $n$ is chosen to be sufficiently large. Graphs of order less than $n$, say $m < n$, can be extended to order $n$ by including isolated vertices with attribute zero. For practical issues, it is important to note that limiting the maximum order to some arbitrarily large number $n$ and extending smaller graphs to graphs of order $n$ are purely technical assumptions to simplify mathematics. For pattern recognition problems, these limitations should have no practical impact, because neither the bound $n$ needs to be specified explicitly nor an extension of all graphs to an identical order needs to be performed. When applying the theory, all we actually require is that the graphs are finite.

A graph $X$ is completely specified by its matrix representation $X = (x_{ij})$ with elements $x_{ij} = \alpha(i, j)$ for all $1 \leq i, j \leq n$. Let $\mathcal{X} = \mathbb{E}^{n \times n}$ be the Euclidean space of all $(n \times n)$-matrices with elements from $E$ and let $\Pi^n$ be the set of all $(n \times n)$-permutation matrices. For each $P \in \Pi^n$ we define a mapping

$$\gamma_P : \mathcal{X} \to \mathcal{X}, \quad X \mapsto P^T X P.$$ 

Then $G = \{ \gamma_P : P \in \Pi^n \}$ is a finite group acting on $\mathcal{X}$. For $X \in \mathcal{X}$, the orbit of $X$ is the set defined by $[X] = \{ \gamma(X) : \gamma \in G \}$. The quotient set

$$\mathcal{X}_G = \mathcal{X}_\mathcal{G} = \{ [X] : X \in \mathcal{X} \}$$

consisting of all orbits is a graph orbifold. The natural projection

$$\pi : \mathcal{X} \to \mathcal{X}_G, \quad X \mapsto [X]$$

maps each matrix representation $X$ to its orbit $[X]$.

Suppose that $X$ is a matrix representation of some attributed graph $X$. Then the orbit $[X]$ consists of all possible matrices that represent $X$. By identifying the attributed graphs $X$ with the orbits $[X]$, we can regard graphs as point in the graph orbifold $\mathcal{X}_G$. The map $\pi : \mathcal{X} \to \mathcal{X}_G$ projects matrices $X$ to the graphs $X$ they represent.

For notational convenience, we identify $\mathcal{X}$ with $\mathbb{E}^N$, where $N = n^2$ and consider vector-rather than matrix representations of graphs. We obtain a vector representation $x$ of graph $X$ by concatenating the columns of a matrix $X$ representing $X$. We write $x \in X$ if $x \in \mathcal{X}$ projects to $X \in \mathcal{X}_G$ via the projection $\pi(x) = X$.

3.2. Geometrical Graph Metric

In principle, any distance function on $\mathcal{X}_G$ can be used to define a radial function. In order to preserve the analytical and statistical properties of the standard radial functions as
far as possible, we restrict to distance functions on orbifolds that extend the Euclidean metric. For this, we define our graph metric by

\[ d(X, Y) = \min \{ \|x - y\| : x \in X, y \in Y \}, \tag{1} \]

where \(\|\cdot\|\) is the Euclidean metric on \(X\). As shown in [13], \(d(\cdot, \cdot)\) is indeed a metric. We call a pair \((x, y) \in X \times Y\) with \(\|x - y\| = d(X, Y)\) an optimal alignment of \(X\) and \(Y\). By \(A(X, Y)\) we denote the set of all optimal alignments of \(X\) and \(Y\).

As shown in [13], the distance \(d(X, Y)\) can be equivalently expressed by

\[ d(X, Y) = \min \{ \|x - y\| : y \in Y \}, \tag{2} \]

where \(x \in X\) is a fixed but arbitrarily chosen representation of \(X\). By symmetry of \(d(\cdot, \cdot)\), the roles of \(X\) and \(Y\) in eqn. (2) can be interchanged.

Note that the above defined graph metric is not a artificial construction for analytical purposes but rather is based on a generalized concept of maximum common subgraph and therefore appears in different guises as a common choice of proximity measure for graphs [5, 10, 31].

4. Radial functions

From now on we assume that \(X_G\) is a graph orbifold obtained and \(d(\cdot, \cdot)\) is the graph metric of \(X_G\) induced by the Euclidean metric of \(X\).

4.1. Radial functions

An orbifold function is a function of the form

\[ f : X_G \to \mathbb{R}, \quad X \mapsto f(X). \]

We consider the following examples of orbifold functions:

- Radial functions
  \[ h(X|C, \sigma) = \exp \left( -\frac{d(X, C)^2}{2\sigma^2} \right) \]
  with center \(C\) and width \(\sigma\).

- Weighted radial functions
  \[ h_w(X|C, \sigma) = w \cdot h(X|C, \sigma) \]
  with weight \(w > 0\) that controls the height of the bell.

- Radial densities
  \[ p(X|C, \sigma) = h_w(X|C, \sigma) = a \cdot h(X|C, \sigma), \]
  with weight \(a > 0\) that scales the radial function \(h(X|C, \sigma)\) to a density.
Observe that radial functions and radial densities are special cases of weighted radial functions. For the sake of convenience we occasionally refer to all types of weighted and unweighted radial functions briefly as radial function.

Since the weight of a radial density depends on the center and the width, studying radial densities is more laborious than studying generic weighted radial functions, where the weight is an independent parameter. For this reason, we first study weighted radial functions with constant weight and then turn to radial densities.

### 4.2. Lifting weighted radial functions

To study orbifold functions, we lift them to $G$-invariant functions on $\mathcal{X}$. A lift of an orbifold function $f : \mathcal{X}_G \to \mathbb{R}$ is a function $f^* : \mathcal{X} \to \mathbb{R}$ satisfying

$$f^*(x) = f(\pi(x)) = f(x).$$

By definition, the lift $f^*$ is invariant under the action of $G$, that is $f^*(x) = f^*(\gamma x)$ for all $\gamma \in G$.

The lift $h_w^*$ of a radial function $h_w$ is of the form

$$h_w^*(x|C,\sigma) = w \cdot \exp\left(-\frac{\min_{c \in C} \|x - c\|^2}{2\sigma^2}\right)$$

$$= \max_{c \in C} w \cdot \exp\left(-\frac{\|x - c\|^2}{2\sigma^2}\right)$$

$$= \max_{c \in C} w \cdot h(x|c,\sigma) = \max_{c \in C} h_w(x|c,\sigma)$$

where

$$h(x|c,\sigma) = \exp\left(-\frac{\|x - c\|^2}{2\sigma^2}\right)$$

is a standard radial function with center $c \in \mathcal{X}$ and width $\sigma > 0$ and $h_w(x|c,\sigma)$ is the weighted standard radial function defined on $\mathcal{X}$. Note that neither $h$ nor $h_w$ are lifts of $h$ and $h_w$.

1. Again, we occasionally refer to all types of weighted and unweighted standard radial functions briefly as standard radial functions.

