Q-MKL: Matrix-induced Regularization in Multi-modality Learning for Neuroimaging of Alzheimer's Disease *

Chris Hinrichs[†‡] **Vikas Singh**[†‡] **Jiming Peng**[§] **Sterling C. Johnson**[†‡]

[†]**University of Wisconsin** Madison, WI [§]**University of Illinois** Urbana-Champaign, IL [‡]**Geriatric Research Education & Clinical Center** Wm. S. Middleton Memorial VA Hospital, Madison, WI

{ hinrichs@cs, vsingh@biostat, scj@medicine }.wisc.edu pengj@illinois.edu

1 Successive approach using Newton's method

Here, we describe a successive method which may be more appropriate for some applications. Recall that in order to solve **Q**-MKL in an iterative fashion, we need only to solve for β with all other parameters fixed, according to the following problem:

$$\begin{aligned} \min_{\beta \geq 0} \quad & \sum_m \frac{\|\mathbf{w}_m\|^2_{\mathcal{H}_m}}{\beta_m} \\ \text{s.t.} \quad & \beta^T \mathbf{Q} \beta \leq 1 \end{aligned} \tag{1}$$

Notice that in general, while the number of kernels may be large, it is unlikely that the size of $\mathbf{Q}$ (quadratic in the number of kernels, not examples), will dominate the combined size of all of the kernels. If so (as in a majority of computer vision problems), it may be advantageous to employ second-order methods to solve (1) for β in terms of $\mathbf{w}$. Perhaps the best known of these second order methods is Newton's method [1] with the following update: $\beta \leftarrow (\beta - H^{-1} g)$ where the Hessian H and gradient g are

$$H = \left[\mathbf{Q} + 2\,\mathrm{Diag}\left(\mathbf{W}\beta^{-3}\right)\right], \quad g = \left(\mathbf{Q}\beta - \mathbf{W}\beta^{-2}\right)$$

where $\mathbf{W} = \mathrm{Diag}(\|\mathbf{w}_1\|^2_{\mathcal{H}_1}, \ldots, \|\mathbf{w}_M\|^2_{\mathcal{H}_M})$, and β is exponentiated element-wise. Note that taking the inverse of H has roughly cubic time complexity in the number of kernels, so this method may be more appropriate for cases where the number of kernels is dominated by their size. In order to compute H and g, we need only to compute $\|\mathbf{w}_m\|^2_{\mathcal{H}_m}$ for each base kernel, which is given as $\|\mathbf{w}_m\|^2_{\mathcal{H}_m} = \frac{1}{2}\beta_m^2 \left((\alpha \circ \mathbf{y})^T K_m (\alpha \circ \mathbf{y})\right)$. Here, we use β from the previous iteration. Again, standard SVM implementations can be used to solve for $\mathbf{w}_{1,\ldots,M}$, (or in terms of α). This mechanism comes with a pitfall: there is no guarantee that $\beta \geq 0$ at the optimum, in which case we must substitute $\beta_m \leftarrow 0$, $\forall \beta_m < 0$, essentially projecting β back into the non-negative orthant (here, gradient and Hessian terms must also be set to zero where the corresponding $\beta = 0$ to rule out infinite values). Nonetheless, this method works well experimentally, and Alg. 1 converges in about 10 iterations.

1.1 Controlling the scale of β

Finally, we remark on normalizing the scale of β. If $\mathbf{Q}$ has any eigen-values $\lambda_i \approx 0$, then β is effectively unregularized along the direction of the corresponding eigen-vectors v_i, which has the effect of allowing the scale of β to grow quite large. This will scale the kernel matrices in such a way that the C parameter may lose its interpretation. That is, the C parameter reflects a trade-off between regularizer and loss, however, under standard regularizers the regularizer also controls the units in which the loss is measured, meaning that the two are not completely unconnected. Therefore, if

*Supported by NIH (R01AG040396), (R01AG021155); NSF (RI 1116584), (DMS 09-15240 ARRA), and (CMMI-1131690); Wisconsin Partnership Proposal; UW ADRC; UW ICTR (1UL1RR025011); AFOSR (FA9550-09-1-0098); and NLM (5T15LM007359). The authors would like to thank Maxwell Collins and Sangkyun Lee for many helpful discussions.

the regularizer does not control the units in which the loss is measured, then this role falls to the loss function itself, at which point the C parameter's meaning is altered. Since β is constrained to be nonnegative, eigen-vectors must be nonnegative as well in order to be fully unconstrained – which is guaranteed to be the case when $\mathbf{Q}$ is a graph Laplacian. We recommend two separate approaches to combatting this problem: one is to set $\mathbf{Q} = \mathbf{Q} + \mathbb{I}^{M \times M}$, which effectively adds a $\|\beta\|_2^2$ regularizer term. It also happens that this will guarantee that the eigen-values of $\mathbf{Q}$ are greater than one, and hence the eigen-values of $\mathbf{Q}^{-1}$ are all less than one, which is required for the theoretical guarantees of Section 3.1. Alternatively, one could add $\mathbf{1}^{M \times M}$ to $\mathbf{Q}$ to directly penalize the least eigen-vector when $\mathbf{Q}$ is a graph Laplacian. Secondly, we can scale β to unit 1- or 2-norm at each iteration, which affects only the scale of the combined kernel. This can be thought of as mixing penalty-based and constraint-based regularizers. In practice, this approach did not affect convergence.

