Fast and Accurate Multi-class Protein Fold Recognition with Spatial Sample Kernels

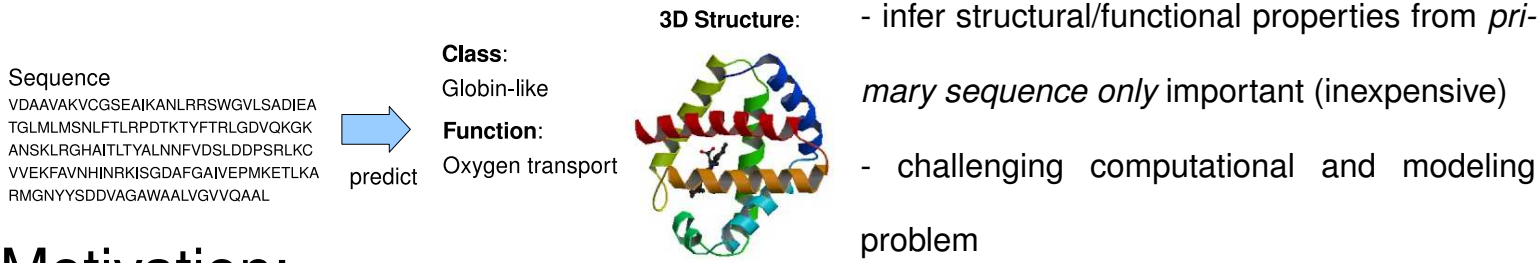
Pavel Kuksa, Pai-Hsi Huang, Vladimir Pavlovic

Department of Computer Science
Rutgers University

CSB 2008

Introduction: problem formulation

- Goal: accurate and fast sequence classification in the *remote* similarity (low sequence identity) and multi-class (diverse) setting:
  - proteins sharing fold, e.g. same fold,$<$ 10% seq.identity
  - proteins from different families



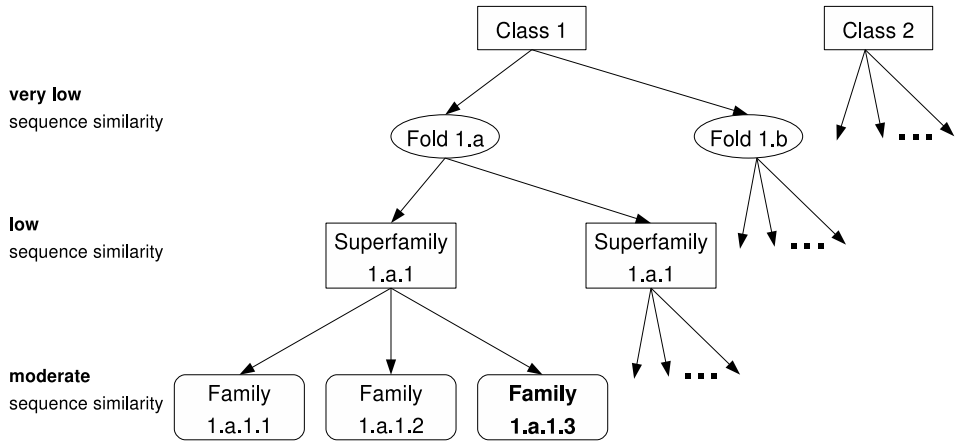
- infer structural/functional properties from *primary sequence only* important (inexpensive)
- challenging computational and modeling problem

- Motivation:
  - remote similarity problems occur in many different domains *e.g.* biological, text, images, *etc.*
  - existing methods still have suboptimal performance
- Challenges:
  - low sequence similarity, very diverse sequences
  - limited labeled data

Introduction: problem formulation (Cont'd)

- Protein sequences: primary AA sequences ($|\Sigma| = 20$)
- SCOP (Structural Classification of Proteins [LCAH+00])

- Remote, *multi-class* (harder) setting:
  - remote homology (superfamily) prediction: test on *unseen* families
  - fold prediction: test on *unseen* superfamilies
  - Challenges:
    - *primary sequence*
    - diverse, low similarity
    - variable-length
    - multi-class



- Goal: accurately predict *superfamily* or *fold* of unknown sequences

Main results summary: Preview

- state-of-the art performance for *multi-class remote* fold prediction and remote homology
  - supervised (labeled)
  - semi-supervised (unlabeled+labeled)
- order-of-magnitude faster than many existing approaches
- Key: new sparse spatial sequence feature, highly discriminative and efficient to compute.

