Kernel methods for predicting protein-protein interactions

Hitomi Hasegawa

April 23, 2008

T-61.6070 Special Course in Bioinformatics I
Modeling of proteomics data

Outline

- Protein-Protein interactions
- Kernels and Support Vector Machines
- Spectrum Kernel
- Motif Kernel
- Pfam Kernel
- Pairwise Kernel
- Comparison

Protein-Protein Interactions

- Important for many biological functions
  - Proteins perform functions by interacting with other proteins
  - Information about interactions can help understand biological processes and diseases
- Interactions are of central importance for virtually every process in a living cell
- Many methods for detecting interaction available
  - High throughput methods based on Mass spectrometry can only provide partial information
  - Computational methods aim at discovering interactions not accessible to high throughput methods

Kernel Methods

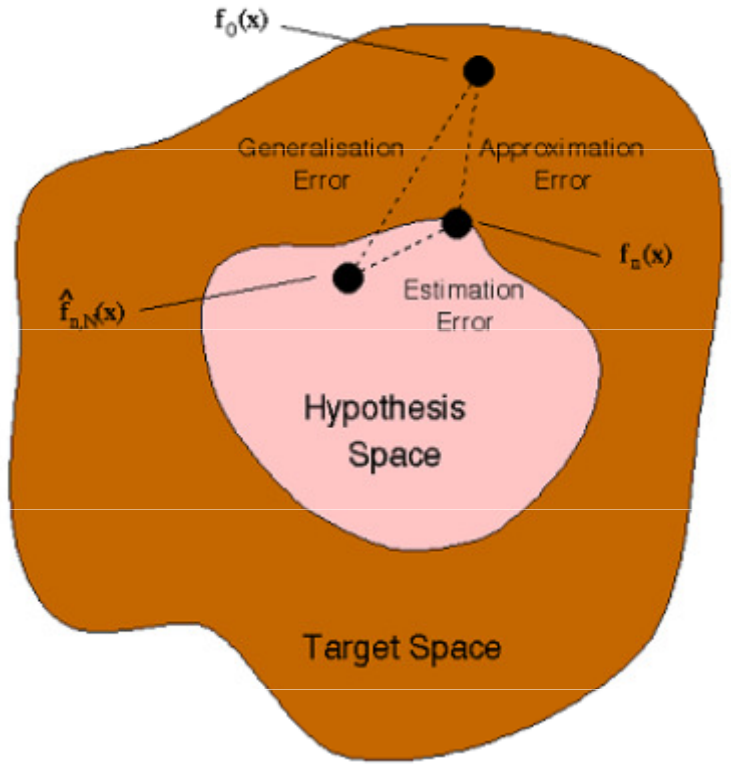
- Kernel methods and particularly support vector machines (SVM) proven useful in many classification problems in Bioinformatics

- In contrast to other machine learning methods, kernels such as SVM can handle non-vector inputs, such as variable length sequences or graph

- They derive their power form the ability to incorporate prior knowledge via their kernel function

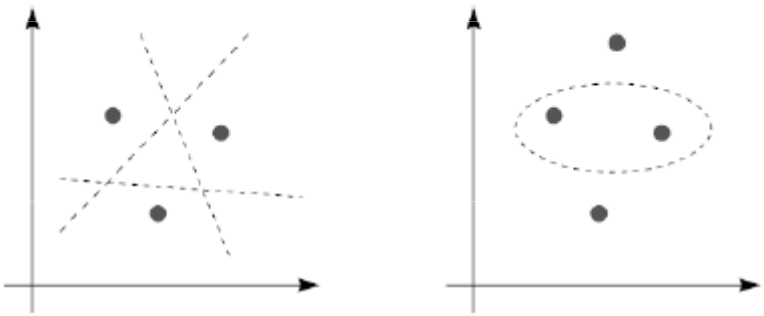
Support Vector Machines (SVM)

