Bayesian network Fisher kernel for categorical feature spaces

Janne Leppä-aho, Tomi Silander & Teemu Roos

Behaviormetrika

ISSN 0385-7417

Behaviormetrika
DOI 10.1007/s41237-019-00103-6

ONLINE FIRST

Springer

Behaviormetrika
https://doi.org/10.1007/s41237-019-00103-6

ORIGINAL PAPER

Bayesian network Fisher kernel for categorical feature spaces

Janne Leppä-aho[1] · Tomi Silander[2] · Teemu Roos[1]

Received: 21 August 2019 / Accepted: 22 December 2019

Abstract

We address the problem of defining similarity between vectors of possibly dependent categorical variables by deriving formulas for the Fisher kernel for Bayesian networks. While both Bayesian networks and Fisher kernels are established techniques, this result does not seem to appear in the literature. Such a kernel naturally opens up the possibility to conduct kernel-based analyses in completely categorical feature spaces with dependent features. We show experimentally how this kernel can be used to find subsets of observations that we see as representative for the underlying Bayesian network model.

Keywords Fisher kernel · Bayesian network · Similarity

1 Introduction

Variables such as color, nationality and species that are measured on a nominal scale do not allow for an obvious definition of distance. Consequently, there is no obvious solution for measuring similarities between random vectors whose components are measured on the nominal scale. In this paper, we propose computing similarities of the nominal scale random vectors using a Fisher kernel based on discrete Bayesian networks (Pearl 1988).

Communicated by Joe Suzuki.

J.L. and T.R. were supported by the Academy of Finland (COIN CoE and Projects TensorML and WiFiUS). J.L. was supported by the DoCS doctoral programme of the University of Helsinki.

✉ Janne Leppä-aho
janne.leppa-aho@helsinki.fi

[1] University of Helsinki/Helsinki Institute for Information Technology, P.O. Box 68, 00014 Helsinki, Finland

[2] NAVER LABS Europe, 6 Chemin de Maupertuis, 38240 Meylan, France

Published online: 08 January 2020

Fisher kernels (Jaakkola and Haussler 1998) were originally motivated by the need to build classifiers for objects of different lengths/sizes. They have since been used to classify proteins, speakers, documents, images, tree structured data, etc. A survey by Sewell (2011) provided a brief review. Much of the development has been concentrating on classification of different kinds of objects, but kernels can also be used for many other things such as regression, clustering or dimensionality reduction. Each different model requires a separate derivation of the formulas that show how the Fisher kernel is computed for the data items modeled by the particular model. To our knowledge, the derivation of the Fisher kernel for general Bayesian networks has not been presented before. We will also present formulas for measuring similarity of two sets of observations modeled by a Bayesian network.

In practice, Bayesian network models are often learned automatically from the data. However, common model selection criteria do not necessarily identify the Bayesian network model completely, but they generally return a class of "equivalent", but differently parametrized, Bayesian networks or a random member of such a class. While the Fisher kernel in general is known to be sensitive to the parametrization, we will show that any member of such an equivalence class yields the same Fisher kernel.

In the following, we will first review the related work on the multivariate categorical similarity/distance measures. We will then quickly introduce notation for Bayesian networks and state the main formula for computing a Fisher kernel based on it. We present the derivation and an important result of insensitivity to the choice of equivalent structure. Then, we show how the kernel generalizes to comparing sets of observations and discuss the connection to a kernel-based distance measure called maximum mean discrepancy. We also describe a simple greedy algorithm which can be used for finding representative sets of observations with the help of the set kernel. Finally, we make use of the algorithm in experiments, where we try find representative sets for various Bayesian network structures.

2 Related work

Boriah et al. (2008) have reviewed several similarity measures for discrete vectors, many of which have implemented some kind of weighting of different variables, but none of which took dependencies between variables into account. The work on similarities that takes dependencies into account seems to do so by considering pairwise similarities only (Niitsuma and Okada 2005; McCane and Albert 2008; Desai et al. 2011; Ring et al. 2015). This is in sharp contrast to the Fisher kernel for Bayesian networks that can model dependencies between many variables.

Fisher kernel for discrete data is not a new idea. Fisher kernels have been derived, for example, for hidden Markov models (Jaakkola et al. 2000) and probabilistic latent semantic indexing (Chappelier and Eckard 2009). While these models can be used for vectors of categorical dimensions, their bias is designed with respect to a special latent variable that implements labeling, clustering or topic detection. As dependency models, their structure implies that all the observed variables are (marginally) dependent on each other. Since Bayesian networks try to capture a more explicit dependency structure, the Fisher kernel-based similarity also becomes different.

In addition to Fisher kernels, other approaches to building kernels with the aid of probabilistic models include probability product kernels (Jebara et al. 2004) and diffusion kernels (Lafferty and Lebanon 2005).

3 Fisher kernel for Bayesian networks

Bayesian networks are multivariate dependency models over n random variables $D = (D_1, \ldots, D_n)$. In our setting, each D_i is a categorical variable taking one of the r_i values from $\{1, \ldots, r_i\}$. The Bayesian network $B = (G, \theta)$ consists of a directed acyclic graph (DAG) G and parameters θ. In the graph G, the nodes $1, \ldots, n$ correspond to the components of D and the arcs encode dependence structure among the variables. We let π_i denote the parent set of node i (i.e., the nodes from which there are direct arcs to node i). Letting $\pi_i(D)$ denote the parent variables and enumerating all the possible parent variable configurations from 1 to q_i, this structure is then populated with parameters θ so that each value k of variable D_i and each value configuration j of $\pi_i(D)$ is attached to a parameter θ_{ijk} so that:

$$P(D_i = k|\pi_i(D) = j;\theta) = \theta_{ijk},$$

where $\pi_i(D) = j$ means that the parent variables take values according to the jth configuration. For a data point d, we use $P(d;\theta)$ as a shorthand for $P(D = d;\theta)$.

Using an indicator function $I_d(x)$, that takes value 1, if $d = x$, and 0 otherwise, we can write the likelihood function for d as:

$$P(d;\theta) = \prod_{i=1}^{n} P(d_i \mid \pi_i(d);\theta) = \prod_{i=1}^{n}\prod_{j=1}^{q_i}\prod_{k=1}^{r_i} \theta_{ijk}^{I_{d_i}(k)\cdot I_{\pi_i(d)}(j)}. \tag{1}$$

We assume all the parameters θ_{ijk} to be strictly positive.

3.1 Fisher kernel

A Fisher kernel is a way to define the similarity of a pair of items based on a parametric statistical model. We will first review the definition of Fisher kernel and then right after that present the form that the Fisher kernel takes when this parametric model is a Bayesian network over discrete variables. The actual derivation is presented in the following section.

Definition 1 For a model $P(d;\theta)$, parametrized by a vector θ of parameters, a Fisher kernel $K(x, y;\theta)$ is defined as:

$$K(x, y;\theta) = s^T(x;\theta)\mathcal{I}^{-1}(\theta)s(y;\theta).$$

In the definition, the score $s(x;\theta)$ is a column vector of partial derivatives of the log likelihood, i.e., $s(x;\theta)_i = \frac{\partial}{\partial\theta_i}\log P(x;\theta)$, and the $\mathcal{I}(\theta)$ is a so-called Fisher information matrix (Cover and Thomas 2006) with elements:

$$\mathcal{I}(\theta)_{ij} = -\mathbb{E}_d\left[\frac{\partial^2 \log P(d;\theta)}{\partial\theta_i\partial\theta_j}\right],$$

where $\mathbb{E}_d[f(d)] = \sum_d P(D = d)f(D = d)$.