2 Q-SVM Experiments with MNIST digits

We briefly mentioned in section 3.2 that a special case of $\mathbf{Q}$-MKL reduces to a form of SVM, which we refer to as $\mathbf{Q}$-SVM, which we will expand upon here. Consider a setting where all base kernels are rank one (e.g., derived from a single feature). If so, then each coefficient β_m effectively becomes a feature weight, and a 2-norm penalty on β is a 2-norm penalty on weights. Thus, 2-norm MKL reduces to a standard SVM. In the case of $\mathbf{Q}$-MKL we can interpret $\mathbf{Q}$ as specifying a covariance (or other quadratic) structure between *features*, rather than *kernels*. Thus, we can reduce the $\mathbf{Q}$-MKL model to a simple QP:

$$\min_{\mathbf{w},\xi \geq 0, b} \quad \mathbf{w}^T \mathbf{Q} \mathbf{w} + C \sum_i \xi_i \tag{2}$$
$$\text{s.t.} \;\; y_i \left(\mathbf{w}^T \mathbf{x}_i + b \right) \geq 1 - \xi_i$$

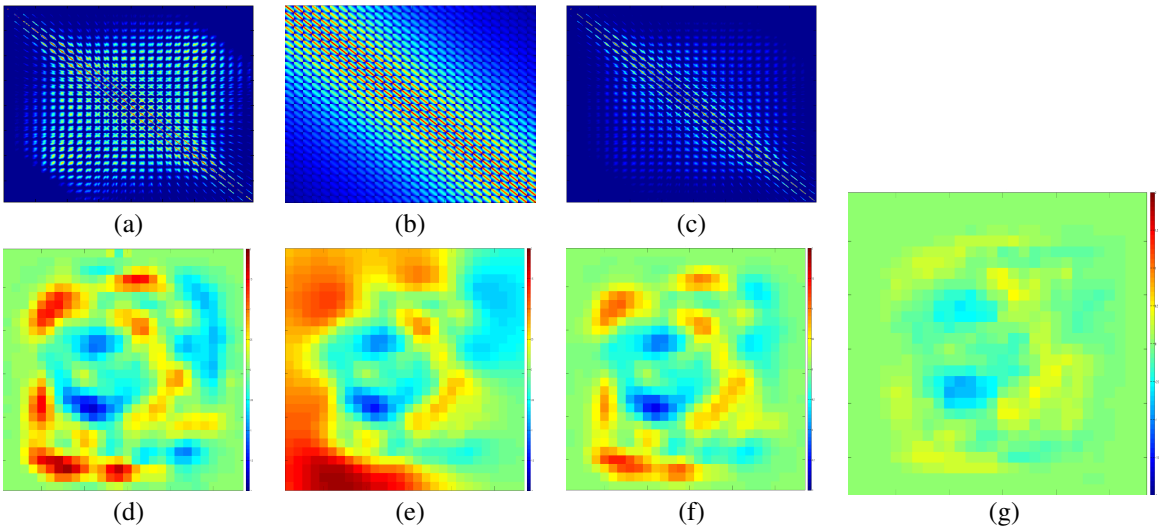

Figure 1: Top Row: Three $\mathbf{Q}$-matrices for pixel-wise features in the MNIST dataset. (Each is 784×784.) (a): pixel-wise covariances over the entire MNIST data set; (b): Gaussian kernel between pixel coordinates; (c): element-wise product of (a) and (b). Middle Row: (d – f) Pseudo-Inverses of the above $\mathbf{Q}$-matrices (a – c) were used as $\mathbf{Q}$-SVM regularizers, and the resulting weight vectors are depicted here as 28×28 images. Last Column: (g) Weight vector trained by a standard SVM. Color scale in all images goes from $-2 \times 10^{-3} - -2 \times 10^{-3}$. All $\mathbf{Q}$-matrices were normalized to have trace equal to the number of rows, to facilitate comparison with the Identity matrix (*i.e.*, standard SVM).

The $\mathbf{Q}$-SVM model turns out to be very similar to several other models that have been proposed in the machine learning literature recently. We note that the model of [3] is similar in concept to (2), except that their formulation is an adaptation of ADABoost (for fMRI voxels), while $\mathbf{Q}$-SVM is a direct extension of the SVM model. A recent

work for natural language processing[4] is also a special case of (2), in which we set $\mathbf{Q} = \mathbb{I}^{M\times M} - pp^T$, where $p_i = \frac{1}{M},\ \forall i$, (or some other distribution if it can be specified through domain knowledge). The authors of [5] proposed a regularization scheme in which the classifier $\mathbf{w}$ is transformed according to $\mathbf{w} \rightarrow e^{\frac{1}{2}L}\mathbf{w}$, where L is a graph Laplacian (or Laplace-Beltrami operator) based on distances between voxel-wise features and exponentiation is applied element-wise. We can see that the norm on $\mathbf{w}$ becomes $\mathbf{w}^T(L^TL)\mathbf{w}$ which is a case of $\mathbf{Q}$-SVM.

