Computational Pattern Analysis and Statistical Learning

Lecture 6: Advanced topics

Tijl De Bie, Konstantin Tretyakov
(Largely based on joint work with Nello Cristianini and John Shawe-Taylor)

Tartu, Estonia

November 2006

1. Lecture 6A: Kernels on structured data
   - Kernels on vectors
   - Kernel on texts and strings
   - A kernel on graphs
   - A kernel on data with a probabilistic model

2. Lecture 6B: Kernel methods for data fusion
   - Why data fusion?
   - Combining complementary data sources
   - Canonical Correlation Analysis

3. Wrap-up

Backward look

- Kernels may be useful when the feature vectors are high-dimensional
- Examples:
  - feature vector representation corresponding to a Gaussian kernel
  - graphs represented as adjacency matrices
  - text over a large vocabulary
- However, *a kernel is only useful when it is more efficient to compute than the features themselves*
- Here we will discuss some examples of such kernels

We have seen a few

- RBF kernel, linear kernel, polynomial kernel:

$$
\begin{aligned}
k_{\mathrm{RBF}}(x_i, x_j) &= \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)\\
k_{\mathrm{linear}}(x_i, x_j) &= x_i' \cdot x_j\\
k_{\mathrm{polynomial},d}(x_i, x_j) &= (x_i' \cdot x_j + 1)^d = \sum_{k=1}^{d} \binom{d}{k} (x_i' \cdot x_j)^k
\end{aligned}
$$

- In order to understand what these mean, consider that the resulting projection of a feature vector on the weight vector can always be written as $\mathbf{x}'\mathbf{w} = \sum_{i=1}^{n} \alpha_i k(x_i, x)$
- Hence, RBF $\rightarrow$ sum of Gaussians // linear $\rightarrow$ sum of inner products // polynomial $\rightarrow$ sum of powers up to d of inner products

Texts and strings

- I consider a text an ordered list/sequence of distinct words from a dictionary (usually large)
- I consider a string an ordered list/sequence of symbols from an alphabet (usually quite small)
- In fact the difference is artificial, but often it's useful and intuitive...

The bag-of-words kernel

- Consider a text (let's say a natural language sentence)
- What are the essential ingredients?
- The words! (We ignore the grammar / word ordering for now)
- Imagine a feature vector $\mathbf{x}$ with as ith entry the number of occurrences of the ith word in the vocabulary
- The *bag-of-words* representation...

The bag-of-words kernel

- Text (e.g. bag of words)
- Sentence i is x_i
- E.g. $x_i =$"This is a sentence containing the words: this, and, a, and and"
- Vocabulary: {a, and, containing, is, sentence, the, this, words}
- (Usually, the vocabulary is much larger than the number of words used)

- Vector representation:

$$\mathbf{x}_i = \begin{pmatrix} 2 \\ 3 \\ 1 \\ 1 \\ 1 \\ 1 \\ 2 \\ 1 \end{pmatrix}$$

- (Usually an extremely sparse vector...)

The bag-of-words kernel

- How to compute this kernel efficiently?
- I.e. the inner product between two such feature vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ without ever actually computing them
- One approach:
  - Sort the words in each sentence alphabetically, remove duplicates, and remember counts
  - Go through the lists of words left to right, take product, add up...
- In practice, this is much faster (because vocabulary size is much larger than word use in texts)

The bag-of-words kernel

x_1=Today is the last lecture.
x_2=The weather is great today.
x_3=The sun is shining.

The bag-of-words kernel

x_1=Today is the last lecture.
x_2=The weather is great today.
x_3=The sun is shining.

is, last, lecture, the, today
great, is, the, today, weather
is, shining, sun, the

great, is, last, lecture, shining, sun, the, today, weather

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

The bag-of-words kernel

x_1=Today is the last lecture.
x_2=The weather is great today.
x_3=The sun is shining.

is, last, lecture, the, today
great, is, the, today, weather
is, shining, sun, the

great, is, last, lecture, shining, sun, the, today, weather

$$\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \qquad \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

The bag-of-words kernel

x_1=Today is the last lecture.
x_2=The weather is great today.
x_3=The sun is shining.

is, last, lecture, the, today
great, is, the, today, weather
is, shining, sun, the

great, is, last, lecture, shining, sun, the, today, weather

$$\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \qquad \begin{pmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

The bag-of-words kernel

x_1=Today is the last lecture.
x_2=The weather is great today.
x_3=The sun is shining.