The first key observation to be made is that from

$$h_w^*(x|C,\sigma) = \max_{c \in C} h_w(x|c,\sigma)$$

follows that studying radial functions on structures is equivalent to studying point-wise maximizers of standard radial functions on vectors with the same weight $w$ and width $\sigma$ but different centers. The centers of the standard radial functions are the different representations that project to the same center of the radial function on structures.

1. To see this, observe that $(x, c)$ needs not to be an optimal alignment of $X$ and $C$. In this case, we have $d(X, C) < \|x - y\|$. From this follows that $h(x|c,\sigma) \neq h(X|C,\sigma)$ and $h_w(x|c,\sigma,a) \neq h_w(X|C,\sigma,a)$. Thus the assumption follows by definition of a lift of an orbifold function.
4.3. Restricting lifts to Dirichlet domains

Our goal is to cast the analysis of radial functions $h_w(x|C,\sigma)$ on $X_G$ to the analysis of standard radial functions $h_w(x|c,\sigma)$ restricted to the Dirichlet domain

$$D_c = \{ x \in X : \|x - c\| \leq \|x - c'\|, \ c' \in C \} \subseteq X$$

of $c$. A standard radial function $h_w(x|c,\sigma)$ restricted to its Dirichlet domain $D_c$ is a truncated radial function of the form

$$h^t_w(x|c,\sigma) = \begin{cases} h_w(x|c,\sigma) & : x \in D_c \\ 0 & : x \notin D_c \end{cases}.$$ 

Thus, our goal is to cast the analysis of radial functions on structures to the analysis of truncated radial functions on vectors. This is only possible for centers $C \in X_G$ that are regular in the following sense: For $x \in X$, we define the isotropy group of $x$ as the set of all elements of $G$ that fix $x$, that is

$$G_x = \{ \gamma \in G : \gamma x = x \}.$$ 

The isotropy group of $x$ is trivial if $G_x = \{\text{id}\}$. Obviously, if the isotropy group of $x$ is trivial, then the isotropy group of $\gamma x$ is trivial for all $\gamma \in G$. We say a structure $X \in X_G$ is regular if the isotropy group of any of its vector representations is trivial.

Assuming that a center $C$ is regular is not a restriction for the following reason: Suppose that $C$ of a radial function $h_w$ is not regular. Then the support of the lift $h_w$ of $h$ is nonzero in some subspace, where $C$ is regular. Therefore, we can assume that the center of an extended Gaussian is regular without loss of generality.

For each representation $c$ that projects to a regular center $C$, we have

1. $\pi(D_c) = X_G$
2. $\text{int}(D_c) \cap \text{int}(D_{\gamma c}) = \emptyset$ for all $\gamma \in G \setminus \{\text{id}\}$,

where $\text{int}(S)$ denotes the interior of a set $S$. The first property states that each structure has at least one representation in the Dirichlet domain of $c$. From the second property we can conclude that representations from the interior of a Dirichlet domain are in one-to-one correspondence with structures from some subset of $X_G$. Combining both properties yields that almost all structures have exactly one representation in the Dirichlet domain of $c$.

Thus, our second key observation is that we can cast the study of radial functions with regular center $C$ to the study of truncated radial functions defined on the Dirichlet domain centered around an arbitrary representation $c$ of $C$.

4.4. Densities

So far we have considered weighted radial functions, where the weight is a parameter independent of the center and width. In what follows, we consider radial densities, where the weight $a$ depends on the parameters of the underlying radial function. For this, we
assume that \((X, \|\cdot\|)\) is the Euclidean space and \((X, \mathcal{B}, \lambda)\) is the Lebesgue-Borel measure space. The action of the finite group \(G\) on \(X\) induces the measure space \((X_G, \mathcal{B}_G, \lambda_G)\), where

\[
\mathcal{B}_G = \{ \mathcal{B} \subset X_G : \pi^{-1}(\mathcal{B}) \in \mathcal{B} \}
\]

and \(\lambda_G\) is the induced quotient measure. By setting

\[
a = \left( \int_{X_G} h(X|C, \sigma) \lambda_G(dX) \right)^{-1}, \tag{4}
\]

the radial function \(h(X|C, \sigma)\) scales to density, written as \(p(X|C, \sigma)\).

As shown in the previous section, the weight \(a\) of a weighted radial function \(h_w\) remains identical for its lift \(h^*\) as well as for the associated truncated radial function \(h^t\). It remains to show that the weight \(a\) defined in eqn. (4) scales the lift \(h^*\) of a radial function \(h^*\) and a truncated radial function \(h^t\) on a Dirichlet domain to a density. For this, we apply the following disintegration formula [7]: For any \(\lambda\)-integrable function \(f\) on \(X\) we have

\[
\int_X f(x) \lambda(dx) = \int_{X_G} f(X) \lambda_G(dX), \tag{5}
\]

where \(f\) is the unique function defined by

\[
f(\pi(x)) = \frac{1}{|G|} \sum_{\gamma \in G} f(\gamma x). \tag{6}
\]

Equation (6) holds for any \(G\)-invariant function and therefore for lifts of orbifold functions. Applying the disintegration formula (5) to the lift

\[
h^*(x|C, \sigma) = \max_{c \in \mathcal{C}} h(x|c, \sigma) = \max_{\gamma \in \mathcal{G}} h(\gamma x|c, \sigma)
\]

of the radial function \(h(X|C, \sigma)\) yields

\[
\int_X h^*(x|C, \sigma) \lambda(dx) = \int_{X_G} h(X|C, \sigma) \lambda_G(dX).
\]

The last equation shows that the weight \(a\) defined in eqn. (4) indeed scales the lift \(h^*\) to a density.

Next we want to show that weight \(a\) in eqn. (4) also scales the truncated radial function \(h^t\) to a density. For this, we assume that \(c\) projects to a regular center \(C\). We have \(\lambda(D_c) = \lambda(D_{\gamma c})\) for all \(\gamma \in \mathcal{G}\). From this follows

\[
\int_X h^*(x|C, \sigma) \lambda(dx) = \sum_{\gamma \in \mathcal{G}} \int_{D_c} h(x|\gamma c, \sigma) \lambda(dx)
\]

\[
= |G| \int_{D_c} h(x|c, \sigma) \lambda(dx)
\]

\[
= \int_{D_c} h(x|c, \sigma) \lambda'(dx)
\]
where $\mathcal{X}'$ is the measure defined by some cross section $\psi : \mathcal{X} \to D_c$ of $\pi$. A cross section of $\pi$ is a map $\psi : \mathcal{X}' \to \mathcal{X}$ such that $\pi(\psi(X)) = X$ for all $X \in \mathcal{X}'$. Note that the integral is well-defined, because two cross sections into the same Dirichlet domain are equal almost everywhere. This shows that the weight $a$ defined in eqn. (4) also scales the truncated Gaussian to a density on its Dirichlet domain.