As an illustrative example, let us consider hand-written digit recognition from the MNIST digits dataset [2]. When examples correspond to images, features may correspond to pixels – there is a natural correlational structure inherent in the data, as neighboring pixels are expected to be highly correlated. Ordinarily, the relation between neighboring pixels is lost when converting images to feature vectors – however by encoding this relation in $\mathbf{Q}$, we can bias the SVM classifier to choose a smoothly varying pattern of pixels, which is more likely to correspond to the true separation between digit classes. Three such $\mathbf{Q}$-matrices are shown in the top row of Fig. 1. The first is the covariance function between pixels, which can be thought of as an indication for a given pair of pixels of how likely they are to both be included in a given character. This is similar to a spatial smoothness prior, except that it also encodes directionality. The second is a Gaussian kernel between 2D pixel coordinates, and the third is their element-wise product, (which is also p.s.d.). We then trained a $\mathbf{Q}$-SVM classifier to distinguish between '3' and '8' characters. The resulting weight vectors are shown in the second row of Fig. 1. For comparison, the weight vector chosen by a standard SVM is shown in the far right column.

It is clear from the figure that the corresponding regularizers differ, and they lead to different classifiers. We can view $\mathbf{Q}$-MKL as serving the same purpose – it provides a mechanism by which existing domain knowledge about relations not only between features, but between higher dimensional kernels as well, can be directly incorporated in the regularizer. Interactions between kernels are inherently more complex than those between single-dimensional features, which makes $\mathbf{Q}$-MKL a significant generalization of the above models.

3 Guaranteeing p.s.d. virtual kernels

As mentioned in Section 3.1.1, Thm. 2 requires that for each virtual kernel $\widetilde{\mathbb{K}}_i \succeq 0$, in order for it to define a valid RKHS. We can ensure this by choosing $\mathbf{Q}$ as follows. Note that there may be other ways of guaranteeing this condition; the procedure below is given as a demonstration that such cases do exist. Also note that in practice, even if we do not ensure that the virtual kernels are positive semi-definite (p.s.d.), $\mathbf{Q}$-MKL often performs as well as, or better than, 2-norm MKL.

We will construct $\mathbf{Q}$ by constructing its eigen-vectors directly, and choosing the eigen-values arbitrarily so as to be non-uniform. Let V be the matrix of eigen-vectors as columns, such that $\mathbf{Q} = V^T\Lambda V$, where Λ is an arbitrary, non-uniform, diagonal matrix. We begin by setting $V = \mathbb{I}^{M\times M}$. Next, we arbitrarily choose a pair of base kernels K_1 and K_2, and find the minimum $c \in \mathbb{R}$ such that $K_1 + cK_2 \succeq 0$ and $K_2 + cK_1 \succeq 0$. We then put c in $V(1,2)$, and $-c$ in $V(2,1)$, and renormalize the first 2 columns of V. This way, the two updated columns of V are normalized, orthogonal, and define P.S.D. virtual kernels. Let this updated V be denoted as $V_{1,2}$. If desired, we may construct $V_{i,j}$ for all other pairs of kernels K_i and K_j in a similar fashion, and combine the resulting orthonormal matrices.

4 UCI datasets

In this set of experiments we evaluate $\mathbf{Q}$-MKL 's performance on benchmark UCI datasets [6]. We present these experiments in order to show that our proposed regularization scheme does not worsen the model's performance relative to existing MKL models, and in some cases it may actually be beneficial to have a data-driven regularizer. This is an important validation because if one supposes that the purpose of regularization is to push the model *away* from sampling artifacts in the data, then a data-driven regularizer might defeat this purpose. The UCI results show that this is not the case, and in fact there can in some case be a benefit to doing so.

In order to facilitate comparison, we followed the methods of the SimpleMKL experiments [7]. Briefly, we used the same five repositories, (*Liver*, *Pima Diabetes*, *Ionosphere*, *WPBC Breast Cancer*, and *Sonar*). We whitened each feature by mean centering and normalizing to unit standard deviation, and normalized kernels to unit trace. The C parameter was set to 100. For kernels we used polynomials of degree one through three, and Gaussians with ten different bandwidths: $\{1, 2.5, 5\} * 10^{-1,0,1}$, and 100. We used 4-fold cross validation with 20 iterations to approximate the 70% training sets used in [7]. For each data set, we repeated the entire process with several different $\mathbf{Q}$-functions:

Liver

Regularizer	Accuracy	Sensitivity	Specificity	Area under ROC
Inv(Cov)	**0.717** $\pm$0.04	**0.530** $\pm$0.09	0.852 $\pm$0.05	**0.760** $\pm$0.01
Lap(Cov)	**0.722** $\pm$0.04	**0.530** $\pm$0.08	0.860 $\pm$0.05	**0.761** $\pm$0.01
Inv(Err)	**0.716** $\pm$0.04	**0.530** $\pm$0.07	0.851 $\pm$0.05	**0.760** $\pm$0.01
Lap(Err)	**0.719** $\pm$0.04	**0.529** $\pm$0.08	0.859 $\pm$0.04	**0.765** $\pm$0.01
$\mathbb{I}^{M\times M}$	0.701 $\pm$0.05	**0.518** $\pm$0.10	0.835 $\pm$0.06	0.750 $\pm$0.02
$\mathbf{1}^{M\times M}$	0.644 $\pm$0.07	0.271 $\pm$0.25	**0.913** $\pm$0.09	0.717 $\pm$0.05
Err	0.682 $\pm$0.04	0.477 $\pm$0.08	0.833 $\pm$0.06	0.731 $\pm$0.02

Pima

Regularizer	Accuracy	Sensitivity	Specificity	Area under ROC
Inv(Cov)	0.761 $\pm$0.03	0.871 $\pm$0.03	0.559 $\pm$0.06	0.821 $\pm$0.01
Lap(Cov)	**0.767** $\pm$0.03	0.878 $\pm$0.03	0.562 $\pm$0.06	0.822 $\pm$0.01
Inv(Err)	**0.764** $\pm$0.03	0.873 $\pm$0.03	**0.564** $\pm$0.06	0.822 $\pm$0.00
Lap(Err)	0.763 $\pm$0.03	0.877 $\pm$0.03	0.554 $\pm$0.06	0.823 $\pm$0.00
$\mathbb{I}^{M\times M}$	**0.766** $\pm$0.02	0.866 $\pm$0.03	**0.582** $\pm$0.06	0.821 $\pm$0.00
$\mathbf{1}^{M\times M}$	0.651 $\pm$0.03	**0.998** $\pm$0.02	0.007 $\pm$0.06	**0.993** $\pm$0.03
Err	**0.771** $\pm$0.02	0.890 $\pm$0.03	0.551 $\pm$0.05	0.829 $\pm$0.00

Ionosphere

Regularizer	Accuracy	Sensitivity	Specificity	Area under ROC
Inv(Cov)	0.939 $\pm$0.03	0.866 $\pm$0.06	**0.980** $\pm$0.02	**0.980** $\pm$0.01
Lap(Cov)	0.927 $\pm$0.06	0.829 $\pm$0.18	**0.982** $\pm$0.02	0.973 $\pm$0.02
Inv(Err)	0.940 $\pm$0.02	0.869 $\pm$0.06	**0.980** $\pm$0.02	0.981 $\pm$0.00
Lap(Err)	0.934 $\pm$0.05	0.845 $\pm$0.15	**0.982** $\pm$0.02	**0.976** $\pm$0.02
$\mathbb{I}^{M\times M}$	**0.948** $\pm$0.03	**0.887** $\pm$0.07	**0.982** $\pm$0.02	**0.982** $\pm$0.00
$\mathbf{1}^{M\times M}$	**0.940** $\pm$0.07	**0.866** $\pm$0.18	**0.983** $\pm$0.02	**0.974** $\pm$0.02
Err	**0.950** $\pm$0.02	**0.897** $\pm$0.05	**0.981** $\pm$0.02	**0.982** $\pm$0.00

WPBC

Regularizer	Accuracy	Sensitivity	Specificity	Area under ROC
Inv(Cov)	**0.759** $\pm$0.05	**0.025** $\pm$0.05	0.989 $\pm$0.02	0.625 $\pm$0.03
Lap(Cov)	**0.762** $\pm$0.06	**0.025** $\pm$0.05	0.992 $\pm$0.02	0.616 $\pm$0.03
Inv(Err)	**0.763** $\pm$0.04	**0.028** $\pm$0.05	0.992 $\pm$0.02	0.616 $\pm$0.03
Lap(Err)	**0.755** $\pm$0.06	**0.027** $\pm$0.05	0.984 $\pm$0.03	0.605 $\pm$0.03
$\mathbb{I}^{M\times M}$	**0.756** $\pm$0.06	**0.033** $\pm$0.06	0.982 $\pm$0.03	0.614 $\pm$0.03
$\mathbf{1}^{M\times M}$	**0.762** $\pm$0.05	0.000 $\pm$0.00	**0.999** $\pm$0.00	**0.957** $\pm$0.09
Err	**0.764** $\pm$0.05	0.006 $\pm$0.02	**1.000** $\pm$0.00	0.534 $\pm$0.04

