Workshop on Kernel Methods in Bioinformatics

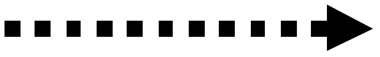
New String Kernels for Biosequence Data

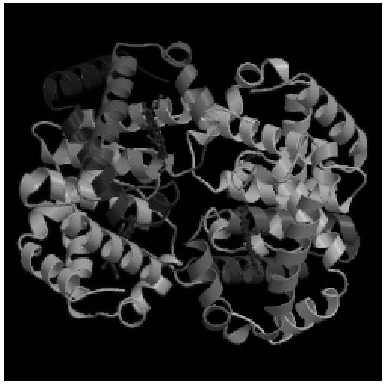
Christina Leslie
Department of Computer Science
Columbia University

Biological Sequence Classification Problems

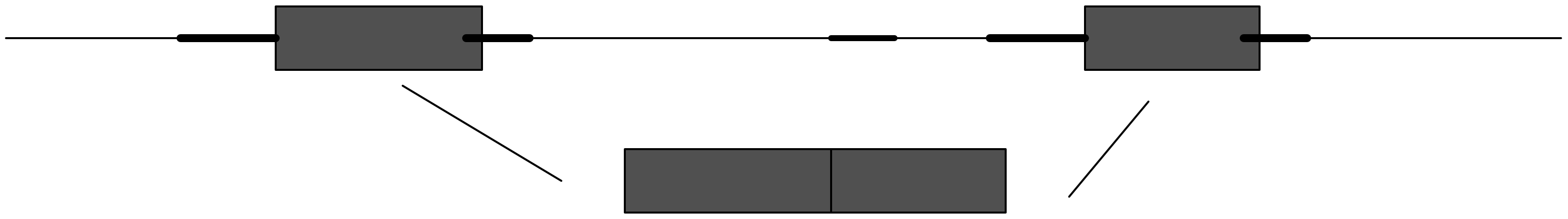
- *Protein classification:* Learn how to classify protein sequence data into families and superfamilies defined by structure/function relationships

```
VLSPADKTNVKAAWGKVGAHAGEYGAEALER
MFLSFPTTKTYFPHFDLSHGSAQVKGHGKKV
ADALTNAVAHVDDMPNALSALSDLHAHKLRV
DPVNFKLLSHCLLVTLAAHLPAEFTPAVHAS
LDKFLASVSTVLTSKYR
```





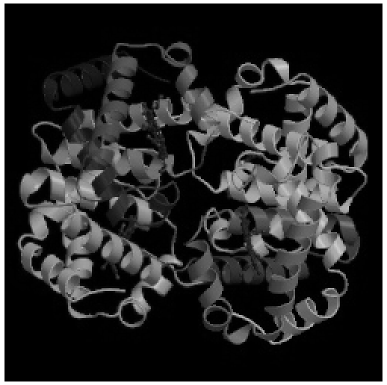
- *Pre-mRNA splicing prediction:* Learn to distinguish exons from pseudo exons based on their splice/pseudo splice signals and intronic flanking regions



Biological Sequence Classification Problems

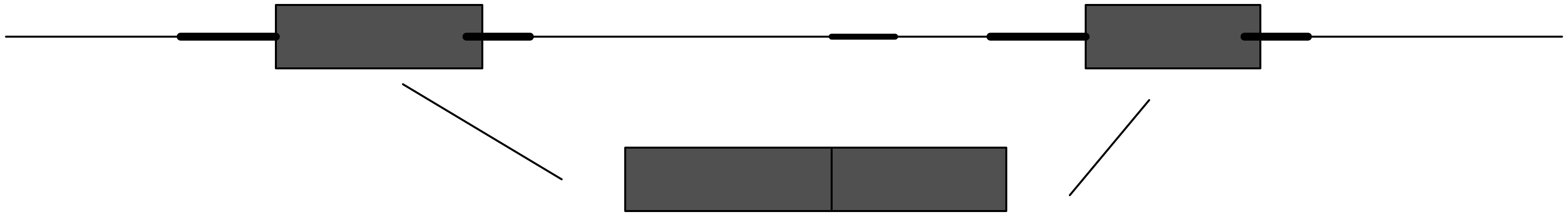
- *Protein classification:* Learn how to classify protein sequence data into families and superfamilies defined by structure/function relationships

```
VLSPADKTNVKAAWGKVGAHAGEYGAEALER
MFLSFPTTKTYFPHFDLSHGSAQVKGHGKKV
ADALTNAVAHVDDMPNALSALSDLHAHKLRV
DPVNFKLLSHCLLVTLAAHLPAEFTPAVHAS
LDKFLASVSTVLTSKYR
```

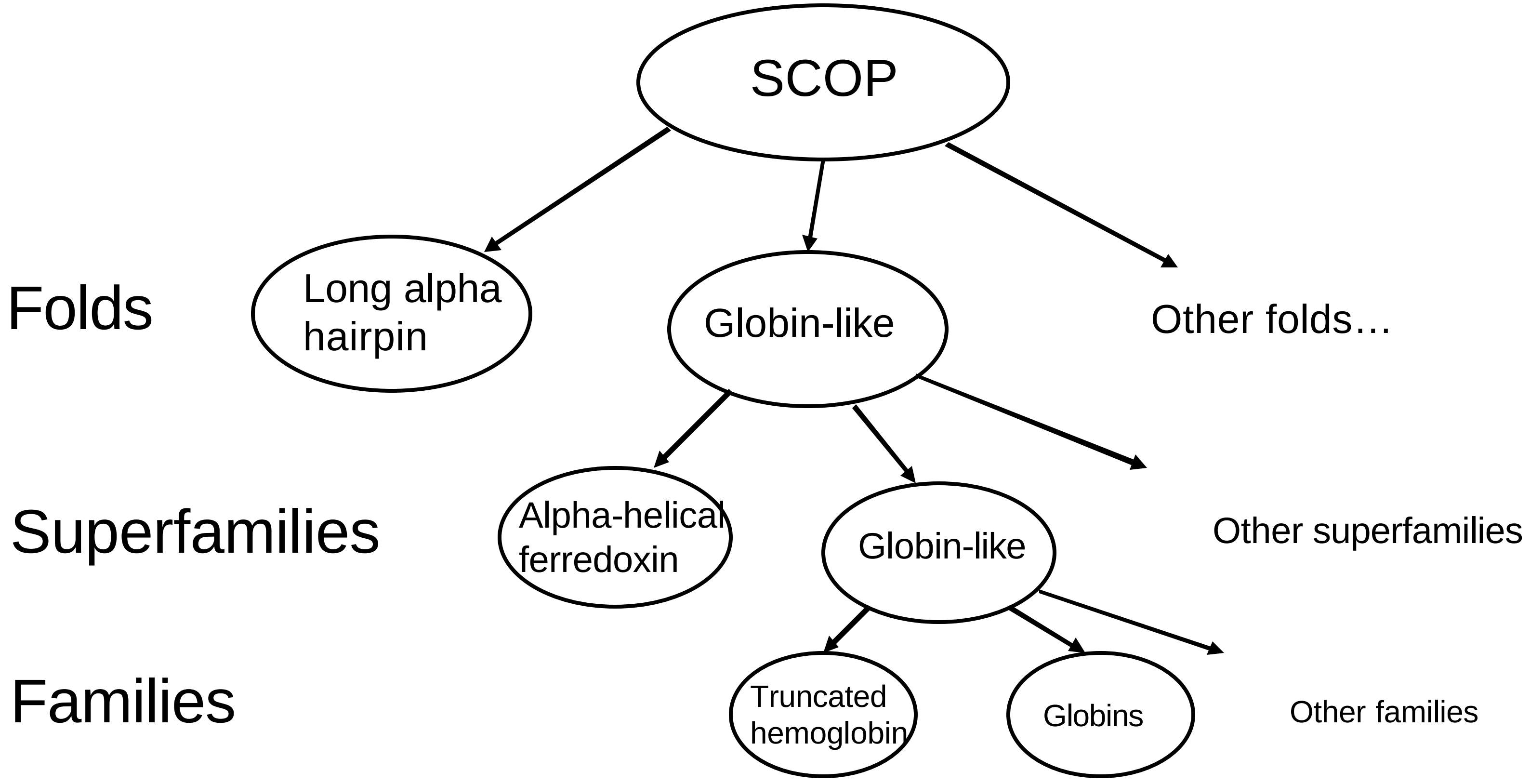


✓

- *Pre-mRNA splicing prediction:* Learn to distinguish exons from pseudo exons based on their splice/pseudo splice signals and intronic flanking regions



Protein Classification



- *Remote homologs*: sequences that belong to the same superfamily but not the same family – remote evolutionary relationship
- Use *discriminative supervised learning* approach (SVMs) to *train* a classifier for remote homology detection

Kernels for Discrete Objects

- Can define kernels for sequences, graphs, other *discrete* objects for use with kernel-based classifiers:

$$\{ \text{sequences} \} \xrightarrow{F} R^N$$

For sequences x, y, feature map F, kernel value is inner product in feature space

$$K(x, y) = \langle F(x), F(y) \rangle$$

- Original string kernels [Watkins, Haussler, later Lodhi *et al*.] require quadratic time in sequence length, $O(|x|\,|y|)$, to compute each kernel value $K(x, y)$

String Kernels for Biosequences

- We'll define new fast *string kernels* for biological sequence data
  - Biologically-inspired underlying feature map
  - Kernels scale linearly with sequence length, $O(c_K(|x| + |y|))$ to compute
  - Strong protein classification performance
  - Many models for *inexact sequence matching*

