# Sequence Data Mining: Techniques and Applications

Sunita Sarawagi
IIT Bombay
http://www.it.iitb.ac.in/~sunita

## What is a sequence?

- Ordered set of elements: $s = a_1, a_2, .. a_n$
- Each $a_i$ could be
  - Categorical: domain a finite set of symbols $\Sigma$, $|\Sigma|$=m
  - Numerical
  - Multiple attributes
- The length *n* of a sequence is not fixed
- Order determined by time or position and could be regular or irregular

## Motivation

- Several real-life mining applications on sequence data
- Classical applications
  - Speech, language, handwritten are all complex sequences
- Newer applications
  - Bio-informatics: DNA and proteins
  - Telecommunication: Network alarms, network packet data
  - Retail data mining: Customer behavior

## Outline

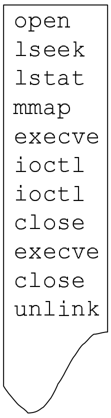
- Three case studies
  - Intrusion detection
  - Information Extraction
  - Bio-informatics: protein classification
- Sequence mining operators
- Approaches to sequence mining
- Conclusions and future work

## Case study: intrusion detection

- Intrusions could be detected at
  - Host-level (attacks on privileged programs like lpr, sendmail)
  - Network-level (denial-of-service attacks, port-scans, etc)
    - TCP-dumps
- Method
  - Signature-based (match signature of previous attacks)
    - cannot detect new intrusions
  - Anomaly-based (model normal usage and detect deviation)
- Automatic Vs Manual:
  - Manual:
    - Might miss patterns, may not evolve as normal usage pattern slowly drifts.
  - Automated:
    - Use historical audit trails and a learning algorithm
    - May not provide full coverage

## Host-level attacks on privileged programs

- Attacks exploit a loophole in the program to do illegal actions
  - Example: exploit buffer over-flows to run user-code
- What to monitor of an executing privileged program to detect attacks?
- Sequence of system calls
  - |Σ| = set of all possible system calls ~100
- Mining problem: given traces of previous normal execution, monitor a new execution and flag attack or normal
- Challenge: is it possible to do this given widely varying normal conditions?



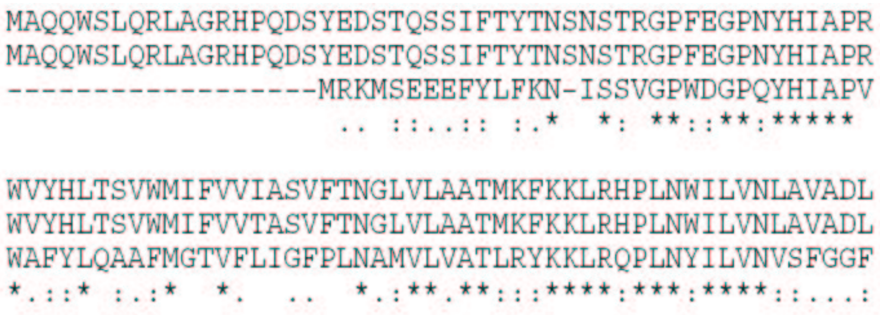
```
open
lseek
lstat
mmap
execve
ioctl
ioctl
close
execve
close
unlink
```

## Bio-informatics

- Many recent advances in sequence analysis due to bio-informatics
- Two main kinds of sequences:
  - Genes:
    - Sequence of 4 possible nucleotides, |Σ|=4
    - AACTGACCTGGGCCCAATCC
  - proteins:
    - Sequence of 20 possible amino-acids, |Σ|=20
    - Length of sequence n varies between 100s to 10,000
- Sequence analysis in bio-informatics: rich and varied, we will concentrate on one problem
  - Protein family classification

## Protein family classification

- Protein families characterized by common occurrence of a few scattered amino acids in a background of other unrelated symbol
- Example: three aligned sequences of a family