is, last, lecture, the, today
great, is, the, today, weather
is, shining, sun, the

great, is, last, lecture, shining, sun, the, today, weather

$$\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \qquad \begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

The bag-of-words kernel

x_1=Today is the last lecture.
x_2=The weather is great today.
x_3=The sun is shining.

is, last, lecture, the, today
great, is, the, today, weather
is, shining, sun, the

great, is, last, lecture, shining, sun, the, today, weather

$$\begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} \qquad \begin{pmatrix} 5 & 3 & 2 \\ 3 & 4 & 2 \\ 2 & 2 & 4 \end{pmatrix}$$

The bag-of-words kernel

x_1=Today is the last lecture.
x_2=The weather is great today.
x_3=The sun is shining.

is, last, lecture, the, today
great, is, the, today, weather
is, shining, sun, the

great, is, last, lecture, shining, sun, the, today, weather

$$\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \qquad \begin{pmatrix} 5 & 3 & 2 \\ 3 & 5 & 2 \\ 2 & 2 & 4 \end{pmatrix}$$

The k-mer kernel

- A kernel for strings, not containing distinct words
- Count substrings, e.g. all substrings up to length $k = 3$
- $AGTCGTC \rightarrow \left\{ \begin{array}{c} 1 \times ACT, 2 \times GTC, \\ 1 \times TCG, 1 \times CGT \end{array} \right\}$
- Dimensionality of the feature space: $(\text{alphabet size})^k$, here $4^3 = 64$
- Usually very sparse (especially for large k)

$$\begin{pmatrix} AAA \\ \vdots \\ ACT \\ \vdots \\ CGT \\ \vdots \\ GTC \\ \vdots \\ TCG \\ \vdots \\ TTT \end{pmatrix} \rightarrow \begin{pmatrix} 0 \\ 1 \\ 1 \\ 2 \\ 1 \\ 0 \end{pmatrix}$$

The k-mer kernel

- Algorithm: based on Jaak's algorithm to find the longest frequent substring
- Traverse the trie-structured substring space
- Along the way, keep pointers to the occurrences of the substring, in all strings between which the kernel needs to be computed
- Once reached the required depth (e.g. depth 3 for the 3-mer kernel), multiply the numbers of occurrences in the different strings

The k-mer kernel

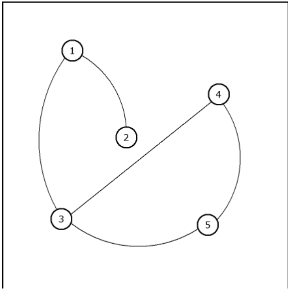
References:

- Jaak Vilo: Pattern Discovery from Biosequences. PhD Thesis, Department of Computer Science, University of Helsinki, Finland. Series of Publications A, Report A-2002-3 Helsinki, November 2002, 149 pages.
- Christina S. Leslie, Rui Kuang: Fast String Kernels using Inexact Matching for Protein Sequences. Journal of Machine Learning Research 5: 1435-1455 (2004).

The diffusion kernel

- Consider an undirected graph, unweighted (for simplicity here)
- Graph Laplacian: $-$ the degree on the diagonal elements, and 1's if there is an edge

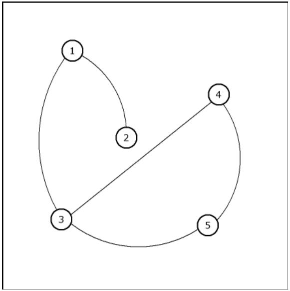
$$\mathbf{L} = \begin{pmatrix} -2 & 1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 \\ 1 & 0 & -3 & 1 & 1 \\ 0 & 0 & 1 & -2 & 1 \\ 0 & 0 & 1 & 1 & -2 \end{pmatrix}$$



The diffusion kernel

- Let's consider a lazy random walk over the graph

$$
\begin{aligned}
P(i \rightarrow i) &= 1 - d_i \cdot \Delta t \\
P(i \rightarrow j) &= \Delta t \text{ if } (i,j) = \text{edge}
\end{aligned}
$$



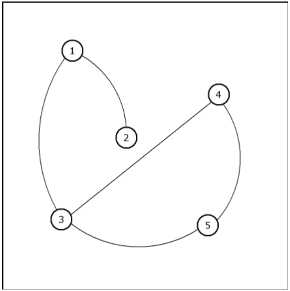
- Then, the probability to go from i to j after time period t is the element at (i,j) of

$$\mathbf{K} = \exp(t\mathbf{L})$$

The diffusion kernel

For $t = 0.1$:

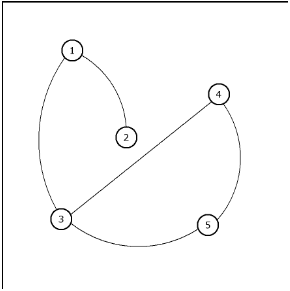
$$\mathbf{K} = \begin{pmatrix} 0.98 & 0.01 & 0.01 & 0.00 & 0.00 \\ 0.01 & 0.99 & 0.00 & 0.00 & 0.00 \\ 0.01 & 0.00 & 0.97 & 0.01 & 0.01 \\ 0.00 & 0.00 & 0.01 & 0.98 & 0.01 \\ 0.00 & 0.00 & 0.01 & 0.01 & 0.98 \end{pmatrix}$$



The diffusion kernel

For $t = 0.1$:

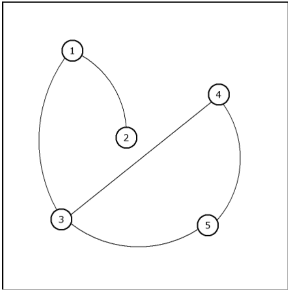
$$\mathbf{K} = \begin{pmatrix} 0.83 & 0.09 & 0.08 & 0.00 & 0.00 \\ 0.09 & 0.91 & 0.00 & 0.00 & 0.00 \\ 0.08 & 0.00 & 0.75 & 0.08 & 0.08 \\ 0.00 & 0.00 & 0.08 & 0.83 & 0.09 \\ 0.00 & 0.00 & 0.08 & 0.09 & 0.83 \end{pmatrix}$$



The diffusion kernel

For $t = 1$:

$$\mathbf{K} = \begin{pmatrix} 0.32 & 0.30 & 0.18 & 0.10 & 0.10 \\ 0.30 & 0.52 & 0.10 & 0.04 & 0.04 \\ 0.18 & 0.10 & 0.24 & 0.24 & 0.24 \\ 0.10 & 0.04 & 0.24 & 0.34 & 0.29 \\ 0.10 & 0.04 & 0.24 & 0.29 & 0.34 \end{pmatrix}$$



The diffusion kernel

- Reference: R. I. Kondor and J. Lafferty (2002). Diffusion Kernels on Graphs and Other Discrete Input Spaces. ICML 2002.

The marginalised kernel

- Assume a probabilistic model for the data (a graphical model / bayesian network / markov random field)
- For example: an HMM for strings:
  - hidden chain variables are $h(k)$
  - visible chain variables are $x(k)$ (the string itself)
- Define the kernel as:

$$k(x_i, x_j) = \sum_{h_i, h_j} P(h_i|x_i)\, P(h_j|x_j)\, k^*((x_i, h_i), (x_j, h_j))$$

The marginalised kernel

- Intuition: check whether there are plausible explanations for data x_i and x_j that are similar (according to k^*)
- Example: $k^*\left(\left(x_i, h_i\right), \left(x_j, h_j\right)\right) = \delta\left(h_i, h_j\right)$
- Then:

$$k\left(x_i, x_j\right) = \sum_h P\left(h|x_i\right) P\left(h|x_j\right)$$

- Intuition: are there hidden chains h that are likely both under x_i and x_j?
- Very generally applicable – also to more general probabilistic models
- Reference: Tsuda K, Kin T, Asai K. Marginalized kernels for biological sequences. 1: Bioinformatics. 2002;18 Suppl 1:S268-75.

Data fusion?

- Remember, we had data objects x
- We represented them using vectors $\mathbf{x}$
- But: there were often several ways to do this vector representation (either explicitly, or implicitly by using a specific choice of kernel)
- So which choice to make?

Data fusion?

- Examples:
    - we have seen there are several ways of representing nodes in a graph – also implicitly by using the diffusion kernel
    - nonlinear kernels on vectorial data: many many choices...
- More fundamentally:
    - Genes can represented by the DNA sequence, the AA sequence, the 3-D structure of the protein, microarray expression data, motif data,...
    - the content of a text can be represented by the bag-of-words representation in a chosen language (there are many languages – all contain the same information)
- But, why make a choice here? Use all if possible!

Data fusion?

Two major ideas:

1. Extract what different representations have in common
   - what do translations of the same text have in common?
   - not the grammar, not the vocabulary...
   - the semantics – the meaning!
2. Combine how different representations are complementary
   - Microarray data, gene sequence, motif data,... all may tell you a different story about the gene

Convex combinations of kernels

- Let us compute a kernel for each data source: $\mathbf{K}_j$
- Then, we can compute a convex combination of those:

$$\mathbf{K} = \sum \mu_j \mathbf{K}_j \text{ with } \sum \mu_j = 1$$

- This is again a valid kernel!
- There are heuristic ways of doing this...
- There are ways of doing this which minimise the (Rademacher) complexity bound, and which are based on convex optimisation theory

Convex combinations of kernels

References:

- Lanckriet, G.R.G., Cristianini, N., Bartlett, P., El Ghaoui, L., Jordan, M.I. (2004). Learning the Kernel Matrix with Semidefinite Programming. Journal of Machine Learning Research, 5, 27-72, 2004.
- Lanckriet, G.R.G., De Bie, T., Cristianini, N. , Jordan, M.I., Noble, W.S. (2004). A statistical framework for genomic data fusion. Bioinformatics, 20, 2626-2635, 2004.

Canonical correlation analysis

- Given:
  - 2 representations $\mathbf{X}$ and $\mathbf{Z}$ for the same objects x_i:
    $\mathbf{X} = \begin{pmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_n \end{pmatrix}'$, $\mathbf{Z} = \begin{pmatrix} \mathbf{z}_1 & \mathbf{z}_2 & \ldots & \mathbf{z}_n \end{pmatrix}'$
  - assume these data are centred (i.e., their means are in the origin)
- Find:
  - weight vectors $\mathbf{w}_x$ and $\mathbf{w}_z$ such that the projection of $\mathbf{x}$ on $\mathbf{w}_x$ strongly correlates with the projection of the corresponding $\mathbf{z}$ on $\mathbf{w}_z$
  - intuitively: common 'factors' or 'features' underlying both representations

Canonical correlation analysis

- Covariance between projections:
  $\sigma_{xz} = \sum_{i=1}^{n} \mathbf{x}_i'\mathbf{w}_x \cdot \mathbf{z}_i'\mathbf{w}_z = \mathbf{w}_x'\mathbf{X}'\mathbf{Z}\mathbf{w}_z$
- Variance of projection of $\mathbf{X}$:
  $\sigma_x^2 = \sum_{i=1}^{n} \mathbf{x}_i'\mathbf{w}_x \cdot \mathbf{x}_i'\mathbf{w}_x = \mathbf{w}_x'\mathbf{X}'\mathbf{X}\mathbf{w}_x$
- Variance of projection of $\mathbf{Z}$:
  $\sigma_z^2 = \sum_{i=1}^{n} \mathbf{z}_i'\mathbf{w}_z \cdot \mathbf{z}_i'\mathbf{w}_z = \mathbf{w}_z'\mathbf{Z}'\mathbf{Z}\mathbf{w}_z$
- Correlation defined as: $\rho_{xz} = \frac{\sigma_{xz}}{\sigma_x \sigma_z}$
- The correlation on the training set can be written as

$$\rho_{xz} = \frac{\mathbf{w}_x'\mathbf{X}'\mathbf{Z}\mathbf{w}_z}{\sqrt{\mathbf{w}_x'\mathbf{X}'\mathbf{X}\mathbf{w}_x}\sqrt{\mathbf{w}_z'\mathbf{Z}'\mathbf{Z}\mathbf{w}_z}}$$

Canonical correlation analysis

- Optimisation problem:

$$\max_{\mathbf{w}_x,\mathbf{w}_y} \frac{\mathbf{w}_x'\mathbf{X}'\mathbf{Z}\mathbf{w}_z}{\sqrt{\mathbf{w}_x'\mathbf{X}'\mathbf{X}\mathbf{w}_x}\sqrt{\mathbf{w}_z'\mathbf{Z}'\mathbf{Z}\mathbf{w}_z}}$$

- Seems hard... but note: invariant with respect to scalings of $\mathbf{w}_x$ and $\mathbf{w}_y$
- Get rid of this by restating the problem as

$$\begin{aligned} \max_{\mathbf{w}_x,\mathbf{w}_y} \quad & \mathbf{w}_x'\mathbf{X}'\mathbf{Z}\mathbf{w}_z \\ \text{s.t.} \quad & \mathbf{w}_x'\mathbf{X}'\mathbf{X}\mathbf{w}_x + \mathbf{w}_z'\mathbf{Z}'\mathbf{Z}\mathbf{w}_z = 2 \end{aligned}$$

Canonical correlation analysis

- Solve by means of method of Lagrange multipliers: introduce Lagrange multiplier $\frac{\lambda}{2}$ (divided by 2 for convenience only)

$$\max_{\mathbf{w}_x,\mathbf{w}_y} \mathbf{w}_x'\mathbf{X}'\mathbf{Z}\mathbf{w}_z - \frac{\lambda}{2}\left(\mathbf{w}_x'\mathbf{X}'\mathbf{X}\mathbf{w}_x + \mathbf{w}_z'\mathbf{Z}'\mathbf{Z}\mathbf{w}_z - 2\right)$$

- Take gradient with respect to the weight vectors and equate to $\mathbf{0}$:

$$\begin{aligned} \mathbf{X}'\mathbf{Z}\mathbf{w}_z - \lambda\mathbf{X}'\mathbf{X}\mathbf{w}_x &= 0 \\ \mathbf{Z}'\mathbf{X}\mathbf{w}_x - \lambda\mathbf{Z}'\mathbf{Z}\mathbf{w}_z &= 0 \end{aligned}$$

Canonical correlation analysis

- (The result again:)

$$\begin{aligned} \mathbf{X}'\mathbf{Z}\mathbf{w}_z - \lambda\mathbf{X}'\mathbf{X}\mathbf{w}_x &= 0 \\ \mathbf{Z}'\mathbf{X}\mathbf{w}_x - \lambda\mathbf{Z}'\mathbf{Z}\mathbf{w}_z &= 0 \end{aligned}$$

- In matrix notation:

$$\begin{pmatrix} \mathbf{0} & \mathbf{X}'\mathbf{Z} \\ \mathbf{Z}'\mathbf{X} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{w}_x \\ \mathbf{w}_z \end{pmatrix} = \lambda \begin{pmatrix} \mathbf{X}'\mathbf{X} & \mathbf{0} \\ \mathbf{0} & \mathbf{Z}'\mathbf{Z} \end{pmatrix} \begin{pmatrix} \mathbf{w}_x \\ \mathbf{w}_z \end{pmatrix}$$

- An easily solvable generalised eigenvalue problem

Canonical correlation analysis

- By left multiplication of the first equation with $\mathbf{w}_x$ and the second with $\mathbf{w}_z$, we can see that

$$\begin{aligned}\mathbf{w}_x'\mathbf{X}'\mathbf{Z}\mathbf{w}_z - \lambda\mathbf{w}_x'\mathbf{X}'\mathbf{X}\mathbf{w}_x &= 0\\ \mathbf{w}_z'\mathbf{Z}'\mathbf{X}\mathbf{w}_x - \lambda\mathbf{w}_z'\mathbf{Z}'\mathbf{Z}\mathbf{w}_z &= 0\end{aligned}$$

  and hence $\mathbf{w}_x'\mathbf{X}'\mathbf{X}\mathbf{w}_x = \mathbf{w}_z'\mathbf{Z}'\mathbf{Z}\mathbf{w}_z$ ($= 1$ to satisfy the constraint)
- $\longrightarrow$ normalise $\mathbf{w}_x$ and $\mathbf{w}_z$ such that $\mathbf{w}_x'\mathbf{X}'\mathbf{X}\mathbf{w}_x = \mathbf{w}_z'\mathbf{Z}'\mathbf{Z}\mathbf{w}_z = 1$ after solving the eigenvalue problem
- Furthermore:

$$\lambda = \frac{\mathbf{w}_x'\mathbf{X}'\mathbf{Z}\mathbf{w}_z}{\mathbf{w}_x'\mathbf{X}'\mathbf{X}\mathbf{w}_x} = \frac{\mathbf{w}_x'\mathbf{X}'\mathbf{Z}\mathbf{w}_z}{\mathbf{w}_z'\mathbf{Z}'\mathbf{Z}\mathbf{w}_z} = \frac{\mathbf{w}_x'\mathbf{X}'\mathbf{Z}\mathbf{w}_z}{\sqrt{\mathbf{w}_x'\mathbf{X}'\mathbf{X}\mathbf{w}_x}\sqrt{\mathbf{w}_z'\mathbf{Z}'\mathbf{Z}\mathbf{w}_z}}$$

  the correlation along those directions

Regularised CCA

- In high dimensional spaces, there is too much freedom to find large correlation weight vectors on a given training set
- Assume $\mathbf{X}$ and $\mathbf{Z}$ are full rank (i.e. dimensionality $d \geq n$), then by choosing $\mathbf{Z}\mathbf{w}_z = \mathbf{X}\mathbf{w}_x \Leftrightarrow \mathbf{w}_x = \mathbf{X}^{-1}\mathbf{Z}\mathbf{w}_z$ we can always achieve a correlation $\lambda = 1$
- This means that a correlation of 1 is in any case non-significant (also it would not be stable)
- In other words: overfitting with bad generalisation as a consequence
- $\rightarrow$ reduce the norms of the weight vectors (constrain the capacity...)