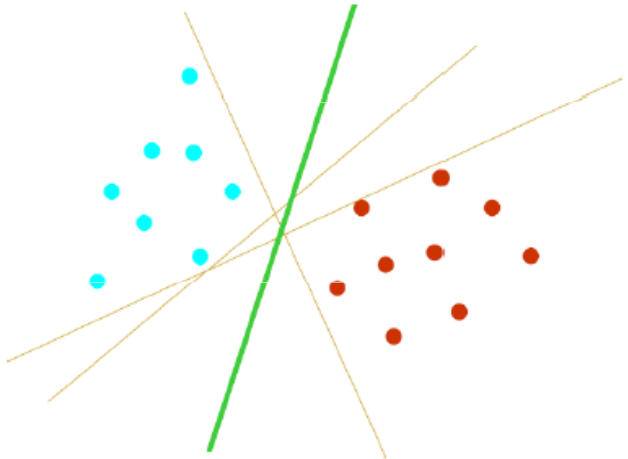
- Minimizes the generalization error rather than the error in the training data
- Its goal is to produce a classifier that will work well on unseen examples (it generalizes well)
- The generalization error is based on both the approximation error and the estimation error



VC Dimension

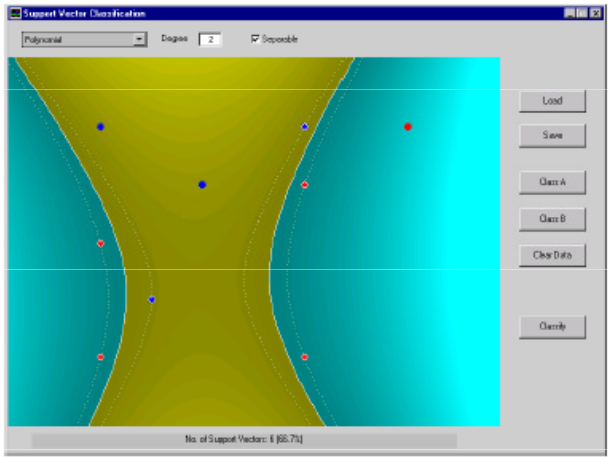
- VC dimension measures the capacity of a set of functions in which they either belong or do not belong to a certain group
- For protein-protein interactions consider a two class problem without loss of generality
- Proteins either interact with each other or not

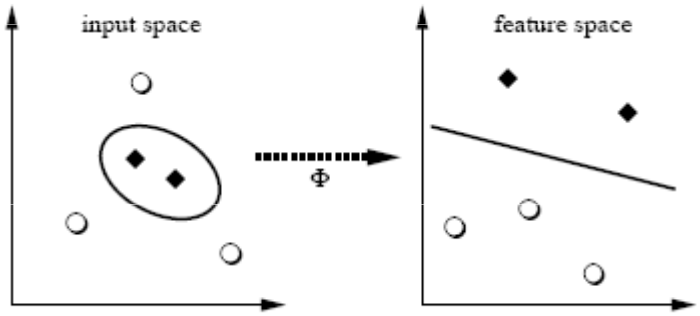




VC dimension cont'd

- In some cases linear functions do not suffice
- It can be generalized with the high dimensional feature space → curse of dimensionality
- Instead use a kernel function that performs operations on input space instead of feature space
- Then map input space to feature space via dot products





Kernels

- Large number of kernels available
- Selection via cross validation and bootstrapping
- Kernels also useful for
  - Microarray data
  - DNA
  - Graphs
  - Trees
  - Pattern recognition

Kernel methods for protein-protein interaction

- The methods I present today
  - Spectrum kernel
  - Motif kernel
  - Pfamm kernel
  - Pairwise kernel
- Discriminative approaches, either positive or negative
- These algorithms when given labeled training vectors learn a decision boundary to discriminate between the two classes

Spectrum Kernel

- Is a simple string kernel
- Composition computed with respect to length-k substrings (*kmers)*
- Its features counts the number of times that a common *kmer* appears in two sequences
  - K=5 & alphabet = 20 → each vector 5^20 elements
- The spectrum of the input sequence is the set of all k-length subsequences that it contains

Spectrum Kernel cont'd

- Feature map is indexed by all possible subsequences of length K form the alphabet
- The kernel can be computed efficiently with a trie data structure
- There is a variation to this kernel in which mismatches are allowed
- Complexity *O(pm^2)*
  - *Where p is the length of one protein and m is the number of proteins in the training set*

Motif Kernel

- Detects homology based on discrete sequence motifs
- Kernel defined by a dot product between sequences over the alphabet of amino acids → defined in terms of the sequence motifs that appear in a pair of sequences
- Counts how many times a discrete sequence motif matches a sequence
  - *Motif [AS}.DKF[FILMV] → matches ACDKFF, SRDKFI and SADKFV*
- Then mapping to the feature space takes place with i.e. PyML software

Motif Kernel cont'd

- Complexity *O(pqm^2)*
  - *Where p is the length of one protein, q is the number of motifs in the database, and m is the number of proteins in the training set*

Pfam Kernel

- Assumes that evolutionary conserved features within each protein of an interacting pair are responsible for the interaction

- Interacting pairs found in one region of the network will often have homologs in another region of the same network

- Evolutionary redundancy implies that features that have predictive value for a portion of a large network will also have a predictive value in the unknown part of the same network

Pfam Kernel cont'd

- Uses a set of hidden Markov models to represent the domain structure of a protein
- Computed by comparing each protein sequence with every HMM in Pfam database
- Each comparison results in an *E-value*
- *E-values* are used to reduce the size of the data set by eliminating matches that are below a certain level (i.e. 0.01)
- Data is mapped with the use of software such as Gist

Pairwise Kernels

- The previous methods only compare a pair of sequences (proteins)
- This can extended to solve pairs of proteins
  - *K((X1, X2), (X1', X2'))*
- The pairwise kernel can be constructed to express the similarity between pairs of proteins in terms of similarities between individual proteins
  - *K((X1, X2), (X1', X2')) = K'(X1, X1')K'(X2, X2') + K'(X1, X2') K'(X2, X1')*

Ben-Hur and Noble Pairwise Kernel

- In addition to combining Kernels, other data can be used to improve the methods

- I.e. Gene Ontology annotations, local properties of the network and homologous interactions in other species

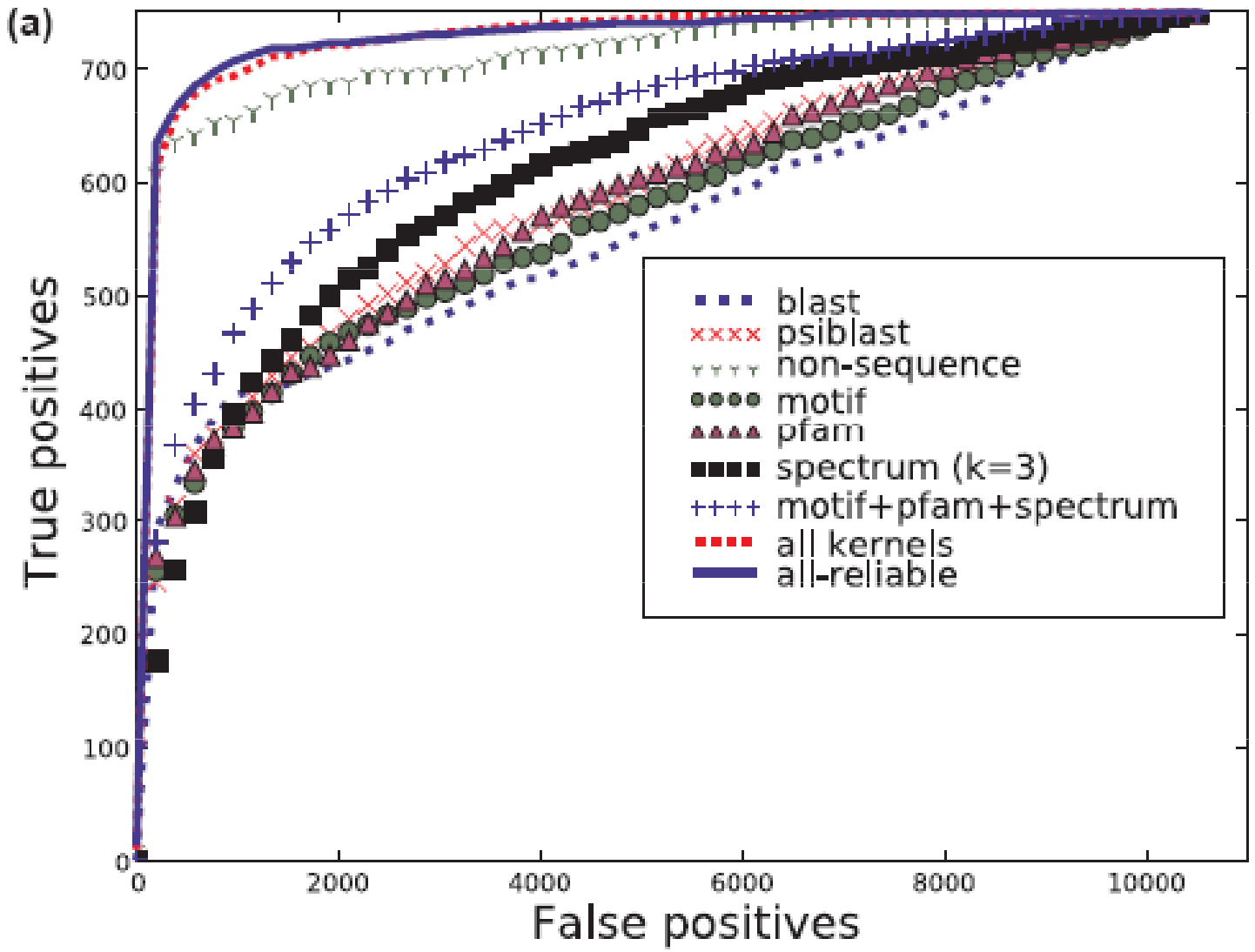
Kernel Comparison Experiment

- Predict yeast interactions using BIND database
  - BIND includes published interaction from high throughput experiments as well as curated entries derived from published papers
- Selected positive and negative examples
  - Positive, 10 517 physical interactions among 4233 yeast proteins → then elimnated self interactions
  - Negative, random non interacting pairs from the 4233 interacting proteins → such alarge set is likely to contain only few proteins that interact
- High throughput experiments can contain a large number of false positives
  - The source for the experiments were assays estimated to be correct 95% of the time