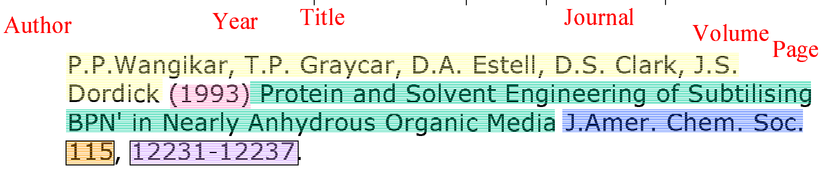
```
MAQQWSLQRLAGRHPQDSYEDSTQSSIFTYTNSNSTRGPFEGPNYHIAPR
MAQQWSLQRLAGRHPQDSYEDSTQSSIFTYTNSNSTRGPFEGPNYHIAPR
------------------MRKMSEEEFYLFKN-ISSVGPWDGPQYHIAPV
                   .. ::..:: :.*  *: **::**:*****

WVYHLTSVWMIFVVIASVFTNGLVLAATMKFKKLRHPLNWILVNLAVADL
WVYHLTSVWMIFVVTASVFTNGLVLAATMKFKKLRHPLNWILVNLAVADL
WAFYLQAAFMGTVFLIGFPLNAMVLVATLRYKKLRQPLNYILVNVSFGGF
*.::* :.:*  *.  ..  *.:**.**:::****:***:****::...:
```

## Information extraction

Sequence: text string with elements as words

- Example: Addresses, bib records

| House number | Building | Road | Area | City | Zip |
|---|---|---|---|---|---|
| 156 | Hillside ctype | Scenic drive | Powai | Mumbai | 400076 |



P.P.Wangikar, T.P. Graycar, D.A. Estell, D.S. Clark, J.S. Dordick (1993) Protein and Solvent Engineering of Subtilising BPN' in Nearly Anhydrous Organic Media J.Amer. Chem. Soc. 115, 12231-12237.

Mining problem:

Given a set of tags (labels) e.g. address fields, classify parts of the sequence to different labels

## Outline

- Three case studies
- Sequence mining operators
  - Whole sequence classification
  - Partial sequence classification (Tagging)
  - Predicting next symbol of a sequence
  - Clustering sequences
  - Finding repeated patterns in a sequence
- Approaches to sequence mining
- Conclusion and future work

## Classification of whole sequences

Given:

- a set of classes C and
- a number of example of instances in each class c,

train a model so that for an unseen sequence we can say to which class it belongs

Example:

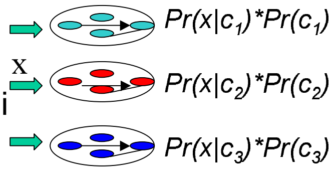
- Given a set of protein families, find family of new protein
- Given a sequence of packets, predict session as intrusion or not
- Given several utterances of a set of words, classify a new utterance to the right word

## Existing methods of classification

- Generative classifiers
- Discriminatory classifiers
- Distance based classifiers: (Nearest neighbor)
- Kernel-based classifiers

# Generative models

- For each class *i*,
  - train a generative model $M_i$ to maximize likelihood over all training sequences in the class i
- Find $\Pr(c_i)$ as fraction of training instances in class i
- For new sequence x,
  - find $Pr(x|c_i)$ for each i
  - choose *i* with largest value of $Pr(x|c_i)*P(c_i)$



x

$Pr(x|c_1)*Pr(c_1)$

$Pr(x|c_2)*Pr(c_2)$

$Pr(x|c_3)*Pr(c_3)$

Need a generative model for sequence data

# Discriminatory methods

- Treat training data as points in n-dimensional space
- Create boundaries such that all points in the same region are in the same class
- Examples:
  - Decision trees
  - Neural networks
  - Regression methods

Need to embed sequence data in a fixed coordinate space

# Kernel-based classifiers

- Define function $K(x_i, x)$ that intuitively defines similarity between two sequences and satisfies two properties
  - K is symmetric i.e., $K(x_i, x) = K(x, x_i)$
  - K is positive definite
- Each class c computes $f(x,c) = \Sigma\, w_{ic}K(x_i, x)+b_c$ where $x_i$, is a training sequence
- Predicted class is c with highest value f(x,c)
- Well-known kernel classifiers
  - Nearest neighbor classifier
  - Support Vector Machines
  - Radial Basis functions

Need to define similarity functions between sequences that also satisfy kernel properties