Regularised CCA

- Regularised optimisation problem:

$$\begin{aligned} \max_{\mathbf{w}_x,\mathbf{w}_y} \quad & \mathbf{w}_x'\mathbf{X}'\mathbf{Z}\mathbf{w}_z \\ \text{s.t.} \quad & (1-\gamma)\left(\mathbf{w}_x'\mathbf{X}'\mathbf{X}\mathbf{w}_x + \mathbf{w}_z'\mathbf{Z}'\mathbf{Z}\mathbf{w}_z\right) + \\ & \gamma\left(\mathbf{w}_x'\mathbf{w}_x + \mathbf{w}_z'\mathbf{w}_z\right) = 2 \end{aligned}$$

- This ensures that the norms of $\mathbf{w}_x$ and $\mathbf{w}_z$ are bounded (and small)
- I.e. we reduce the pattern space!
- (In a somewhat different way as before...)

Regularised CCA

- Solve by means of method of Lagrange multipliers: introduce Lagrange multiplier $\frac{\lambda}{2}$ (divided by 2 for convenience only)

$$\max_{\mathbf{w}_x,\mathbf{w}_y} \mathbf{w}_x'\mathbf{X}'\mathbf{Z}\mathbf{w}_z - \frac{\lambda}{2} \quad \left(\mathbf{w}_x' \left((1-\gamma)\,\mathbf{X}'\mathbf{X} + \gamma\mathbf{I}\right)\mathbf{w}_x \right.$$
$$\left. + \mathbf{w}_z' \left((1-\gamma)\,\mathbf{Z}'\mathbf{Z} + \gamma\mathbf{I}\right)\mathbf{w}_z - 2\right)$$

- Optimality conditions:

$$\begin{aligned} \mathbf{X}'\mathbf{Z}\mathbf{w}_z - \lambda\left((1-\gamma)\,\mathbf{X}'\mathbf{X} + \gamma\mathbf{I}\right)\mathbf{w}_x &= 0 \\ \mathbf{Z}'\mathbf{X}\mathbf{w}_x - \lambda\left((1-\gamma)\,\mathbf{Z}'\mathbf{Z} + \gamma\mathbf{I}\right)\mathbf{w}_z &= 0 \end{aligned}$$

- Now there holds that $\mathbf{w}_x'\left(\mathbf{X}'\mathbf{X} + \gamma\mathbf{I}\right)\mathbf{w}_x = \mathbf{w}_z'\left(\mathbf{Z}'\mathbf{Z} + \gamma\mathbf{I}\right)\mathbf{w}_z$ and $\lambda = \frac{\mathbf{w}_x'\mathbf{X}'\mathbf{Z}\mathbf{w}_z}{\sqrt{\mathbf{w}_x'\left((1-\gamma)\mathbf{X}'\mathbf{X}+\gamma\mathbf{I}\right)\mathbf{w}_x}\sqrt{\mathbf{w}_z'\left((1-\gamma)\mathbf{Z}'\mathbf{Z}+\gamma\mathbf{I}\right)\mathbf{w}_z}}$

Regularised CCA

- In matrix notation:

$$\begin{pmatrix} \mathbf{0} & \mathbf{X}'\mathbf{Z} \\ \mathbf{Z}'\mathbf{X} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{w}_x \\ \mathbf{w}_z \end{pmatrix}$$

$$= \lambda \begin{pmatrix} (1-\gamma)\,\mathbf{X}'\mathbf{X} + \gamma\mathbf{I} & \mathbf{0} \\ \mathbf{0} & (1-\gamma)\,\mathbf{Z}'\mathbf{Z} + \gamma\mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{w}_x \\ \mathbf{w}_z \end{pmatrix}$$

- By increasing γ we reduce the size of the 'pattern space', and the stability of the correlation found increases
- On the other hand, we introduce a bias: we do not really maximise the correlation anymore