Putting all pieces together, we observe that the study of radial densities $p$ with regular center $C$ can be casted to the study of standard radial densities $p^*$ on vectors and their associated truncated radial densities $p_t$ on Dirichlet domains centered around some arbitrary representation $c$ of $C$. The weight $a$ that scales all radial functions to densities is identical and shown in eqn. (4). Since truncated radial densities can be identified with truncated Gaussian distributions, we find that the problem of estimating the parameters of radial densities on graphs is equivalent to the problem of estimating the parameters of a truncated Gaussian.

We conclude this subsection with a final remark on the expectation and variance. In general, neither the center $c$ nor the squared width $\sigma^2$ of a truncated radial density coincides with its expectation or variance, respectively. We obtain the expectation and variance by setting

$$\mathbb{E}[x] = c + \delta_E(c, \sigma) \quad (7)$$
$$\mathbb{V}[x] = \sigma^2 + \delta_V(c, \sigma), \quad (8)$$

where $\delta_E(c, \sigma)$ and $\delta_V(c, \sigma)$ are the respective adjustments for the truncation. Intuitively, we expect that the adjustment $\delta_E(c, \sigma)$ shifts the expectation $\mathbb{E}[x]$ into the appropriate tail of the distribution.

5. Maximum Likelihood

This section adopts the maximum likelihood method for parameter estimation of radial densities. We assume that $S = \{X_1, \ldots, X_N\} \subseteq \mathcal{X}'$ is a sample of $N$ structures drawn independently from a radial density with unknown regular center $C$ and width $\sigma$. We derive the necessary conditions for optimality and then present an algorithm for estimating the unknown parameters. Finally, we plug the proposed maximum likelihood method into a parametric Bayes classifier.

5.1. Necessary conditions of optimality

The likelihood to be maximized is an orbifold function given by

$$L(C, \sigma|S) = \prod_{i=1}^{N} p(X_i|C, \sigma).$$

Lifting the likelihood yields

$$L^*(C, \sigma|S) = \prod_{i=1}^{N} p^*(x_i|C, \sigma).$$
where $x_i$ projects to $X_i \in S$. Thus, the lifted likelihood is the likelihood of the lift $p^\star$. We pick an arbitrary representation $c$ that projects to $C$. Then we can reduce the lifted likelihood to the likelihood

$$L^\ell(c, \sigma|S) = \prod_{i=1}^{N} p^\ell(x_i|c, \sigma),$$

of the truncated radial function $p^\ell$ on the Dirichlet domain $D_c$, where $x_i \in D_c$ projects to $X_i \in S$. Hence, we can estimate the parameters $C$ and $\sigma$ by adopting the maximum likelihood method for estimating the parameters $c$ and $\sigma$ of a truncated Gaussian. For this, we maximize the log-likelihood of the truncated radial density given by

$$\ell(c, \sigma) = \ln L^\ell(c, \sigma|S) = \sum_{i=1}^{N} \ln p^\ell(x_i|c, \sigma).$$

Since $C$ is regular, the log-likelihood $\ell(c, \sigma)$ is differentiable with respect to its parameters in the interior of the Dirichlet domain $D_c$. The partial derivatives are of the form

$$\nabla_c \ell(c, \sigma) = -\frac{N}{\sigma^2} \left\{ \delta_E(c, \sigma) + c - \frac{1}{N} \sum_{i=1}^{N} x_i \right\},$$

$$\nabla_\sigma \ell(c, \sigma) = -\frac{N}{\sigma^3} \left\{ \delta_V(c, \sigma) + \sigma^2 - \frac{1}{N} \sum_{i=1}^{N} \|x_i - \hat{c}\|^2 \right\}.$$

For a proof, we refer to Appendix A. Recall that $\delta_E(c, \sigma)$ and $\delta_V(c, \sigma)$ are the adjustments defined in (7) and (8) that shift the center $c$ and the squared width $\sigma^2$ to the expectation $E[x]$ and variance $V[x]$.

Setting the derivatives to zero and solving the equations accordingly yields the necessary conditions for optimality

$$\hat{c} = \frac{1}{N} \sum_{i=1}^{N} x_i - \delta_E(c, \sigma) \quad (9)$$

$$\hat{\sigma}^2 = \frac{1}{N} \sum_{i=1}^{N} \|x_i - \hat{c}\|^2 - \delta_V(\hat{c}, \sigma) \quad (10).$$

Equation (9) shows that the maximum likelihood estimate of the center is a sample mean estimate of the unknown expectation $E[x]$ corrected by an adjustment from the expectation to the center. Similarly, from equation (10) follows that the maximum likelihood estimate of the squared variance is an estimate of the unknown variance $V[x]$ corrected by an adjustment from the variance to the width.

### 5.2. Algorithm

Equations (9) and (10) are not closed-form solutions, because the adjustments $\delta_E(c, \sigma)$ and $\delta_V(\hat{c}, \sigma)$ involve integrals depending on $c$ and $\sigma$, respectively. We can – at least
in principle – approximate the adjustments \( \delta_E(c, \sigma) \) and \( \delta_V(\hat{c}, \sigma) \) using Markov-Chain-Monte-Carlo integration [9]. In a practical setting, however, this approach turns out to be computationally intractable for attributed graphs. We sacrifice exactness of the solution for the sake of computational efficiency and discard both adjustments \( \delta_E(c, \sigma) \) and \( \delta_V(\hat{c}, \sigma) \). Then we obtain

\[
\hat{c} = \frac{1}{N} \sum_{i=1}^{N} x_i \\
\hat{\sigma}^2 = \frac{1}{N} \sum_{i=1}^{N} \|x_i - \hat{c}\|^2. \tag{12}
\]

showing that we approximate the unknown center and squared width of the truncated radial density by estimating its mean and variance. A limitation of this approximation is that the resulting error is large for centers close to the boundary of their Dirichlet domain and for large widths. In both cases, the truncation is large.

By definition of the Dirichlet domain \( D_{\hat{c}} \), the vectors \( x_i \) are optimally aligned against the center \( \hat{c} \). As proved in [16, 17], if \((x_i, \hat{c})\) are optimal alignments, then the graph \( \hat{C} = \pi(\hat{c}) \) is a sample Frechet mean of the sample graphs \( X_i = \pi(x_i) \), that is

\[
\hat{C} = \arg \min_{Y \in S_\theta} \sum_{i=1}^{N} d(X_i, Y)^2.
\]

An efficient method to approximate a sample Frechet mean is the incremental arithmetic mean method proposed by [14]

\[
c_1 = x_1 \\
c_i = \frac{i-1}{i} c_{i-1} + \frac{1}{i} x_i, \quad 1 < i \leq N,
\]

where \((x_i, c_{i-1})\) is an optimal alignment for all \( i \in \{2, \ldots, N\} \) and \( x_1 \) is an arbitrarily chosen vector representation that projects to sample graph \( X_1 \in S \). We obtain the maximum likelihood estimate of the center by setting \( \hat{c} = c_N \) and projecting to \( \hat{C} = \pi(\hat{c}) \).