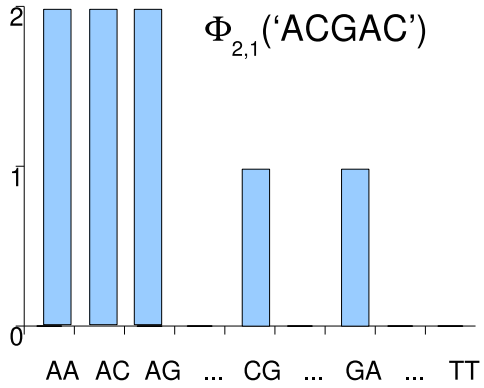
Classification of Sequences using Kernel Methods

- Show some of the best results in many sequence analysis tasks
- Infer class labels via similarity measures (kernels) $K(x, y)$:
  - measures similarity between two objects (e.g. bio-sequences)
  - has a corresponding vector (dot-product) feature (kernel) space $\phi(x)$
  - the problem is mapped from the original space (may be arbitrary, non-vector) to *vector feature* space

$$K(X, Y) = \Phi(X)^T \Phi(Y)$$

Prior state-of-the-art methods for protein classification

- mismatch(k,m) method [LEWN02]:
  - compares sequences using common *observed* k-mers, with up to m mismatches
  - induced feature set has exponential size ($|\Sigma|^k$)
  - expensive (due to inexact matching)



- profile(k,σ) method [KIW+04]:
  - considers common (in *probabilistic* sense, $\log P > \sigma$) k-mers derived from *profiles* (describe amino acid probabilities at each position)
  - induced feature set has exponential size
  - expensive
    - need to estimate profiles
    - inexact *probabilistic* matching

Multi-class classification problem

- assign a label $\hat{y} \in Y$ where $|Y| > 2$
- can formulate a multi-class optimization problem directly as in [WW99, Vap98] (slow, expensive)
- can combine *binary* predictors:
  - *one-vs-rest*: estimate $|Y|$ *binary classifiers*, decision rule is $\hat{y} = \text{argmax}_{y \in Y} f_y(x)$ (simple)
  - *one-vs-one*: voting (estimate $\binom{|Y|}{2}$ classifiers)
  - for hard problems (eg. multi-class protein classification), complex coding schemes suggested, e.g. *adaptive codes* [MIW+07] (expensive):
    - Need at least $n_{fold}+n_{supfam}+n_{fam}$ independent binary classifiers
- we use simple *one-vs-rest* decision rule (inexpensive) combined with a class of efficient string-based kernels (SSSK)

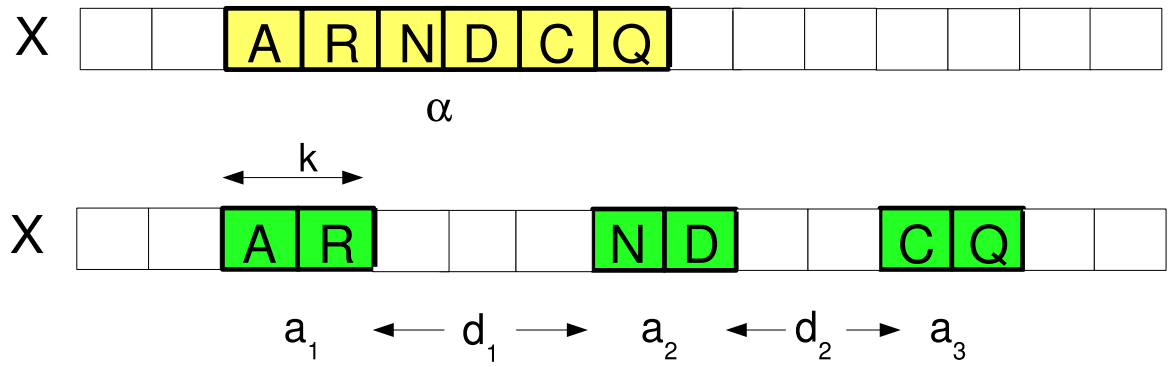
Summary: state-of-the-art methods for protein classification

- Good performance is achieved at high computational cost
- Complex optimization schemes used (eg. adaptive codes)
- Performance of the best available methods is still too low for reliable sequence annotation

New sequence classification method

- Inspired by a need to account for complex processes of sequence transformations (e.g. evolution) without explicitly modeling the process
- Based on a class of efficient string-based kernels (sparse spatial sample kernels, SSSK)
- can be effeciently used in multi-class setting with simple (eg. one-vs-rest) classification schemes
- Has linear $O(cn)$ in the sequence length (n) time complexity and small alphabet-independent constants
- Provides significant improvements over existing state-of-the-art methods in both performance and running times