Regularised CCA

- Limit case 1: for $\gamma = 0$: unregularised CCA is retrieved
- Limit case 2: for $\gamma = 1$:

$$\begin{pmatrix} \mathbf{0} & \mathbf{X}'\mathbf{Z} \\ \mathbf{Z}'\mathbf{X} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{w}_x \\ \mathbf{w}_z \end{pmatrix} = \lambda \begin{pmatrix} \mathbf{w}_x \\ \mathbf{w}_z \end{pmatrix}$$

- This amounts to maximising the *covariance* between the projections on the respective weight vectors:

$$\lambda = \frac{\mathbf{w}_x'\mathbf{X}'\mathbf{Z}\mathbf{w}_z}{\sqrt{\mathbf{w}_x'\mathbf{w}_x}\sqrt{\mathbf{w}_z'\mathbf{w}_z}}$$

Regularised CCA

Some notes concerning framework and statistics:

- 'Correlation' cannot be written as an averaging pattern function
- For this reason it seems harder to study using a Rademacher type of analysis
- What *can* be studied is the covariance $\mathbf{w}_x'\mathbf{X}'\mathbf{Z}\mathbf{w}_z$ (this is an averaging pattern function)
- Hence, often this is what is done, even when this is not of direct interest in the optimisation problem

Kernel CCA

- Can we kernelise this?

$$\begin{pmatrix} \mathbf{0} & \mathbf{X}'\mathbf{Z} \\ \mathbf{Z}'\mathbf{X} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{w}_x \\ \mathbf{w}_z \end{pmatrix} = \lambda \begin{pmatrix} (1-\gamma)\,\mathbf{X}'\mathbf{X} + \gamma \mathbf{I} & \mathbf{0} \\ \mathbf{0} & (1-\gamma)\,\mathbf{Z}'\mathbf{Z} + \gamma \mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{w}_x \\ \mathbf{w}_z \end{pmatrix}$$

- Let's apply our general scheme...
- *First step:* the representer theorem (shown for $\mathbf{w}_x$ only):

$$\mathbf{w}_x = \mathbf{X}' \left(\frac{1}{\lambda\gamma} \left(\mathbf{Z}\mathbf{w}_z - \lambda\,(1-\gamma)\,\mathbf{X}\mathbf{w}_x \right) \right) = \mathbf{X}'\boldsymbol{\alpha}_x$$

- Similarly $\mathbf{w}_z = \mathbf{Z}'\boldsymbol{\alpha}_z$

Kernel CCA

Second step: plug this all in, and left-multiply:

$$\begin{pmatrix} \mathbf{X} & \mathbf{0} \\ \mathbf{0} & \mathbf{Z} \end{pmatrix} \begin{pmatrix} \mathbf{0} & \mathbf{X}'\mathbf{Z} \\ \mathbf{Z}'\mathbf{X} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{X}'\boldsymbol{\alpha}_x \\ \mathbf{Z}'\boldsymbol{\alpha}_z \end{pmatrix} =$$
$$\lambda \begin{pmatrix} \mathbf{X} & \mathbf{0} \\ \mathbf{0} & \mathbf{Z} \end{pmatrix} \begin{pmatrix} (1-\gamma)\,\mathbf{X}'\mathbf{X} + \gamma\mathbf{I} & \mathbf{0} \\ \mathbf{0} & (1-\gamma)\,\mathbf{Z}'\mathbf{Z} + \gamma\mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{X}'\boldsymbol{\alpha}_x \\ \mathbf{Z}'\boldsymbol{\alpha}_z \end{pmatrix}$$

$$\begin{pmatrix} \mathbf{0} & \mathbf{X}\mathbf{X}'\mathbf{Z}\mathbf{Z}' \\ \mathbf{Z}\mathbf{Z}'\mathbf{X}\mathbf{X}' & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\alpha}_x \\ \boldsymbol{\alpha}_z \end{pmatrix} =$$
$$\lambda \begin{pmatrix} (1-\gamma)\,\mathbf{X}\mathbf{X}'\mathbf{X}\mathbf{X}' + \gamma\mathbf{X}\mathbf{X}' & \mathbf{0} \\ \mathbf{0} & (1-\gamma)\,\mathbf{Z}\mathbf{Z}'\mathbf{Z}\mathbf{Z}' + \gamma\mathbf{Z}\mathbf{Z}' \end{pmatrix} \begin{pmatrix} \boldsymbol{\alpha}_x \\ \boldsymbol{\alpha}_z \end{pmatrix}$$

