Analysing the Immune System with Fisher Features

John Shawe-Taylor

Department of Computer Science
University College London

WITMSE, Helsinki, September 2016

Experiment

- β chain CDR3 TCR repertoire sequenced from CD4 spleen cells.
  - unimmunised mice
  - 'early' immunised mice (5,7 and 14 days) immunised with Complete Freund's Adjuvant (CFA) or ovalbumin (OVA) with CFA
  - Isolate CD4 cells, amplify (NO barcoding, multiplex) and sequence on HiSeq
- Can we detect OVA vs. no-OVA from the sequences collected?
- Can we detect from a sample of T-cell sequences if an animal has been exposed to a pathogen?

Experiment

- β chain CDR3 TCR repertoire sequenced from CD4 spleen cells.
  - unimmunised mice
  - 'early' immunised mice (5,7 and 14 days) immunised with Complete Freund's Adjuvant (CFA) or ovalbumin (OVA) with CFA
  - Isolate CD4 cells, amplify (NO barcoding, multiplex) and sequence on HiSeq
- Can we detect OVA vs. no-OVA from the sequences collected?
- Can we detect from a sample of T-cell sequences if an animal has been exposed to a pathogen?

Experiment

- β chain CDR3 TCR repertoire sequenced from CD4 spleen cells.
  - unimmunised mice
  - 'early' immunised mice (5,7 and 14 days) immunised with Complete Freund's Adjuvant (CFA) or ovalbumin (OVA) with CFA
  - Isolate CD4 cells, amplify (NO barcoding, multiplex) and sequence on HiSeq
- Can we detect OVA vs. no-OVA from the sequences collected?
- Can we detect from a sample of T-cell sequences if an animal has been exposed to a pathogen?

Experiment

- β chain CDR3 TCR repertoire sequenced from CD4 spleen cells.
  - unimmunised mice
  - 'early' immunised mice (5,7 and 14 days) immunised with Complete Freund's Adjuvant (CFA) or ovalbumin (OVA) with CFA
  - Isolate CD4 cells, amplify (NO barcoding, multiplex) and sequence on HiSeq
- Can we detect OVA vs. no-OVA from the sequences collected?
- Can we detect from a sample of T-cell sequences if an animal has been exposed to a pathogen?

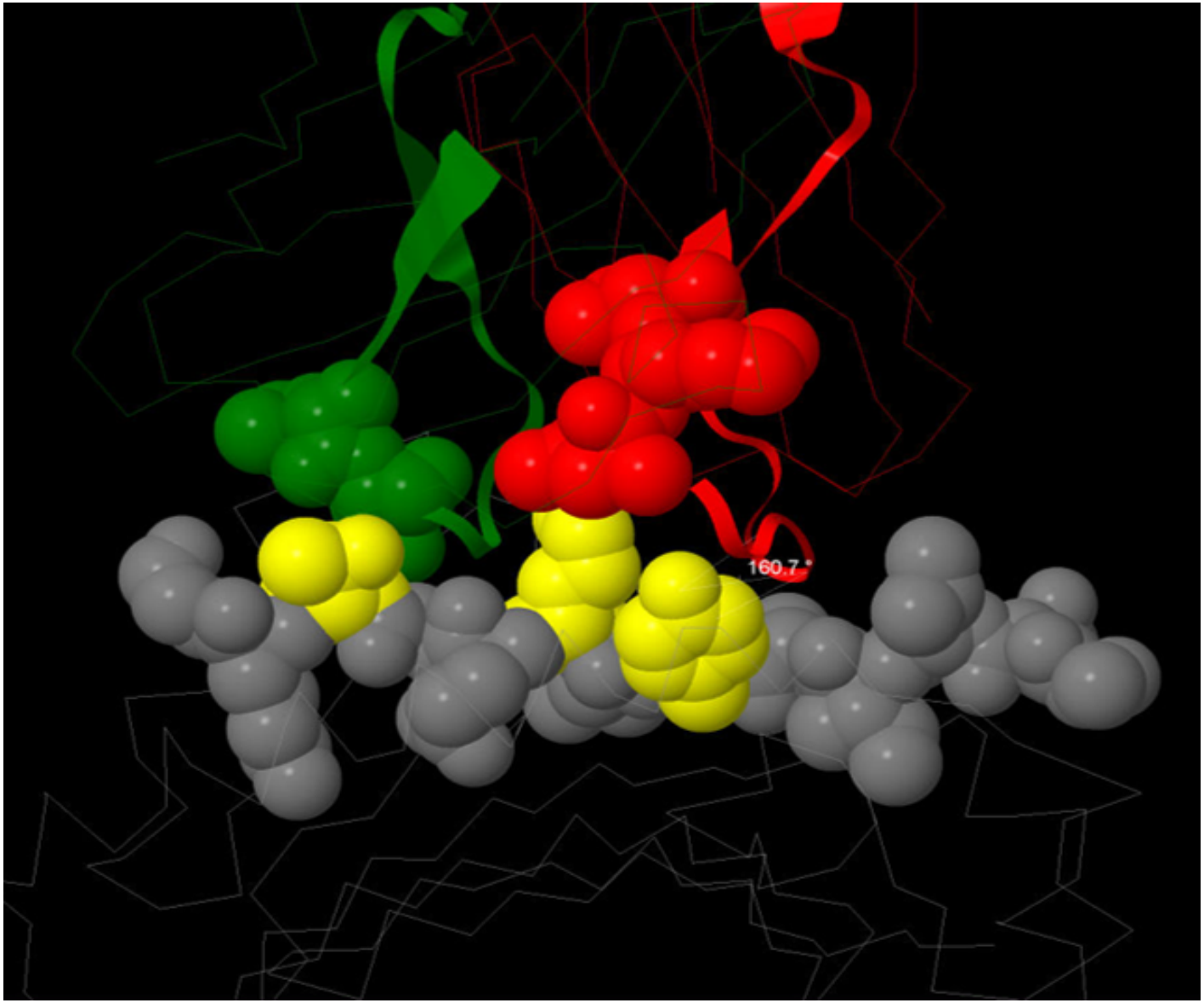
Experiment

- β chain CDR3 TCR repertoire sequenced from CD4 spleen cells.
  - unimmunised mice
  - 'early' immunised mice (5,7 and 14 days) immunised with Complete Freund's Adjuvant (CFA) or ovalbumin (OVA) with CFA
  - Isolate CD4 cells, amplify (NO barcoding, multiplex) and sequence on HiSeq
- Can we detect OVA vs. no-OVA from the sequences collected?
- Can we detect from a sample of T-cell sequences if an animal has been exposed to a pathogen?