# Partial sequence classification (Tagging)

- The tagging problem:
  - Given:
    - A set of tags L
    - Training examples of sequences showing the breakup of the sequence into the set of tags
  - Learn to breakup a sequence into tags
  - (classification of parts of sequences)
- Examples:
  - Text segmentation
    - Break sequence of words forming an address string into subparts like Road, City name etc
  - Continuous speech recognition
    - Identify words in continuous speech

## Approaches used for tagging

- Rule-based local models
- Adapt state-based generative models
  - Separate model per tag
  - Combined model with states labeled with tags
    - Normal Generative models
    - Special Conditional models (Collins 02)

## Sequence clustering

- Given a set of sequences, create groups such that similar sequences in the same group
- Three kinds of clustering algorithms
  - Distance-based:
    - K-means
    - Various hierarchical algorithms
  - Model-based algorithms
    - Expectation Maximization algorithm
  - Density-based algorithms

Need similarity function

Need generative models

Need dimensional embedding

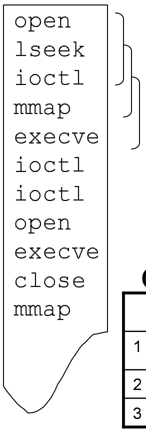
# Outline

- Three case studies
- Sequence mining operators
- Approaches to sequence mining: Three primitives
  - Embed sequence in a fixed dimensional space
    - All conventional record mining techniques will apply
  - Distance between two sequences
    - Sequence classification: SVM and NN
    - Clustering sequences: distance-based approach
  - Generative models for sequence
    - Sequence classification: whole and partial
    - Clustering sequences: model-based approach
- Conclusion and future work

# Embedding sequences in fixed dimensional space

- Extract aggregate features
  - Real-valued elements: Fourier coefficients, Wavelet coefficients, Auto-regressive coefficients
  - Categorical data: number of symbol changes
- Ignore order, each symbol a dimension
  - extensively used in text classification and clustering
- Sliding window techniques (k: window size)
  - Define a coordinate for each possible k-gram $\alpha$
    - □ $\alpha$-th co-ordinate is number of times $\alpha$ in sequence
    - (k,m) mismatch score: $\alpha$-th co-ordinate is number of k-grams in sequence with m mismatches with $\alpha$
  - Define a coordinate for each of the k-positions

# Sliding window examples



open
lseek
ioctl
mmap
execve
ioctl
ioctl
open
execve
close
mmap

One symbol per column

| | o | c | l | i | e | m |
|---|---|---|---|---|---|---|
| 1 | 2 | 1 | 1 | 3 | 2 | 1 |
| 2 | .. | .. | .. | .. | .. | .. |
| 3 | .. | .. | .. | .. | .. | .. |

Sliding window: window-size 3

One row per trace

| | ioe | cli | oli | lie | lim | ... |
|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 1 | 0 | 1 | |
| 2 | .. | .. | .. | .. | .. | .. |
| 3 | .. | .. | .. | .. | .. | .. |

mis-match scores: m=1

| | ioe | cli | oli | lie | lim | ... |
|---|---|---|---|---|---|---|
| 1 | 2 | 1 | 1 | 0 | 1 | |
| 2 | .. | .. | .. | .. | .. | .. |
| 3 | .. | .. | .. | .. | .. | .. |

Multiple rows per trace

| | A1 | A2 | A3 |
|---|---|---|---|
| 1 | o | l | i |
| 1 | l | i | m |
| 1 | i | m | e |
| 1 | .. | .. | .. |
| 1 | e | c | m |

# Detecting attacks on privileged programs

- Short sequences of system calls made during normal execution of system calls are very consistent, yet different from the sequences of its abnormal executions
- Each execution a trace of system calls:
  - ignore online traces for the moment
- Two approaches
  - STIDE
    - Create dictionary of unique k-windows in normal traces, count what fraction occur in new traces and threshold.
  - IDS
    - next...