    $\rightarrow$ Mismatches

    $\rightarrow$ Gaps, substitutions, wildcards

Outline

1. Mismatch kernel
   - Feature maps indexed by k-mers
   - Inexact matching through mismatches
   - Efficient computation of mismatch kernel
   - Fast prediction
2. Experimental results on SCOP dataset
3. Other models for inexact matching
   - Kernels from gaps, substitutions, wildcards
   - Results for new kernels on SCOP experiments

Spectrum-based Feature Map

- Idea: feature map based on *spectrum* of a sequence
  - The k-spectrum of a sequence is the set of all k-length contiguous subsequences that it contains
  - Feature map is indexed by all possible k-length subsequences (“k-mers”) from the alphabet of amino acids
  - Dimension of feature space = $|\Sigma|^k$ ($|\Sigma|$ = 20 for amino acids)

```
AKQDYYYYEI

AKQ
 KQD
  QDY
   DYY
    YYY
     YYY
      YYE
       YEI
```

k-Spectrum Feature Map

- Feature map for k-spectrum with no mismatches:

  For sequence x, $F_{(k)}(x) = (F_t(x))_{\{\text{k-mers } t\}}$,

  where $F_t(x)$ = #occurrences of t in x

AKQDYYYYEI

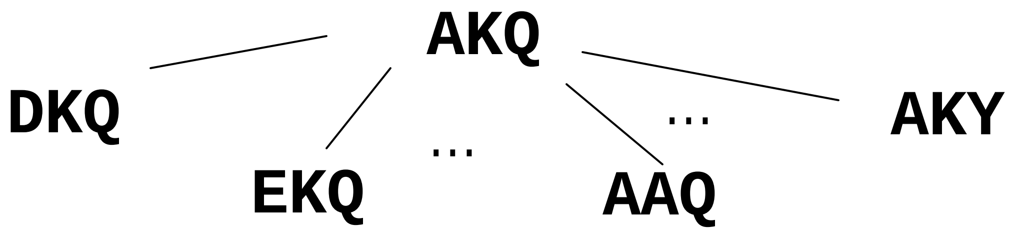
$$\Downarrow$$

```
( 0 , 0 , … , 1 , … , 1 ,      …    , 2 )
 AAA AAC  …  AKQ  …  KQD       …     YYY
```

C. Leslie, E. Eskin, and W. Noble, *The Spectrum Kernel: A String Kernel for SVM Protein Classification.* Pacific Symposium on Biocomputing, 2002.

Inexact Matching through Mismatches

- For k-mer s, the *mismatch neighborhood* $N_{(k,m)}(s)$ is the set of all k-mers t within m mismatches from s
- Size of mismatch neighborhood is $O(|\Sigma|^m k^m)$



(k,m)-Mismatch Feature Map

- Feature map for k-spectrum, allowing m mismatches:

  For a k-mer s, $F_{(k,m)}(s) = (F_t(s))_{\{\text{k-mers } t\}}$,

  where $F_t(s) = 1$ if t is in neighborhood $N_{(k,m)}(s)$,

  $F_t(s) = 0$ otherwise

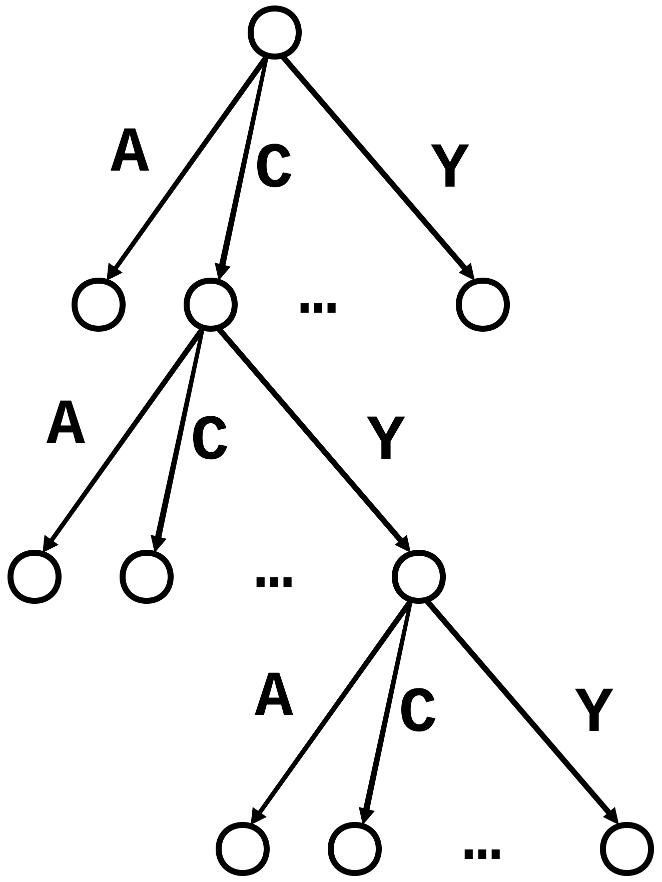
$$\texttt{AKQ} \longrightarrow \begin{array}{ccccccccccc} (\,0\,, & \ldots\,, & 1\,, & \ldots\,, & 1\,, & \ldots\,, & 1\,, & \ldots\,, & 1\,, & \ldots\,, & 0\,) \\ & & \texttt{AAQ} & & \texttt{AKQ} & & \texttt{DKQ} & & \texttt{EKQ} & & \end{array}$$

- Extend additively to longer sequences x by summing over all k-mers s in x

C. Leslie, E. Eskin, J. Weston and W. Noble, *Mismatch String Kernels for SVM Protein Classification.* Neural Information Processing Systems 2002.

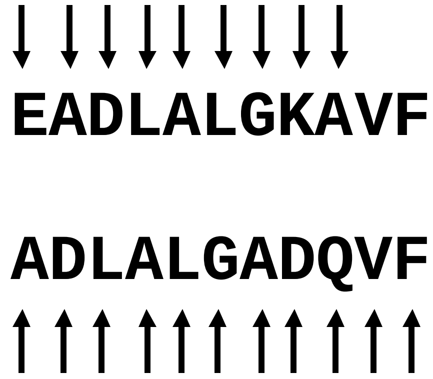
Computing the (k,m)-Mismatch Kernel

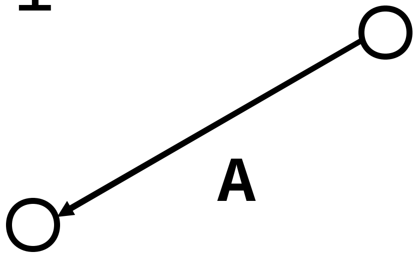
- Use *mismatch tree* to organize lexical traversal of all instances of k-mers (with mismatches) in the training data
  - Each path down to a leaf corresponds to a coordinate in feature map
  - Kernel values for all training sequences updated at each leaf node
  - Depth-first traversal can be accomplished with recursive function