Experiment

- β chain CDR3 TCR repertoire sequenced from CD4 spleen cells.
  - unimmunised mice
  - 'early' immunised mice (5,7 and 14 days) immunised with Complete Freund's Adjuvant (CFA) or ovalbumin (OVA) with CFA
  - Isolate CD4 cells, amplify (NO barcoding, multiplex) and sequence on HiSeq
- Can we detect OVA vs. no-OVA from the sequences collected?
- Can we detect from a sample of T-cell sequences if an animal has been exposed to a pathogen?

Nice picture



Observations

- Immune reaction occurs by T-cells binding to the antigen and then becoming amplified
- Single sequence signatures are not present
- Immune reaction of each mouse is different to the same antigen
- Reaction appears to be distributed: i.e. many different T-cells are amplified

Thomas N, Best K, Cinelli M, Reich-Zeliger S, Gal H, Shifrut E, Madi A, Friedman N, Shawe-Taylor J, Chain B. Tracking global changes induced in the CD4 T-cell receptor repertoire by immunization with a complex antigen using short stretches of CDR3 protein sequence. Bioinformatics. 2014 ;30(22):3181-8.

Observations

- Immune reaction occurs by T-cells binding to the antigen and then becoming amplified
- Single sequence signatures are not present
- Immune reaction of each mouse is different to the same antigen
- Reaction appears to be distributed: i.e. many different T-cells are amplified

Thomas N, Best K, Cinelli M, Reich-Zeliger S, Gal H, Shifrut E, Madi A, Friedman N, Shawe-Taylor J, Chain B. Tracking global changes induced in the CD4 T-cell receptor repertoire by immunization with a complex antigen using short stretches of CDR3 protein sequence. Bioinformatics. 2014 ;30(22):3181-8.

Observations

- Immune reaction occurs by T-cells binding to the antigen and then becoming amplified
- Single sequence signatures are not present
- Immune reaction of each mouse is different to the same antigen
- Reaction appears to be distributed: i.e. many different T-cells are amplified

Thomas N, Best K, Cinelli M, Reich-Zeliger S, Gal H, Shifrut E, Madi A, Friedman N, Shawe-Taylor J, Chain B. Tracking global changes induced in the CD4 T-cell receptor repertoire by immunization with a complex antigen using short stretches of CDR3 protein sequence. Bioinformatics. 2014 ;30(22):3181-8.

Observations

- Immune reaction occurs by T-cells binding to the antigen and then becoming amplified
- Single sequence signatures are not present
- Immune reaction of each mouse is different to the same antigen
- Reaction appears to be distributed: i.e. many different T-cells are amplified

Thomas N, Best K, Cinelli M, Reich-Zeliger S, Gal H, Shifrut E, Madi A, Friedman N, Shawe-Taylor J, Chain B. Tracking global changes induced in the CD4 T-cell receptor repertoire by immunization with a complex antigen using short stretches of CDR3 protein sequence. Bioinformatics. 2014 ;30(22):3181-8.

Aim of this talk

- Introduce Fisher kernels as a method of identifying features for solving this problem
- Show how feature selection learning algorithms are effective in identifying useful sets of features
- Explore how we can search large feature sets efficiently
- Consider searching feature sets that are not a priori computed
- Results support biological hypothesis that short protein sequences are critical to the T-cell function

Aim of this talk

- Introduce Fisher kernels as a method of identifying features for solving this problem
- Show how feature selection learning algorithms are effective in identifying useful sets of features
- Explore how we can search large feature sets efficiently
- Consider searching feature sets that are not a priori computed
- Results support biological hypothesis that short protein sequences are critical to the T-cell function

Aim of this talk

- Introduce Fisher kernels as a method of identifying features for solving this problem
- Show how feature selection learning algorithms are effective in identifying useful sets of features
- Explore how we can search large feature sets efficiently
- Consider searching feature sets that are not a priori computed
- Results support biological hypothesis that short protein sequences are critical to the T-cell function

Aim of this talk

- Introduce Fisher kernels as a method of identifying features for solving this problem
- Show how feature selection learning algorithms are effective in identifying useful sets of features
- Explore how we can search large feature sets efficiently
- Consider searching feature sets that are not a priori computed
- Results support biological hypothesis that short protein sequences are critical to the T-cell function

Aim of this talk

- Introduce Fisher kernels as a method of identifying features for solving this problem
- Show how feature selection learning algorithms are effective in identifying useful sets of features
- Explore how we can search large feature sets efficiently
- Consider searching feature sets that are not a priori computed
- Results support biological hypothesis that short protein sequences are critical to the T-cell function

Kernels from Probabilistic Models

- If we consider learning a representation as a pre-processing stage, it is natural to consider modelling the data with a probabilistic model
- There are then two main methods of defining kernels from probabilistic models:
  - Averaging over a model class - i.e. each model gives one feature:

$$\kappa(x, z) = \sum_{m \in M} P(x|m)P(z|m)P_M(m)$$

    also known as the marginalisation kernel.
  - Fisher kernels for cases where the model is determined by a real parameter vector
- Give a quick (tutorial) example of the Fisher kernel

Kernels from Probabilistic Models

- If we consider learning a representation as a pre-processing stage, it is natural to consider modelling the data with a probabilistic model
- There are then two main methods of defining kernels from probabilistic models:
  - Averaging over a model class - i.e. each model gives one feature:

$$\kappa(x, z) = \sum_{m \in M} P(x|m)P(z|m)P_M(m)$$

    also known as the marginalisation kernel.
  - Fisher kernels for cases where the model is determined by a real parameter vector
- Give a quick (tutorial) example of the Fisher kernel