## Classification models on k-grams trace data

- When both normal and abnormal data available
  - class label = normal/abnormal:

| *7-grams* | class labels |
|---|---|
| 4 2 66 66 4 138 66 | "normal" |
| 5 5 5 4 59 105 104 | "abnormal" |
| … | … |

- When only normal trace,
  - class-label=k-th system call

| *6 attributes* | Class labels |
|---|---|
| 4 2 66 66 4 138 | "66" |
| 5 5 5 4 59 105 | "104" |
| … | … |

Learn rules to predict class-label [RIPPER]

## Examples of output RIPPER rules

- Both-traces:
  - if the 2nd system call is *vtimes* and the 7th is *vtrace*, then the sequence is "normal"
  - if the 6th system call is *lseek* and the 7th is *sigvec*, then the sequence is "normal"
  - …
  - if none of the above, then the sequence is "abnormal"
- Only-normal:
  - if the 3rd system call is *lstat* and the 4th is *write*, then the 7th is *stat*
  - if the 1st system call is *sigblock* and the 4th is *bind*, then the 7th is *setsockopt*
  - …
  - if none of the above, then the 7th is *open*

## Experimental results on sendmail

- The output rule sets contain ~250 rules, each with 2 or 3 attribute tests

- Score each trace by counting fraction of mismatches and thresholding

Summary: Only normal traces sufficient to detect intrusions

| *traces* | *Only-normal* | *BOTH* |
|---|---|---|
| sscp-1 | 13.5 | 32.2 |
| sscp-2 | 13.6 | 30.4 |
| sscp-3 | 13.6 | 30.4 |
| syslog-remote-1 | 11.5 | 21.2 |
| syslog-remote-2 | 8.4 | 15.6 |
| syslog-local-1 | 6.1 | 11.1 |
| syslog-local-2 | 8.0 | 15.9 |
| decode-1 | 3.9 | 2.1 |
| decode-2 | 4.2 | 2.0 |
| sm565a | 8.1 | 8.0 |
| sm5x | 8.2 | 6.5 |
| sendmail | 0.6 | 0.1 |

## More realistic experiments

| | STIDE | | RIPPER | |
|---|---|---|---|---|
| | threshold | %false-pos | threshold | %false-pos |
| Site-1 lpr | 12 | 0.0 | 3 | 0.0016 |
| Site-2 lpr | 12 | 0.0013 | 4 | 0.0265 |
| named | 20 | 0.0019 | 10 | 0.0 |
| xlock | 20 | 0.00008 | 10 | 0.0 |

- Different programs need different thresholds
- Simple methods [stide] work as well
- Results sensitive to window size
- Is it possible to do better with sequence specific methods?

## Outline

- Three case studies
- Sequence mining operators
- Approaches to sequence mining: Three primitives
  - Embed sequence in a fixed dimensional space
  - Distance between two sequences
  - Generative models for sequence
    - Sequence classification: whole and partial
    - Clustering sequences: model-based approach
- Conclusion and future work

## Modeling sequences

- Most sequences are naturally generated and may not follow a well-defined statistical model
- Complete modeling not possible
- Approximate modeling still possible in many applications because
  - Sequences have short-term memory
  - A partial aspect of the sequence might need to be modeled

## Probabilistic models for sequences

$$\Pr(a_1, a_2, \ldots, a_n) = \prod_{i=1}^{n} \Pr(a_i | a_1 \ldots a_{i-1})$$

- Independent model

  $$\Pr(a_i | a_1 \ldots a_{i-1}) = \Pr(a_i)$$
- One-level dependence (Markov chains)

  $$\Pr(a_i | a_1 \ldots a_{i-1}) = \Pr(a_i | a_{i-1})$$
- Fixed memory (Order-$l$ markov chains)

  $$\Pr(a_i | a_1 \ldots a_{i-1}) = \Pr(a_i | a_{i-1} \ldots a_{i-l})$$
- Variable memory models

  $$\Pr(a_i | a_1 \ldots a_{i-1}) = \Pr(a_i | a_{i-1} \ldots a_{i-l_i}), \quad l_i < l$$
- More complicated models
  - Hidden Markov Models

## Independent model

$$\Pr(a_i | a_1 \ldots a_{i-1}) = \Pr(a_i)$$

- Model structure
  - A parameter for each symbol in Σ
- Probability of a sequence *s* being generated from the model
  - example: Pr(AACA)

    = P(A) P(A) P(C) P(A) = P(A)$^3$ P(C)

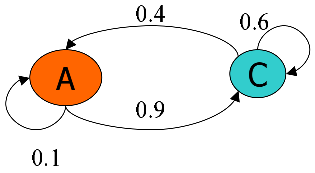
    = 0.1$^3$£ 0.9
- Training transitions probability between states
  - Data T : set of sequences
  - Count(s ε T): total number of times substring s appears in training data T