Finally, we use the maximum likelihood estimate \( \hat{c} \) to determine the maximum likelihood estimate of the squared width according to

\[
\hat{\sigma}^2 = \frac{1}{N} \sum_{i=1}^{N} \|x_i - \hat{c}\|^2 = \frac{1}{N} \sum_{i=1}^{N} d(X_i, \hat{C})^2,
\]

where \((x_i, \hat{c})\) are optimal alignments. Algorithm 1 summarizes the maximum likelihood method for parameter estimation of radial densities of structures.

5.3. Bayes Classifier

The probability model for a classifier is given by conditional probability distributions \( p(c_j | X) \) for each class label \( c_j \in C = \{c_1, \ldots, c_k\} \). The input graph \( X \) is then assigned to the class whose conditional distribution is maximum.
Algorithm 1 (Maximum likelihood method)

Input:
Sample $S = \{X_1, \ldots, X_N\} \subseteq X_G$ of $N$ graphs drawn from a radial density

Procedure:
1. Set $\hat{c} = x_1$, where $\pi(x_1) = X_1$
2. for each $i \in \{2, \ldots, N\}$ do
   2.1. Find optimal alignment $(x_i, \hat{c})$
   2.2. Update $\hat{c}$ according to
         $\hat{c} \leftarrow \frac{i - 1}{i} \cdot \hat{c} + \frac{1}{i} \cdot x_i$
3. Set $\hat{C} = \pi(\hat{c})$
4. Estimate squared width $\hat{\sigma}^2$ according to
         $\hat{\sigma}^2 = \frac{1}{N} \sum_{i=1}^{N} \|x_i - \hat{c}\|^2$
where $(x_i, \hat{c})$ are optimal alignments

Output:
Center $\hat{C}$ and width $\hat{\sigma}$

A Bayes classifier infers the conditional distributions $p(c_j|X)$ from Bayes’ theorem

$$p(c_j|X) = \frac{p(X|c_j) \cdot p(c_j)}{p(X)},$$

where $p(X|c_j)$ is the likelihood, and $p(c_j)$ is the prior probability of class $c_j$. The probability $p(X)$ scales $p(c_j|X)$ to a density and can be ignored for classification problems.

The priors $p(c_k)$ can be estimated from a given training set

$$S = \{(X_1, c_1), \ldots, (X_n, c_n)\} \subseteq X_G \times C$$

in the usual way by the relative frequency of class $c_j$. To estimate the likelihoods $p(X|c_j)$, we assume that the data are generated by bell-shaped densities. We approximate these densities by radial densities, and estimate their centers and widths by applying the maximum likelihood method on the training examples.
6. Experiments

We investigated the performance and behavior of the proposed maximum likelihood method for classification problems using Bayes rule. Under the assumption that the graphs of each class are generated by a radial density, we considered two variants:

1. Bayes($C$): We assume that the widths $\sigma$ are identical for each class such that only the centers $C$ need to be estimated.

2. Bayes($C, \sigma$): Both, centers $C$ and widths $\sigma$ are unknown and need to be estimated.

6.1. Random Graphs

In our first series of experiments, we explored the properties of both Bayes classifiers using random weighted graphs.

6.1.1. Data Generation Process

For each trial, we generated the data for a $k$-class problem according to the following procedure:

For each of the $k$ classes do,

a) randomly generate a symmetric matrix $C = (c_{ij}) \in \mathbb{R}^{5 \times 5}$ by drawing the elements of $c_{ij} = c_{ji}$ from a uniform distribution in $[0, 1]$.

b) randomly generate 100 symmetric matrices $X = (x_{ij})$ by adding Gaussian noise $N(0, \sigma^2)$ to $C = (c_{ij})$ according to the rule

$$x_{ij} = x_{ji} = c_{ij} + N(0, \sigma^2).$$

c) select half of the matrices $X$ for the training set and the other half for the test set.

The $k$ matrices $C$ represent the unknown center graphs $C$ of the $k$ classes and the matrices $X$ represent the data graphs $X$. All graphs are complete weighted graphs. After generation of the data set, we randomly permuted the vertices of the data graphs $X$.

It is important to note that the data generated by the above procedure only approximates a radial density with center graph $C$ and a width determined by the variance $\sigma^2$ of the Gaussian noise. This can be seen as follows: Suppose that $C$ is a matrix representation of $C$. Adding Gaussian noise to $C$ results in a matrix representation $X$ projecting to data graph $X$. Two cases can occur: either $X$ is a member of the Dirichlet domain $D_C$ of $C$ or it is not. In the latter case, the maximum likelihood method selects a representation $X'$ of the same graph $X$ that lies in $D_C$. In doing so, parts of the radial density are superimposed by their truncated parts. Thus, the larger the width, the more likely the radial function will be deteriorated given a fixed sample of graphs.
6.1.2. Graph-based Bayes Classifiers

We applied the proposed maximum likelihood method for estimating the parameters of radial densities in a class-wise manner on the training set of each data set. Given the likelihoods and the identical class priors, we applied Bayes rule for classification on the test set. For distance calculations and alignments, we used an extended version of the Bron-Kerbosch algorithm proposed by [18]. The extended Bron-Kerbosch algorithm returns exact graph distances and optimal alignments.

6.1.3. Vector-based Bayes Classifiers

We compared the graph-based Bayes classifiers Bayes($C$) and Bayes($C, \sigma$) with their vectorial counterparts. For this, we considered the weight matrices of the data graphs before permuting the vertices. In doing so, the weight matrices can be regarded as vectors generated from a Gaussian distribution with mean $\mu$ and covariance matrix $\Sigma = \sigma^2 I$ according to the data generating process described above. The parameters were estimated using the standard maximum likelihood method. As for graphs, we considered two variants, Bayes($\mu$) and Bayes($\mu, \sigma$), respectively.

6.1.4. Results: Varying the Noise Level

In our first experiment, we assessed the performance of both graph-based classifiers depending on the standard deviation $\sigma$ of the Gaussian noise. We set the number $k$ of classes (centers) to $k = 5$. We added Gaussian noise with zero mean and identical standard deviation $\sigma$ to each vertex and edge weight. For each standard deviation $\sigma \in \{0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5\}$, we conducted 100 trials giving a total of 700 classification problems. Figure 1 shows the results.

We make the following observations:

- As expected, the performance of both graph-based classifiers decreases with increasing width of the graph distribution.

- Both graph-based classifiers Bayes($C$) and Bayes($C, \sigma$) perform comparable. The results indicate that the proposed maximum likelihood method applied in Bayes($C, \sigma$) does not destabilize due to increased model complexity (unknown width).