Kernel CCA

Third step: apply kernel trick

$$
\begin{pmatrix} \mathbf{0} & \mathbf{K}_x\mathbf{K}_z \\ \mathbf{K}_z\mathbf{K}_x & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\alpha}_x \\ \boldsymbol{\alpha}_z \end{pmatrix} =
$$

$$
\lambda \begin{pmatrix} (1-\gamma)\,\mathbf{K}_x^2 + \gamma\mathbf{K}_x & \mathbf{0} \\ \mathbf{0} & (1-\gamma)\,\mathbf{K}_z^2 + \gamma\mathbf{K}_z \end{pmatrix} \begin{pmatrix} \boldsymbol{\alpha}_x \\ \boldsymbol{\alpha}_z \end{pmatrix}
$$

Finally, assuming full rank of the kernels:

$$
\begin{pmatrix} \mathbf{0} & \mathbf{K}_z \\ \mathbf{K}_x & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\alpha}_x \\ \boldsymbol{\alpha}_z \end{pmatrix} =
$$

$$
\lambda \begin{pmatrix} (1-\gamma)\,\mathbf{K}_x + \gamma\mathbf{I} & \mathbf{0} \\ \mathbf{0} & (1-\gamma)\,\mathbf{K}_z + \gamma\mathbf{I} \end{pmatrix} \begin{pmatrix} \boldsymbol{\alpha}_x \\ \boldsymbol{\alpha}_z \end{pmatrix}
$$

Kernel CCA

- Note: it makes sense to normalise the weight vectors such that $\mathbf{w}_x'\mathbf{X}'\mathbf{X}\mathbf{w}_x = \mathbf{w}_z'\mathbf{Z}'\mathbf{Z}\mathbf{w}_z = 1$, or in terms of the dual vectors

$$\begin{aligned} \alpha_x'\mathbf{X}\mathbf{X}'\mathbf{X}\mathbf{X}'\alpha_x = \alpha_z'\mathbf{Z}\mathbf{Z}'\mathbf{Z}\mathbf{Z}'\mathbf{w}_z &= 1 \\ \alpha_x'\mathbf{K}_x^2\alpha_x = \alpha_z'\mathbf{K}_z^2\alpha_z &= 1 \end{aligned}$$

- In summary:
  - $\mathbf{w}_x = \mathbf{X}'\alpha_x$ and $\mathbf{w}_z = \mathbf{Z}'\alpha_z$ can be used to project new data points on these weight vectors (as before), as

$$\mathbf{x}'\mathbf{w}_x = \sum_{i=1}^{n} \alpha_{x,i} k_x(x, x_i) \text{ and } \mathbf{z}'\mathbf{w}_z = \sum_{i=1}^{n} \alpha_{z,i} k_z(x, x_i)$$

  - The algorithm itself finds the dual vectors relying on kernels only

Kernel CCA – extracting several features

- All this was for just the maximal correlation
- Often different views on the same objects have more than one ‘factor’ in common
- Hence, we want more than one such pair of weight vectors (or equivalently dual vectors)
- Easily achieved by taking more than one eigenvector
- Consecutive eigenvectors correspond to weight vectors with decreasing correlations (but potentially still large)

Kernel CCA – applications

- Potential applications:
- Cross-language retrieval: which features underly different translated versions of the same texts? (See project!)
  Way to approach it:
  - find a set of eigenvectors of the CCA eigenvalue problem
  - project all documents on these eigenvalues
  - use these as representations
  - this is a more language-independent representation, where semantically similar texts have similar representations
- Image retrieval: which features underly both images and their captions?
- Which features explain both the DNA upstream region of a gene and its expression behaviour?

I hope you learned something about...

1. Little Green Men

I hope you learned something about...

1. Little Green Men
2. Prophecies (by whoever or whatever...)

I hope you learned something about...

1. Little Green Men
2. Prophecies (by whoever or whatever...)
3. Where to find the corn crake

I hope you learned something about...

1. Little Green Men
2. Prophecies (by whoever or whatever...)
3. Where to find the corn crake
4. And some other things about pattern analysis and statistical learning...

Thanks!

- Questions? (if there is time)
- I'll be here still on Monday and Tuesday until noon (catch me if you have more questions)
- Important general references:
  - John Shawe-Taylor and Nello Cristianini: Kernel methods for pattern analysis, Cambridge University Press, 2004
  - Other joint work with JST and NC...
  - More references to come on the website