  Pr(σ) = Count(σ ε T) / length(T)

Pr(A) = 0.1<br>
Pr(C) = 0.9

## Markov chains (Order(1))

$$\text{Pr}(a_i|a_1 \ldots a_{i-1}) = \text{Pr}(a_i|a_{i-1})$$



- Model structure
  - A state for each symbol in Σ
  - Edges between states with probabilities
- Probability of a sequence *s* being generated from the model
  - example: Pr(AACA)

    = P(A|A) P(A|A) P(C|A) P(A|C)

    = 0.1*0.1*0.9*0.4
- Training transitions probability between states

  Pr(σ|β) = Count(βσ **ε** T) / Count(β **ε** T)

## Higher order Markov Chains

$$\text{Pr}(a_i|a_1 \ldots a_{i-1}) = \text{Pr}(a_i|a_{i-1} \ldots a_{i-l})$$

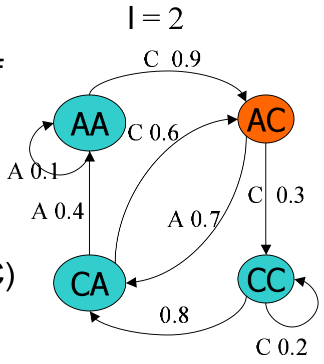
l = memory of sequence

- Model
  - A state for each possible suffix of length l ➔ $|\Sigma|^l$ states
  - Edges between states with probabilities and single symbols
- P(AACA)

  = P(A|AC) P(A|CA)P(C|AA) P(A|AC)

  = 0.7*0.4*0.9*0.7
- Training model

  Pr(σ|s) = count(sσ **2** T) / count(s **2** T)

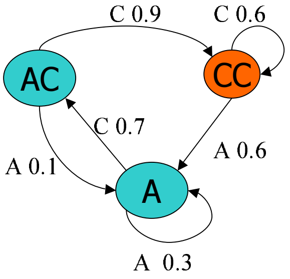


# Variable Memory models

- Probabilistic Suffix Automata (PSA)

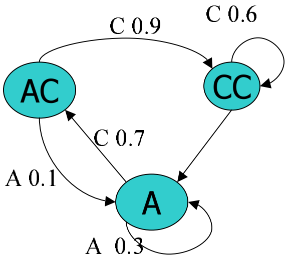
$$\mathrm{Pr}(a_i|a_1 \ldots a_{i-1}) = \mathrm{Pr}(a_i|a_{i-1} \ldots a_{i-l_i}), \quad l_i < l$$

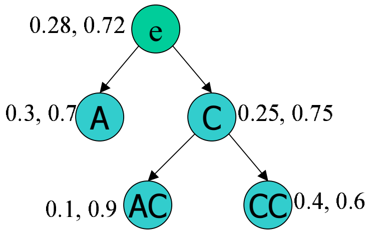
- Model
  - States: substrings of size no greater than l where no string is suffix of another
- Calculating Pr(AACA):
  - = P(A|CC)P(A|A)P(C|A)P(A|AC)
  - = 0.6*0.3*0.7*0.1
- Training: not straight-forward
  - Eased by Prediction Suffix Trees
  - PSTs can be converted to PSA after training



# Prediction Suffix Trees (PST)

- Suffix trees with emission probabilities of observation attached with each tree node





P(AACA)=0.28*0.3*0.7*0.1

- Linear time algorithms exist for constructing such PSTs from training data [Apostolico 2000]

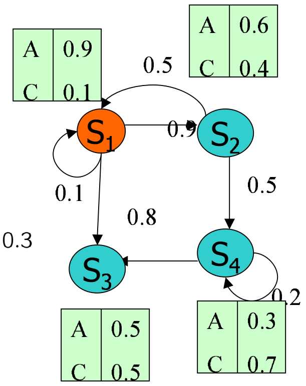
# Hidden Markov Models

- Doubly stochastic models

$\text{Pr}(AACA) = \sum_{ijkl} \text{Pr}(AACA, S_i S_j S_k S_l)$

$\text{Pr}(AACA, S_i S_j S_k S_l) = \text{Pr}(S_i)\,\text{Pr}(A|S_i)\,\text{Pr}(S_j|S_i)..\,\text{Pr}(A|S_l)$