- For small widths, the performance of the graph-based classifiers is comparable to their vector-based counterparts. With increasing width, however, the average accuracy of the graph-based classifiers decreases more strongly than the one of the vector-based classifiers.

The degraded performance of the graph-based classifiers directly follows from the disintegration formula (5) and the subsequent discussion. For graphs of order 5, the volume of the graph orbifold $\mathcal{X}_G$ is $|G| = 5! = 120$ times smaller than its ambient Euclidean space $\mathcal{X}$. Figure 2 contrasts the results of the graph-based and vector-based classifiers, where we conducted the same experiment with $k = 5$ classes for...
Figure 1: Average classification accuracies and their standard deviations of Bayes classifiers as a function of the Gaussian noise level $\sigma$ (standard deviation) added to vertex and edge weights. The average is taken over 100 trials for each $\sigma$. Red refers to the case, where only the center/mean is unknown. Blue refers to the case where both, center/mean and width/variance are unknown. Shown are the results of the graph-based classifiers $\text{Bayes}(C)$ (red filled) and $\text{Bayes}(C,\sigma)$ (blue filled) as well as their vector-based counterparts $\text{Bayes}(\mu)$ (red patterned) and $\text{Bayes}(\mu,\sigma)$ (blue patterned). Error bars show the standard deviations. Error bars not shown are due to tiny standard deviations.

Graphs and $k = 5 \times 120 = 600$ classes for vectors. The results show that with comparable volume of both spaces, the graph-based classifiers perform better than their vector-based counterparts. Here, the superior performance of the graph-based classifiers is due to the data generation procedure. The randomly generated graphs follow a deteriorated radial distribution as discussed in Section 6.1.1, whereas their vectorial counterparts follow a Gaussian distribution. Deterioration results in representations that are optimally aligned against their center. This in turn leads to better performance of the graph-based classifiers.

From our observations, we conclude that the graph-based classifier behave similarly as their vector-based counterparts given the circumstances of a ‘smaller’ graph space due to folding the underlying Euclidean space by a finite group action.

6.1.5. Results: Varying the Widths

In this experiment, we assessed the performance of both graph-based classifiers depending on different widths of the class distributions. For this, we use the following experimental setup:

- We set the number $k$ of classes (centers) to $k = 5$.
- According to our sampling procedure, the noise level $\sigma$ determines the width of a class distribution. For class $c \in \{0, \ldots, 4\}$, we considered the noise levels $\sigma_c = \sigma - c \cdot \delta$
Figure 2: Shown are the average classification accuracies of graph-based and vector-based Bayes classifiers as a function of the Gaussian noise level $\sigma$. The setting is almost identical to the one in Figure 1 with the exception that the number $k$ of classes differs for graphs ($k = 5$) and vectors ($k = 600$).

with $\delta \in \{0, 0.05, 0.1, 0.15\}$ as well as $\sigma_c = 2^{-c}\sigma$. Thus, for a given noise level $\sigma$, we considered 5 different types of class-widths.

- We considered the noise levels $\sigma \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$,

For each type of class-width and for each noise level $\sigma$, we conducted 100 trials giving a total of 2500 classification problems. Table 1 summarizes the results. Note that the results for $\sigma_c = \sigma - c \cdot \delta$ with $\delta = 0$ coincide with those from the previous section.

As expected, the performance of both graph-based classifiers increases with uniformly decreasing class widths $\sigma_c$ regardless how largely they vary over all classes. Bayes($C, \sigma$) outperforms Bayes($C$) the larger the class widths $\sigma_c$ are and the more the class widths $\sigma_c$ vary over all classes. For example, Bayes($C, \sigma$) performs slightly better than Bayes($C$) in the case of $\sigma_c = 2^{-c}\sigma$. In this example, the class widths vary widely. In at least three out of five classes, however, the class widths are small, because they are determined by noise levels about 0.1 and less. This results in good performance of both graph-based classifiers. Similarly, we also obtain comparable results at low noise levels $\sigma \in \{0.1, 0.2\}$ regardless of how widely the class widths $\sigma_c$ vary. On the other hand, Bayes($C, \sigma$) clearly outperforms Bayes($C$) in the case of $\sigma_c = \sigma - 0.15c$, where $\sigma \in \{0.4, 0.5\}$. Here, the class widths vary widely and the noise level is high.

Suppose that $\beta$ is the percentage of trials in which Bayes($C$) performs better than Bayes($C, \sigma$). The corresponding column of Table 1 shows that $\beta$ is all the larger the less the class widths vary. In other words, the more the class widths vary, the less likely it is that Bayes($C$) achieves a better result than Bayes($C, \sigma$). In addition, if Bayes($C$) performs better than Bayes($C, \sigma$) on some trial, then the difference between both accuracies is all the less the larger the class widths vary, as shown in the last column of Table 1.

Apart from the findings of our first experiments, these results demonstrate that the proposed maximum likelihood approach is sensitive to class distributions of varying widths.
Table 1: Average classification accuracy of the graph-based classifiers Bayes($C$) and Bayes($C, \sigma$) as a function of class width $\sigma_c$ and noise level $\sigma$. The column labeled with $\beta$ shows the percentage of trials in which Bayes($C$) performed better than Bayes($C, \sigma$). The last column labeled with $\max \beta$ shows the maximum difference between the classification accuracies of Bayes($C$) and Bayes($C, \sigma$) given that Bayes($C$) performs better than Bayes($C, \sigma$).
Table 2: Summary of main characteristics of the data sets. Shown are the size of the training (\(tr\)), validation (\(va\)), and test set (\(te\)), respectively, the number of classes (\(|C|\)), the average number of nodes (\(\bar{|V|}\)), the maximum number of nodes (max \(|V|\)), the average number of edges (\(\bar{|E|}\)), and the maximum number of edges (max \(|E|\)).

| data set    | size (\(tr, va, te\)) | \(|C|\) | \(\bar{|V|}\) | max \(|V|\) | \(\bar{|E|}\) | max \(|E|\) |
|-------------|-----------------------|--------|-------------|-----------|-------------|-----------|
| Letter      | 750, 750, 750         | 15     | 4.7         | 8         | 3.1         | 9         |
| Fingerprint | 500, 300, 2000         | 4      | 5.42        | 26        | 4.42        | 24        |
| GREC        | 286, 286, 528         | 22     | 11.5        | 25        | 12.2        | 30        |
| COIL-4      | 48, 20, 40            | 4      | 5.9         | 26        | 8.8         | 48        |

Despite the fact that estimating the true widths is intractable due to truncation, their relative sizes can be roughly estimated, which in turn is sufficient for a Bayes classifier.

6.2. Image Recognition

6.2.1. Data.

We selected four data sets described in [23]. Each data set is divided into a training, validation, and test set. The description of the data sets are excerpts from [23]. Table 2 provides a summary of the main characteristics of the data sets.

**Letter graphs (high distortion level)** The letter data set compiles distorted letter drawings from the Roman alphabet that consist of straight lines only (A, E, F, H, I, K, L, M, N, T, V, W, X, Y, Z). The graphs are uniformly distributed over the 15 classes (letters). The letter drawings are obtained by distorting prototype letters at high distortion level. Lines of a letter are represented by edges and ending points of lines by vertices. Each vertex is labeled with a two-dimensional vector giving the position of its end point relative to a reference coordinate system. Edges are labeled with weight 1. Finally, we standardized the attributes.

**Fingerprint graphs** This data set represents fingerprint images of the NIST-4 database [32] from four classes arch, left, right, and whorl. Fingerprint images are converted into graphs by filtering the images and extracting regions that are relevant [21]. Relevant regions are binarized and a noise removal and thinning procedure is applied. This results in a skeletonized representation of the extracted regions. Ending points and bifurcation points of the skeletonized regions are represented by vertices. Additional vertices are inserted in regular intervals between ending points and bifurcation points. Finally, undirected edges are inserted to link vertices that are directly connected through a ridge in the skeleton. Each vertex is labeled with a two-dimensional attribute giving its position. Edges are attributed with an angle denoting the orientation of the edge with respect to the horizontal direction. Finally, we standardized the attributes.
GREC graphs  This data set [6] consists of graphs representing symbols from architectural and electronic drawings. The images occur at five different distortion levels. Depending on the distortion level, either erosion, dilation, or other morphological operations are applied. The result is thinned to obtain lines of one pixel width. Finally, graphs are extracted from the resulting denoised images by tracing the lines from end to end and detecting intersections as well as corners. Ending points, corners, intersections and circles are represented by vertices and labeled with a two-dimensional attribute giving their position. The vertices are connected by undirected edges which are labeled as line or arc. An additional attribute specifies the angle with respect to the horizontal direction or the diameter in case of arcs. Finally, we standardized the attributes.

COIL-4 graphs  The COIL-100 database [20] consists of colored images of 100 different objects. In [22], a subset of 32 objects that are hard to recognize has been identified. Following [28], we selected four of the 32 objects, two spherical fruit like objects and two toy car like objects. The chosen objects correspond to indices 1, 3, 18, and 41 of the COIL-100 data set, where the first index is 0.

Images of the objects are taken at pose intervals of five degrees. Objects are represented by graphs as follows: First, a Harris corner detection algorithm extracts the corners. Next, the images are segmented using a Delauney triangulation applied on the corner points. The result of the triangulation is then converted into a graph by representing lines by undirected edges and ending points of lines by nodes. Each node is labeled with a two-dimensional attribute giving its position, while edges are unlabeled.

The training set is composed of 12 images per object, acquired every 15 degree of rotation. From the remaining images we randomly select 5 images per object for the validation set, and 10 images per object for the test set. This results in a training set of size 48, a validation set of size 20, and a test set of size 40. The resulting dataset is referred to as COIL-4.

6.2.2. Bayes Classifiers

We applied the proposed maximum likelihood method for estimating the parameters of radial densities in a class-wise manner on the training and validation set of each data set. In addition, we used both, the training and validation set to estimate the class priors. Given the likelihoods and the class priors, we applied Bayes rule for classification.

For distance calculations and alignments, we used the graduated assignment algorithm [10]. The graduated alignment algorithm constructs an approximate bi-stochastic match matrix. We applied Munkres algorithm for converting the match matrix to a permutation sub-matrix. For this, we converted the rectangular match matrix to a squared cost matrix by padding rows and columns with a pre-specified maximum cost accordingly. Lower cost refer to more likely vertex correspondences and vice versa. Munkres algorithm returns a permutation matrix, which we convert to a sub-matrix by deleting the padding rows and columns. We then aligned the first graph towards the second given the permutation sub-matrix and then computed the squared Euclidean distance.
6.2.3. Methods

We compared the proposed maximum likelihood classifier (ML-Bayes) against the following methods:

1. kNN: k-Nearest Neighbor [28]
2. SK+SVM: a family of similarity kernels in conjunction with the SVM [24].
3. LE+SVM: a family of Lipschitz embeddings in conjunction with SVM [24].
4. FSel-SVM: SVM recursive feature selection [3].
5. FSel+kPCA: feature selection through component analysis [3].
6. LGQ, LGQ2.1: learning graph quantization methods proposed by [15, 28].

In the following, we briefly describe the above methods. The description of SK+SVM, LE+SVM, FSel-SVM, and FSel+kPCA is an excerpt of [28].

**kNN** As the conceptually simplest algorithm, k-nearest neighbor serves as our reference method. In [28], the value of $k$ is selected based on the training and validation set of the underlying dataset. For graph matching, the graduated assignment algorithm [10] has been applied followed by Munkres algorithm as a post-processing step.

**SK+SVM** Similarity kernels [24] transform graph distances $d(X,Y)$ to a similarity measure $\exp(-\gamma d(X,Y)^2)$, which is a kernel for a subsequent SVM classifier. The parameter $\gamma$ is the meta-parameter that needs to be determined based on performance on the validation set. The standard training paradigm is applied to the SVM classifier in the next stage. Here, SK+SVM refers to a family of similarity kernels that differ in the choice of how prototypes are selected.

**LE+SVM** Lipschitz embedding classifiers are a family of methods proposed by [24]. Given a training set $S = \{X_1, ..., X_n\}$ of graphs, we define a set $R = \{R_1, ..., R_k\}$ consisting of $k$ subsets $R_i$ of $S$, called reference set for embedding. The Lipschitz embedding $\phi_{R,f} : S \rightarrow \mathbb{R}^k$ with respect to $R$ is defined as $\phi_{R,f}(X) = (f(X,R_1), ..., f(X,R_k))$, where $f(X,R_i)$ could be the minimum, maximum or mean distance of $X$ to elements in the set $R_i$. The meta-parameters are the number $k$ of reference sets and the size of a reference set. The optimal parameters are chosen in conjunction with optimizing the SVM classifier in the next stage. Similarly, as for SK+SVM, Lipschitz embeddings LE+SVM refer to a family of methods that differ in the choice of the reference set.

---

2 The description of SK+SVM is taken from [28].
Table 3: Classification rates (%) on the IAM Graph Database. References show where the results are taken from. For the families SK+SVM and LE+SVM the best result on each test set is shown. Classification results on COIL-4 are unavailable for SK+SVM, LE+SVM, FSel-SVM, and FSel-kPCA.