Kernels from Probabilistic Models

- If we consider learning a representation as a pre-processing stage, it is natural to consider modelling the data with a probabilistic model
- There are then two main methods of defining kernels from probabilistic models:
  - Averaging over a model class - i.e. each model gives one feature:

$$\kappa(x, z) = \sum_{m \in M} P(x|m)P(z|m)P_M(m)$$

    also known as the marginalisation kernel.
  - Fisher kernels for cases where the model is determined by a real parameter vector
- Give a quick (tutorial) example of the Fisher kernel

Fisher kernels

- We assume the model is parametrised according to some parameters: consider the simple example of a 1-dim Gaussian distribution parametrised by μ and σ:

$$M = \left\{ P(x|\theta) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2} \right) : \theta = (\mu, \sigma) \in \mathbb{R}^2 \right\}.$$

- The Fisher score vector is the derivative of the log likelihood of an input x wrt the parameters:

$$\log \mathcal{L}_{(\mu,\sigma)}(x) = -\frac{(x-\mu)^2}{2\sigma^2} - \frac{1}{2}\log(2\pi\sigma).$$

Fisher kernels

- We assume the model is parametrised according to some parameters: consider the simple example of a 1-dim Gaussian distribution parametrised by μ and σ:

$$M = \left\{ P(x|\theta) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2} \right) : \theta = (\mu, \sigma) \in \mathbb{R}^2 \right\}.$$

- The Fisher score vector is the derivative of the log likelihood of an input x wrt the parameters:

$$\log \mathcal{L}_{(\mu,\sigma)}(x) = -\frac{(x-\mu)^2}{2\sigma^2} - \frac{1}{2}\log(2\pi\sigma).$$

Fisher kernels

- Hence the score vector is given by:

$$\mathbf{g}\left(\theta^0, x\right) = \left(\frac{(x-\mu_0)}{\sigma_0^2}, \frac{(x-\mu_0)^2}{\sigma_0^3} - \frac{1}{2\sigma_0}\right).$$

- Taking $\mu_0 = 0$ and $\sigma_0 = 1$ the feature embedding is given by:

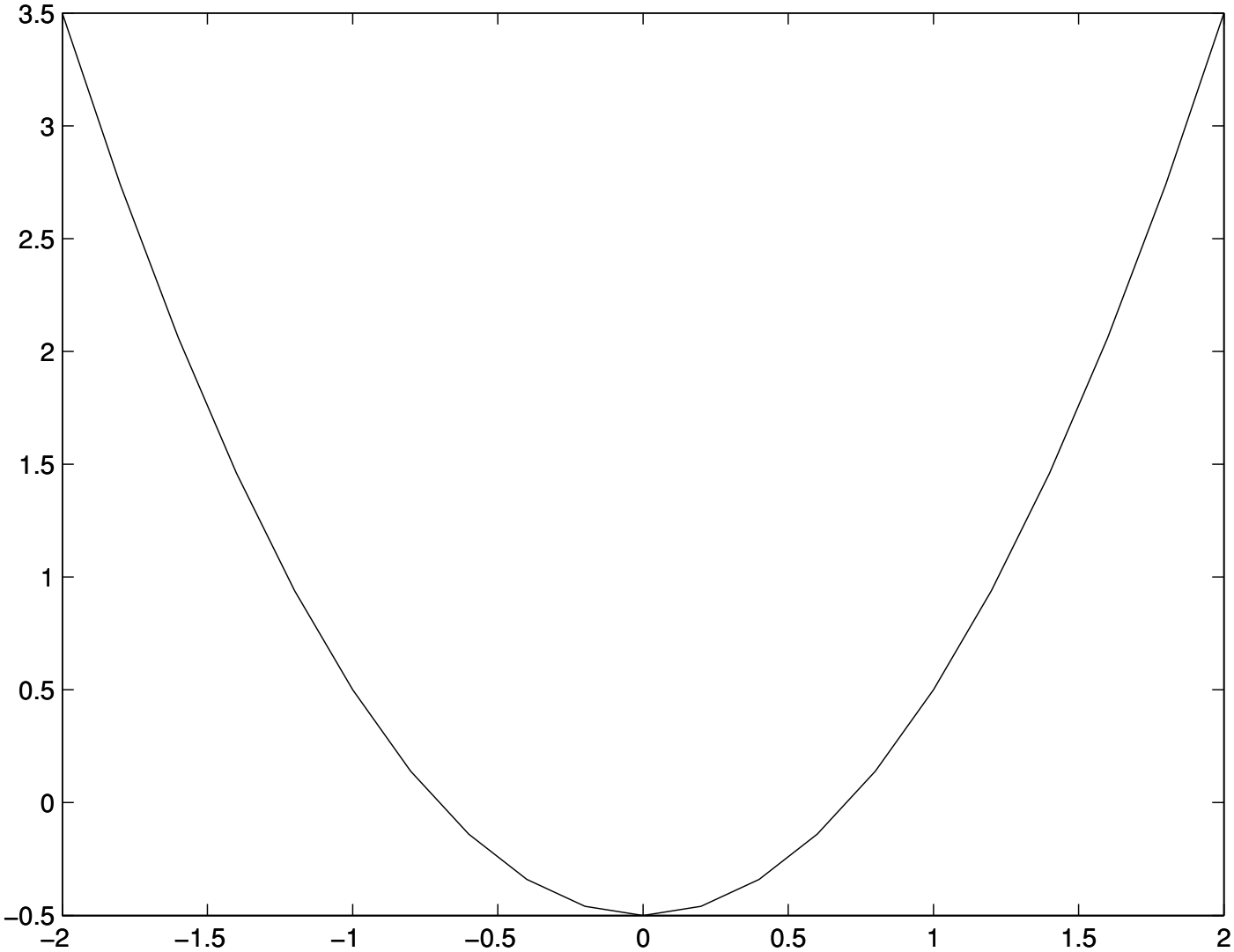
Fisher kernels

- Hence the score vector is given by:

$$\mathbf{g}\left(\theta^0, x\right) = \left(\frac{(x-\mu_0)}{\sigma_0^2}, \frac{(x-\mu_0)^2}{\sigma_0^3} - \frac{1}{2\sigma_0}\right).$$

- Taking $\mu_0 = 0$ and $\sigma_0 = 1$ the feature embedding is given by:

Fisher kernels



String kernels as Fisher kernels

- We can consider a Markov model of generating text conditioned on the previous $k-1$-characters The probability of a string d being generated by the model is therefore

$$P(d) = \prod_{j=k}^{|d|} p_{d[j-k+1:j-1] \to d_j},$$

- Taking the uniform distribution model gives the class of string kernels - but these can now be learned based on a corpus
- can extend to probabilistic Finite State Automata learned from the corpus
- results competitive with tfidf BoWs on Reuters, with some improvements in average precision

⋆ C. Saunders, J. Shawe-Taylor and A. Vinokourov (2003) String Kernels, Fisher Kernels and Finite State Automata, NIPS 15.