For the Bayesian networks, we obtain the following:

Theorem 1 *Let $M = (G, \theta)$ be a Bayesian network over variables*

$D = (D_1, \ldots, D_n)$ with parent sets $(\pi_1, \ldots, \pi_n)$. Now, the Fisher similarity for two n-dimensional discrete vectors x and y modeled by M, can be computed as a variable-wise sum

$$K(x, y;\theta) = \sum_{i=1}^{n} K_i(x, y;\theta),$$

where

$$K_i(x, y;\theta) = \begin{cases} 0 & \text{if } \pi_i(x) \neq \pi_i(y), \\ \frac{-1}{P(\pi_i(x);\theta)} & \text{if } \pi_i(x) = \pi_i(y) \\ & \wedge\, (x_i \neq y_i), \\ \frac{1}{P(\pi_i(x);\theta)} \cdot \frac{1-\theta_{i\pi_i(x)x_i}}{\theta_{i\pi_i(x)x_i}} & \text{if } \pi_i(x) = \pi_i(y) \\ & \wedge\, (x_i = y_i). \end{cases} \tag{2}$$

The formula above may lead to dividing by zero, but only in case of comparing items that the model deems impossible by assigning them the zero probability (which is however excluded under our assumption $\theta_{ijk} > 0$).

The joint probabilities $P(\pi_i(x);\theta)$ are not readily available in Bayesian networks, but must be computed, which in general is an NP-hard problem (Cooper 1990). However, many practical implementations of the Bayesian network inference are based on the junction tree algorithm (Lauritzen and Spiegelhalter 1988), which features a static data structure that contains the joint probabilities for sets (called cliques) that contain parent sets as their subsets. Therefore, the probabilities $P(\pi_i(x);\theta)$ can be marginalized from the clique probabilities. When such an exact inference is computationally too demanding, approximate inference techniques can be used.

4 Outline of the derivation

In this section, we will go through the derivation of Bayesian network Fisher kernel on a high level. To reduce clutter, some details of the derivations are postponed to Appendices.

4.1 Univariate Fisher kernel

While our interest is in a multivariate model, much of the derivation can be reduced to computing the Fisher kernel for a single categorical variable with r possible values, since a Bayesian network can be represented as a collection of conditional probability tables, and the tables themselves consist of univariate categorical distributions. Therefore, we will first study the simple univariate case.

The main result of this section is formulated as follows.

Lemma 1 *Assume D is a single categorical variable taking values from 1 to r with probabilities $\theta = (\theta_1, \dots, \theta_r)$ where $\theta_k > 0$ and $\sum_{k=1}^{r} \theta_k = 1$. Now, the Fisher similarity for two observations x and y is given as*:

$$K(x, y;\theta) = \begin{cases} \frac{1-\theta_x}{\theta_x} & \text{if } x = y, \text{ and} \\ -1 & \text{if } x \neq y, \end{cases}$$

where $\theta_x = P(D = x;\theta)$.

Proof We will proceed in a straightforward fashion, dividing our proof into three parts: (1) We first compute the gradient of the log likelihood; (2) then find an expression for the inverse of Fisher information matrix; (3) and lastly, combine these to get the statement of Lemma 1.

4.1.1 Gradient of the log likelihood

Let d denote a value of D. We can express the likelihood as:

$$P(d;\theta) = \prod_{k=1}^{r} \theta_k^{I_d(k)},$$

where θ_r is not a real parameter of the model but a shorthand notation $\theta_r \equiv 1 - \sum_{k=1}^{r-1} \theta_k$. Thus, the log-likelihood function is $\log P(d;\theta) = \sum_{k=1}^{r} I_d(k) \log \theta_k$. Taking the partial derivatives wrt. θ_k yields:

$$s(d;\theta)_k = \frac{\log P(d;\theta)}{\partial \theta_k} = I_d(k)\theta_k^{-1} - I_d(r)\theta_r^{-1}.$$

Using e_k to denote k^{th} standard basis vector of $\mathbb{R}^{r-1}$, and $\mathbf{1}$ for the vector of all ones, the whole vector $s(d;\theta)$ can be written as:

$$s(d;\theta) = \begin{cases} \theta_k^{-1} e_k & \text{if } d = k < r, \text{ and} \\ -\theta_r^{-1} \mathbf{1} & \text{if } d = r, \end{cases}$$

or using indicators

$$s(d;\theta) = \sum_{k=1}^{r-1} I_d(k) \theta_k^{-1} e_k - I_d(r) \theta_r^{-1} \mathbf{1}. \tag{3}$$

It is worth emphasizing that each partial derivative depends on the data vector d, but is a function of the parameter θ. It may be useful to further reveal the structure by writing:

$$s(d;\theta)_k = \begin{cases} 0 & \text{if } d \neq k < r, \\ \theta_k^{-1} & \text{if } d = k < r, \text{ and} \\ -\theta_r^{-1} & \text{if } d = r. \end{cases}$$

So depending on the data item d, the partial derivative $s(d;\theta)_k$ is one of the three functions of θ above.

4.1.2 The Fisher information and its inverse

The Fisher information matrix for the multinomial model (which is equivalent to our categorical case when the number of trials is 1) takes the form (Bernardo and Smith 1994, p. 336):

$$\mathcal{I}(\theta) = \frac{1}{\theta_r} \begin{bmatrix} \theta_r \theta_1^{-1} + 1 & 1 & \cdots & 1 \\ 1 & \theta_r \theta_2^{-1} + 1 & \cdots & 1 \\ \cdots & \cdots & \ddots & \vdots \\ 1 & 1 & \cdots & \theta_r \theta_{r-1}^{-1} + 1 \end{bmatrix} \tag{4}$$

and its inverse is given as:

$$\mathcal{I}^{-1}(\theta) = \begin{bmatrix} \theta_1(1-\theta_1) & -\theta_1\theta_2 & \cdots & -\theta_1\theta_{r-1} \\ -\theta_1\theta_2 & \theta_2(1-\theta_2) & \cdots & -\theta_2\theta_{r-1} \\ \cdots & \cdots & \ddots & \vdots \\ -\theta_1\theta_{r-1} & -\theta_2\theta_{r-1} & \cdots & \theta_{r-1}(1-\theta_{r-1}) \end{bmatrix}. \tag{5}$$

Bernardo and Smith (1994) omit the explicit derivations of (4) and (5). We will present the derivation of $\mathcal{I}(\theta)$ in Appendix A. The inverse can be obtained from this by noting that $\mathcal{I}(\theta)$ is expressible as a sum consisting of two terms: an invertible (diagonal) matrix and a rank one matrix. Inverting a matrix with this structure is straightforward (Miller 1981). Even more straightforwardly, computing the matrix product, $\mathcal{I}(\theta)\mathcal{I}^{-1}(\theta)$, and verifying that it gives an identity matrix proves that inverse of $\mathcal{I}(\theta)$ has to have the form given in (5).

For further purposes, it is useful to write the elements of $\mathcal{I}(\theta)$ as:

$$\mathcal{I}(\theta)_{kl} = \delta_{kl}\theta_k^{-1} + \theta_r^{-1}, \tag{6}$$

where we used Kronecker delta symbol δ_{xy} that equals 1, if $x = y$, and 0, otherwise. □

4.1.3 The kernel

The last thing left to do is to combine our results to get the expression for the univariate kernel $K(x, y) = s^T(x;\theta)\mathcal{I}^{-1}(\theta)s(y;\theta)$. As the gradient $s(d;\theta)$ takes different forms depending on the value d, it is maybe the easiest to enumerate all the cases corresponding to different combinations for values x and y. After considering all the cases, it turns out that the kernel will take different forms depending on whether $x = y$ or $x \neq y$. This derivation is done explicitly in Appendix A. We will just state the final result here, which is also the statement of Lemma 1:

$$\begin{aligned} K(x, y;\theta) &= \delta_{xy}\theta_x^{-1} - 1 \\ &= \begin{cases} \frac{1-\theta_x}{\theta_x} & \text{if } x = y, \text{ and} \\ -1 & \text{if } x \neq y. \end{cases} \end{aligned}$$

We notice that dissimilar items are always dissimilar at level −1, but similar items are similar at level $[0, \infty]$. The similarity of a value to itself is greater for rare values (rare means that $\theta_x = P(D = x;\theta)$ is small).