$\text{Pr}(AACA, S_1 S_2 S_4 S_4) = 1 * 0.9 * 0.9 * 0.6 * 0.5 * 0.7 * 0.2 * 0.3$



- Efficient dynamic programming algorithms exist for
  - Finding Pr(S)
  - The highest probability path P that maximizes Pr(S|P) (Viterbii)
- Training model
  - Baum-Welch algorithm

# Discriminative training of HMMs

- Models trained to maximize likelihood of data might perform badly when
  - Model not representative of data
  - Training data insufficient
- Alternatives to Maximum-likelihood/EM
  - Objective functions:
    - Minimum classification error
    - Maximum posterior probability of actual label Pr(c|x)
    - Maximum mutual information with class
  - Harder to train above functions, number of alternatives to EM proposed
    - Generalized probabilistic descent [Katagiri 98]
    - Deterministic annealing [Rao 01]

## HMMs for profiling system calls

- Training:
  - Initial number of states = 40 (roughly equals number of distinct system calls)
  - Train using Baum Welch on normal traces
- Methods of testing:
  - Need to handle variable length and online data
  - For each call, find the total probability of outputting given all calls before it.
    - If probability below a threshold call it abnormal.
  - Trace is abnormal if fraction of abnormal calls are high

## More realistic experiments

| | STIDE | | RIPPER | | HMM | |
|---|---|---|---|---|---|---|
| | threshold | %false-pos | threshold | %false-pos | threshold | %false-pos |
| Site-1 lpr | 12 | 0.0 | 3 | 0.0016 | $10^{-7}$ | 0.0003 |
| Site-2 lpr | 12 | 0.0013 | 4 | 0.0265 | $10^{-7}$ | 0.0015 |
| named | 20 | 0.0019 | 10 | 0.0 | $10^{-7}$ | 0.0 |
| xlock | 20 | 0.00008 | 10 | 0.0 | $10^{-7}$ | 0.0 |

[from Warrender 99]

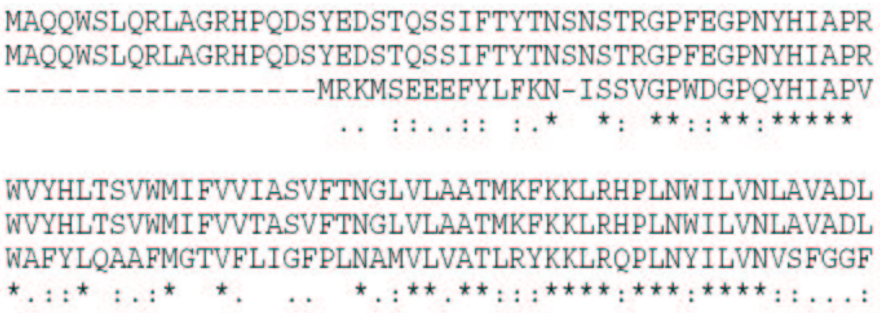
- HMMs
  - Take longer time to train
  - Less sensitive to thresholds, no window parameter
  - Best overall performance
- VMM and Sparse Markov Transducers also shown to perform significantly better than fixed window methods [Eskin 01]

## Case study: classifying protein sequences

- Classifying proteins into its functional/structural classes based on its sequence of amino acids
- Methods proposed
  - Nearest neighbor classifiers based on pair-wise sequence alignment as the distance measure
  - Consensus patterns using Motifs
  - Profile Hidden Markov Models
  - Support Vector Machines with various kernels
    - Fisher's kernel (Fisher-SVM)
    - Mismatch string kernels