Sparse Spatial Sample Kernels

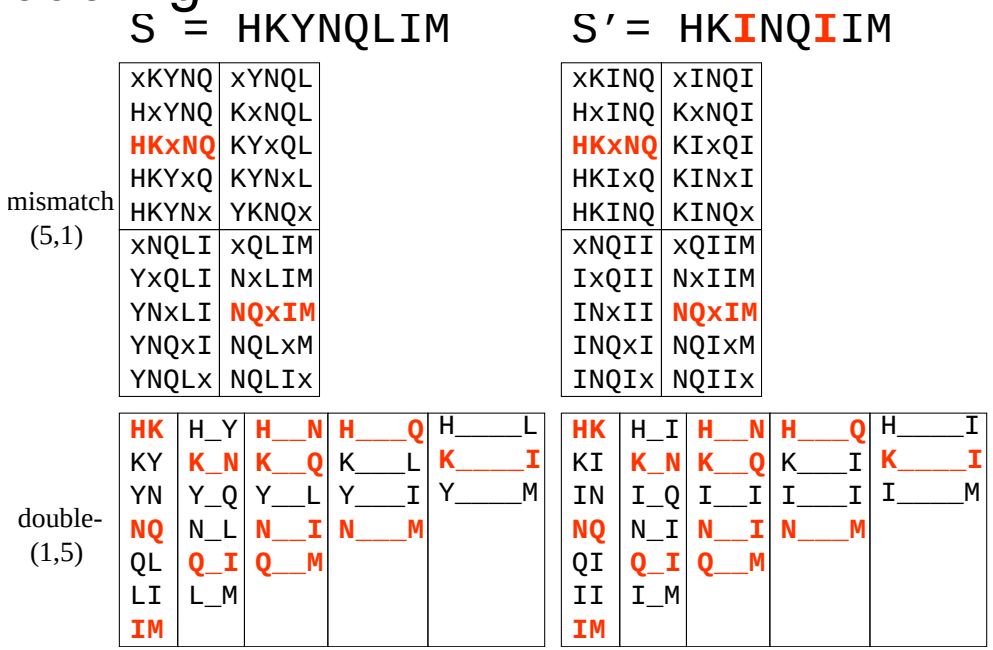


Contiguous k-mer feature α of a traditional mismatch/spectrum kernel (top) contrasted with the spatial(k, t, d) sample features (bottom).

$$K^{k,t,d}_{SSSK}(X, Y) = \sum_{\substack{(a_1,d_1,a_2,\ldots,d_{t-1},a_t) \\ a_i \in \Sigma^k \\ 0 \leq d_i \leq d-1}} C((a_1, d_1, \ldots, d_{t-1}, a_t)|X)C((a_1, d_1, \ldots, d_{t-1}, a_t)|Y)$$

- We use: double-(k=1,d=3) (t=2), and triple-(k=1,d=3) (t=3)

Sequence Modeling



Differences in modeling substitutions by spectrum-like and spatial kernels

- Mismatch: very few features retained
- Spatial kernels: still significant overlap in sequence features
- For the spectrum-like kernel, the number of allowed mismatches needs to be increased (at a high computational cost)

Computing Spatial Sample Kernels

- Can be efficiently computed using Sorting and Counting

Input: set of strings $S = \{s_1, \ldots, s_N\}$, parameters k, t, d

1: $\mathbf{K} = \mathbf{0}$
2: **for all** $(d_1, \ldots, d_{t-1}) \in \{1, \ldots, d\}^{t-1}$ **do**
3: Build sets $L_{s_i}, i = 1, \ldots, N$ of SSS features for each string in the sequence set
4: Construct complete set of features in the sequence set $L = \bigcup_{i=1}^{N} L_{s_i}$ with each feature containing the index of the sequence it comes from
5: Obtain a permutation π_L that lexicographically orders L using t rounds of counting sort
6: Obtain counts $\mathbf{c}(f) = \{c(f, s_i)\}_{i=1,\ldots,N}$ for each distinct feature f in L_{π_L} in a single pass over the list and update kernel $\mathbf{K} = \mathbf{K} + \mathbf{c}(f)\mathbf{c}(f)^T$
7: **end for**

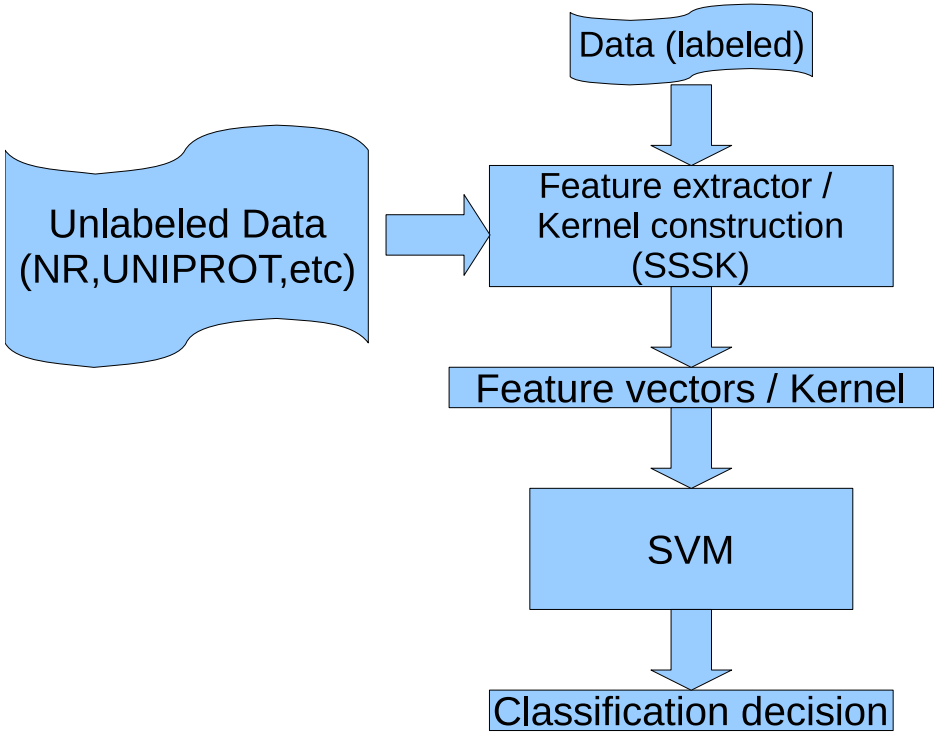
- can work with unlabeled data as well (Semi-supervised learning)
- scales well to large datasets