String kernels as Fisher kernels

- We can consider a Markov model of generating text conditioned on the previous $k-1$-characters The probability of a string d being generated by the model is therefore

$$P(d) = \prod_{j=k}^{|d|} p_{d[j-k+1:j-1] \to d_j},$$

- Taking the uniform distribution model gives the class of string kernels - but these can now be learned based on a corpus
- can extend to probabilistic Finite State Automata learned from the corpus
- results competitive with tfidf BoWs on Reuters, with some improvements in average precision

⋆ C. Saunders, J. Shawe-Taylor and A. Vinokourov (2003) String Kernels, Fisher Kernels and Finite State Automata, NIPS 15.

String kernels as Fisher kernels

- We can consider a Markov model of generating text conditioned on the previous $k-1$-characters The probability of a string d being generated by the model is therefore

$$P(d) = \prod_{j=k}^{|d|} p_{d[j-k+1:j-1] \to d_j},$$

- Taking the uniform distribution model gives the class of string kernels - but these can now be learned based on a corpus
- can extend to probabilistic Finite State Automata learned from the corpus
- results competitive with tfidf BoWs on Reuters, with some improvements in average precision

⋆ C. Saunders, J. Shawe-Taylor and A. Vinokourov (2003) String Kernels, Fisher Kernels and Finite State Automata, NIPS 15.

String kernels as Fisher kernels

- We can consider a Markov model of generating text conditioned on the previous $k-1$-characters The probability of a string d being generated by the model is therefore

$$P(d) = \prod_{j=k}^{|d|} p_{d[j-k+1:j-1] \to d_j},$$

- Taking the uniform distribution model gives the class of string kernels - but these can now be learned based on a corpus
- can extend to probabilistic Finite State Automata learned from the corpus
- results competitive with tfidf BoWs on Reuters, with some improvements in average precision

⋆ C. Saunders, J. Shawe-Taylor and A. Vinokourov (2003) String Kernels, Fisher Kernels and Finite State Automata, NIPS 15.

Finite State Automata Fisher Kernels

- The generation is now over the transitions: the probability of a string d being generated by the model is therefore

$$P(d) = \prod_{j=1}^{|d|} p_{s_{j-1} \to s_j},$$

where s_j is the state of the automata after reading symbol d_j.

- Note that the state s_j will be indexed by some suffix of $d[1:j]$ and the transition probabilities from any state sum to 1.
- The structure of the FSA and the transition probabilities can be learned from data in order to tune the model to a particular application
- Using uniform transition probabilities corresponds to the string kernel

Finite State Automata Fisher Kernels

- The generation is now over the transitions: the probability of a string d being generated by the model is therefore

$$P(d) = \prod_{j=1}^{|d|} p_{s_{j-1} \to s_j},$$

  where s_j is the state of the automata after reading symbol d_j.
- Note that the state s_j will be indexed by some suffix of $d[1:j]$ and the transition probabilities from any state sum to 1.
- The structure of the FSA and the transition probabilities can be learned from data in order to tune the model to a particular application
- Using uniform transition probabilities corresponds to the string kernel

Finite State Automata Fisher Kernels

- The generation is now over the transitions: the probability of a string d being generated by the model is therefore

$$P(d) = \prod_{j=1}^{|d|} p_{s_{j-1} \to s_j},$$

  where s_j is the state of the automata after reading symbol d_j.
- Note that the state s_j will be indexed by some suffix of $d[1:j]$ and the transition probabilities from any state sum to 1.
- The structure of the FSA and the transition probabilities can be learned from data in order to tune the model to a particular application
- Using uniform transition probabilities corresponds to the string kernel

Finite State Automata Fisher Kernels

- The generation is now over the transitions: the probability of a string d being generated by the model is therefore

$$P(d) = \prod_{j=1}^{|d|} p_{s_{j-1} \to s_j},$$

  where s_j is the state of the automata after reading symbol d_j.
- Note that the state s_j will be indexed by some suffix of $d[1:j]$ and the transition probabilities from any state sum to 1.
- The structure of the FSA and the transition probabilities can be learned from data in order to tune the model to a particular application
- Using uniform transition probabilities corresponds to the string kernel

Creating Fisher Features

- We use the Fisher kernel to create a set of features: these correspond to particular transitions in the probabilistic model, eg particular subsequence counts in the string kernel or the subsequence corresponding to a given transition in the FSA.
- Our hypothesis for our particular application is that the immune reaction will be characterised by a small subset of short sequences.
- We will use a 1-norm regularised learning algorithm to perform feature selection.
- this has the advantage of performing feature selection, but also of being able to learn effectively in the presence of large numbers of features.

Creating Fisher Features

- We use the Fisher kernel to create a set of features: these correspond to particular transitions in the probabilistic model, eg particular subsequence counts in the string kernel or the subsequence corresponding to a given transition in the FSA.
- Our hypothesis for our particular application is that the immune reaction will be characterised by a small subset of short sequences.
- We will use a 1-norm regularised learning algorithm to perform feature selection.
- this has the advantage of performing feature selection, but also of being able to learn effectively in the presence of large numbers of features.

Creating Fisher Features

- We use the Fisher kernel to create a set of features: these correspond to particular transitions in the probabilistic model, eg particular subsequence counts in the string kernel or the subsequence corresponding to a given transition in the FSA.
- Our hypothesis for our particular application is that the immune reaction will be characterised by a small subset of short sequences.
- We will use a 1-norm regularised learning algorithm to perform feature selection.
- this has the advantage of performing feature selection, but also of being able to learn effectively in the presence of large numbers of features.