Computing the Kernel for Pair of Sequences

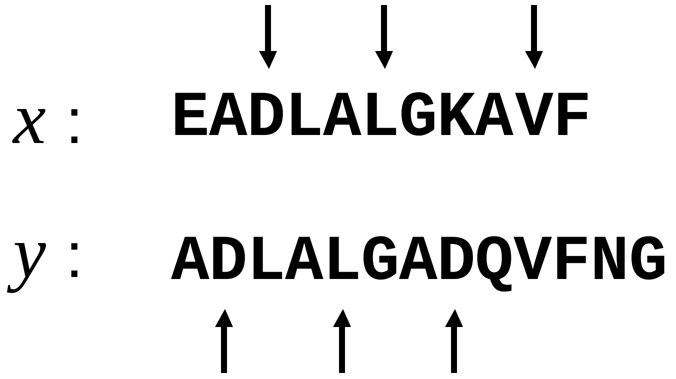
- Traversal of trie for k=3, m=1

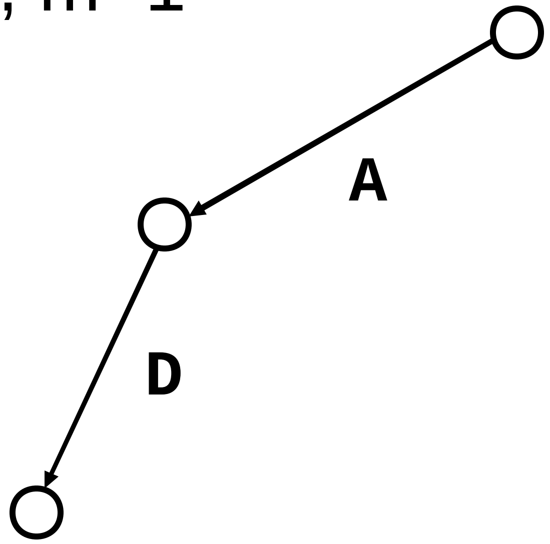




Computing the Kernel for Pair of Sequences

- Traversal of trie for k=3, m=1



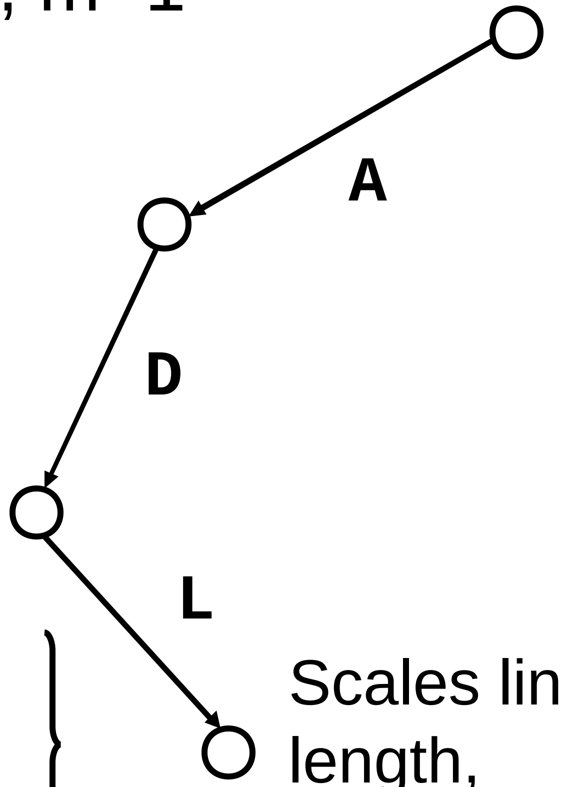


Computing the Kernel for Pair of Sequences

- Traversal of trie for k=3, m=1

x : **EADLALGKAVF**

y : **ADLALGADQVFNG**



Update kernel value for K(x, y) by adding contribution for feature **ADL**

Scales linearly with length, $O(k^{m+1}|\Sigma|^m(|x|+|y|))$

SVM Classifiers

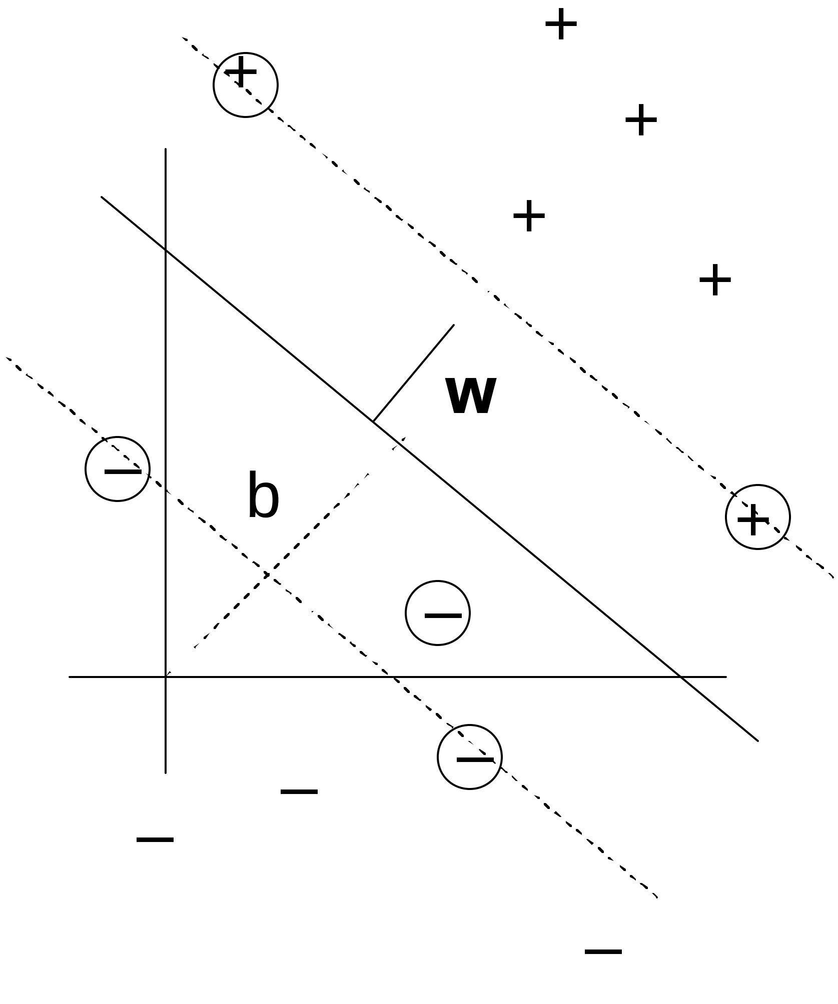
- Linear classifier defined in feature space by

  $$f(x) = \langle \mathbf{w}, F(x) \rangle + b$$

  where sign(f(x)) gives prediction
- SVM solution gives normal vector

  $$\mathbf{w} = \Sigma_i \, y_i \, \alpha_i \, F(x_i)$$

  as a linear combination of *support vectors*, involving weights α_i and labels y_i



Fast prediction

- SVM training determines subset of training sequences corresponding to *support vector sequences* and their weights: (x_i, α_i)
- Linear decision rule in feature space:

$$f(x) = \Sigma_i \, y_i \, \alpha_i \langle F(x_i), F(x) \rangle + b$$

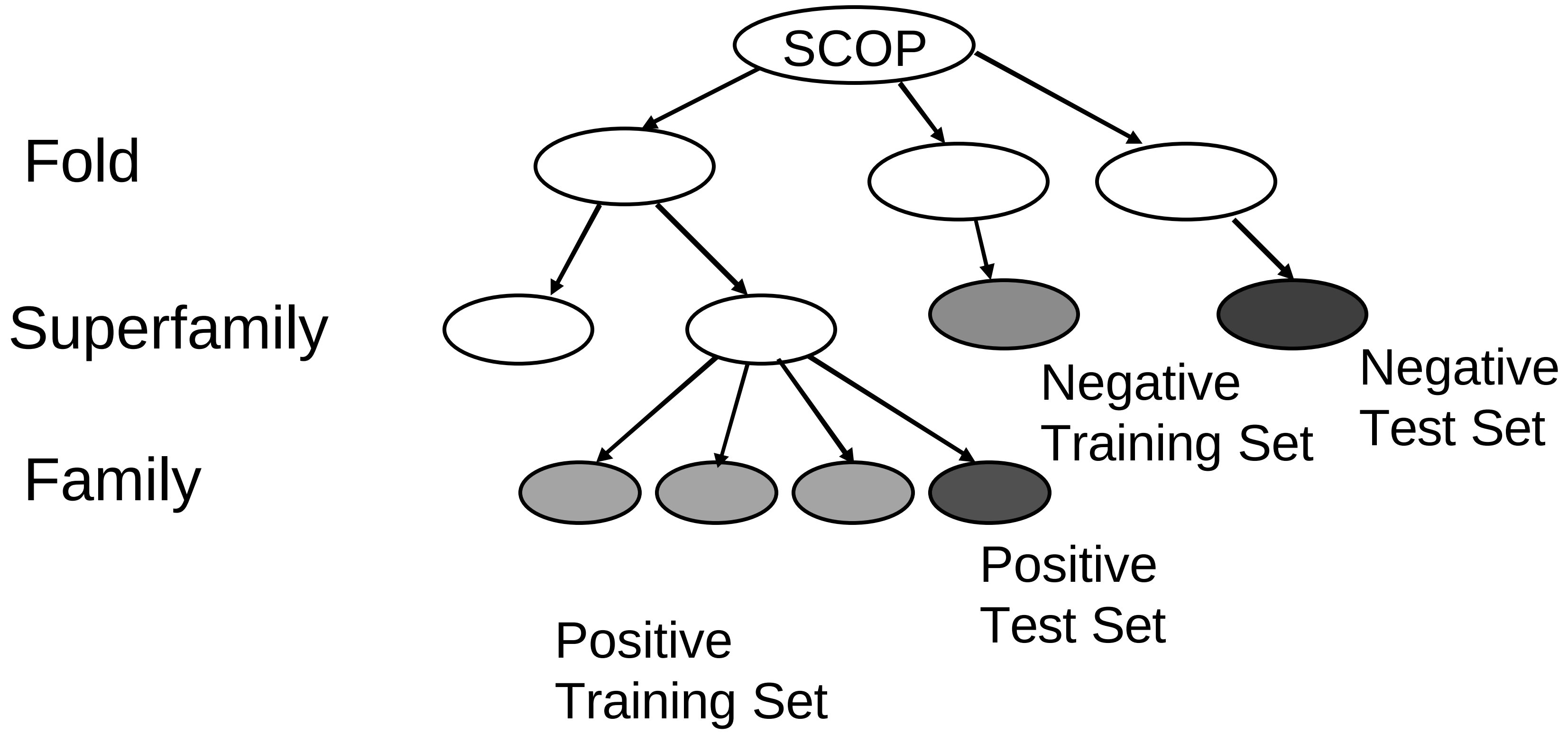
- $F(x)$ is sum of feature vectors $F(s)$ for k-mers s in x

$\rightarrow$ Precompute per k-mer scores for classifier
$\rightarrow$ Test sequences can be classified in *linear time* via lookup of k-mers

Outline

1. Mismatch kernel
   - Feature maps indexed by k-mers
   - Inexact matching through mismatches
   - Efficient computation of mismatch kernel
   - Fast prediction
2. Experimental results on SCOP dataset
3. Other models for inexact matching
   - Kernels from gaps, substitutions, wildcards
   - Results for new kernels on SCOP experiments