Sonar

Regularizer	Accuracy	Sensitivity	Specificity	Area under ROC
Inv(Cov)	0.834 $\pm$0.05	0.789 $\pm$0.10	0.878 $\pm$0.08	0.917 $\pm$0.01
Lap(Cov)	0.816 $\pm$0.06	0.764 $\pm$0.16	0.863 $\pm$0.07	0.896 $\pm$0.04
Inv(Err)	0.783 $\pm$0.08	0.670 $\pm$0.18	0.899 $\pm$0.13	0.888 $\pm$0.03
Lap(Err)	0.817 $\pm$0.06	0.765 $\pm$0.13	0.869 $\pm$0.07	0.901 $\pm$0.04
$\mathbb{I}^{M\times M}$	0.836 $\pm$0.06	**0.805** $\pm$0.10	0.868 $\pm$0.08	0.923 $\pm$0.01
$\mathbf{1}^{M\times M}$	**0.858** $\pm$0.05	**0.817** $\pm$0.08	0.898 $\pm$0.07	**0.947** $\pm$0.01
Err	0.756 $\pm$0.08	0.523 $\pm$0.19	**0.973** $\pm$0.06	0.881 $\pm$0.03

Table 1: Performance measures for several $\mathbf{Q}$ functions on UCI datasets. Bold numerals indicate measures which are not significantly different from the maximum under a paired t-test. Lap = Laplacian; Inv = Inverse; Cov = Matrix covariance (Table 1, row 1); Err = Training error covariance.

(Pseudo-) Inverse and Graph-Laplacian of matrix covariance, training error covariance, and covariance of training-set

Function	$\mathbf{Q}(i,j) =$	Arguments	Uses
Covariance	$\frac{\langle K_i, K_j \rangle}{\|K_i\| \|K_j\|}$	$K_{1,\ldots,M}$	Unsupervised covariance (matrix cosine)
Eigen-space alignment	$\frac{1}{N} \sum_k^N v_k^{(i)^T} v_k^{(j)}$	$K_{1,\ldots,M}$	Mean cosine of the first N eigen-vectors
Histogram intersection	$\left\| \min(K_i, K_j) \right\|_1$	$K_{1,\ldots,M}$	Kernels $\rightarrow$ histograms; $\min(\cdot,\cdot)$ applied entry-wise
Training error covariance	$\text{err}(K_i, \mathbf{y})^T \text{err}(K_j, \mathbf{y})$	$K_{1,\ldots,M}, \mathbf{y}$	Covariance of training errors
α covariance	$\alpha_i^T \alpha_j$	$K_{1,\ldots,M}, \mathbf{y}$	Covariance of SVM α parameters

Table 2: $\mathbf{Q}$-functions, their arguments, and their uses.

SVM parameters α. For comparison, we also used Identity (2-norm MKL) and $\mathbf{1}^{M \times M}$ (1-norm MKL). Accuracy sensitivity, specificity and area under ROC curve are shown in Table 1.

Several trends can be seen to emerge from these results: first, 1-norm MKL significantly underperforms the other methods in two of the data sets (*Liver, Pima*), while slightly over-performing on one of them (*Sonar*). The greater variance of 1-norm MKL's predictive performance is attributable to the sparsity it encourages at the kernel level. Next, we note that on *Pima, Ionosphere*, and *Breast Cancer* 2-norm MKL is comparable to the two $\mathbf{Q}$-functions used, (on *Sonar* Train-Error covariance slightly underperforms the other two), while on *Liver* the two $\mathbf{Q}$-functions significantly outperform 2-norm MKL. From this, we conclude that in general, $\mathbf{Q}$-MKL does not induce any significant risk the way 1-norm does, while performing competitively (or more favorably) that 2-norm MKL. It is interesting to note that the UCI datasets are essentially *unimodality*; there is no *a priori* information which tells us how the kernels are related.

5 More interpretation of Q from AD experiments

Next, we provide a more detailed exploration of the covariance structures found in our neuroimaging data, as well as some other $\mathbf{Q}$-functions, shown in Table 2. Recall that the images were divided into four groups: Voxel-Based Morphometry (VBM) of MR images, longitudinal Tensor-Based Morphometry (TBM) of MR images, Fluoro-Deoxy Glucose PET (FDG-PET) images at baseline, and FDG-PET images at 24 months. For each set of images we sorted the voxels by t-statistic of separation between AD and control subjects, (excluding test subjects), and constructed separate kernels using 250,000, 150,000, 100,000, 65,000, 25,000, 10,000, 5000, and 2000 voxels. For each set of voxels we constructed linear, quadratic, and Gaussian kernels, giving a total of $8 \times 3 \times 4 = 96$ kernels. The bandwidth parameter for the Gaussian kernels was chosen as $2\times$ the number of features used, which is a proxy for the mean distance between examples.

5.1 Covariance Q

From Figure 2(a) we can see that there are four main blocks (sets of images), each with 3 sub-blocks (kernel functions), each consisting of an 8×8 block of kernels which differ only in terms of the features used to construct them. As discussed in section 5, some notable covariance behaviors are represented as individual eigen-vectors $v_1, \ldots, v_5$, as shown in Figure 2(b–f). Some of these are expected, such as the block structure seen in v_2, (Figure. 2(c)) corresponding to the block of TBM-derived kernels, while some are surprising, such as the covariance between the quadratic kernels in all kernels except for those derived from TBM MR images as seen in v_3, (Figure. 2(d)). This relationship is echoed somewhat in v_6, (Figure. 2(g)). Additionally, a strong covariance relationship exists between the linear and Gaussian kernels of VBM MR images in v_4, (Figure. 2(e)), and among the FDG-PET kernels in v_5, (Figure. 2(f)).