<table>
<thead>
<tr>
<th>Method</th>
<th>Refs.</th>
<th>Letter</th>
<th>Fingerprint</th>
<th>GREC</th>
<th>COIL-4</th>
</tr>
</thead>
<tbody>
<tr>
<td>kNN</td>
<td>[28]</td>
<td>82.0</td>
<td>76.7</td>
<td>95.5</td>
<td>87.5</td>
</tr>
<tr>
<td>SK+SVM</td>
<td>[24]</td>
<td>79.1</td>
<td>41.0</td>
<td>94.9</td>
<td>–</td>
</tr>
<tr>
<td>LE+SVM</td>
<td>[24]</td>
<td>92.5</td>
<td>82.8</td>
<td>96.8</td>
<td>–</td>
</tr>
<tr>
<td>FSel-SVM</td>
<td>[3]</td>
<td>92.8</td>
<td>81.7</td>
<td>92.2</td>
<td>–</td>
</tr>
<tr>
<td>FSel-kPCA</td>
<td>[3]</td>
<td>90.3</td>
<td>82.6</td>
<td>91.6</td>
<td>–</td>
</tr>
<tr>
<td>LGQ</td>
<td>[28]</td>
<td>80.9</td>
<td>79.2</td>
<td>94.7</td>
<td>90.0</td>
</tr>
<tr>
<td>LGQ2.1</td>
<td>[28]</td>
<td>83.7</td>
<td>82.2</td>
<td>97.3</td>
<td>92.5</td>
</tr>
<tr>
<td>Bayes(C)</td>
<td>–</td>
<td>84.9</td>
<td>75.9</td>
<td>93.4</td>
<td>27.5</td>
</tr>
<tr>
<td>Bayes(C,σ)</td>
<td>–</td>
<td>83.5</td>
<td>77.4</td>
<td>87.3</td>
<td>92.5</td>
</tr>
</tbody>
</table>

**FSel-SVM**  This approach applied by [3] on graphs uses the most significant features depending upon how crucial they are for classification. Consider the decision surface of a linear SVM given by $f(x) = \langle w, x \rangle + b$, where $w, x \in \mathbb{R}^n, b \in \mathbb{R}$. The influence a feature $x_i$ exerts on classification depends on weight $w_i$. Hence, a SVM is trained, the features ranked based on their corresponding weights and low ranking features recursively eliminated until an optimal number of features remain.

**FSel-kPCA**  Principal Component Analysis (or its nonlinear extension kernel Principal Component Analysis) is a popular technique for projecting original data into a space spanned by eigenvectors of its covariance matrix. The most directions (or components) with high variances correspond to higher eigenvalues. Now a feature ranking strategy is implemented by choosing dimensions which are crucial for classification. This approach has been applied on graphs by [3].

**LGQ and LGQ2.1**  Both methods extend the respective learning vector quantization algorithms LVQ and LVQ2.1 [19] to the graph domain.

In [28], the k-means algorithm for graphs [13] has been applied to initialize the prototypes in a class-wise manner. The optimal number $k$ of prototypes, the initial learning rate $\eta_0$, and the window width of LGQ2.1 have been selected using the training and validation set of the dataset under consideration. The learning rate was set to be $\eta_t = 0.01(1 - t/T)$ at every iteration step $t$, where $T$ is the maximum number of iterations. For distance calculation and graph alignment, the graduated assignment algorithm [10] has been applied followed by Munkres algorithm as a post-processing step.
6.2.4. Results

Table 3 presents the classification rates of each method. Since SK+SVM and LE+SVM are families of methods, we summarized their performance as follows: For each dataset, we selected the best model of a family and recorded its performance in Table 2. This gives an overly optimistic bias towards both families, which is justified by the fact that our main findings remain unaffected.

Table 4 shows the number of prototypes used by each method. For SK+SVM, LE+SVM, FSeL-SVM, and FSeL-kPCA the size of the prototype sets was inferred from information available in [3, 24, 25] and may only be approximate.

Except for Bayes\((C)\) on COIL-4, the results show that both Bayes classifiers perform reasonably well, though the proposed maximum likelihood method may provide poor probability estimates due to (1) ignoring truncation, and (2) assuming spherical distributions, which are unimodal. As shown on random graphs, for applying the Bayes rule it is sufficient to estimate the parameters relative to one another. Then good classification accuracy can be expected when the centers are sufficiently distant and the widths are sufficiently small.

The failure of Bayes\((C)\) on COIL-4 is due to the fact that the widths vary more widely than for the other classes, which is recognized by Bayes\((C, \sigma)\). Though the class widths vary for each data set, the better performance of Bayes\((C)\) could be due to the specific characteristics of the particular data sets as shown in our experiments on random graphs (see the last two columns of Table 1).

The benefit of the Bayes classifiers is that they do not require parameter tuning and that they are much faster in training and classification. The computation time of Bayes\((C, \sigma)\) for both, training and evaluating the model, range from about 1.3 seconds for COIL-4 up to about 11 seconds for the GREC data set using Java jdk6 under a MacBookPro8,2 with a 2.3 GHz IntelCore i7 processor. All other methods are of magnitudes slower during training and/or classification due to larger prototype sets, parameter selection procedures and/or cycling several iterations through the training set (LGQ methods). In contrast, Bayes\((C)\) and Bayes\((C, \sigma)\) learn one prototype (center) per class and require only one cycle through the training set to learn the model.

The results suggest to extend the proposed approach to mixture models of radial densities for classification as well as clustering problems.

7. Conclusion

In this contribution we proposed a maximum likelihood method for estimating the location parameters of bell-shaped densities on attributed graphs. Lifted to the underlying Euclidean space, radial densities on graphs reduce to truncated Gaussian distributions. For estimating the parameters of a truncated Gaussian, we derived the gradients of the log-likelihood. By ignoring computation of integrals, we applied the proposed method to random graphs and four benchmark data sets. The results suggest to develop efficient algorithms for estimating the adjustments from expectation and variance to center and
Table 4: Prototype set sizes in different algorithms and the estimated speedup factor (sf) with respect to the algorithm with the largest prototype set. Approximate sizes of the prototype sets are labelled accordingly.

width due to truncation. In addition, extending the proposed approach to mixture models of radial densities is another direction of future research.