4.2 Multivariate Fisher kernel

We will now move on to the derivation in the multivariate case, leaving again some details to Appendix B. As seen from Eq. (1), the log-likelihood function for Bayesian networks is:

$$\begin{aligned} \log P(d;\theta) &= \sum_{i=1}^{n}\sum_{j=1}^{q_i}\sum_{k=1}^{r_i} I_{d_i}(k) \cdot I_{\pi_i(d)}(j) \log \theta_{ijk} \\ &= \sum_{i=1}^{n}\sum_{j=1}^{q_i} I_{\pi_i(d)}(j) \sum_{k=1}^{r_i} I_{d_i}(k) \log \theta_{ijk}, \end{aligned} \tag{7}$$

where θ_{ijr_i} is never a real parameter but just a shorthand $\theta_{ijr_i} \equiv 1 - \sum_{k=1}^{r_i-1} \theta_{ijk}$. From Eq. (7), it is easy to compute the components of score using the calculations already made in the univariate case. This gives:

$$s(d;\theta)_{ijk} = I_{\pi_i(d)}(j)[I_{d_i}(k)\theta_{ijk}^{-1} - I_{d_i}(r_i)\theta_{ijr_i}^{-1}], \tag{8}$$

which also has the familiar form from the univariate case, apart from the extra indicator function due to the presence of parent variables. To get some intuition from the above equation, we can write it case by case as:

$$s(d;\theta)_{ijk} = \begin{cases} 0 & \text{if } \pi_i(d) \neq j, \\ 0 & \text{if } \pi_i(d) = j \wedge d_i \neq k < r_i \wedge d_i \neq r_i, \\ \theta_{ijk}^{-1} & \text{if } \pi_i(d) = j \wedge d_i = k < r_i, \text{ and} \\ -\theta_{ijr_i}^{-1} & \text{if } \pi_i(d) = j \wedge d_i = r_i. \end{cases}$$

4.2.1 Fisher information matrix

The derivation of Fisher Information matrix also proceeds along the same lines as the univariate case. The indicator function corresponding to the parent configuration results in the marginal probability of parent variables appearing in the formula. Details can be found in Appendix B. The end result is as follows:

$$\mathcal{I}(\theta)_{(xyz),(ijk)} = \delta_{ij,xy} P(\pi_i(D) = j)(\delta_{kz}\theta_{ijk}^{-1} + \theta_{ijr_i}^{-1}), \tag{9}$$

where $\delta_{ij,xy} = \delta_{ix} \cdot \delta_{jy}$ is generalization of Kronecker delta notation to a series of integers.

By looking at Eq. (9), we see that the matrix has a block diagonal structure. There is one block corresponding to each parent configuration for every variable. For variable D_i, the jth block is a $(r_i - 1) \times (r_i - 1)$ matrix, where on the diagonal we have values $\theta_{ijk}^{-1} + \theta_{ijr_i}^{-1}$ for $k = 1, \ldots, r_i - 1$ and outside the diagonal $\theta_{ijr_i}^{-1}$ (with all the values multiplied by $P(\pi_i(D) = j)$). So each block looks just like the matrix in the univariate case (see Eq. (4)), which we have already inverted.

4.2.2 Inverse of the Fisher information matrix

The inverse of a block diagonal matrix is obtained by inverting the blocks separately. Applying the univariate result for each of the blocks, we obtain that the ijth block has the form:

$$^{ij}[\mathcal{I}^{-1}(\theta)] = \frac{1}{P(\pi_i(D) = j)}[^{ij}A -^{ij} F],$$

where ^{ij}A is a diagonal matrix with entries $^{ij}A_{kk} = \theta_{ijk}$ and $^{ij}F_{kl} = \theta_{ijk}\theta_{ijl}$.

4.2.3 Fisher kernel

In calculating the $K(x, y;\theta) = s(x;\theta)^T \mathcal{I}^{-1}(\theta)\, s(y;\theta)$, the block diagonality of the $\mathcal{I}^{-1}(\theta)$ will save us a lot of computation. We therefore partition the score vector into sub-vectors $s(d;\theta)_i$ that contain partial derivatives for all the parameters $\theta_i = \cup_{j=1}^{q_i} \cup_{k=1}^{r_i-1} \{\theta_{ijk}\}$, and then further subdivide these into parts $s(d;\theta)_{ij}$.

Since the kernel $K(x, y;\theta)$ can be expressed as an inner product $\langle s(x;\theta)^T \mathcal{I}^{-1}(\theta),\ s(y;\theta)\rangle$, we may factor this sum as:

$$K(x, y;\theta) = \sum_{i=1}^{n} K_i(x, y;\theta),$$

where $K_i(x, y;\theta) = \langle [s(x;\theta)^T \mathcal{I}^{-1}(\theta)]_i,\ s(y;\theta)_i \rangle$. It is therefore sufficient to study the terms K_i.

Since in $s(x;\theta)_i$ only the entries corresponding to the indices $(i, \pi_i(x))$ may be nonzero, the same applies to the row vectors $[s(x;\theta)^T \mathcal{I}^{-1}(\theta)]_i$ due to the block diagonality of $\mathcal{I}^{-1}(\theta)$.

That automatically implies that if $\pi_i(x) \neq \pi_i(y)$ then $K_i(x, y;\theta) = 0$. If $\pi_i(x) = \pi_i(y) = j$, the computation reduces to the univariate case for $K(x_i, y_i;\theta_{ij})$, after taking the factors $P(\pi_i(D) = j)$ in a Fisher information matrix blocks into account. Here $\theta_{ij} = \{\theta_{ijk} \mid k = 1, \ldots, r_i\}$.

Summarizing these results, we get $K(x, y;\theta) = \sum_{i=1}^{n} K_i(x, y;\theta)$, where

$$K_i(x, y;\theta) = \begin{cases} 0 & \text{if } \pi_i(x) \neq \pi_i(y), \\ -\frac{1}{P(\pi_i(x);\theta)} & \text{if } \pi_i(x) = \pi_i(y) \\ & \wedge\ (x_i \neq y_i), \ \text{ and} \\ \frac{1}{P(\pi_i(x);\theta)} \frac{1-\theta_{i\pi_i(x)x_i}}{\theta_{i\pi_i(x)x_i}} & \text{if } \pi_i(x) = \pi_i(y) \\ & \wedge\ (x_i = y_i), \end{cases}$$

or more compactly

$$\begin{aligned} K_i(x, y;\theta) &= \frac{\delta_{\pi_i(x)\pi_i(y)}}{P(\pi_i(x);\theta)} \left[\frac{\delta_{x_i y_i}}{\theta_{i\pi_i(x)x_i}} - 1 \right] \\ &= \frac{\delta_{\pi_i(x)\pi_i(y)} K(x_i, y_i;\theta_{i\pi_i(x)})}{P(\pi_i(x);\theta)}. \end{aligned}$$

5 Invariance of Fisher kernel for equivalent network structures

A distribution may sometimes be represented by different parametrizations of different Bayesian network structures. We say that two network structures G_1 and G_2 are equivalent if for every parametrization θ^1 of G_1 there exists θ^2 for G_2 so that the networks represent the same probability distribution, and vice versa. Verma and Pearl (1990) showed that two network structures are equivalent if they have the same skeleton (i.e., they are the same if directed arcs are turned into undirected ones), and if they have the same set of V-structures (colliding arcs from two parents that are non-adjacent to each other).

5.1 Fisher kernel for non-equivalent structures

It is known that in general the Fisher kernel depends on the parametrization of the model. We will show by an example that this is also the case in Bayesian network models when the network structures are not equivalent.

Let us assume two independent binary variables, A and B, with following distributions: $P(A) = (0.7, 0.3)$ and $P(B) = (0.4, 0.6)$. If we present the joint distribution with a network without any arc, the Fisher similarity for the point $(A = 0, B = 0)$ is:

$$K((0,0),(0,0)) = \frac{1-0.7}{0.7} + \frac{1-0.4}{0.4} = 1\frac{13}{14}.$$

However, we may also express the same joint distribution in a network in which there is an arc from A to B. In this case, $P(A) = (0.7, 0.3)$, $P(B \mid A = 0) = P(B \mid A = 1) = (0.4, 0.6)$. Unlike in the previous case, the Fisher kernel term K_B now takes into account the probability of the parent A, which leads to a result different from the previous case:

$$K((0,0),(0,0)) = \frac{1-0.7}{0.7} + \frac{1}{0.7}\frac{1-0.4}{0.4} = 2\frac{4}{7}.$$

5.2 Fisher kernel for equivalent structures

Assume next that network structures of models M_1 and M_2 are equivalent, meaning that the two graphs imply exactly the same assertions of conditional independence. We would like to now show that in terms of the Fisher kernel, it does not matter which of these equivalent structures we use. To show this, we make use of the following property of Fisher kernels.

Theorem 2 *Let $\theta^1 \in \mathbb{R}^q$ and $\theta^2 \in \mathbb{R}^q$ denote two equivalent parametrizations for Fisher kernel K such that $\theta^2 = g(\theta^1)$ for some differentiable and invertible function $g : \mathbb{R}^q \to \mathbb{R}^q$. Now $K(x, y; \theta^1) = K(x, y; \theta^2)$, $\forall x, y$.*

In other words, Fisher kernel is invariant under one-to-one reparametrizations. For the proof, see Shawe-Taylor and Cristianini (2004). To show the invariance under equivalent Bayesian network structures, it suffices to prove the following statement:

Theorem 3 *Let $M_1 = (G_1, \theta^1)$ and $M_2 = (G_2, \theta^2)$ represent two Bayesian networks with equivalent structures. Let q denote the number of free parameters in M_1 and M_2. Given the parameters θ^1 for M_1, there exists a differentiable and invertible mapping $g : \mathbb{R}^q \to \mathbb{R}^q$ such that $\theta^2 = g(\theta^1)$.*

In fact, this result is used as an assumption by Heckerman and Geiger (1995), and stated to hold in our setting where conditional distributions in the Bayesian network are unrestricted multinomials. However, an explicit proof does not appear in the paper. We give one in Appendix C. This allows us to formulate our main result of this section:

Theorem 4 *Let $M_1 = (G_1, \theta^1)$ and $M_2 = (G_2, \theta^2)$ be two Bayesian networks with equivalent structures such that they represent the same distribution over D. Let x and y denote two observations on D. Now,*

$$K(x, y;\theta^1) = K(x, y;\theta^2),\ \forall x, y.$$

6 Comparing sets of observations

Having derived the Fisher kernel for measuring similarity among pairs of observations and studied its properties, we next show how this kernel easily generalizes to comparing two sets of observations. This set kernel also allows us to define a distance function which has a connection to a quantity called maximum mean discrepancy. We will also present a simple greedy algorithm which can be used to find a prototypical sets of points from a larger set of candidates as measured by the similarity induced by Fisher kernel.

6.1 The univariate set kernel

Let $X = (x^1, \dots, x^N)$ and $Y = (y^1, \dots, y^M)$, where x^l and y^m are i.i.d. observations on a categorical random variable. Since the log likelihood of i.i.d set of cases is the sum of member log likelihoods, it is rather easy to show that the Fisher kernel for two data sets is the average of Fisher kernels of pairs in the sets (Chappelier and Eckard 2009).

If we compare a data set X of size N with N_k occurrences of the value k, and another data set Y of size M which has M_k occurrences of the value k, we can collect the common terms of the sum and get:

$$\begin{aligned} K(X, Y;\theta) &= \frac{1}{NM} \sum_{l=1}^{N} \sum_{m=1}^{M} K(x^l, y^m) \\ &= \frac{1}{NM} \sum_{(l,m)=(1,1)}^{(N,M)} [\delta_{x^l y^m} \theta_{x^l}^{-1} - 1] \\ &= \sum_{k=1}^{r} \left[\frac{N_k}{N} \frac{M_k}{M} \theta_k^{-1} \right] - 1. \end{aligned}$$

6.2 The multivariate set kernel

The i.i.d assumption applied to n-dimensional data vectors x^l and y^i allows us to present the multivariate set kernel as:

$$K(X, Y;\theta) = \frac{1}{NM} \sum_{i=1}^{n} \sum_{(l,m)=(1,1)}^{(N,M)} K_i(x^l, y^m;\theta).$$

Now to express the inner sum more compactly, we need to obtain the counts of the variable configurations (as determined by the Bayesian network structure) in sets X and Y. We denote with N_{ijk} the number of data vectors x in X for which $x_i = k$ and the parent configuration $\pi_i(x)$ of is the jth possible one out of $q_i = \prod_{p \in \pi_i} r_p$. We write $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$. For corresponding counts in the set Y, we denote the counts with M_{ijk}. We may now develop the kernel as $K(X, Y;\theta) = \frac{1}{NM} \sum_{i=1}^{n} K_i(X, Y;\theta)$, where

$$\begin{aligned} K_i(X, Y;\theta) &= \sum_{(l,m)=(1,1)}^{(N,M)} \frac{\delta_{\pi_i(x^l)\pi_i(y^m)}}{P(\pi_i(x^l);\theta)} \left[\frac{\delta_{x_i^l y_i^m}}{\theta_{i\pi_i(x^l)x_i^l}} - 1 \right] \\ &= \sum_{(l,m)=(1,1)}^{(N,M)} \frac{\delta_{\pi_i(x^l)\pi_i(x^m)}}{P(\pi_i(x^l);\theta)} \frac{\delta_{x_i^l y_i^m}}{\theta_{i\pi_i(x^l)x_i^l}} - \frac{\delta_{\pi_i(x^l)\pi_i(y^m)}}{P(\pi_i(x^l);\theta)} \\ &= \sum_{j=1}^{q_i} \sum_{k=1}^{r_i} \frac{N_{ijk} M_{ijk}}{P(\pi_i(x) = j;\theta)\theta_{ijk}} - \sum_{j=1}^{q_i} \frac{N_{ij} M_{ij}}{P(\pi_i(x) = j;\theta)} \\ &= \sum_{j=1}^{q_i} \frac{1}{P(\pi_i(x) = j;\theta)} \left[\left[\sum_{k=1}^{r_i} \frac{N_{ijk} M_{ijk}}{\theta_{ijk}} \right] - N_{ij} M_{ij} \right]. \end{aligned}$$

We notice that the formula in brackets is a version of the univariate set kernel, so we could write:

$$K_i(X, Y;\theta) = \sum_{j=1}^{q_i} \frac{K(x_{ij}, y_{ij};\theta_{ij})}{P(\pi_i(x) = j;\theta)},$$

where x_{ij} denotes the vector of values of x_i in those vectors of X where parents of x_i have an index j. We need the convention that the value of set kernel is zero if either of the sets is empty.

6.3 Maximum mean discrepancy

Applying set kernel to Euclidean distance with the kernel trick (see, for instance, Shawe-Taylor and Cristianini 2004), we can compute (squared) distances between two data sets according to:

$$d(X, Y) = \frac{1}{N^2} \sum_{i,j} K(x^i, x^j) + \frac{1}{M^2} \sum_{i,j} K(y^i, y^j) - \frac{2}{NM} \sum_{i,j} K(x^i, y^j), \tag{10}$$

where $K(\cdot, \cdot)$ refers to the multivariate Fisher kernel with the dependence on parameters omitted for brevity.

We note that distance defined in (10) coincides with the empirical estimate of maximum mean discrepancy (MMD) (Gretton et al. 2012) with the kernel function being the Fisher kernel. MMD provides a test statistic for comparing whether two sets of data come from the same distribution. With the help of a kernel function and the corresponding reproducing kernel Hilbert space (RKHS), it is defined as the maximal difference in expectations over functions in the unit ball of this RKHS. MMD is expected to be close to zero when two data sets are drawn from the same distribution and larger otherwise.

We note that the theory developed by Gretton et al. (2012) builds on an assumption that RKHS (with the associated kernel $K(\cdot,\cdot)$ defined on elements of space $\mathcal{X}$) is universal. This technical assumption guarantees that the population version of the MMD is a metric for the probability distributions over $\mathcal{X}$. For instance, Gaussian kernel would define such RKHS and allow us to do two-sample testing for sets of observations in $\mathbb{R}^n$. We use the distance defined in (10) in our experiments, and refer to it as MMD, but leave further theoretical considerations (for instance, if the large sample bounds derived in Gretton et al. (2012) would be readily available to use with our Fisher kernel) to future work.

6.4 Greedy algorithm for optimizing MMD

We describe a simple greedy algorithm for finding a subset of points with small MMD from a larger candidate set. Given a set of N training examples X on variables D, and a Fisher kernel K, we try find a subset $X_S = \{x^i \mid i \in S\} \subset X$ of size k, where $S \subset \{1,\dots,N\} \equiv \lceil N \rceil$ so that $d(X_S, X)$ is minimized. If the Bayesian network from which the Fisher kernel is constructed is learned from X, this subset X_S minimizing the distance can be taken to be a representative sample of the full set X as deemed by the model. We will put this statement under closer scrutiny in the experiments where we make use of the algorithm. This type of an application of using MMD for selecting prototypical samples has been also considered in Kim et al. (2016).

The greedy optimization procedure is described in Algorithm 1. The algorithm proceeds by starting from the initially given set of points X_S. It then goes through the k points in X_S one at a time, changing the current data point to one that results in the minimal $d(X_S, X)$. The algorithm converges after it has looped through the k points once without being able to obtain a reduction in $d(X_S, X)$.

Algorithm 1 Greedy algorithm.

```
Require: Initial index set $S$, $|S| = k$
  $min\_d \leftarrow d(X_S, X)$
  while $S$ has changed do
    for $i \in S$ do
      $S \leftarrow S \setminus \{i\}$
      $j^* = \arg\min_{j \in \lceil N \rceil \setminus (S \cup \{i\})} d(X_{S \cup \{j\}}, X)$
      $best\_d \leftarrow d(X_{S \cup \{j^*\}}, X)$
      if $best\_d < min\_d$ then
        $S \leftarrow S \cup \{j^*\}$
        $min\_d \leftarrow best\_d$
      else
        $S \leftarrow S \cup \{i\}$
      end if
    end for
  end while
```

7 Experiments

The aim of our experiments is to illustrate the nature of the similarity that the Bayesian network Fisher kernel measures. By providing a valid kernel function, our kernel would be readily available for use in supervised tasks like classification or regression through any kernel-based method. The only requirement would be being able to come up with a Bayesian network over the domain. A straightforward approach would use the available training data to learn the structure of a Bayesian network and then its parameters. A score-based approach for learning the network might involve maximizing a penalized likelihood, and the parameters would be set to ones maximizing the likelihood given the structure. As noted by van der Maaten (2011), it is not guaranteed that the Fisher kernel extracted from this generatively trained model would be optimal for discriminative purposes. A generative model is trained to model the input data well, which would drive the gradients of training instances towards zero and not necessarily mean that these representations would be particularly good for discriminating between the labels of instances.

In our experiments, we use Fisher kernel as mean of gaining insight on the underlying Bayesian network by trying to identify sets of observation that are important for the model. In general, there has been lately a surge of interest in understanding and interpreting various machine learning models (Murdoch et al. 2019). As Fisher kernel measures similarity of data items from the point of view of the model, it provides a natural tool for various tasks related to model interpretation (Khanna et al. 2019).

7.1 Data summarization

We study how Fisher kernel performs in summarizing data. This experiment is inspired by the data summarization experiment in Khanna et al. (2019). Outline of the experiment is as follows. We first learn a Bayesian network M from data, build a Fisher kernel and then employ Algorithm 1 to find a subset of points that are important for M. To quantify "important", we re-train the parameters of the model using the found subset and evaluate predictive performance of the newly trained model with hold-out data. We consider different approaches for learning the network and vary the evaluation metric.

We use junction tree algorithm to obtain the parent probabilities required in the Fisher kernel. In case the constructed junction tree has a tree width larger than 10, the parent probabilities are computed by sampling data (10^6 samples) from the Bayesian network and then estimating probabilities as empirical frequencies (with pseudocount one to avoid possible zero probabilities).

As an alternative to Fisher kernel-based prototype selection, we also consider a simple heuristic based on χ^2 test. More in detail, when evaluating goodness of X_S, we look at data on each variable in X_S independently, and compute a p value from a χ^2 test, where observed frequencies are computed from X_S and the expected frequencies from the whole set X. The degrees-of-freedom is one less the number of observed categories. The best X_S would be the one maximizing $\sum_{i=1}^{n} \log \hat{p}_i$, where $\hat{p}_i$ is the p value corresponding to the test on the ith variable in X_S (and in X). Intuition here is that a large p value indicates that the observed frequencies match the expected ones. Algorithm 1 is also used to optimize this heuristic (to be precise, the objective function is $-\sum_{i=1}^{n} \log \hat{p}_i$). Also as a baseline, we show results for randomly selecting the subset X_S.

7.1.1 Network learned using hill climb

In this experiment, we considered three data sets found in UCI repository: letter ($N = 20{,}000$, $n = 17$), nursery ($N = 12{,}960$, $n = 9$) and waveform-5000 ($N = 5000$, $n = 41$). The class variable present in each data set was treated as any other categorical feature. Possible continuous features were discretized to (at most) 4 bins and data was split to training and test sets (50%/50%). We then learned a Bayesian network using hill climbing[1] with BIC as a scoring function using the training data. The parameters of the network were set to smoothed (pseudo-count one) maximum likelihood estimates. The Fisher kernel was constructed using this network. We measured full set accuracy by computing per sample negative log likelihood for the test set. We then proceeded to finding representative subsets from the training data. We considered four different values for subset size k and sampled 1000 of such sets. For the each sampled set, we recorded Fisher kernel MMD distance, χ^2 heuristic criterion and test set log likelihood corresponding to model with parameters learned from this set. We ran Algorithm 1 for each k with MMD distance and χ^2 heuristic. The best of the sampled subsets (according to each criterion) was used as a starting point for the greedy algorithm.

Results are shown in Fig. 1. Random mean represents the average log likelihood computed from the randomly sampled subsets with error bar showing the standard deviation. We can see that the Fisher kernel-based MMD finds more representative subsets than the compared criteria as measured by the log likelihood.

To give a rough idea of the running times[2] of the methods in Fig. 1, it took 17, 32 and 94 seconds to compute the parent probabilities and the Fisher kernel evaluations for training data points in data sets waveform-5000, nursery and letter, respectively. Running the greedy algorithm with the MMD distance for different subsets sizes took from 8 s ($k = 200$) to 3 min ($k = 1000$) in data set waveform-5000. For nursery and letter, the corresponding times were from 0.2 to 5 min and from 0.7 to 19 min, respectively. For the greedy χ^2 method, the corresponding times ranged (in minutes) from 1 to 4, from 0.2 to 0.8 and from 1 to 5, respectively. To find the starting points for the greedy algorithms, sampling 1000 random subsets, recording both the MMD and χ^2 statistics took from 9 to 18 s in waveform-5000, from 20 to 34 seconds in nursery and from 45 to 101 s in letter.

[1] We used the implementation found in R-package 'bnlearn'.

[2] Using a quad-CPU 3.10 GHz desktop computer. Methods were implemented in Python.

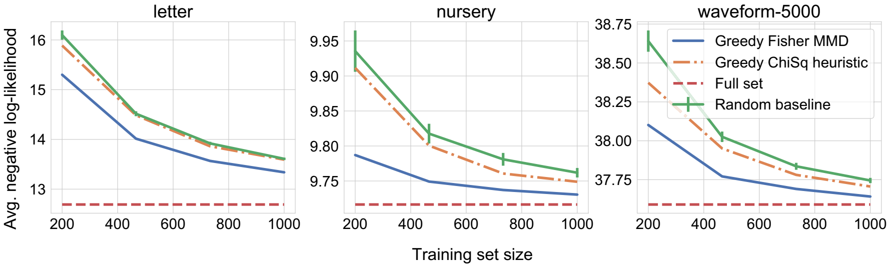

Fig. 1 Negative log likelihoods for different subset sizes and criterions in UCI datasets. Greedy methods refer to Algorithm 1 with the corresponding criterion. Full set represents the accuracy of the model trained with all the available data and random baseline is averaged result over models retrained with random subsets

7.1.2 Bayesian network classifiers

Next, we considered classification experiments with the aforementioned three data sets. Outline of the experiments is exactly the same, except for the network learning and the evaluation metric. After splitting the data in half, we used training data to learn a tree-augmented naive Bayes (TAN) classifier (Friedman et al. 1997) and K-dependence Bayesian (KDB) classifier (Sahami 1996). These both classes of classifiers can be represented as Bayesian networks over the class variable and the features. Each feature is connected to the class variable, and the classifiers differ in their way of modeling the dependency structure of the features. TAN uses conditional mutual information (given class variable) to find the best tree structure for the distribution of features. KDB also makes use of conditional mutual information but learns the structure in a greedy manner, connecting each feature to at most K other features. We used $K = 3$ in the experiments and parameters of both the classifiers were smoothed by adding one pseudocount to empirical frequencies. Given the network structures and the Fisher kernel, the representative subsets were found using the procedure described in the last subsection. Goodness of the found subset was evaluated by computing the classification error and log loss (per sample) on the test set.

The results for the TAN and KDB are presented in Figs. 2 and 3, respectively. We can see that in general the Fisher kernel-based selection criterion is able to find considerably better subsets in terms of both evaluation metrics for both of the classifiers. Especially with TAN, the classification accuracy even with the smallest subset size is close to the full set accuracy in nursery and waveform-5000 data sets. With KDB in data set nursery, the smallest subset sizes produce results close to or worse than the random sampling. On the other hand, TAN does not seem to have similar problems. One thing explaining this could be that KDB (with $K = 3$) contains more parameters, so naturally more data points are required to estimate the parameters accurately. In general, the evaluation metrics considered in this experiment are focused only on the class variable, whereas the similarity measured by Fisher kernel takes into account all the variables through their corresponding parameters. From this point of view, the

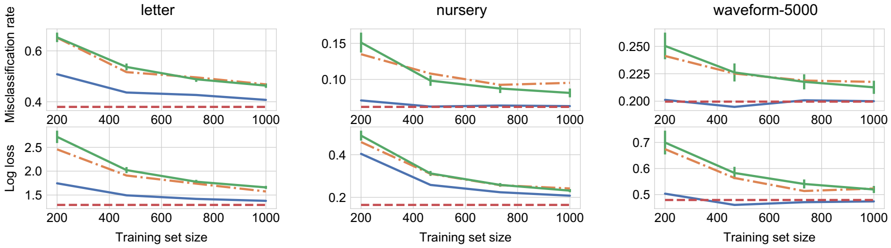

Fig. 2 Results on classification experiments for TAN. Misclassification rate on top and log loss on the bottom. Labeling of the lines follows Fig 1

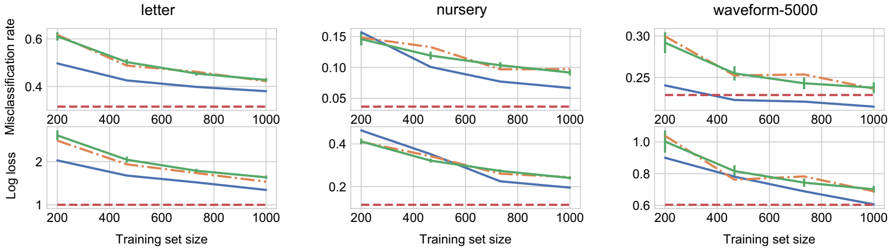

Fig. 3 Results on classification experiments for KDB. Missclassification rate on top and log loss on the bottom. Labeling of the lines follows Fig 1

log likelihood used in the previous subsection provides a more natural evaluation criterion. Nevertheless, this experiment demonstrates that Fisher kernel is in most cases able to extract representative sets for discriminative purposes, too.

8 Conclusion

We have presented a Fisher kernel for discrete Bayesian networks and shown that it makes sense to use it with networks learned from data even if the exact DAG structure is not in general identifiable. We also presented formulas for comparing two sets of cases and connected the resulting distance to maximum mean discrepancy. In the experiments, we illustrated the nature of the similarity the Fisher kernel measures using examples that draw their motivation from interpreting Bayesian network models.

The obvious future work would contain testing this kernel in a huge number of available kernel-based methods and studying further the model interpretation aspect by extending the current experiments.

We have assumed that our model does not contain hidden variables. However, when the modeled vector $z = (x, h)$ is partitioned into an observed part x and a hidden part h, one can easily define a marginalized kernel (Tsuda et al. 2002)

$K(x,x') = \sum_{h,h'} P(h|x)P(h'|x')K(z,z')$, which can be computed efficiently when the hidden dimension is not too large, since $P(h|x) \propto P(x,h)$, which is easy to compute for Bayesian networks.

Acknowledgements Open access funding provided by University of Helsinki including Helsinki University Central Hospital.

Compliance with ethical standards

Conflicts of interest The authors declare that they have no conflict of interest.