5.2 Histogram Q

Next, for comparison we present the histogram $\mathbf{Q}$ (See Table 2, entry 3). Note that remarkably similar patterns appear as in the covariance $\mathbf{Q}$.

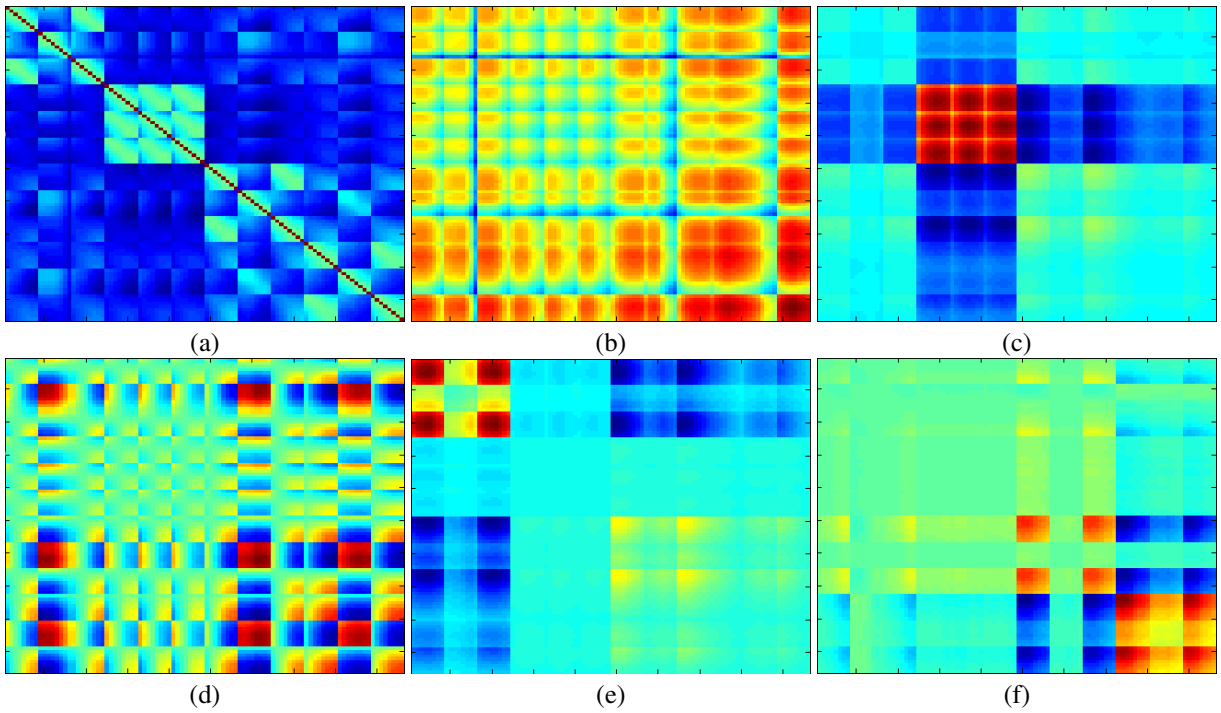

Figure 2: (a) Covariance $\mathbf{Q}$ matrix. (b–f) Least 5 eigen-vectors, represented as outer products.