A. Deriving the gradient of the log-likelihood

Suppose that \( x_1, \ldots, x_N \in D_c \) are \( N \) training examples. The log-likelihood of the truncated radial density to be maximized is defined by

\[
\ell(c, \sigma) = \sum_{i=1}^{N} \ln p^t(x_i | c, \sigma).
\]

The truncated radial function \( p^t \) is differentiable with respect to its parameters in the interior of the Dirichlet fundamental domain \( D_c \). In addition, since \( C \) is regular, we find that any representation \( c \) of \( C \) lies in the interior of \( D_c \). Thus, the gradient of the log-likelihood exists and is of the form

\[
\nabla_{c, \sigma} \ell(c, \sigma) = \left( \begin{array}{c}
\nabla_c \ell(c, \sigma) \\
\nabla_\sigma \ell(c, \sigma)
\end{array} \right) = \left( \begin{array}{c}
\sum_{i=1}^{N} \nabla_c \ln p^t(x_i | c, \sigma) \\
\sum_{i=1}^{N} \nabla_\sigma \ln p^t(x_i | c, \sigma)
\end{array} \right).
\]
To determine the gradient of $\ell$ at $(c, \sigma)$, we first consider a single training example $x \in D_c$. Thus, the log-likelihood is of the form

$$\ell(c, \sigma) = \ln p^t(x|c, \sigma)$$

$$= \ln a \cdot h(x|c, \sigma)$$

$$= \ln \left( a \cdot \exp \left( -\frac{\|x - c\|^2}{2\sigma^2} \right) \right)$$

$$= \ln a - \frac{\|x - c\|^2}{2\sigma^2}.$$ 

where $a$ is the weight that scales the standard radial function $h$ to a density. Suppose that $\alpha = 1/a$, then we have

$$\alpha = \alpha(c, \sigma) = \int_{D_c} h(x|c, \sigma) \lambda'(dx).$$

The gradient of $\ln \alpha(c, \sigma)$ at $(c, \sigma)$ is of the form

$$\nabla_{c, \sigma} \ln \alpha(c, \sigma) = \frac{\nabla_{c, \sigma} \alpha(c, \sigma)}{\alpha(c, \sigma)},$$

where the gradient $\nabla_{c, \sigma} \alpha(c, \sigma)$ can be determined by using Lebesgue’s Dominated Convergence Theorem as follows

$$\nabla_{c, \sigma} \alpha(c, \sigma) = \nabla_{c, \sigma} \int_{D_c} h(x|c, \sigma) \lambda'(dx)$$

$$= \int_{D_c} \nabla_{c, \sigma} h(x|c, \sigma) \lambda'(dx).$$

In the following, we determine the gradients $\nabla_c \ln p^t(x|c, \sigma)$ and $\nabla_\sigma \ln p^t(x|c, \sigma)$ separately. Then we construct the gradient of $\ell$ at $(c, \sigma)$ over all training examples.

**Determining $\nabla_c \ln p^t(x|c, \sigma)$**

We have

$$\nabla_c \alpha(c, \sigma) = \int_{D_c} \nabla_c h(x|c, \sigma) \lambda'(dx)$$

$$= \int_{D_c} \nabla_c \exp \left( -\frac{\|x - c\|^2}{2\sigma^2} \right) \lambda'(dx)$$

$$= \int_{D_c} \frac{x - c}{\sigma^2} \cdot h(x|c, \sigma) \lambda'(dx).$$
From the last equation follows

\[ \nabla_c \ln \alpha(c, \sigma) = \frac{\nabla_c \ln \alpha(c, \sigma)}{a(c, \sigma)} = \frac{\int_{D_c} \frac{x-c}{\sigma^2} \cdot h(x|c, \sigma) \lambda'(dx)}{\int_{D_c} h(x|c, \sigma) \lambda'(dx)} \]

\[ = \frac{\int_{D_c} (x-c) \cdot h(x|c, \sigma) \lambda'(dx)}{\sigma^2 \int_{D_c} h(x|c, \sigma) \lambda'(dx)} \]

\[ = \frac{\mathbb{E}[x] - c}{\sigma^2} \]

\[ = \frac{\delta_E(c, \sigma)}{\sigma^2} \]

where \( \delta_E(c, \sigma) \) is the adjustment defined in (7) that shifts the center \( c \) to the expectation \( \mathbb{E}[x] \).

The gradient of \( \ln \, p^t(x|c, \sigma) \) with respect to \( c \) is then given by

\[ \nabla_c \ln p^t(x|c, \sigma) = \nabla_c \ln a - \nabla_c \frac{\|x - c\|^2}{2\sigma^2} \]

\[ = -\nabla_c \ln \alpha - \nabla_c \frac{\|x - c\|^2}{2\sigma^2} \]

\[ = -\frac{\delta_E(c, \sigma)}{\sigma^2} + \frac{c - x}{\sigma^2}. \]

Note that the second equation follows from \( a = 1/\alpha \).

**Determining \( \nabla_\sigma \ln p^t(x|c, \sigma) \)**

We have

\[ \nabla_\sigma \alpha(c, \sigma) = \int_{D_c} \nabla_\sigma h(x|c, \sigma) \lambda'(dx) \]

\[ = \int_{D_c} \nabla_\sigma \exp \left( -\frac{\|x - c\|^2}{2\sigma^2} \right) \lambda'(dx) \]

\[ = \int_{D_c} \frac{\|x - c\|^2}{\sigma^4} \cdot h(x|c, \sigma) \lambda'(dx). \]
From the last equation follows
\[
\nabla_\sigma \ln \alpha(c, \sigma) = \frac{\nabla_\sigma \alpha(c, \sigma)}{\alpha(c, \sigma)} \\
= \frac{1}{\sigma^3} \left( \frac{1}{\alpha(c, \sigma)} \int_{D_c} \|x - c\|^2 \cdot h(x|c, \sigma) \lambda'(dx) \right) \\
= \frac{\nabla[ x]}{\sigma^3} \\
= \frac{\delta_V(c, \sigma) + \sigma^2}{\sigma^3},
\]

where \( \delta_V(c, \sigma) \) is the adjustment defined in (8) that shifts the squared width \( \sigma^2 \) to the variance \( \nabla[ x] \).

The gradient of \( \ln p^t(x|c, \sigma) \) with respect to \( \sigma \) is then given by
\[
\nabla_\sigma \ln p^t(x|c, \sigma) = \nabla_\sigma \ln a - \nabla_\sigma \frac{\|x - c\|^2}{2\sigma^2} \\
= -\nabla_\sigma \ln a - \nabla_\sigma \frac{\|x - c\|^2}{2\sigma^2} \\
= -\delta_V(c, \sigma) + \sigma^2 - \frac{\|x - c\|^2}{\sigma^3}.
\]

Gradient \( \nabla_{c,\sigma} \ell \) over all training examples

Finally, summing the gradients \( \nabla_c \ln p^t \) and \( \nabla_\sigma \ln p^t \) over all training examples gives the gradients of the log-likelihood
\[
\nabla_c \ell(c, \sigma) = -\frac{N}{\sigma^2} \left\{ \delta_x(c, \sigma) + c - \frac{1}{N} \sum_{i=1}^N x_i \right\} \\
\nabla_\sigma \ell(c, \sigma) = -\frac{N}{\sigma^3} \left\{ \delta_V(c, \sigma) + \sigma^2 - \frac{1}{N} \sum_{i=1}^N \|x_i - c\|^2 \right\}.
\]

References


26


[23] K. Riesen and H. Bunke, ”IAM Graph Database Repository for Graph Based Pattern Recognition and Machine Learning”, SSPR, 2008.