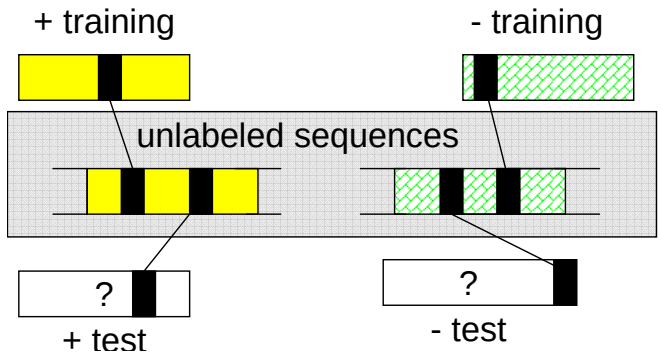
Semi-supervised learning

- very limited labeled data $\Rightarrow$ weak classifiers
- can enhance performance with more data
- Obtain features for each train/test sequence X using neighboring unlabeled sequences N(X) [WLI+05]:

$$\Phi'(X) = \sum_{x \in N(X)} \Phi(x)$$

- Equivalent to computing kernel over *string sets*:
$K'(X,Y) = \sum_{x \in N(X)} \sum_{y \in N(Y)} K_{SSSK}(x,y)$
- can do this very efficiently with SSSK algorithm





Sparse Spatial Sample Methods: Summary

- efficiently model complex sequence transformations (multiple mutations, insertions, deletions, etc), variable length pattern
- explicitly model spatial configuration of features in the sequence
- captures sequence content at multiple scales
- fast $O(cn)$ evaluation (linear in sequence length (n), with small alphabet-independent constant c)
- alphabet-free matching (alphabet size independent)

Experiments

- Multi-Class Remote fold recognition
  - Ding & Dubchak benchmark
  - SCOP
- Multi-Class Remote Homology (Superfamily) prediction
- Fully-supervised (labeled data only)
- Semi-supervised (labeled + unlabeled)

Ding and Dubchak benchmark dataset [DD01]

- *multi-class remote* fold detection problem
- used as benchmark in many studies
- contains sequences from 27 folds divided into two *independent* sets (27**-way classification**)
  - training and test sequences share less than 35% sequence identities
  - within training set, no sequences share more than 40% sequence identities

Remote fold and homology detection data set [MIW+07]

- derived from SCOP 1.65 [LCAH+00]
- *remote fold* detection data set
  - *multi-class* fold detection problem
  - contains 26 folds, 303 superfamilies and 652 families for training
  - 46 superfamilies held out for testing
  - 24**-way classification**
- *remote* homology detection dataset
  - *multi-class* superfamily prediction
  - 74 superfamilies and 544 families for training
  - 110 families held out for testing
  - 74**-way classification**

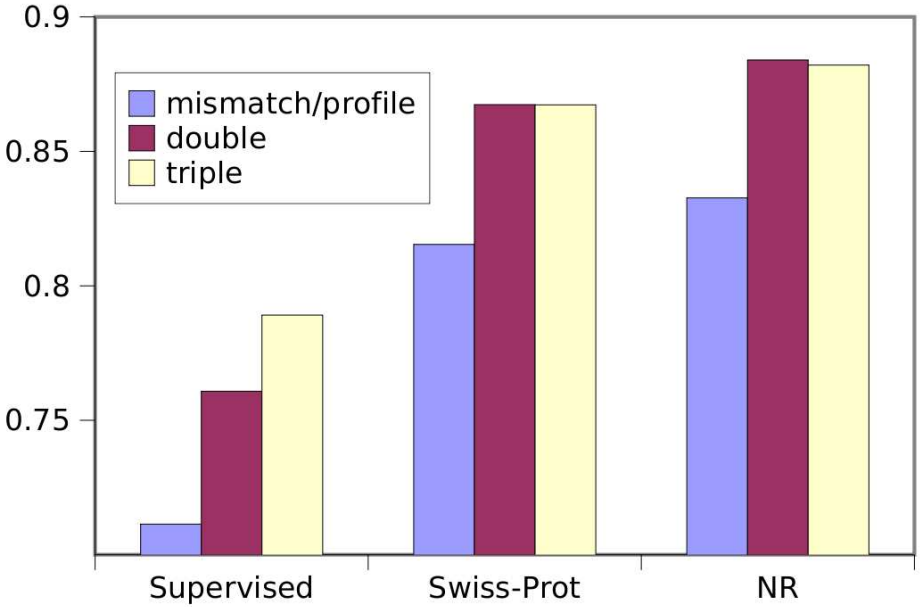
Large-Scale Protein Remote Homology and Fold Prediction

- Use large sequence databases as unlabeled data for semi-supervised learning

Dataset	# Seq	# Neighbors (mean/median/max)
Swiss-Prot	101,602	56/28.5/385
NR	534,936	114/86/490

- neighborhood $N(X)$ for each sequence X (train + test)
  $N(X) = \{X' : eValue(X, X') \leq 0.05\}$ with 2 PSI-BLAST iterations