SCOP Experiments

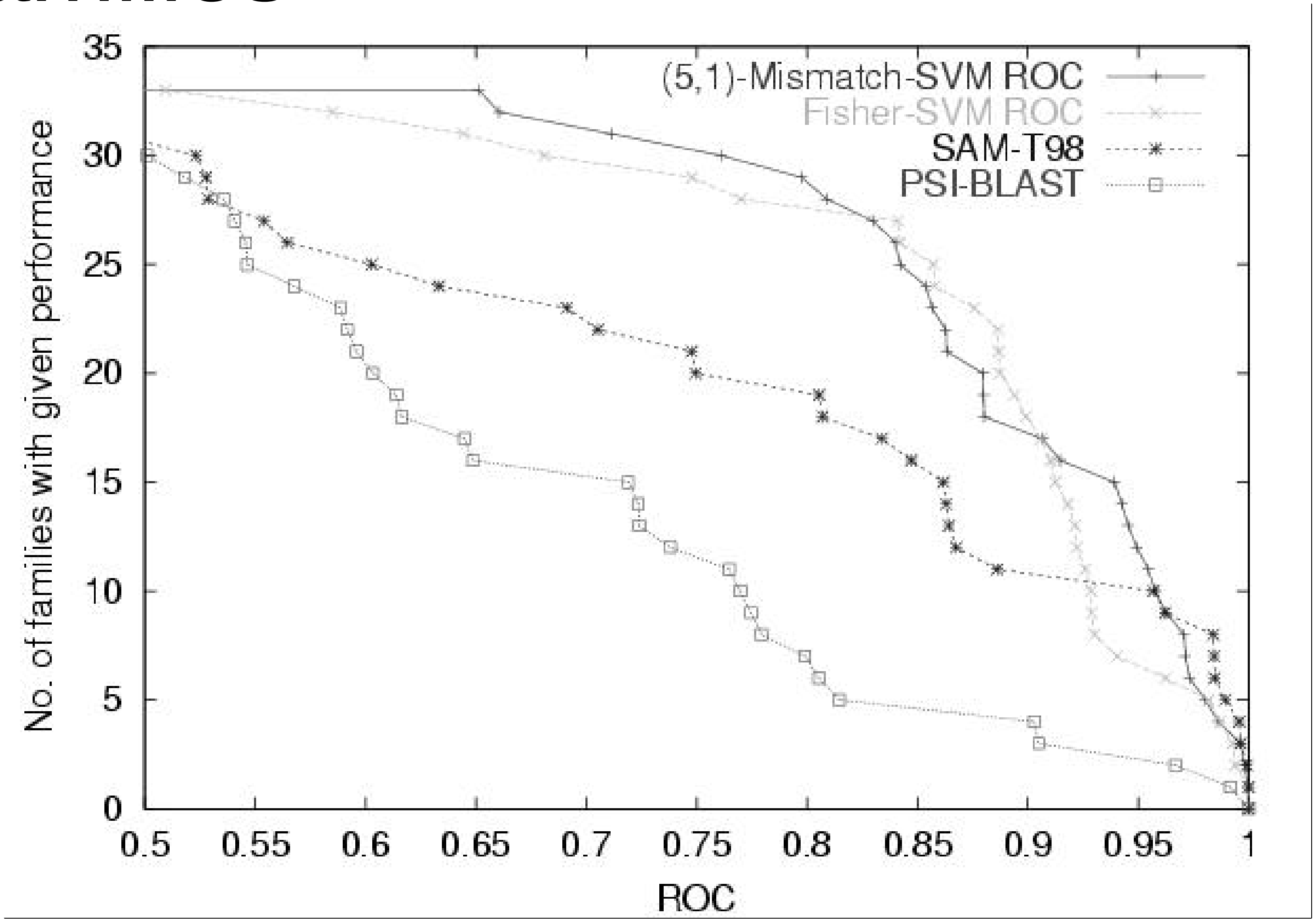


- Tested with experiments on SCOP dataset from Jaakkola *et al.*
- Experiments designed to ask: Could the method discover a new family of a known superfamily?

SCOP Experiments

- 160 experiments for 33 target families from 16 superfamilies
- Compared results against
  - SVM-Fisher (HMM-based kernel)
  - SAM-T98 (profile HMM)
  - PSI-BLAST (heuristic alignment-based method)
- *ROC scores*: area under the graph of true positives as a function of false positives, scaled so that both axes vary between 0 and 1

Results Across All Target Families



Background on Fisher-SVM

- Previous solution [Jaakkola, Diekhans, Haussler]:
  - Use positive examples to train profile HMM, (M_+, θ_0)
  - For each training example x, *Fisher score* is gradient of log-likelihood score for x given M_+ (evaluated at θ_0)

$$x \longrightarrow \nabla_\theta \log P(x \mid M_+, \theta)$$

- Method relies on generative model
  - Requires large amount of data or sophisticated priors to train M_+
  - Expensive: dynamic programming (quadratic in sequence length) – for each sequence x, forward-backward algorithm to compute features

Aside: Connection with Fisher Kernel

- Consider order k-1 Markov chain model for positive sequences, with parameters

  $\theta^{t|s_1..s_{k-1}} = P(x_j = t \mid x_{j-k+1}..x_{j-1} = s_1..s_{k-1})$

- Corresponding Fisher coordinate for x is

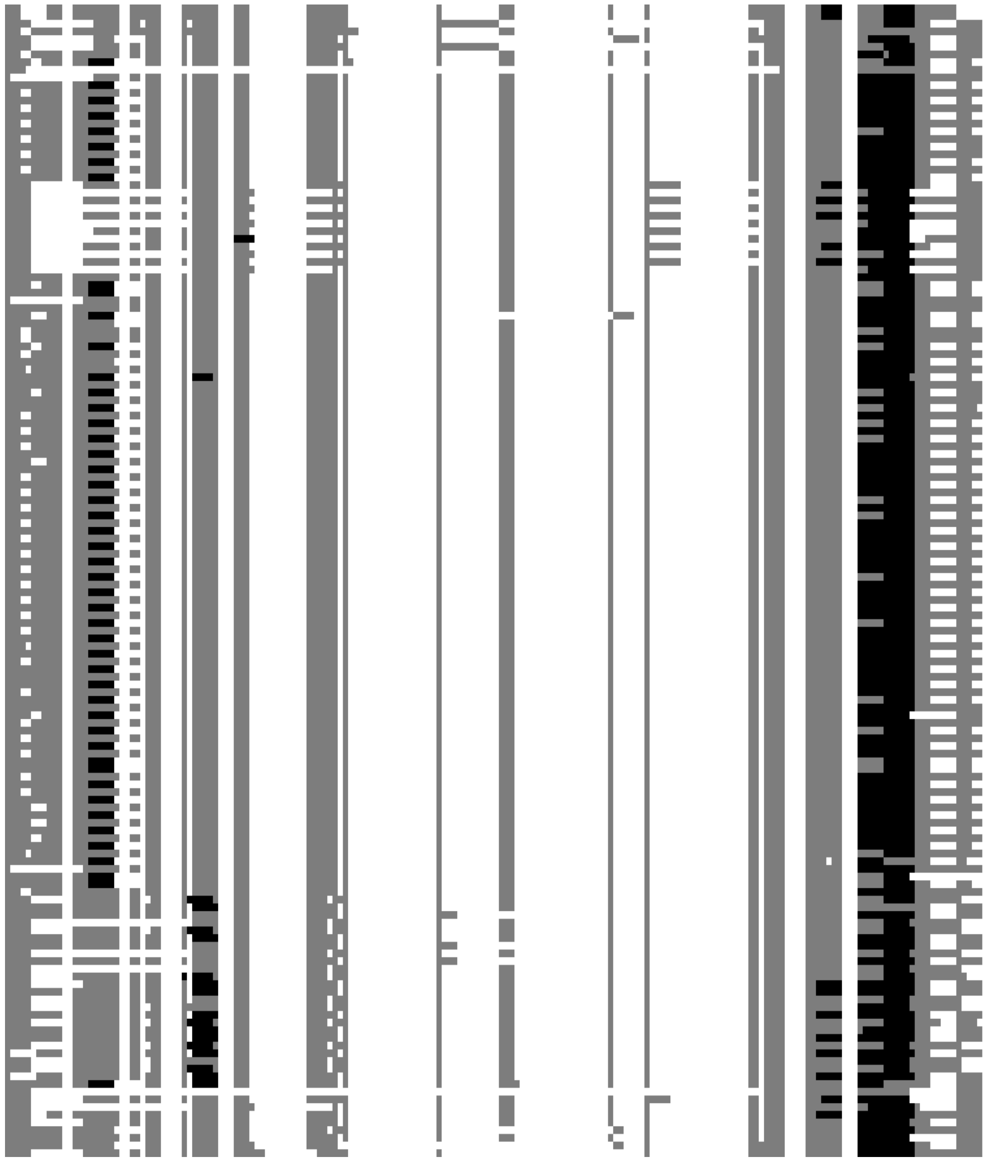
  (#occurrences of $s_1.. s_{k-1}t$ in x)/$\theta^{t|s_1..s_{k-1}}$

  – (#occurrences of $s_1..s_{k-1}$ in x)

- Fisher kernel for Markov chain model similar to k-spectrum kernel

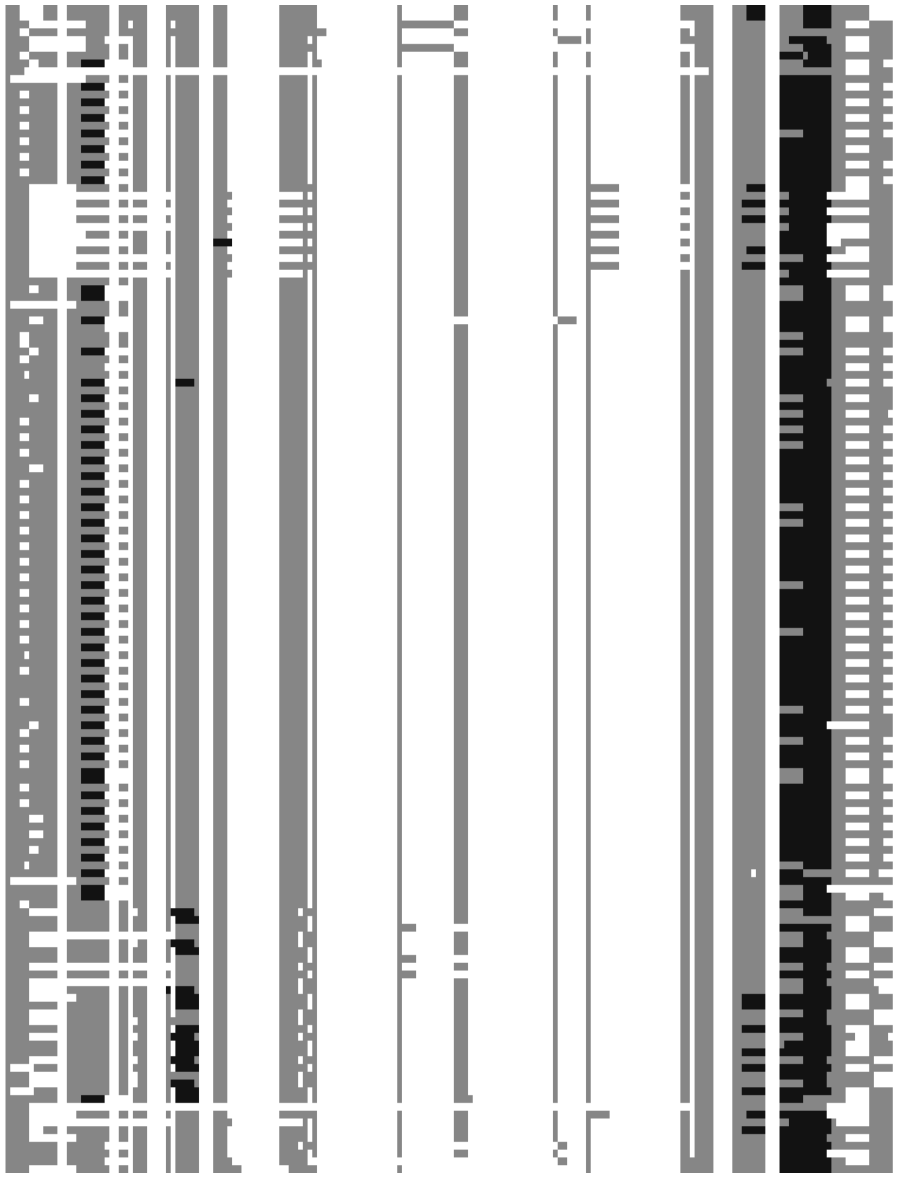
Interpretation of Mismatch-SVM Classifier

- Rank features by $|w_i|$, associate to +/- class by $\mathrm{sign}(w_i)$
- Top positively-weighted k-mer features learned by SVM map to *conserved regions* in the *multiple alignment* of positive training sequences



Interpretation of Mismatch-SVM Classifier

- Rank features by $|w_i|$, associate to +/- class by sign
- Top positively-weighted k-mer features learned by SVM map to *conserved regions* in the *multiple alignment* of positive training sequences



Advantages of Mismatch-SVM

- Mismatch-SVM performs as well as SVM-Fisher but avoids computational expense, training difficulties of profile HMM
- Advantages of string kernel:
  - *Efficient computation*: scales linearly with sequence length
  - *Fast prediction*: classify test sequences in linear time
  - *Interpretation* of learned classifier
  - *General approach* for biosequence data, does not rely on alignment or generative model

Outline

1. Mismatch kernel
   - Feature maps indexed by k-mers
   - Inexact matching through mismatches
   - Efficient computation of mismatch kernel
   - Fast prediction
2. Experimental results on SCOP dataset
3. Other models for inexact matching
   - Kernels from gaps, substitutions, wildcards
   - Results for new kernels on SCOP experiments

Other Fast(er) Kernels for Inexact Matching

- Mismatch kernel is linear in sequence length, but constant $c_K = k^{m+1}|\Sigma|^m$ depends on alphabet size
- Other models for inexact matching can achieve $O(c_K(|x| + |y|))$ with c_K independent of $|\Sigma|$
  - Restricted gaps
  - Probabilistic substitutions
  - Wildcards

C. Leslie and R. Kuang, *Fast String Kernels for Inexact String Matching.*

Inexact Matching through Gaps

- For g-mer s, the *gapped match set* $G_{(g,k)}(s)$ consists of all k-mers t that occur in s with g - k gaps
- Size of gapped match set is $O(g^{g-k})$, independent of $|\Sigma|$

```
                     AKQ__
                     AK_K_
                     AK__L
AKQKL  ———————       A_QK_      }   AKQ, AKK, AKL,
                     …          }   AQK, …, KQK, …
                     _KQK_
                     …
```

(g,k)-Gappy Kernel

- Several possibilities for feature map:

  Unweighted: For a g-mer s, $F_{(g,k)}(s) = (F_t(s))_{\{k\text{-mers } t\}}$,

  where $F_t(s) = 1$ if t is in gapped match set $G_{(g,k)}(s)$,

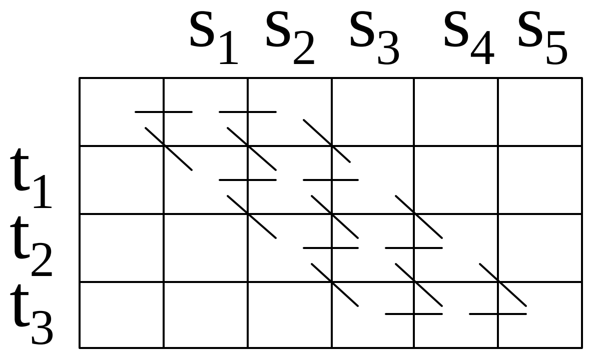
  $F_t(s) = 0$ otherwise

  Weighted: For $0 < \lambda \leq 1$, use instead

  $$F_t(s) = (1/\lambda^k)\, \Sigma_{\{subseq(s) = t\}}\, \lambda^{length(subseq(s))}$$

  where $F_t(s)$ can be computed

  by dynamic programming

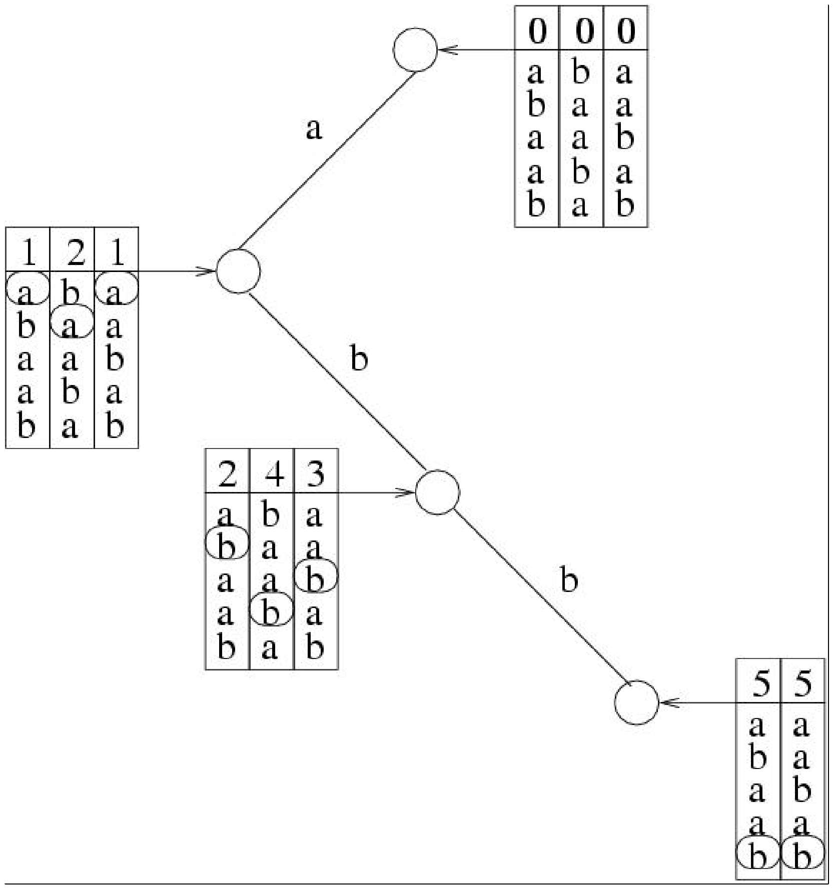


- Extend additively by summing over all g-mers s in x

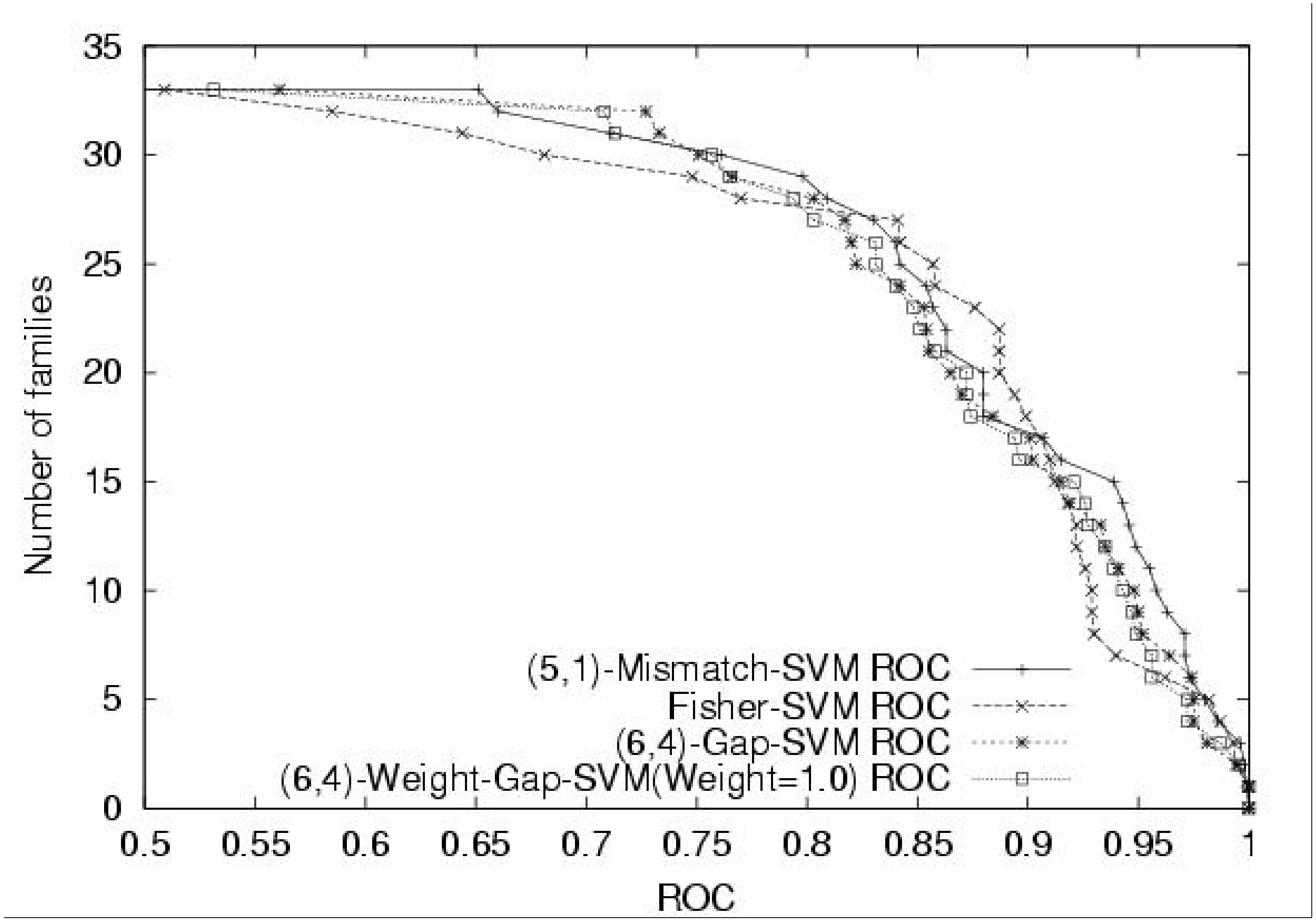
Gappy Kernel Computation

- Traverse instance g-mers in the data, greedily align to k-length paths (k-mer features)
- At leaf node, count instances for each input sequence (unweighted) or perform restricted dynamic programming (weighted)
- *Complexity*: $O(c_K(|x| + |y|))$ with $c_K = g^{g-k+1}$ (unweighted) or

  $(g-k)g^{g-k+1}$ (weighted)



Gappy Kernel SCOP Results



Inexact Matching through Probabilistic Substitutions

- Use *substitution matrices* to obtain P(a|b), substitution probabilities for residues a, b
- The *mutation neighborhood* $M_{(k,\sigma)}(s)$ is the set of all k-mers t such that

$$- \Sigma_{i=1\ldots k} \log P(s_i|t_i) < \sigma$$

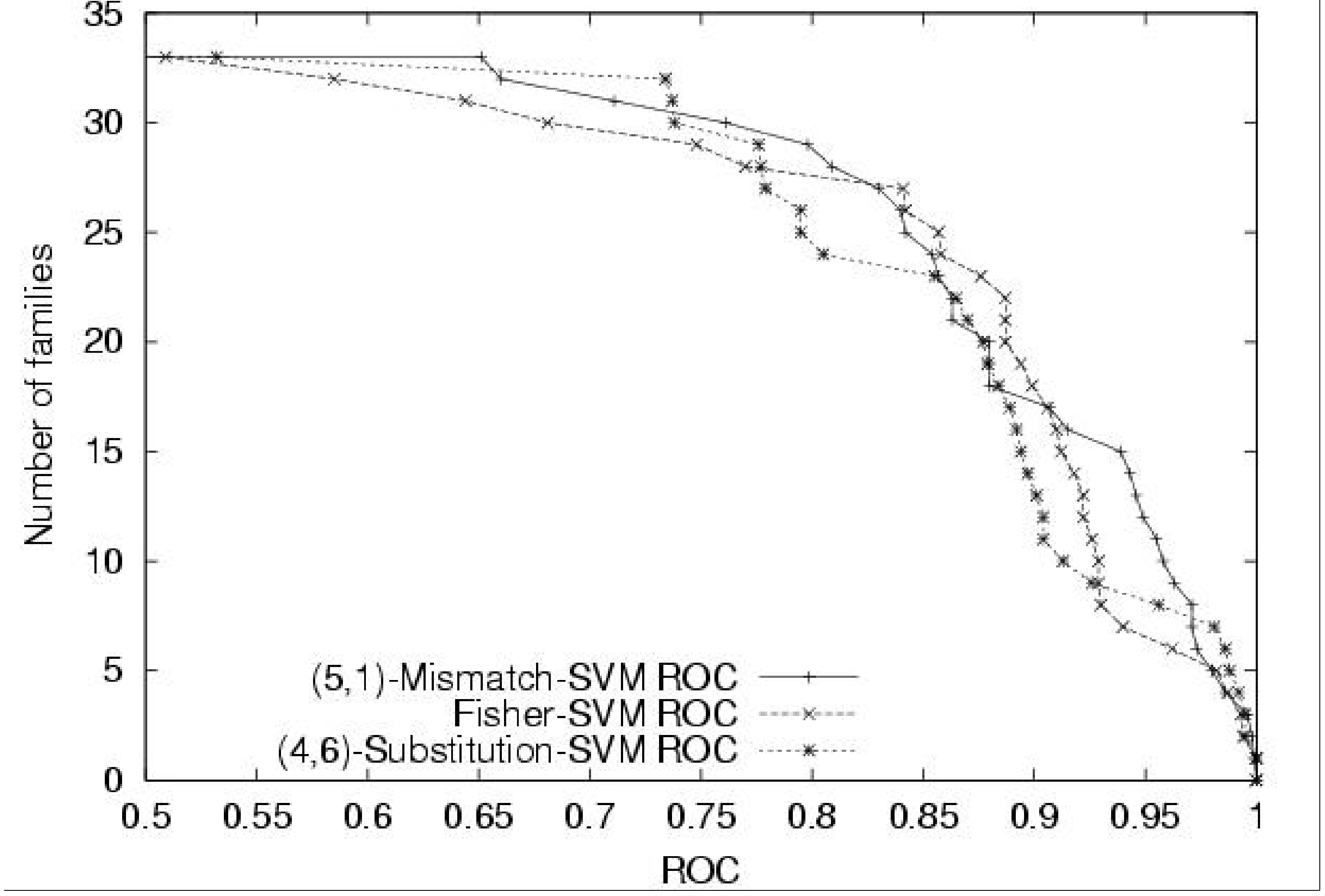
- For a k-mer s, map $F_{(k,\sigma)}(s) = (F_t(s))_{\{\text{k-mers } t\}}$,

  where $F_t(s) = 1$ if t is in neighborhood $M_{(k,\sigma)}(s)$,

  $F_t(s) = 0$ otherwise;

  extend additively
- Trie computation with $c_K = k N_\sigma$, where N_σ is maximum size of mutation neighborhood

Substitution Kernel SCOP Results



Inexact Matching through Wildcards

- Introduce wildcard character “$*$”, define feature space indexed by k-mers from $\Sigma \cup \{*\}$, allowing up to m wildcards
- For a k-mer s, $F_{(k,m)}(s) = (F_t(s))_{\{\text{k-mers } t\}}$,

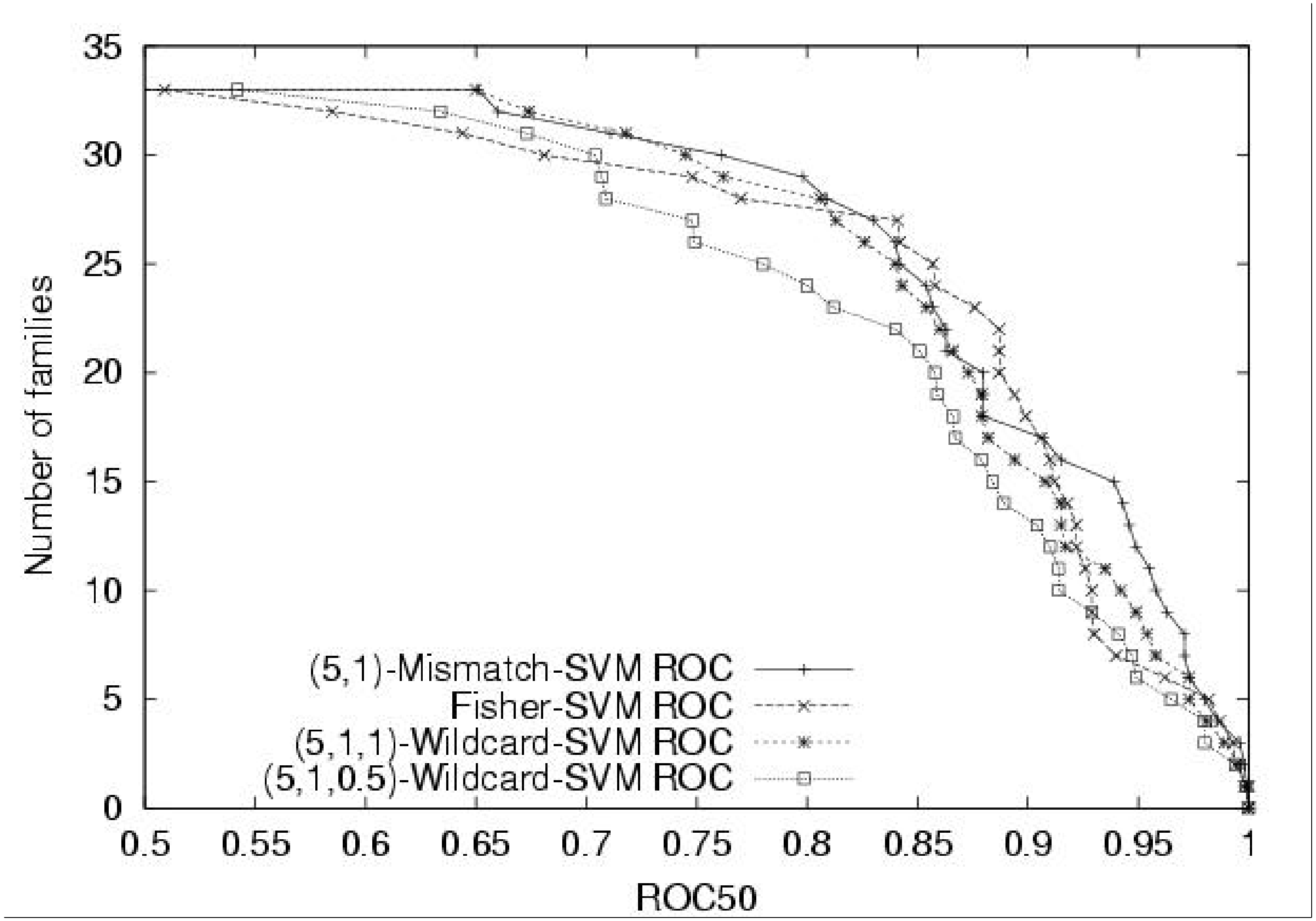
  where $F_t(s) = \lambda^{\text{num}(*,t)}$, if t matches s, $\text{num}(*,t)$ =#wildcards,
  $F_t(s) = 0$ otherwise;

  extend additively

$$\mathbf{AKQ} \longrightarrow (\,0\,,\,\ldots\,,\,\underset{\mathbf{AKQ}}{1}\,,\,\ldots\,,\,\underset{\mathbf{AK*}}{\lambda}\,,\,\ldots\,,\,\underset{\mathbf{A*Q}}{\lambda}\,,\,\ldots\,,\,\underset{\mathbf{*KQ}}{\lambda}\,,\,\ldots\,,\,0\,)$$

- Compute with (pruned) depth k trie over $\Sigma \cup \{*\}$, $c_K = k^{m+1}$
- Alternative weightings introduced elsewhere by Eskin and Snir

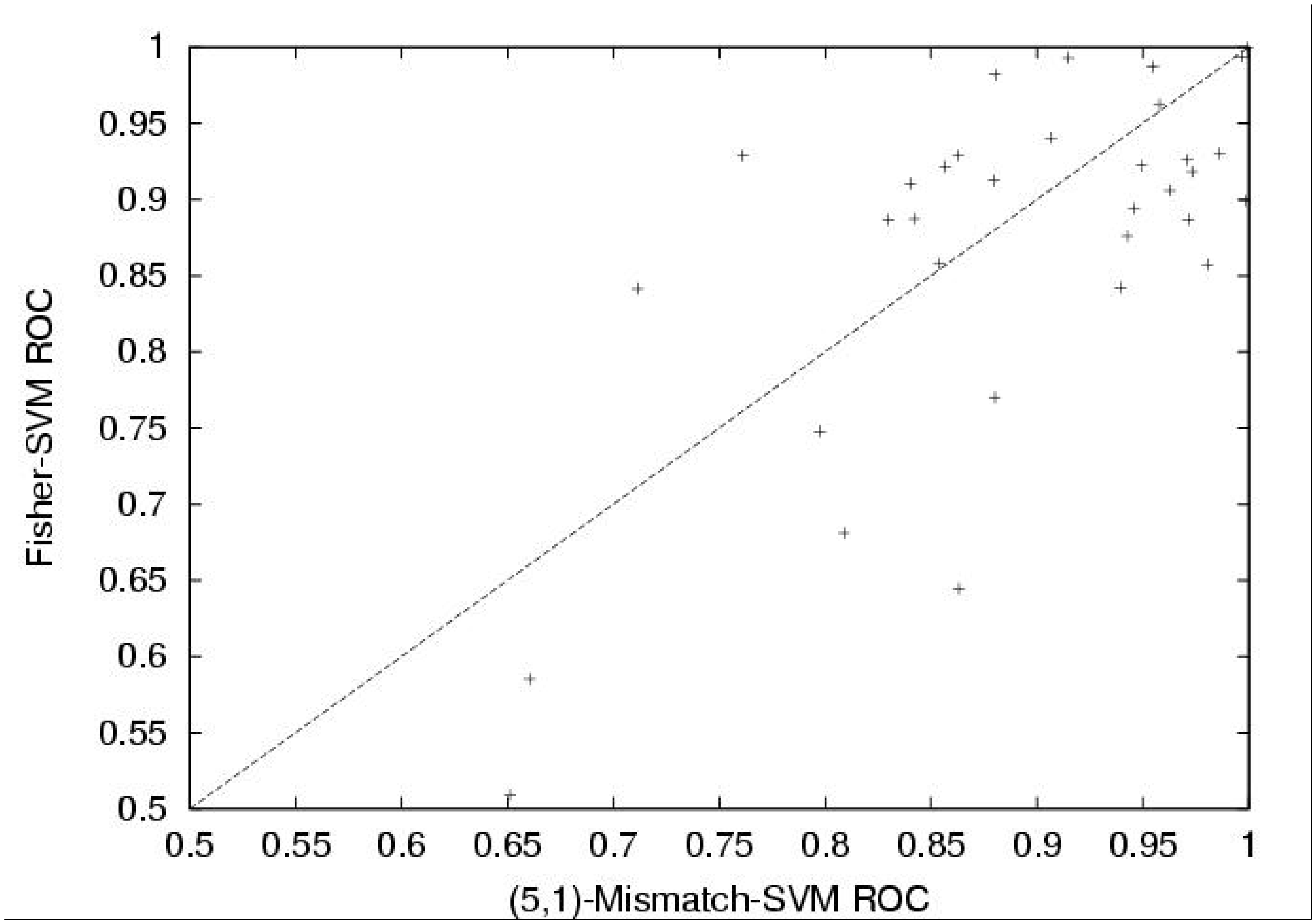
Wildcard Kernel SCOP Results



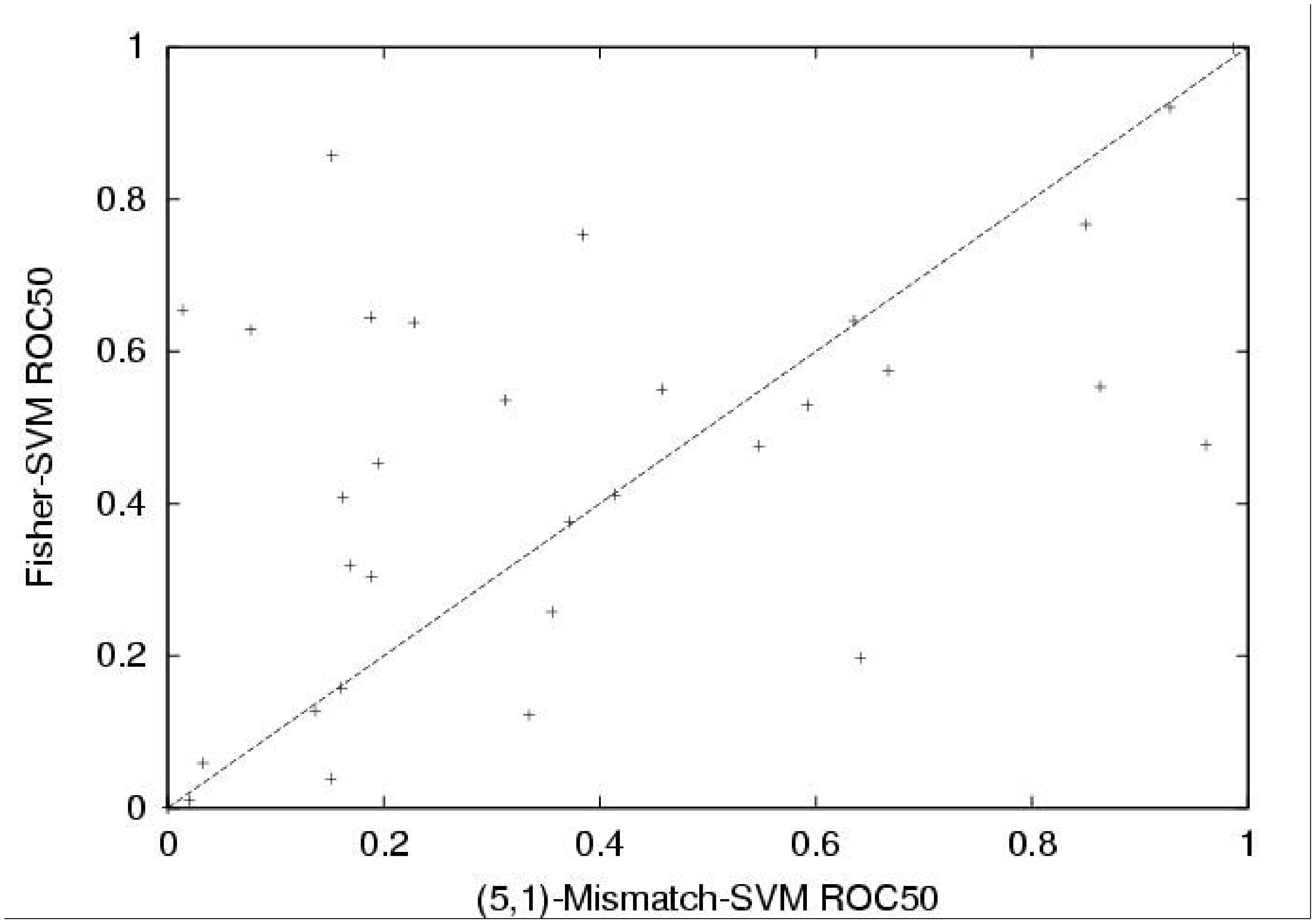
Conclusions and Further Work

- String kernels that incorporate *inexact matching*, used with SVMs, are competitive with best-known methods for protein classification
- Gaps, substitutions, and wildcards lead to computation time $O(c_K(|x| + |y|))$, where c_K is independent of alphabet size
- *Convex combinations* of kernels could lead to improved performance [see Vishwanathan and Smola for exact matching case]
- Can describe all the kernels here using *transducer formalism* of Cortes *et al.*

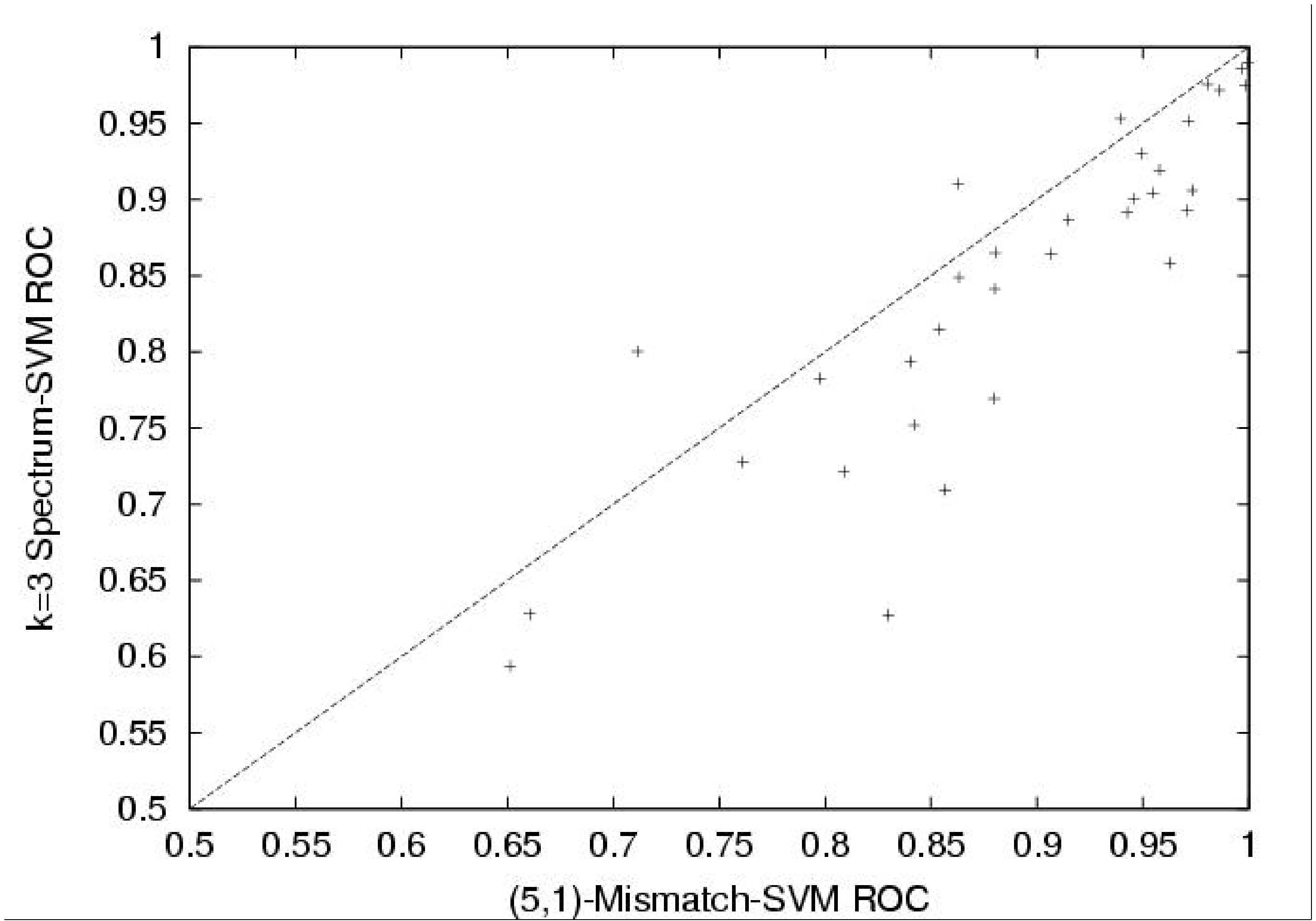
(5,1)-Mismatch vs Fisher Using ROC Scores



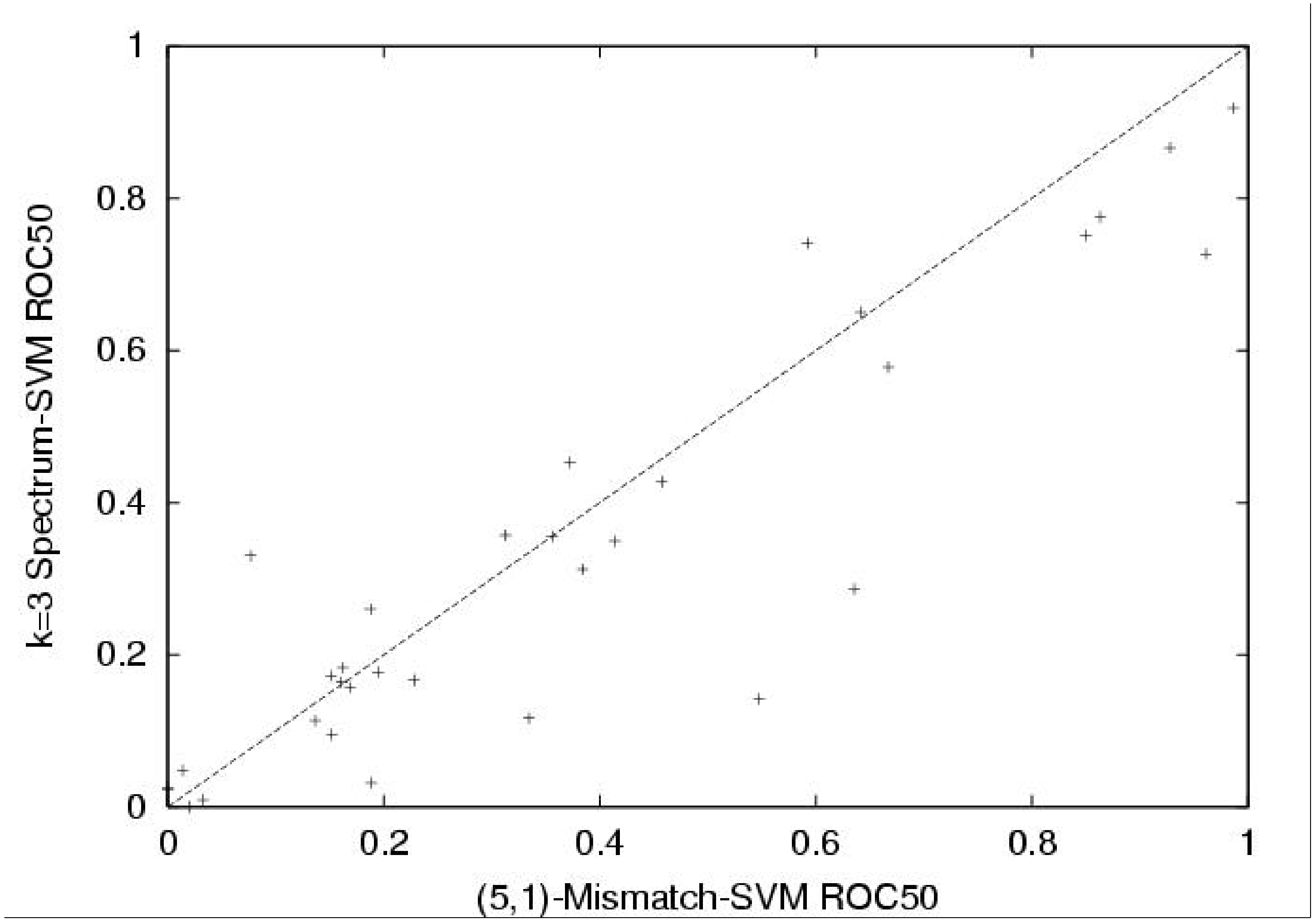
(5,1)-Mismatch vs Fisher Using ROC-50 Scores



(5,1)-Mismatch vs. 3-Spectrum Using ROC Scores



(5,1)-Mismatch vs. 3-Spectrum Using ROC-50 Scores


k=3 Spectrum-SVM ROC50
(5,1)-Mismatch-SVM ROC50