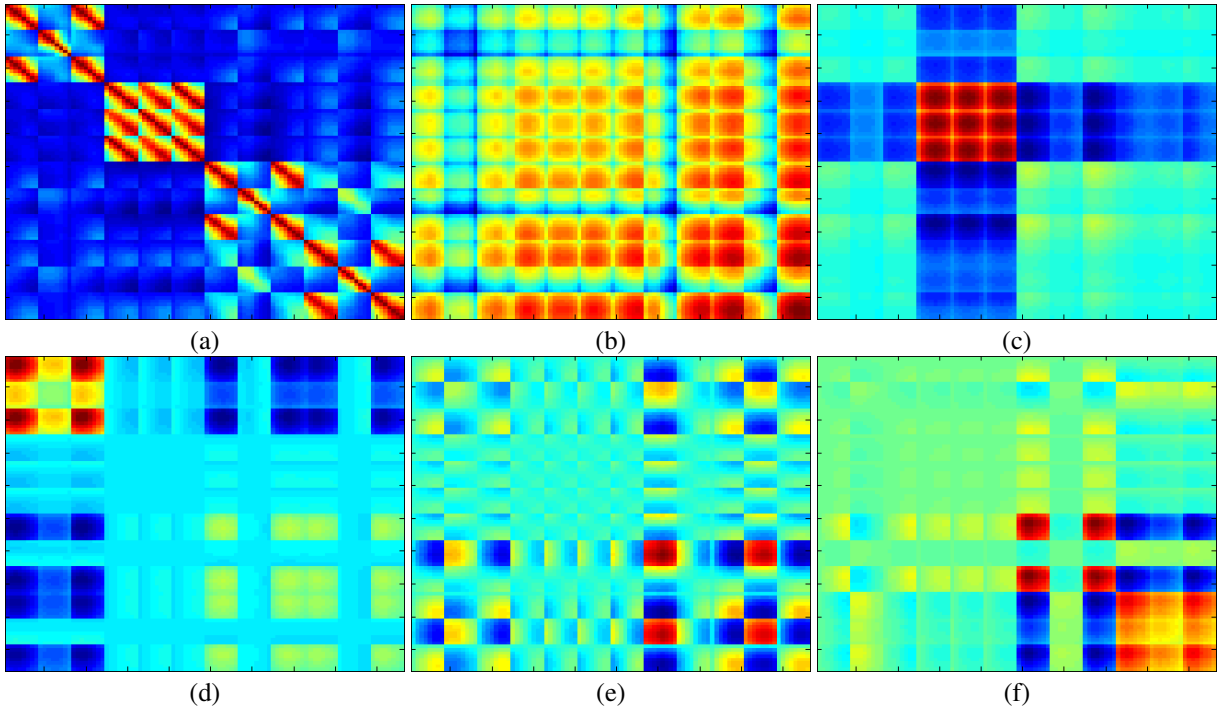

Figure 3: (a) Histogram $\mathbf{Q}$ matrix. (b–f) Least 5 eigen-vectors, represented as outer products.

5.3 Eigen-space alignment Q

Next we present the eigen-space alignment **Q** (see Table 2, entry 2). See Figure 4. Note that some patterns are similar to the covariance and histogram intersection **Q** matrices, some are different – note that both VBM- and TBM-derived kernels show a block structure, while the interaction between quadratic kernels is even more prominent. An interesting trend is that the fewer the number of features shared between kernels, (or the more in some cases,) the less alignment there is between their eigenspaces, regardless of the kernel function used; see especially Figure 4(b,c,d).

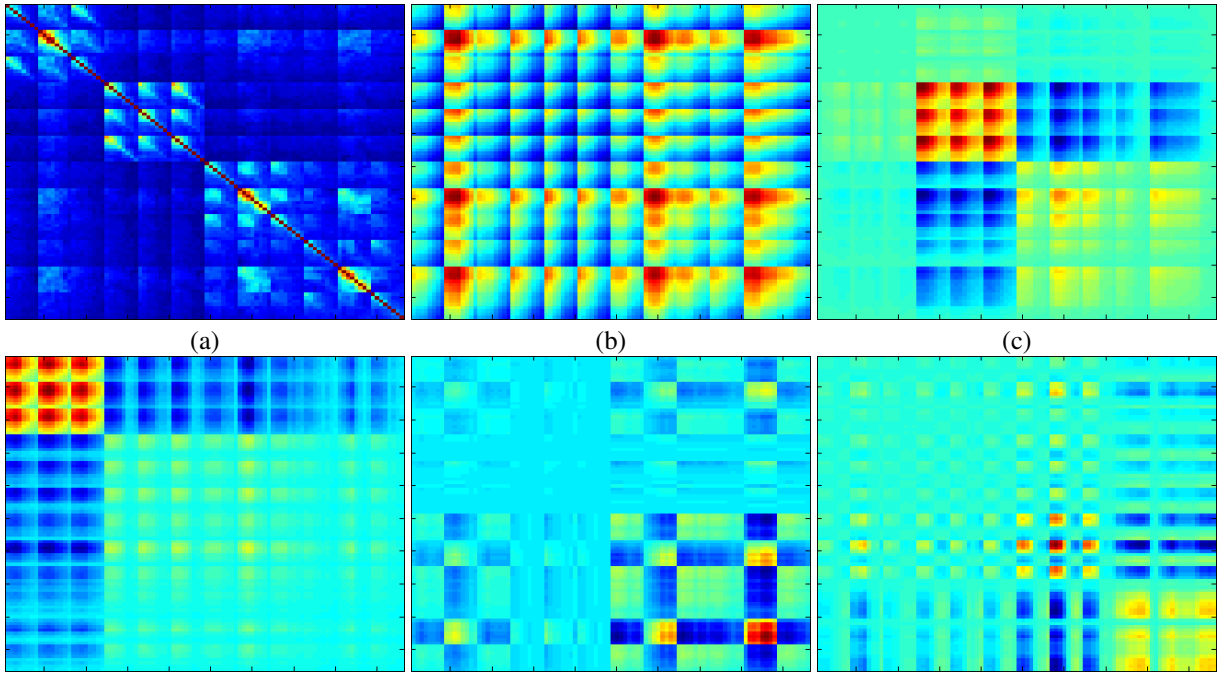

Figure 4: (a) Eigenspace alignment **Q** matrix. (b-f) Least 5 eigen-vectors, represented as outer products.

5.4 Training error covariance

Lastly, we present a similar analysis of the **Q**-function derived from training error covariance. See Figure 5. As with the unsupervised **Q**-functions, there are several block-structures corresponding to similarities in the construction of the kernels, e.g., among VBM, TBM, quadratic, and linear and Gaussian kernels. This is expected, because where features are correlated we can expect there to be some error correlation as well. Notice, however, that in Figure. 5(a), there are some interesting differences. Particularly, some of the FDG-PET quadratic kernels are more strikingly anti-correlated with the rest of the kernels, and within the quadratic kernels there appear to be some sub-clusters as well. (See Figure 5(a,b,e).) Moreover, overlapping features seems to be less of a dominant factor – note the "flatter" appearance of the blocks, with less of a "gradient" moving from upper-left to lower-right within the blocks. (Cf. Figures 2, 4.) An intriguing possibility is that by leveraging the *differences* between the supervised and un-supervised interactions, we may be able to derive a better estimate of the true error covariances, without the confounding influence of data artifacts or normalization issues.

References

[1] J. Nocedal and S.J. Wright. *Numerical optimization*. Springer Verlag, 1999.

[2] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[3] Z. Xiang, Y. Xi, U. Hasson, and P. Ramadge. Boosting with spatial regularization. *NIPS*, 2009.

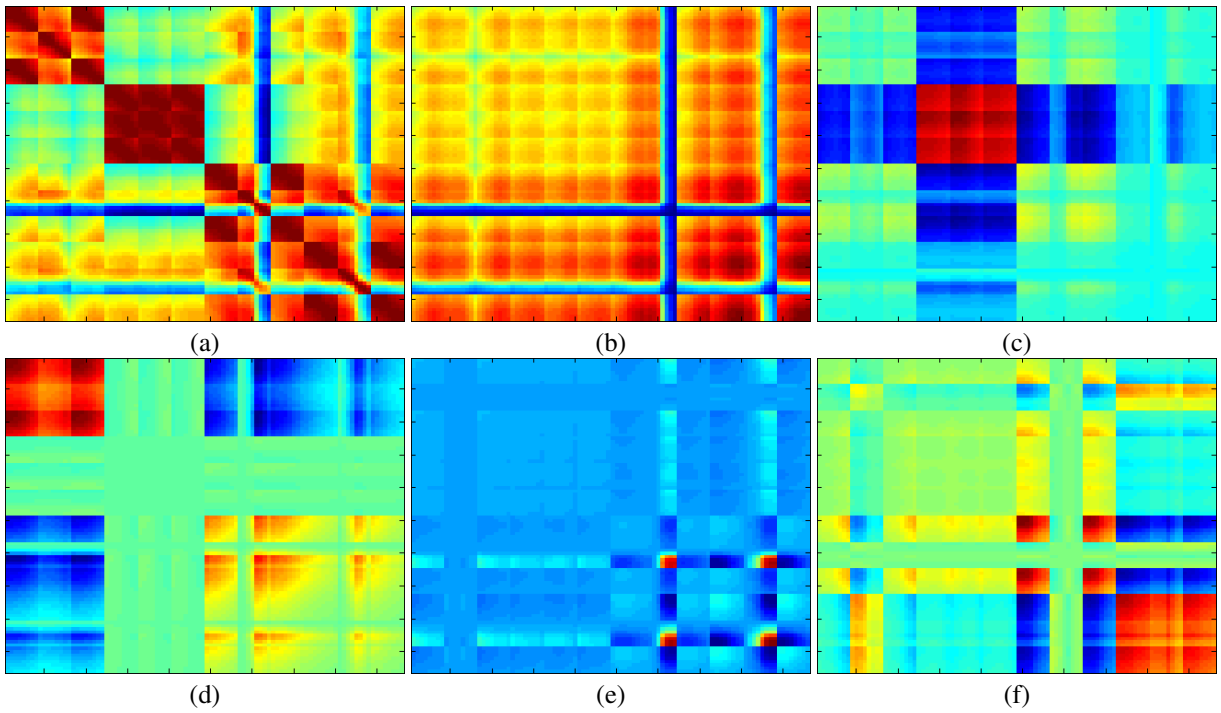

Figure 5: (a) Training error covariance $\mathbf{Q}$ matrix. (b–f) Least 5 eigen-vectors, represented as outer products.

[4] S. Bergsma, D. Lin, and D. Schuurmans. Improved Natural Language Learning via Variance-Regularization Support Vector Machines. *Conference on Natural Language Learning*, 2010.

[5] R. Cuingnet, M. Chupin, H. Benali, and O. Colliot. Spatial and anatomical regularization of svm for brain image analysis. *NIPS*, 2010.

[6] A. Frank and A. Asuncion. UCI machine learning repository, 2010.

[7] A. Rakotomamonjy, F. Bach, S. Canu, and Y. Grandvalet. SimpleMKL. *JMLR*, 9:2491–2521, 2008.