Comparison on Ding and Dubchak data set (27-way classification)



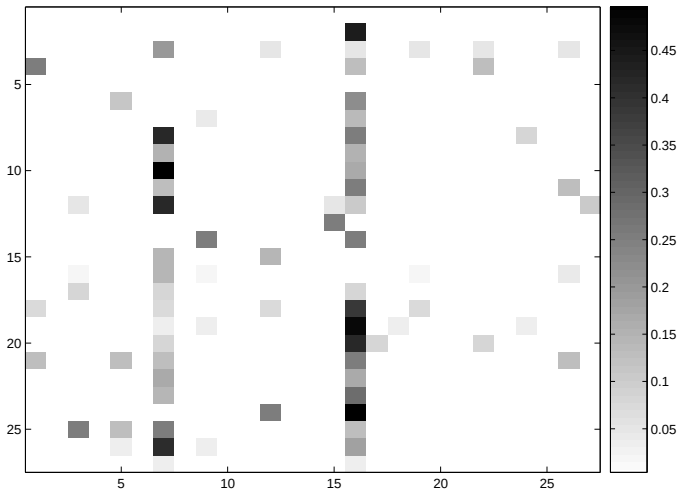
Balanced accuracy

	supervised	Swiss-Prot	NR
mismatch/profile	71.14	81.54	83.57
double	76.08	86.74	**88.40**
triple	**78.91**	**86.73**	88.21

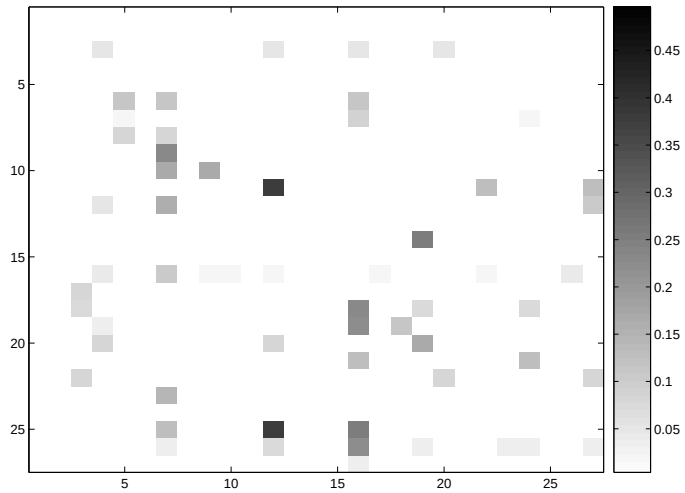
F1 score

	supervised	Swiss-Prot	NR
mismatch/profile	81.45	87.56	88.71
double	77.97	86.40	87.82
triple	**83.74**	**89.31**	**89.80**

Confusion Matrices on Ding & Dubchak dataset



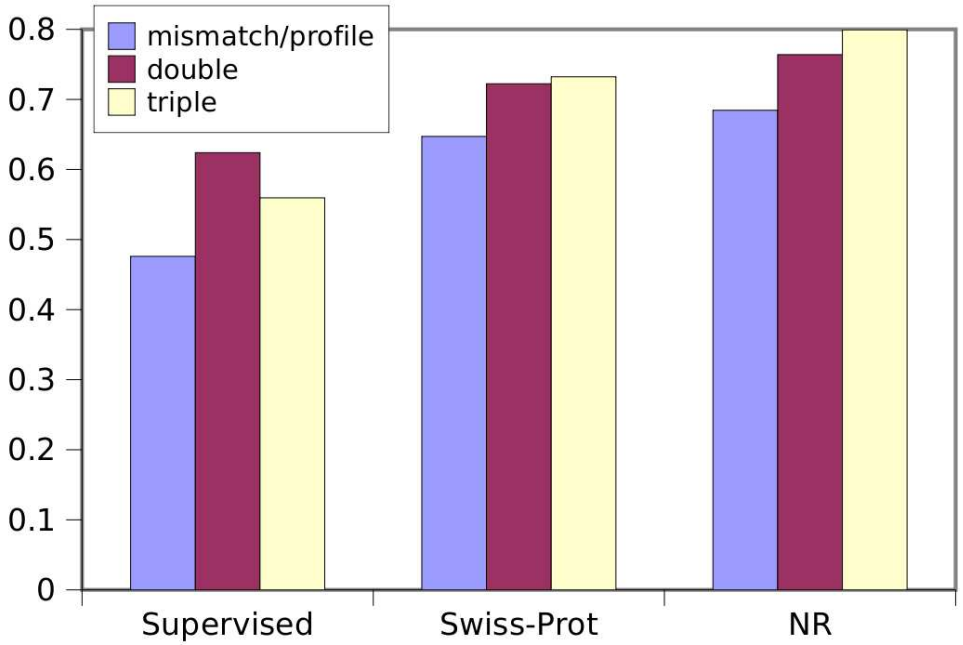
(a) supervised



(b) semi-supervised

- feature: triple(1,3)
- *supervised* (left): testing examples in classes with fewer training examples tend to be incorrectly assigned to two overly represented folds (7 and 16); 3 folds with $>$ 90% accuracy (1 achieves 100%)
- *semi-supervised* (right): alleviated such problem when we enlarge training sets with neighboring sequences; 9 folds with $>$ 90% accuracy (7 achieve 100%)