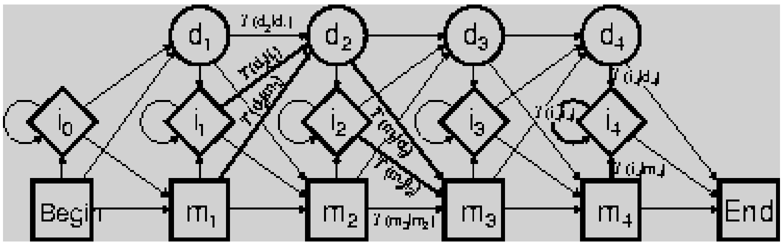
## Profile Hidden Markov Models

- Protein families characterized by common occurrence of a few scattered amino acids in a background of other unrelated symbol



```
MAQQWSLQRLAGRHPQDSYEDSTQSSIFTYTNSNSTRGPFEGPNYHIAPR
MAQQWSLQRLAGRHPQDSYEDSTQSSIFTYTNSNSTRGPFEGPNYHIAPR
------------------MRKMSEEEFYLFKN-ISSVGPWDGPQYHIAPV
                   .. ::..:: :.*  *: **::**:*****

WVYHLTSVWMIFVVIASVFTNGLVLAATMKFKKLRHPLNWILVNLAVADL
WVYHLTSVWMIFVVTASVFTNGLVLAATMKFKKLRHPLNWILVNLAVADL
WAFYLQAAFMGTVFLIGFPLNAMVLVATLRYKKLRQPLNYILVNVSFGGF
*.::* :.:*  *.  ..  *.:**.**:::****:***:****::...:
```

# Profile HMM



Profile HMM of a family has for each aligned symbol three kinds of states:

- Match state: visited when symbol appears in a sequence
- Deletes states: to allow occasional drop of that symbol
- Inserts: to allow insertion of multiple symbols between aligned states

[Above picture from http://www.cse.ucsc.edu/research/compbio/html_format_papers/hughkrogh96/node4.html

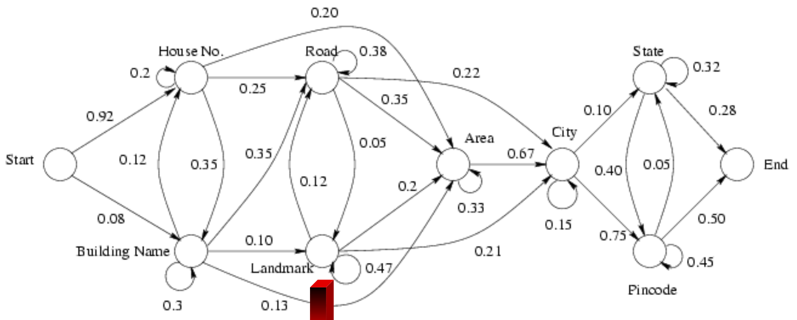
# SVMs on Fisher's kernel

- Train a HMM for the positive class,
  - $\theta$: set of all parameters of the HMM
  - $\theta_m$: the trained values of parameters
- Fisher's score for each sequence s is gradient vector w.r.t $\theta$,

  that is, **r** $Pr(s|\theta)|_{\theta=\theta m}$
- For two sequences $s_j$, $s_k$, kernel is $K(s_j,s_k)$ = similarity between their fisher's score
- Train SVM using this kernel
- Combines biological information in HMM with discriminatory power of SVMs

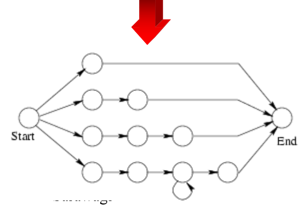
## HMMs for information extraction

- Naïve Model:  One state per element



- Nested model

Each element another HMM



## Summary

- Several applications of sequence mining
- Record mining techniques on sequence data may not be effective
- Many interesting options for sequence-specific generative models
- Case studies on three applications:
  - Intrusion detection
  - Protein classification
  - Information Extraction
- Future work: practical general purpose data mining tools for handling sequence data

# References

- S. F. Altschul, T. L. Madden, A. A. Schaffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman. Gapped BLAST and PSI-BLAST: A new generation of protein database search programs. *Nucleic Acids Research*, 25:3389–3402, 1997.
- Apostolico, A., and Bejerano, G. 2000. Optimal amnesic probabilistic automata or how to learn and classify proteins in linear time and space. In Proceedings of RECOMB2000. http://citeseer.nj.nec.com/apostolico00optimal.html
- R. Durbin, S. Eddy, A. Krogh, and G. Mitchison, *Biological sequence analysis: probabilistic models of proteins and nucleic acids*, Cambridge University Press, 1998.
- Eleazar Eskin, Wenke Lee and Salvatore J. