CS281B/Stat241B: Advanced Topics in Learning & Decision Making

Fisher Kernels and Semidefinite Programming

Lecturer: Michael I. Jordan *Scribe: Barbara Engelhardt*

In this lecture, we introduce Fisher kernels to continue the discussion of kernels based on probability models. Switching topics, an introduction to semidefinite programming is motivated by an interest in combining multiple kernels.

1 Fisher Kernels

In statistics, the *score function* is the first derivative of the log likelihood, $M = \{P_\theta(x) : \theta \in \Theta\}$.

To calculate the score, fix the parameters $\theta = \theta^0$, and take the gradient of the log likelihood with respect to the model parameters,

$$g(\theta^0, x) = \frac{\partial \log p_{\theta^0}(x)}{\partial \theta},$$

where x is a specific value of the observed data. Note that the score is a vector, one component for each parameter, and the log likelihood is a function of a single data point x at a fixed parameter θ^0. The score vector captures the change to the fixed parameter to accommodate the new data point x. For each x, the score (the change in parameters to accommodate that x) is generally different.

1.1 Fisher Information

The Fisher information matrix (I) is the expectation of the negative second derivative of the log likelihood, also known as the (negative) Hessian matrix. It can be computed using the outer product of the score function:

$$I \doteq E_{\theta^0}[g(\theta^0, x), g(\theta^0, x)^T],$$

where in the discrete case the expectation is a sum:

$$I = \sum_{x \in X} p_{\theta^0}(x) g(\theta^0, x) g(\theta^0, x)^T.$$

1.2 Fisher Kernel

The Fisher kernel is defined as

$$\kappa(x, x') = g(\theta^0, x)^T I^{-1} g(\theta^0, x').$$

When the Fisher information matrix is based on the empirical expectation, the Fisher kernel is equivalent to whitening the score function.

It is a simple check to observe that the kernel matrix is positive semidefinite, using the definitions of I and the kernel itself, and using the fact that the expectation of a PSD matrix must also be PSD.

In this setting, x can be any type of data that you can assign a probability distribution to, such as a vector, different length strings, trees, etc. It is the score function $g(\theta^0, x)$ that transforms the data to a fixed length vector, so the data are pairwise comparable.

In practice, the Fisher information matrix is often replaced by the identity matrix to define the kernel $\kappa(x, x') = g(\theta^0, x)^T g(\theta^0, x')$, since it is time consuming to sum over every x to compute the Fisher information. Obviously, this is still a kernel.

One key characteristic of the Fisher kernel is that it is invariant to reparameterization, for invertible and differentiable reparameterizations. A simple proof of this fact can be found on page 429 of the course reader (chapter 12).

The general intuition behind the Fisher kernel is that the score function represents the signature that different x will have on the parameters. In particular, how should we change the parameters to accommodate the current data x? The Fisher kernel measures the similarity between data points by measuring the similarity of their impact on the parameters with respect to the score function.

1.3 Relationship to Marginal Kernels

In Tsuda's paper [4], they present an argument showing that the Fisher kernel is a special case of a marginalized kernel in a model with latent variables.

$$P_\theta(x) = \sum_h P_\theta(x, h)$$

$$\begin{aligned}
g(\theta^0, x) &= \frac{\partial}{\partial\theta} \log p_{\theta^0}(x) \\
&= \frac{\sum_h \frac{\partial}{\partial\theta} p_{\theta^0}(x, h)}{p_{\theta^0}(x)} \\
&= \frac{\sum_h p_{\theta^0}(x, h) \frac{\partial}{\partial\theta} p_{\theta^0}(x, h)}{p_{\theta^0}(x) p_{\theta^0}(x, h)} \\
&= \sum_h p_{\theta^0}(h|x) \frac{\partial}{\partial\theta} \log p_{\theta^0}(x, h).
\end{aligned}$$

As a kernel,

$$\kappa(x, x') = \sum_h \sum_{h'} p_{\theta^0}(h|x) p_{\theta^0}(h'|x') \frac{\partial}{\partial\theta} \log p_{\theta^0}(x, h)^T I^{-1} \frac{\partial}{\partial\theta} \log p_{\theta^0}(x', h')$$

$$\kappa(x, x') = \sum_h \sum_{h'} p_{\theta^0}(h|x) p_{\theta^0}(h'|x') \kappa(z, z'),$$

where $z = (x, h)$, $z' = (x', h')$. In other words, the Fisher kernel is an instance of a marginalized kernel. Note that there is no particular reason to restrict yourself to the Fisher kernel if you are using a marginalized kernel. In general, it is preferable to separate the model from the kernel as much as possible, since the model is imperfect and reliance on an imperfect model can cause problems.

In general, deriving kernels based on graphical models is still a large open area. Strong results have already been produced in some biological applications for HMM-type models, and there are some nice opportunities in document analysis (specifically the latent Dirichlet allocation model) and other areas.

2 Semidefinite Programming

There are a number of reasons to want to combine kernels. In particular, if you are not sure which one you should use, or multiple kernels are realistic for your data, or your data is of many different types, or most generally when you do not want to select one kernel from a handful of possible kernels.

How should multiple kernels be combined? From previous lectures, we have established an algebra of kernels, which gives us that sums and products of kernels are kernels, and also multiplying a kernel by a non-negative scalar is a kernel.

Intuition suggests that we might combine kernels as follows:

$$\kappa(x, x') = \sum_{i=1}^{q} \mu_i \kappa_i(x, x')$$

where the μ_i are to be estimated from data. This can be done using semidefinite programming (SDP).

2.1 Introduction to SDP

C is called a *cone* when, for all $X_i \in C$, then $\sum_i \mu_i X_i \in C$ for $\mu_i \geq 0$, or the conic combination of any set of points in C is also in C.

The set of positive semidefinite (PSD) matrices form a cone. If $K = \sum_i \mu_i K_i$, and K_i is PSD, then $z^T K_i z \geq 0$ for all i, z, and

$$\begin{aligned} z^T K z &= z^T \sum_i \mu_i K_i z \\ &= \sum_i \mu_i z^T K_i z \geq 0. \end{aligned}$$

SDP is the optimization of a linear function over the cone of PSD matrices. A good reference for additional information on combining kernels using SDP, see [2].

A *linear matrix inequality* (LMI) is:

$$F(u) = F_0 + u_1 F_1 + ... + u_q F_q \preceq 0,$$

where the symbol $\preceq 0$ denotes a negative semidefinite matrix. Here all of the F_i are symmetric. The SDP can be defined using LMI constraints:

$$\begin{aligned} &\min_u c^T u \\ \text{subject to } &F^j(u) = F_0^j + u_1 F_1^j + ... + u_q F_q^j \preceq 0 \;\; j = 1, ..., L \\ &Au = b \end{aligned}$$

where $F^j(u)$ is a block matrix composed of all of the individual F_i matrices. LMI are equivalent to an infinite number of scalar constraints, i.e., $F(u) \preceq 0$ if and only if $z^T F(u) z \leq 0 \; \forall z$. Equivalently, the LMI constraint can be written as $\text{tr}[F(u)Z] \leq 0 \; \forall Z$, where Z is a symmetric PSD matrix.

Proof: For $Z = V \Lambda V^T$, where $\lambda_i \geq 0$.

$$\begin{aligned} \text{tr}[F(u)Z] &= \text{tr}[F(u) V \Lambda V^T] = \text{tr}[V^T F(u) V \Lambda] \\ &= \sum_i v_i^T F(u) v_i \lambda_i \geq 0 \Leftrightarrow F(u) \preceq 0. \end{aligned}$$

Two good books on convex optimization are [3] and [1], where the latter can still be downloaded from Boyd's website in PDF.

SDPs can be solved using interior point methods. These methods are reasonably fast and are provably polynomial for the SDP. Available software includes the SeDuMi package, which is free, and MOSEK which costs money.

Quadratic programming is a special case of SDP, as is quadratically constrained quadratic programming (QCQP).

2.2 Duality for SDP's

In order to construct the dual problem for an SDP, we can use Lagrangian duality as before. We again need to check whether it is acceptable to swap the minimization and maximization in the Lagrangian equation. For the LMI constraint, the Lagrangian multiplier is a matrix Z:

$$\mathcal{L}(u, Z) = c^T u + \text{tr}[Z F(u)],$$

so $\max_{Z \succeq 0} \mathcal{L}(u, Z) = c^T u$ only when $F(u) \preceq 0$, and otherwise $\max_{Z \succeq 0} \mathcal{L}(u, Z) = \infty$, since the elements of Z are not bounded in the positive direction.

To recover the primal problem, $\min_u \max_{Z \succeq 0} \mathcal{L}(u, Z)$.

The dual problem is as follows:

$$g(Z) \doteq \min_u \mathcal{L}(u, Z) = c^T u + \text{tr}[Z F_0] + \sum_{i=1}^{q} u_i \text{tr}[Z F_i]$$

$$= \begin{cases} \text{tr}[Z F_0] & tr(Z F_i) = -c \\ -\infty & \text{otherwise} \end{cases}$$

By Slater's condition, there is no duality gap here. In order to show that, it is a worthwhile exercise to prove these constraints are convex, then check Slater's condition. The dual problem is

$$\max \operatorname{tr}[ZF_0]$$
$$\text{subject to } \operatorname{tr}[ZF_i] = -c_i \ \forall i$$
$$Z \succeq 0.$$

Note that the dual problem is also an SDP, since $Z \succeq 0$ is an LMI constraint and the objective function and other constraints are all linear. In order to solve the SDP, normally the dual problem is solved instead of the primal.

2.3 Schur Complement

Let $M = \begin{pmatrix} A & B \\ C & D \end{pmatrix}$, then the Schur complement is $D - CA^{-1}B$.

An important theorem involving the Schur complement which will be useful to us in combining kernels is the following,

$M = \begin{pmatrix} A & B \\ C & D \end{pmatrix} \succeq 0$ if and only if $A \succeq 0$ and $D - CA^{-1}B \succeq 0$.

A similar theorem exists for the invertable property of the matrices. Further discussion of this theorem and how it relates to combining kernels is in [2].

3 Bibliography

References

[1] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[2] G.R.G. Lanckriet, N. Cristianini, L. El Ghaoui, P.L. Bartlett, and M.I. Jordan. Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research*, 5:27–72, 2004.

[3] R. Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

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