A details on derivation of the univariate Fisher kernel

A.1 Fisher information matrix

Computing the Fisher information matrix requires the partial derivatives of the components of $s(d;\theta)_k$ (3). Noting that $\frac{\partial \theta_r^{-1}}{\partial \theta_l} = \theta_r^{-2}$, we get:

$$\begin{aligned}\frac{\partial s(d;\theta)_k}{\partial \theta_l} &= \frac{\partial}{\partial \theta_l}\left(I_d(k)\theta_k^{-1} - I_d(r)\theta_r^{-1}\right)\\ &= -\delta_{kl} I_d(k)\theta_k^{-2} - I_d(r)\theta_r^{-2},\end{aligned}$$

where $\delta_{xy} = 1$, if $x = y$, and 0, otherwise. Since $\mathbb{E}_d[I_d(k)] = P(D = k;\theta) = \theta_k$,

$$\begin{aligned}\mathcal{I}(\theta)_{kl} &= -\mathbb{E}_d[-\delta_{kl} I_d(k)\theta_k^{-2} - I_d(r)\theta_r^{-2}]\\ &= \delta_{kl}\theta_k^{-1} + \theta_r^{-1} = \theta_r^{-1}(\delta_{kl}\theta_k^{-1}\theta_r + 1).\end{aligned}$$

This is the same as Eq. (6) and agrees with the matrix presented in Eq. (4).

A.2 Inverse of Fisher information

Looking at the (5), we see that $\mathcal{I}^{-1}(\theta)_{ij} = \delta_{ij}\theta_i - \theta_i\theta_j$. Using the equation above (the same as Eq. (6)), we get that

$$\begin{aligned}
(\mathcal{I}(\theta)\mathcal{I}^{-1}(\theta))_{ij} &= \sum_{k=1}^{r-1} \mathcal{I}(\theta)_{ik}\mathcal{I}^{-1}(\theta)_{kj} \\
&= \sum_{k=1}^{r-1} \left(\delta_{ik}\theta_i^{-1} + \theta_r^{-1}\right)\left(\delta_{kj}\theta_k - \theta_k\theta_j\right) \\
&= \sum_{k=1}^{r-1} \left(\delta_{ik}\delta_{kj}\theta_i^{-1}\theta_k - \delta_{ik}\theta_i^{-1}\theta_k\theta_j + \delta_{kj}\theta_r^{-1}\theta_k - \theta_k\theta_j\theta_r^{-1}\right) \\
&= \delta_{ij} - \theta_j + \theta_j\theta_r^{-1} - (1-\theta_r)\theta_j\theta_r^{-1} = \delta_{ij},
\end{aligned}$$

which verifies that (5) is the inverse of Fisher information.

A.3 The kernel

We start by first exploring the row vector $s(x;\theta)^T\mathcal{I}^{-1}(\theta)$. We recall, however, that $s(x;\theta)$ looks very different when $x = r$ and $x < r$. It is also useful to write $\mathcal{I}^{-1}(\theta) = A - F$, where $A = \mathrm{diag}(\theta)$ and $F = \theta\theta^T$. We use notation $A_{k\cdot}$ to denote the kth row of matrix A.

Now, if $x = r$, $s(x;\theta) = -\theta_r^{-1}\mathbf{1}$, so multiplying with its transpose sums the rows of the matrix and multiplies the results with $-\theta_r^{-1}$, and we get

$$\begin{aligned}
s(x;\theta)^T\mathcal{I}^{-1}(\theta) &= s(x;\theta)^T A - s(x;\theta)^T F \\
&= -\theta_r^{-1}\sum_{k=1}^{r-1} A_{k\cdot} + \theta_r^{-1}\sum_{k=1}^{r-1} F_{k\cdot} \\
&= -\theta_r^{-1}\theta^T + \theta_r^{-1}(1-\theta_r)\theta^T = -\theta^T.
\end{aligned}$$

When $x < r$, the $s(x;\theta) = \theta_x^{-1}e_x$, so the $s(x;\theta)^T$ picks the xth row from matrices and multiplies it by θ_x^{-1};

$$\begin{aligned}
s(x;\theta)^T\mathcal{I}^{-1}(\theta) &= s(x;\theta)^T A - s(x;\theta)^T F \\
&= \theta_x^{-1}(\theta_x e_x^T) - \theta_x^{-1}(\theta_x\theta^T) = (e_x - \theta)^T.
\end{aligned}$$

To work our way to full $s(x;\theta)^T\mathcal{I}^{-1}(\theta)\, s(y;\theta)$ it is easiest to continue by cases:

(1) If $x = y = r$, we get

$$\begin{aligned}
s(x;\theta)^T\mathcal{I}^{-1}(\theta)\, s(y;\theta) &= (-\theta^T)(-\theta_r^{-1}\mathbf{1}) \;=\; \theta_r^{-1}\theta^T\mathbf{1} \\
&= \theta_r^{-1}(1-\theta_r) = \frac{1-\theta_r}{\theta_r}.
\end{aligned}$$

(2) If $x < r = y$, we get

$$\begin{aligned} s(x;\theta)^T\mathcal{I}^{-1}(\theta)\,s(y;\theta) &= (e_x - \theta)^T(-\theta_r^{-1}\mathbf{1}) \\ &= -\theta_r^{-1}(e_x^T\mathbf{1} - \theta^T\mathbf{1}) \\ &= -\theta_r^{-1}(1 - (1 - \theta_r)) = -1. \end{aligned}$$

(3) Due to the symmetry of the Fisher information, for the case $y < r = x$:

$$s(x;\theta)^T\mathcal{I}^{-1}(\theta)\,s(y;\theta) = -1. \tag{11}$$

(4) Lastly, if $x < r$ and $y < r$, we get:

$$\begin{aligned} s(x;\theta)^T\mathcal{I}^{-1}(\theta)\,s(y;\theta) &= (e_x^T - \theta^T)(\theta_y^{-1}e_y) \\ &= \theta_y^{-1}[e_x^Te_y - \theta^Te_y] \\ &= \theta_y^{-1}[\delta_{xy} - \theta_y] \\ &= \begin{cases} \frac{1-\theta_y}{\theta_y} & \text{if } x = y, \text{ and} \\ -1 & \text{if } x \neq y. \end{cases} \end{aligned}$$

Collecting the results above gives as finally

$$K(x,y;\theta) = \begin{cases} \frac{1-\theta_x}{\theta_x} & \text{if } x = y, \text{ and} \\ -1 & \text{if } x \neq y. \end{cases}$$

B Details on derivation of the multivariate Fisher kernel

B.1 Fisher information matrix

Computing the second derivatives of the log likelihood, thus the partial derivatives of

$$s(d;\theta)_{ijk} = I_{\pi_i(d)}(j)[I_{d_i}(k)\theta_{ijk}^{-1} - I_{d_i}(r_i)\theta_{ijr_i}^{-1}]$$

is straightforward, once we note that $\frac{\partial\theta_{ijr_i}^{-1}}{\partial\theta_{ijz}} = \theta_{ijr_i}^{-2}$. Extending the Kronecker delta notation $\delta_{s,t}$ to test equality of series s and t of integers, the second derivatives are

$$\begin{aligned} \frac{\partial s(d;\theta)_{ijk}}{\partial\theta_{xyz}} &= I_{\pi_i(d)}(j)[-I_{d_i}(k)\delta_{ijk,xyz}\theta_{ijk}^{-2} - I_{d_i}(r_i)\delta_{ij,xy}\theta_{ijr_i}^{-2}] \\ &= I_{\pi_i(d)}(j)\delta_{ij,xy}[-I_{d_i}(k)\delta_{kz}\theta_{ijk}^{-2} - I_{d_i}(r_i)\theta_{ijr_i}^{-2}], \end{aligned}$$

where we used $\delta_{ijk,xyz} = \delta_{ij,xy} \cdot \delta_{kz}$. Then noting that

$$\mathbb{E}_d[I_{d_i}(k) \cdot I_{\pi_i(d)}(j)] = P(D_i = k, \pi_i(D) = j) = P(\pi_i(D) = j)P(D_i = k|\pi_i(D) = j)$$
$$= P(\pi_i(D) = j) \cdot \theta_{ijk},$$

we get

$$\mathcal{I}(\theta)_{(xyz),(ijk)} = -\mathbb{E}_d \left[\frac{\partial s(d;\theta)_{ijk}}{\partial \theta_{xyz}} \right] = \delta_{ij,xy}\delta_{kz}P(\pi_i(D) = j)P(D_i = k|\pi_i(D) = j)\theta_{ijk}^{-2}$$
$$+ \delta_{ij,xy}P(\pi_i(D) = j)P(D_i = r_i|\pi_i(D) = j)\theta_{ijr_i}^{-2}$$
$$= \delta_{ij,xy}P(\pi_i(D) = j)(\delta_{kz}\theta_{ijk}^{-1} + \theta_{ijr_i}^{-1}).$$

C Proof of Theorem 3

Proof Chickering showed that any equivalent structure can be reached from another by a series of covered arc reversal operations without leaving the equivalence class (Chickering 1995). An arc $D_i \rightarrow D_j$ is covered if $\pi_j(D) = \pi_i(D) \cup \{D_i\}$. That is, the nodes share the exactly the same parent set, with the exception that D_i is not its own parent. Without loss of generality, we can now assume that G_1 and G_2 differ by a single covered arc reversal. Assume that this covered arc between D_i and D_j in G_1 is reversed, creating an arc $D_j \rightarrow D_i$ in G_2. Let Z denote the common parents of D_i and D_j. We treat Z as a single categorical variable whose cardinality is the number of all the possible parent combinations. Reversing the arc affects only the parameters defining the conditional distributions of D_i (θ_i^1 and θ_i^2) and D_j (θ_j^1 and θ_j^2). All the other parameters in θ^1 can be mapped to θ^2 using an identity function.

If we multiply the conditional distributions of D_i and D_j together, we get the distribution $p(D_i, D_j \mid Z)$. Given $Z = z$, we can treat this as a complete Bayesian network (does not imply any conditional independencies) over D_i and D_j under both M_1 and M_2. Starting from parameters θ^1 we can get the parameters for the joint $P(D_i, D_j \mid Z = z)$ uniquely. The joint parameters refer here to probabilities $P(D_i = x, D_j = y \mid Z = z)$. Mapping is simply given by the chain rule and it is invertible. The Jacobian related to the inverse mapping is derived by Heckerman et al. (1995) (Theorem 10). The Jacobian is easily seen to be non-zero assuming $\theta_{ijk}^1 > 0\ \forall i, j, k$. To be more precise, Theorem 10 provides the Jacobian related to the mapping from joint parameters to parameters of any complete Bayesian network model over the same domain. In our case, this means, that after bijectively mapping θ^1 to joint parameters, we can continue and map the joint parameters bijectively to parameters θ^2 of M_2. This holds for any z, and as a composition of two bijections is also a bijection, we have proved our claim. □

References

Bernardo JM, Smith AFM (1994) Bayesian theory. Wiley, Chichester

Boriah S, Chandola V, Kumar V (2008) Similarity measures for categorical data: a comparative evaluation. In: SIAM data mining conference, pp 243–254

Chappelier JC, Eckard E (2009) PLSI: The true Fisher kernel and beyond. In: Machine learning and knowledge discovery in databases. Springer, Berlin, pp 195–210

Chickering DM (1995) A transformational characterization of equivalent bayesian network structures. In: Proceedings of the eleventh annual conference on uncertainty in artificial intelligence. Morgan Kaufmann, pp 87–98

Cooper GF (1990) The computational complexity of probabilistic inference using Bayesian belief networks. Artif. Intell. 42(2–3):393–405

Cover TM, Thomas JA (2006) Elements of information theory (Wiley series in telecommunications and signal processing). Wiley-Interscience, New Jersey

Desai A, Singh H, Pudi V (2011) DISC: Data-intensive similarity measure for categorical data. In: Advances in knowledge discovery and data mining: 15th Pacific-Asia conference. Springer, Berlin, pp 469–481

Friedman N, Geiger D, Goldszmidt M (1997) Bayesian network classifiers. Mach Learn 29(2):131–163

Gretton A, Borgwardt KM, Rasch MJ, Schölkopf B, Smola A (2012) A kernel two-sample test. JMLR 13:723–773

Heckerman D, Geiger D (1995) Likelihoods and priors for Bayesian networks. Tech. rep., MSR-TR-95-54, Microsoft Research, Redmond

Heckerman D, Geiger D, Chickering DM (1995) Learning bayesian networks: the combination of knowledge and statistical data. Mach Learn 20(3):197–243

Jaakkola T, Haussler D (1998) Exploiting generative models in discriminative classifiers. In: Advances in neural information processing systems, vol 11. MIT Press, pp 487–493

Jaakkola TS, Diekhans M, Haussler D (2000) A discriminative framework for detecting remote protein homologies. J Comput Biol 7(1–2):95–114

Jebara T, Kondor R, Howard A (2004) Probability product kernels. J Mach Learn Res 5:819–844

Khanna R, Kim B, Ghosh J, Koyejo S (2019) Interpreting black box predictions using Fisher kernels. Proc Mach Learn Res PMLR 89:3382–3390

Kim B, Khanna R, Koyejo OO (2016) Examples are not enough, learn to criticize! criticism for interpretability. In: Advances in neural information processing systems, vol 29. Curran Associates, Inc., pp 2280–2288

Lafferty J, Lebanon G (2005) Diffusion kernels on statistical manifolds. J Mach Learn Res 6:129–163

Lauritzen SL, Spiegelhalter DJ (1988) Local computations with probabilities on graphical structures and their application to expert systems. J R Stat Soc Ser B (Methodol) 50(2):157–194

van der Maaten L (2011) Learning discriminative Fisher kernels. In: Proceedings of the 28th international conference on international conference on machine learning, Omnipress, pp 217–224

McCane B, Albert M (2008) Distance functions for categorical and mixed variables. Pattern Recogn Lett 29(7):986–993

Miller KS (1981) On the inverse of the sum of matrices. Math Mag 54(2):67–72

Murdoch WJ, Singh C, Kumbier K, Abbasi-Asl R, Yu B (2019) Interpretable machine learning: definitions, methods, and applications. arXiv e-prints 1901.04592

Niitsuma H, Okada T (2005) Covariance and PCA for categorical variables. In: Ho TB, Cheung D, Liu H (eds) Advances in knowledge discovery and data mining: 9th Pacific-Asia conference. Springer, Berlin, pp 523–528

Pearl J (1988) Probabilistic reasoning in intelligent systems: networks of plausible inference. Morgan Kaufmann Publishers, San Mateo

Ring M, Otto F, Becker M, Niebler T, Landes D, Hotho A (2015) ConDist: a context-driven categorical distance measure. In: Machine learning and knowledge discovery in databases: European conference. Springer International Publishing, pp 251–266

Sahami M (1996) Learning limited dependence bayesian classifiers. In: Proceedings of the second international conference on knowledge discovery and data mining. AAAI Press, pp 335–338

Sewell M (2011) The Fisher kernel: a brief review. Tech. rep., UCL Department of Computer Science

Shawe-Taylor J, Cristianini N (2004) Kernel methods for pattern analysis. Cambridge University Press, New York

Tsuda K, Kin T, Asai K (2002) Marginalized kernels for biological sequences. Bioinformatics 18(Suppl 1):268–275

Verma T, Pearl J (1990) Equivalence and synthesis of causal models. In: Proceedings of the eleventh sixth annual conference on uncertainty in artificial intelligence. Elsevier Science, pp 255–268

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.