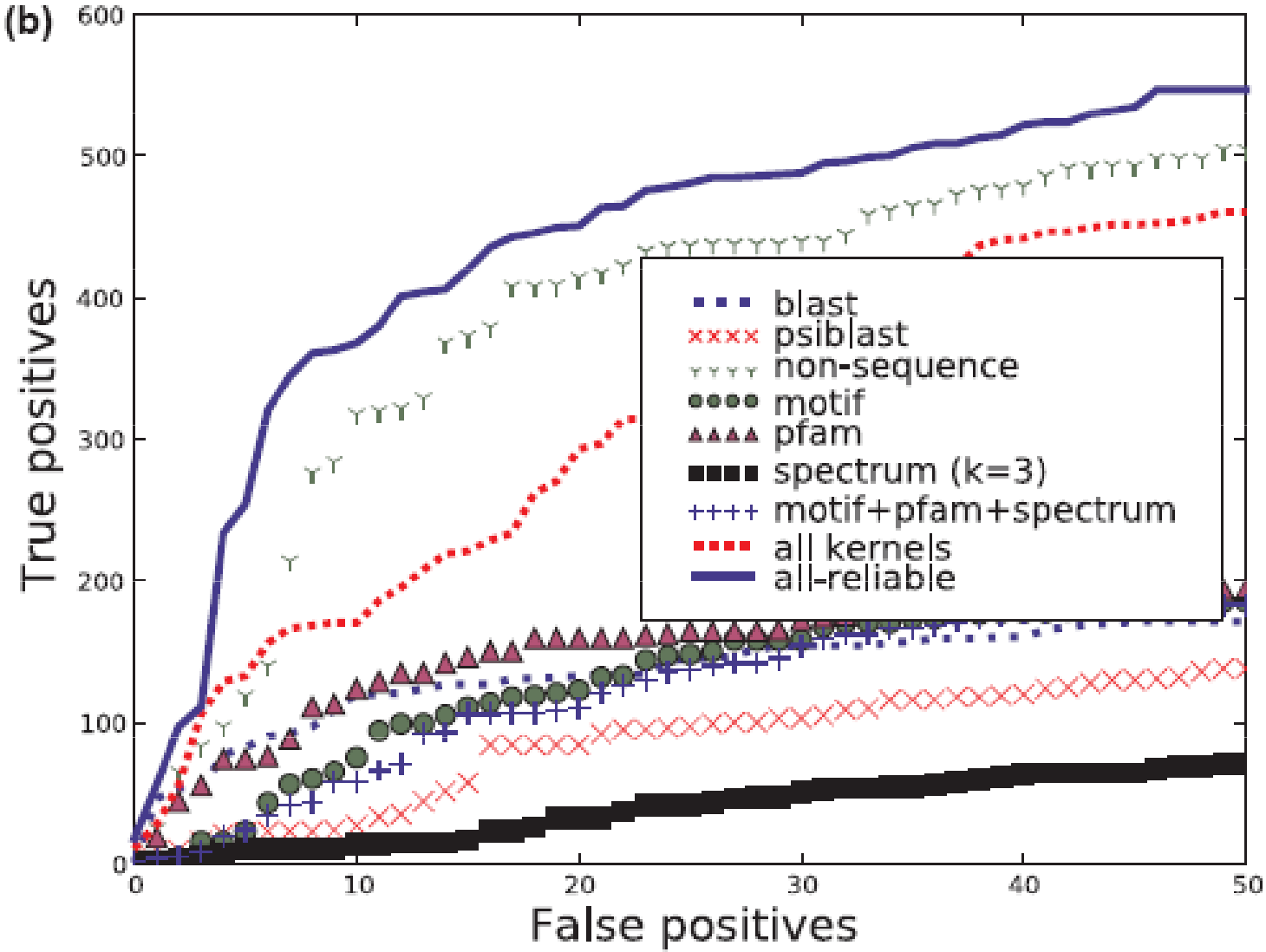
Kernel Comparison Experiment cont'd

- Two main metrics for comparison
  - ROC score, area under the receiver operating characteristic curve
  - ROC50 score, plot of true positives as a function of up to the first 50 false positives
- ROC50 score determines high confidence interactions
- ROC score focuses on high quality rather than high confidence

Kernel Comparison – ROC Score



Kernel Comparison – ROC50 Score



Summary

- SVM Kernel methods have been proven useful for biological problems including prediction of protein-protein interactions
- Many kernels available, the ones showed today are not much better than BLAST or PSI-BLAST by themselves
- However, if methods are combined in pairwise kernels and further data is filtered with some other data such as GO annotations, the results can be much better in some cases

References

- [1] A. Ben-Hur and W. Noble “Kernel methods for predicting protein-protein interactions” Bioinformatics Vol. 21, 2005
- [2] A. Ben-Hur and D. Brutlag “Remote homology detection: a motif based approach” Bioinformatics Vol. 19, 2003.
- [3] S. Gomez, W. Noble and A. Rzhetsky “Learning to predict protein-protein interactions from protein sequences” Vol. 19, 2003
- [4] C. Leslie, E. Ezkin and W. Noble “The spectrum kernel: a string kernel for SVM protein classification” In Proc. of Pacific Symposium in Biocomputing, 2002.
- [5] W. Noble “Support vector machine applications in computational biology” Learning with Kernels, chapter 6, MIT press.
- [6] S. Gunn “Support vector machines for classification and regression” University of Southampton Technical Report, 1998.

Homework

- What is the difference between pairwise models and just plain kernel models?
- What is the curse of dimensionality and how does it relate to SVM kernels?
- Describe briefly to other kernel based approaches that have been proven useful at solving biological problems.
- The motif and spectrum kernel are based on a trie data structure. How does this data structure works and how is it applied to the motif kernel? What is the main difference compared to a tree data structure?