Creating Fisher Features

- We use the Fisher kernel to create a set of features: these correspond to particular transitions in the probabilistic model, eg particular subsequence counts in the string kernel or the subsequence corresponding to a given transition in the FSA.
- Our hypothesis for our particular application is that the immune reaction will be characterised by a small subset of short sequences.
- We will use a 1-norm regularised learning algorithm to perform feature selection.
- this has the advantage of performing feature selection, but also of being able to learn effectively in the presence of large numbers of features.

L_1 sparsity

- Rademacher complexity gives an alternative measure of function class complexity:

$$R_m(\mathcal{H}) = \mathbb{E}_S \mathbb{E}_{\sigma \in \{-1,+1\}^m} \left[\frac{2}{m} \sup_{h \in \mathcal{H}} \sum_{i=1}^{m} \sigma_i h(x_i) \right]$$

  where we assume the class $\mathcal{H}$ is closed under negation.

- Rademacher complexity is not increased by taking the convex closure of $\mathcal{H}$:

$$R_m(B\mathcal{C}(\mathcal{H})) \leq BR_m(\mathcal{H}) \quad \text{for}$$

$$\mathcal{C}(\mathcal{H}) = \left\{ \sum_i \alpha_i h_i : h_i \in \mathcal{H},\ \|\alpha\|_1 = 1 \right\}$$

Generalization error for linear classifiers

Using this definition we can bound the generalisation in terms of the margin distribution as with SVMs

$$\text{generalization error} \leq \sum_{i=1}^{m} \xi_i + BR_m(\mathcal{H}) + 2\sqrt{\frac{\log(1/\delta)}{2m}}$$

where $\mathcal{H}$ is the class of weak learners with range $[-1, 1]$ and $B = \sum_{i=1}^{T} \alpha_i$.
Note the ξ_i are the margin slack variables computed as

$$\xi_i = \left(1 - y_i \sum_{j=1}^{N} \alpha_j h_j(x_i)\right)_+$$

Linear programming machine

- Note that Rademacher complexity of N feature indicators is bounded by $1/m + 4\ln(Nm)/\sqrt{m}$
- The bound suggests an optimisation similar to that of SVMs.
- seeks linear function in a feature space defined explicitly.
- For example using the 1-norm it seeks $\mathbf{w}$ to solve

$$\begin{aligned} \min_{\mathbf{w},b,\xi} \quad & \|\mathbf{w}\|_1 + C\sum_{i=1}^m \xi_i \\ \text{subject to} \quad & y_i\left(\langle \mathbf{w}, \mathbf{x}_i\rangle + b\right) \geq 1 - \xi_i,\ \xi_i \geq 0, \\ & i = 1, \ldots, m. \end{aligned}$$

Dual form

- Can explicitly optimise margin with 1-norm fixed:

$$\begin{aligned} \max_{\rho,\mathbf{a},\xi} \quad & \rho - D\sum_{i=1}^{m}\xi_i \\ \text{subject to} \quad & y_i\mathbf{H}_i\mathbf{a} \geq \rho - \xi_i,\ \xi_i \geq 0, a_j \geq 0 \\ & \sum_{j=1}^{N} a_j = 1. \end{aligned}$$

- Dual has the following form:

$$\begin{aligned} \min_{\beta,\mathbf{u}} \quad & \beta \\ \text{subject to} \quad & \sum_{i=1}^{m} u_i y_i \mathbf{H}_{ij} \leq \beta,\ j = 1,\ldots,N, \\ & \sum_{i=1}^{m} u_i = 1,\ 0 \leq u_i \leq D. \end{aligned}$$

(Demiriz, Bennett and S-T, 2001)

Linear programming boosting

1 initialise $u_i = 1/m, i = 1, \ldots, m$, $\beta = \infty$, $J = \emptyset$
2 choose $j^\star$ that maximises $f(j) = \sum_{i=1}^{m} u_i y_i \mathbf{H}_{ij}$
3 if $f(j^\star) \leq \beta$ solve primal restricted to J and exit
4 $J = J \cup \{j^\star\}$
5 Solve dual restricted to set J to give u_i, β
6 Go to 2

- Note that u_i is a distribution on the examples
- Each j added acts like an additional weak learner
- $f(j)$ is simply the weighted classification accuracy
- Hence gives 'boosting' algorithm - with previous weights updated satisfying error bound
- Guaranteed convergence and soft stopping criteria

Initial Results

- Applying the task of distinguishing OVA from non-OVA mice Fisher features improve accuracy from around 0.70 to 0.74.
- Using selected features in a Gaussian kernel with an SVM increases accuracies to 0.72 for string features and 0.83 for Fisher features.
- The selection criterion for including features into the model is: choose $j^\star$ that maximises $f(j) = \sum_{i=1}^{m} u_i y_i \mathbf{H}_{ij}$, where $\mathbf{u}$ are current dual variables.
- Suggests we may be more ambitious about including features from larger sets

Initial Results

- Applying the task of distinguishing OVA from non-OVA mice Fisher features improve accuracy from around 0.70 to 0.74.
- Using selected features in a Gaussian kernel with an SVM increases accuracies to 0.72 for string features and 0.83 for Fisher features.
- The selection criterion for including features into the model is: choose $j^\star$ that maximises $f(j) = \sum_{i=1}^{m} u_i y_i \mathbf{H}_{ij}$, where $\mathbf{u}$ are current dual variables.
- Suggests we may be more ambitious about including features from larger sets

Initial Results

- Applying the task of distinguishing OVA from non-OVA mice Fisher features improve accuracy from around 0.70 to 0.74.
- Using selected features in a Gaussian kernel with an SVM increases accuracies to 0.72 for string features and 0.83 for Fisher features.
- The selection criterion for including features into the model is: choose $j^\star$ that maximises $f(j) = \sum_{i=1}^{m} u_i y_i \mathbf{H}_{ij}$, where $\mathbf{u}$ are current dual variables.
- Suggests we may be more ambitious about including features from larger sets