Multi-class *remote* fold prediction (26-way classification)



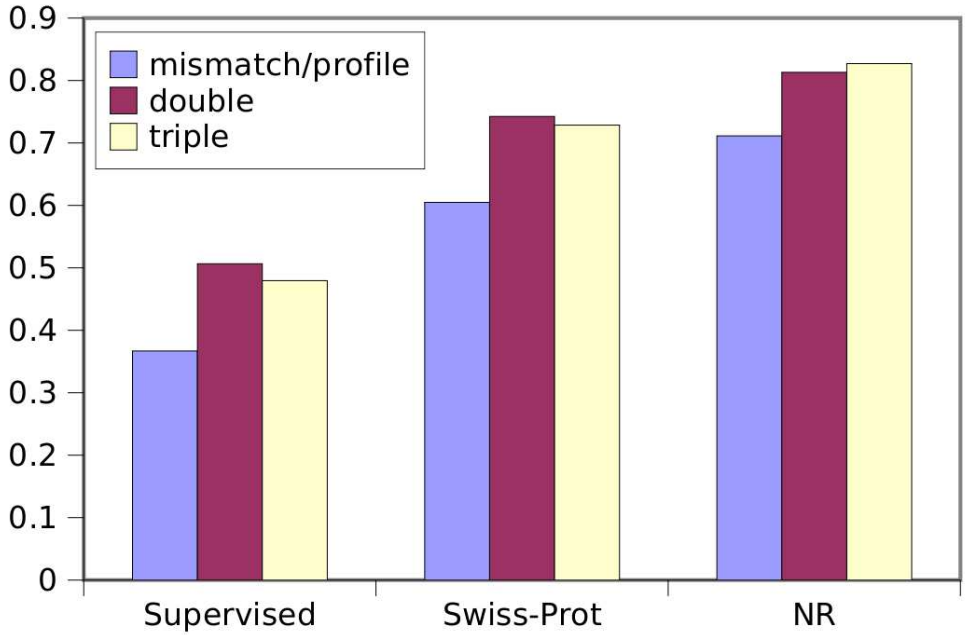
Balanced accuracy

	supervised	Swiss-Prot	NR
mismatch/profile	47.60	64.72	68.45
double	**62.40**	72.24	76.39
triple	55.95	**73.23**	**79.93**

F1 score

	supervised	Swiss-Prot	NR
mismatch/profile	56.67	68.09	75.68
double	**63.26**	71.35	77.51
triple	62.37	**75.08**	**81.33**

Multi-class *remote* homology detection (74-way classification)



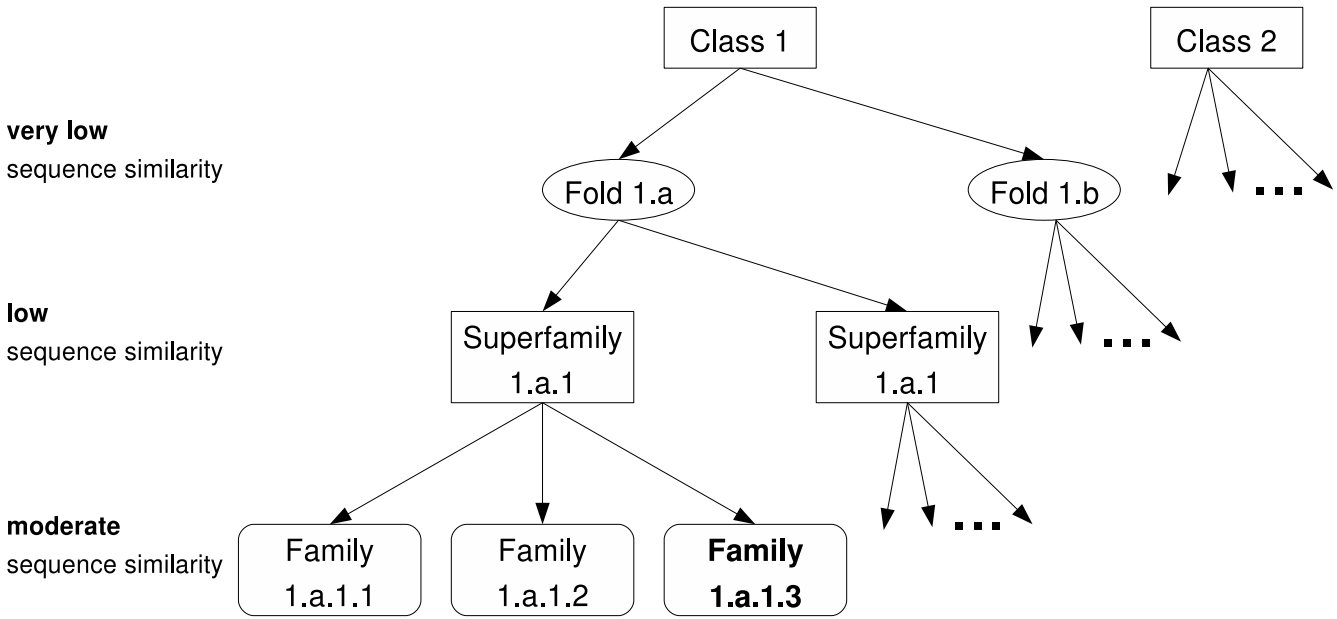
Balanced accuracy

	supervised	Swiss-Prot	NR
mismatch/profile	36.69	60.49	71.14
double	**50.66**	**74.25**	81.31
triple	47.95	72.86	**82.71**

F1 score

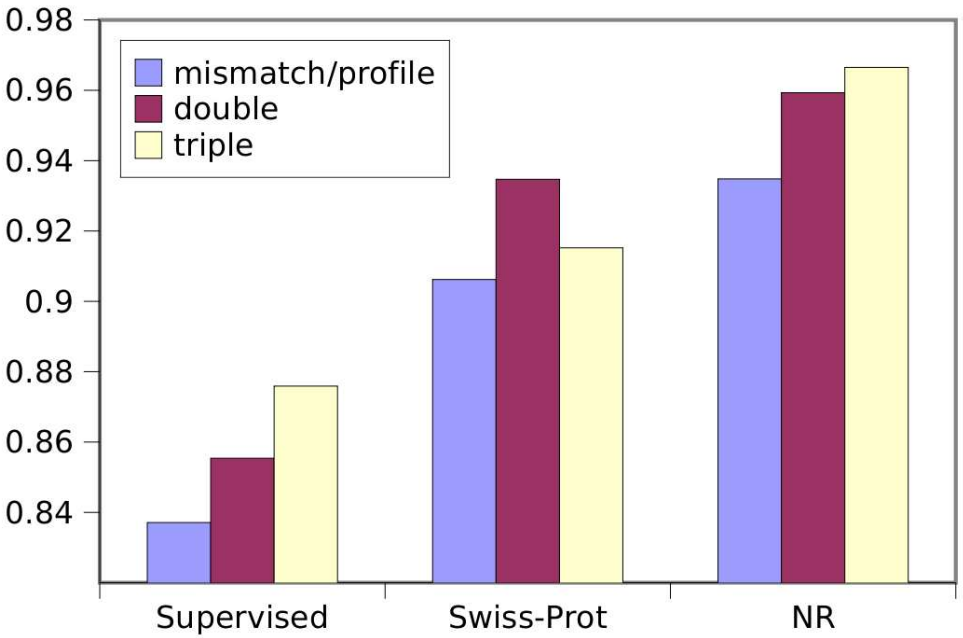
	supervised	Swiss-Prot	NR
mismatch/profile	41.18	62.36	72.19
double	51.72	**76.21**	80.94
triple	**53.75**	74.53	**83.24**

Multi-Class Protein Classification with known tree structure



Multi-class protein fold prediction (10-fold C.V., 26 classes)

- verify performance on multi-class fold prediction when there are train examples from every superfamily
  - match an unknown sequence (assumed to be from one of the known classes/subclasses)



Balanced accuracy

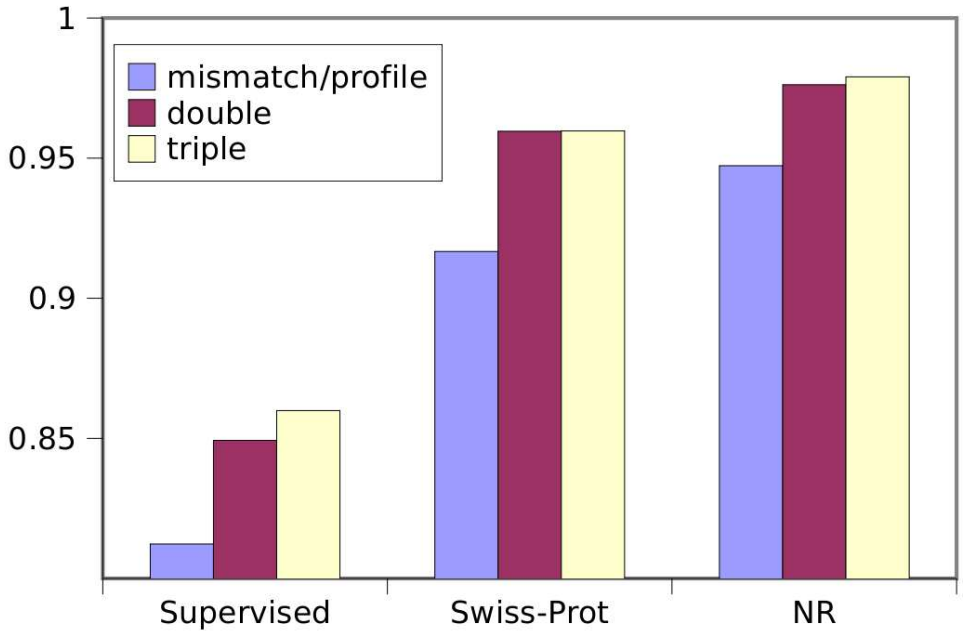
	supervised	Swiss-Prot	NR
mismatch/profile	83.71	90.62	93.48
double	85.54	**93.47**	95.93
triple	**87.59**	91.52	**96.65**