Initial Results

- Applying the task of distinguishing OVA from non-OVA mice Fisher features improve accuracy from around 0.70 to 0.74.
- Using selected features in a Gaussian kernel with an SVM increases accuracies to 0.72 for string features and 0.83 for Fisher features.
- The selection criterion for including features into the model is: choose $j^\star$ that maximises $f(j) = \sum_{i=1}^{m} u_i y_i \mathbf{H}_{ij}$, where $\mathbf{u}$ are current dual variables.
- Suggests we may be more ambitious about including features from larger sets

Sequences of Transitions

- Consider features created by a sequence of transitions:

$$s_1 \rightarrow s_2 \ldots \rightarrow s_k$$

- If $h_{s_j \rightarrow s_{j+1}}$ is the feature corresponding to transition $s_j \rightarrow s_{j+1}$ then

$$\sum_{i=1}^{m} u_i y_i \sum_{j=1}^{k-1} h_{s_j \rightarrow s_{j+1}}(i) = \sum_{j=1}^{k-1} \sum_{i=1}^{m} u_i y_i h_{s_j \rightarrow s_{j+1}}(i) = \sum_{j=1}^{k-1} q_j$$

where q_j is a weighting for each edge of the FSA,

so can use dynamic programming to efficiently find the sequence of a specified length that should be selected.

Sequences of Transitions

- Consider features created by a sequence of transitions:

$$s_1 \to s_2 \ldots \to s_k$$

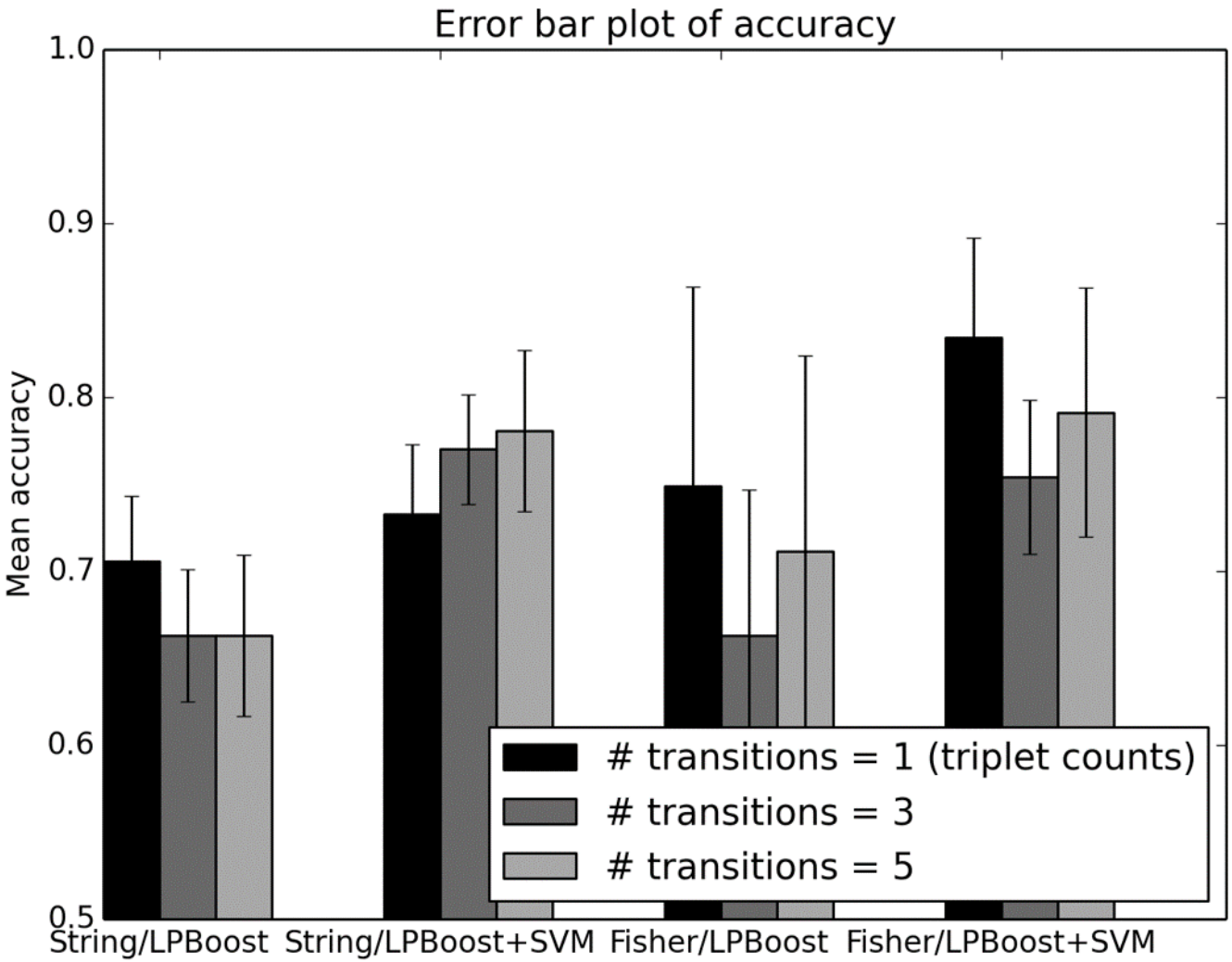
- If $h_{s_j \to s_{j+1}}$ is the feature corresponding to transition $s_j \to s_{j+1}$ then

$$\sum_{i=1}^{m} u_i y_i \sum_{j=1}^{k-1} h_{s_j \to s_{j+1}}(i) = \sum_{j=1}^{k-1} \sum_{i=1}^{m} u_i y_i h_{s_j \to s_{j+1}}(i) = \sum_{j=1}^{k-1} q_j$$

where q_j is a weighting for each edge of the FSA,

so can use dynamic programming to efficiently find the sequence of a specified length that should be selected.

Results with sequences



Sequences of Transitions

- This method efficiently searches a potentially very large space of features: eg 20^7 for the 5 transitions case
- but features correspond to sums of original features as approach does not restrict the sequences to be contiguous
- for interpretability would prefer to restrict to contiguous features
- this can be achieved by using the dynamic programming to suggest pairs of features that might be useful and then introduce a new state/transition to represent the contiguous feature

Sequences of Transitions

- This method efficiently searches a potentially very large space of features: eg 20^7 for the 5 transitions case
- but features correspond to sums of original features as approach does not restrict the sequences to be contiguous
- for interpretability would prefer to restrict to contiguous features
- this can be achieved by using the dynamic programming to suggest pairs of features that might be useful and then introduce a new state/transition to represent the contiguous feature

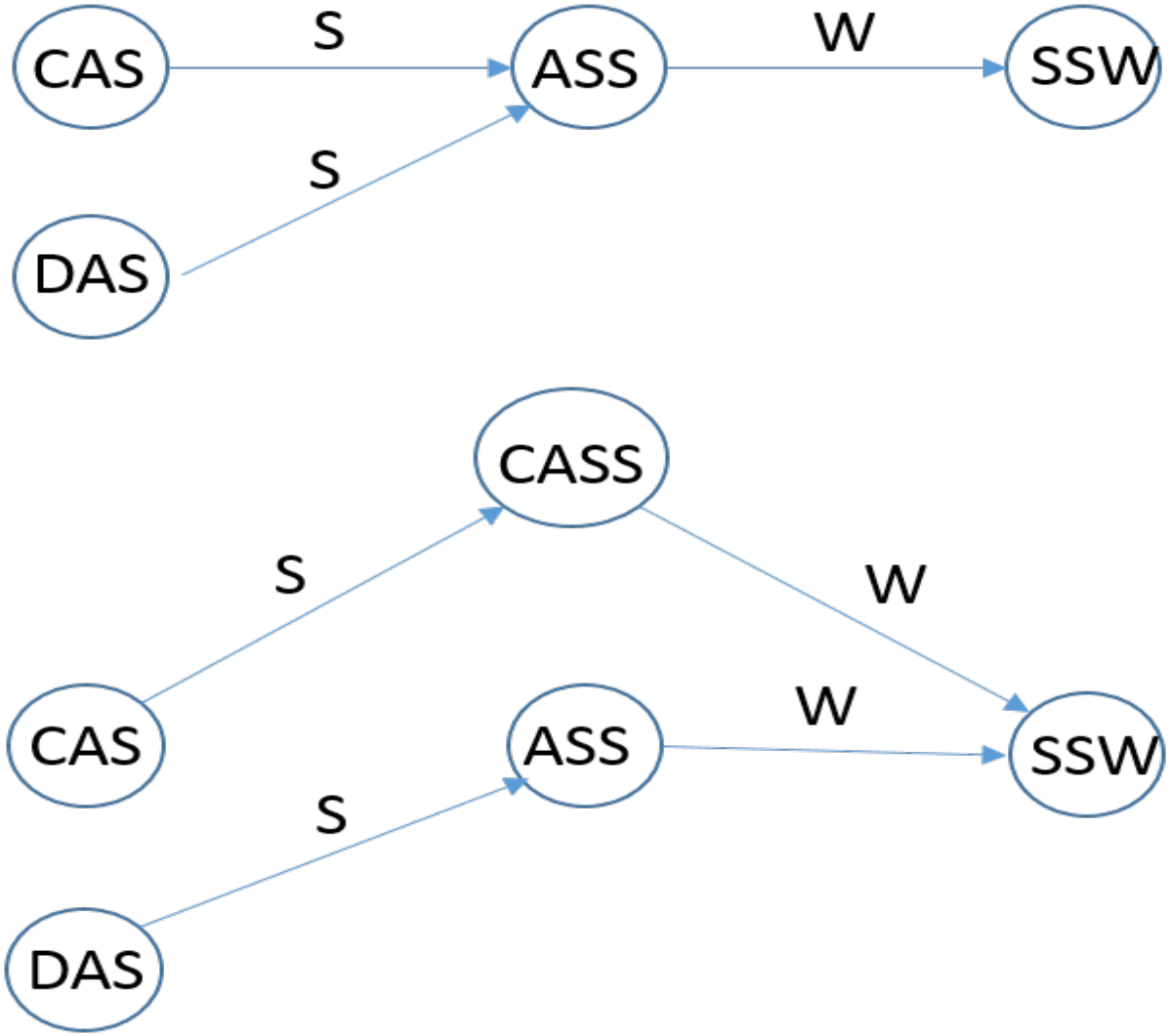
Sequences of Transitions

- This method efficiently searches a potentially very large space of features: eg 20^7 for the 5 transitions case
- but features correspond to sums of original features as approach does not restrict the sequences to be contiguous
- for interpretability would prefer to restrict to contiguous features
- this can be achieved by using the dynamic programming to suggest pairs of features that might be useful and then introduce a new state/transition to represent the contiguous feature

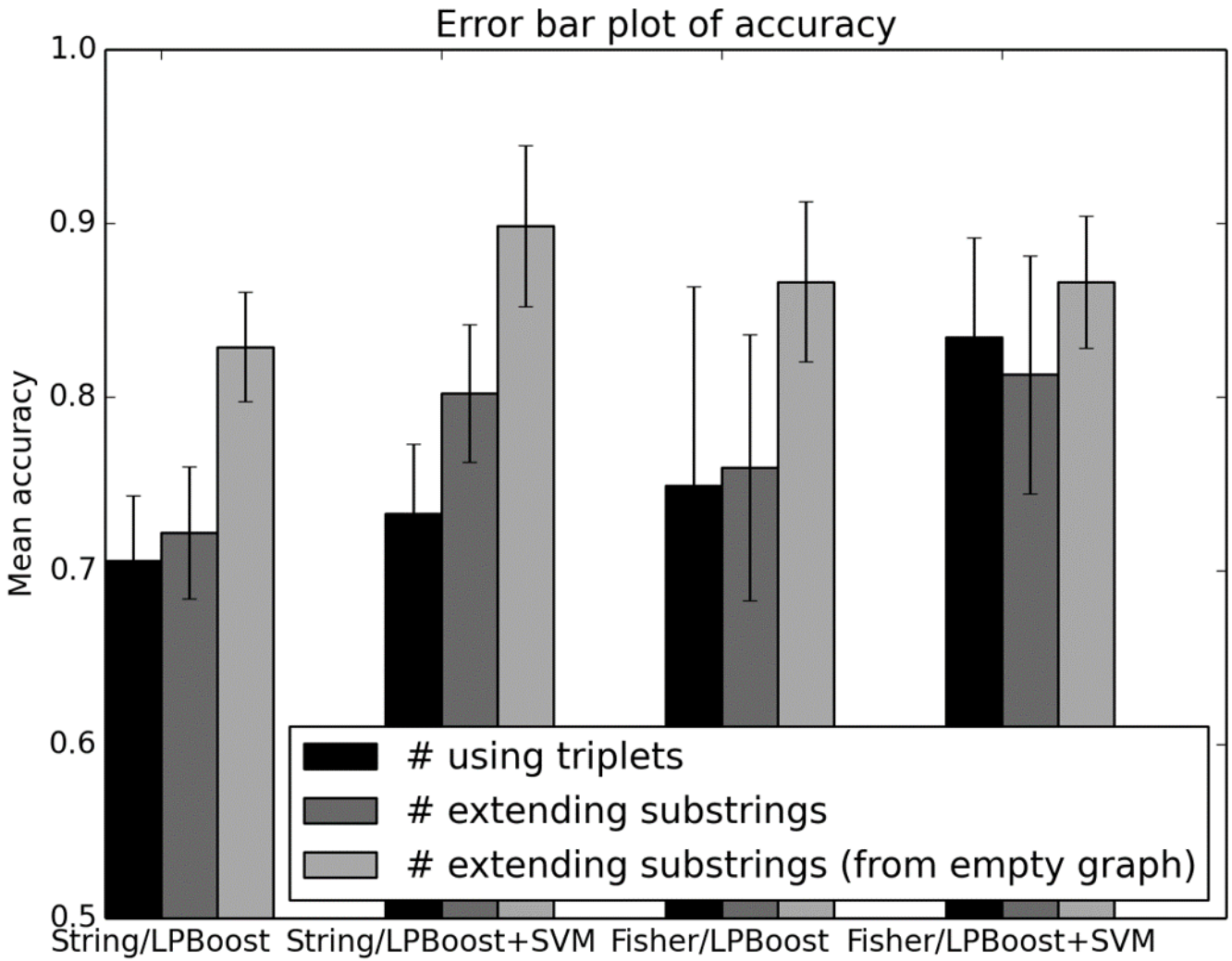
Sequences of Transitions

- This method efficiently searches a potentially very large space of features: eg 20^7 for the 5 transitions case
- but features correspond to sums of original features as approach does not restrict the sequences to be contiguous
- for interpretability would prefer to restrict to contiguous features
- this can be achieved by using the dynamic programming to suggest pairs of features that might be useful and then introduce a new state/transition to represent the contiguous feature

Adding states/transitions



Results with adding states/transitions



Average Size of the Final Graph

	# edges	# nodes
String (from empty)	304	15
Fisher (from empty)	321	16
String	8208	410
String	8368	418

Biological Conclusions

- The OVA response is diverse and predominantly private at the level of CDR3?
- OVA expanded CDR3?s have some sequence similarity
- Amino acid triplets provide features which in combination contribute to defining an OVA response
- Each triplet has a well-defined position along the CDR3
- Many selected triplets are found at the ends of the CDR3, within the sequence coded by V or J region genomic

Biological Conclusions

- The OVA response is diverse and predominantly private at the level of CDR3?
- OVA expanded CDR3?s have some sequence similarity
- Amino acid triplets provide features which in combination contribute to defining an OVA response
- Each triplet has a well-defined position along the CDR3
- Many selected triplets are found at the ends of the CDR3, within the sequence coded by V or J region genomic

Biological Conclusions

- The OVA response is diverse and predominantly private at the level of CDR3?
- OVA expanded CDR3?s have some sequence similarity
- Amino acid triplets provide features which in combination contribute to defining an OVA response
- Each triplet has a well-defined position along the CDR3
- Many selected triplets are found at the ends of the CDR3, within the sequence coded by V or J region genomic

Biological Conclusions

- The OVA response is diverse and predominantly private at the level of CDR3?
- OVA expanded CDR3?s have some sequence similarity
- Amino acid triplets provide features which in combination contribute to defining an OVA response
- Each triplet has a well-defined position along the CDR3
- Many selected triplets are found at the ends of the CDR3, within the sequence coded by V or J region genomic

Biological Conclusions

- The OVA response is diverse and predominantly private at the level of CDR3?
- OVA expanded CDR3?s have some sequence similarity
- Amino acid triplets provide features which in combination contribute to defining an OVA response
- Each triplet has a well-defined position along the CDR3
- Many selected triplets are found at the ends of the CDR3, within the sequence coded by V or J region genomic

Summary and Conclusions

- Consider an intriguing application of machine learning to analysing the immune system
- Consider the use of Fisher kernels as a method of generating potential features
- 1-norm regularisation combined with a hinge loss generates a boosting style algorithm
- Feature selection corresponds to weak learner selection: we have introduced methods for achieving this efficiently in large implicitly defined feature sets.

Summary and Conclusions

- Consider an intriguing application of machine learning to analysing the immune system
- Consider the use of Fisher kernels as a method of generating potential features
- 1-norm regularisation combined with a hinge loss generates a boosting style algorithm
- Feature selection corresponds to weak learner selection: we have introduced methods for achieving this efficiently in large implicitly defined feature sets.

Summary and Conclusions

- Consider an intriguing application of machine learning to analysing the immune system
- Consider the use of Fisher kernels as a method of generating potential features
- 1-norm regularisation combined with a hinge loss generates a boosting style algorithm
- Feature selection corresponds to weak learner selection: we have introduced methods for achieving this efficiently in large implicitly defined feature sets.

Summary and Conclusions

- Consider an intriguing application of machine learning to analysing the immune system
- Consider the use of Fisher kernels as a method of generating potential features
- 1-norm regularisation combined with a hinge loss generates a boosting style algorithm
- Feature selection corresponds to weak learner selection: we have introduced methods for achieving this efficiently in large implicitly defined feature sets.