F1 score

	supervised	Swiss-Prot	NR
mismatch/profile	90.05	94.36	96.04
double	87.24	94.28	96.31
triple	**92.26**	**94.51**	**97.68**

Multi-class superfamily prediction (10-fold C.V.), 74 classes

- match an unknown sequence (assumed to be from one of the known classes/subclasses)



Balanced accuracy

	supervised	Swiss-Prot	NR
mismatch/profile	81.33	91.77	94.73
double	84.93	95.96	97.62
triple	**85.99**	**95.97**	**97.90**

F1 score

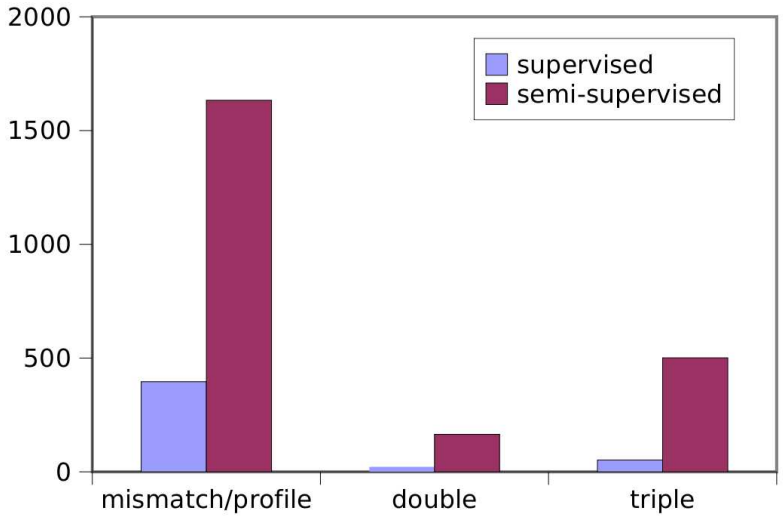
	supervised	Swiss-Prot	NR
mismatch/profile	86.11	93.82	96.16
double	86.50	96.22	97.77
triple	**89.53**	**96.96**	**98.39**

Complexity and Running time analysis

- linear $O(cn)$ complexity
- independent of alphabet set size

Method	Time complexity
Double kernel	$O(dn)$
Triple kernel	$O(d^2 n)$
Mismatch	$O(k^{m+1}\|\Sigma\|^m n)$
Profile kernel	$O(kM_\sigma n)$

Notations:
n - sequence length,
$|\Sigma|$ - alphabet size
k, m - mismatch kernel parameters
($k = 5, 6$ and $m = 1, 2$ in most cases)
M_σ - profile neighborhood size,
$k^m|\Sigma|^m \leq M_\sigma \leq |\Sigma^k|$
d - distance parameter for the spatial kernel.



Running time ratio (semi-supervised), $T_{mismatch} / T_{triple}$

# seq.	10	30	50	100
time ratio	81	93	95	115

Running time (s)

	Supervised	Semi-supervised
Mismatch	396	-
Double	22	165
Triple	52	501

Key Contributions

- state-of-the-art performance on hard multi-class problems in remote similarity settings (superfamily and fold prediction)
- accurate and computationally efficient modeling using sparse spatial sample features
- can learn from large weakly labeled sequence sets
- exploit sparse, invariant(preserved) spatial patterns
- All presented methods can be applied to a wide range of problems on biological sequences, text, as well as in other domains

Future work

- still room for improvement; notice some hard to classify superfamilies
- joint training / feature sharing framework to recover (or exploit) tree hierarchy in sequences
- automatically learn good features for sequence discrimination from data
- general sequence classification: images, text data, *etc.*

References

Chris H.Q. Ding and Inna Dubchak.

Multi-class protein fold recognition using support vector machines an d neural networks .
Bioinformatics, 17(4):349–358, 2001.

Eugene Ie, Jason Weston, William Stafford Noble, and Christina Leslie.

Multi-class protein fold recognition using adaptive codes.
In *ICML '05: Proceedings of the 22nd international conference on Machine learning*, pages 329–336, New York, NY, USA, 2005. ACM.

Rui Kuang, Eugene Ie, Ke Wang, Kai Wang, Mahira Siddiqi, Yoav Freund, and Christina Leslie.

Profile-based string kernels for remote homology detection and motif extraction.
In *CSB '04: Proceedings of the 2004 IEEE Computational Systems Bioinformatics Conference (CSB'04)*, pages 152–160, August 2004.
http://www.cs.columbia.edu/compbio/profile-kernel.

L. Lo Conte, B. Ailey, T.J. Hubbard, S.E. Brenner, A.G. Murzin, and C. Chothia.

SCOP: a structural classification of proteins database.
Nucleic Acids Res., 28:257–259, 2000.

Christina S. Leslie, Eleazar Eskin, Jason Weston, and William Stafford Noble.

Mismatch string kernels for svm protein classification.
In *NIPS*, pages 1417–1424, 2002.

Iain Melvin, Eugene Ie, Jason Weston, William Stafford Noble, and Christina Leslie.

Multi-class protein classification using adaptive codes.
J. Mach. Learn. Res., 8:1557–1581, 2007.

Vladimir N. Vapnik.

Statistical Learning Theory.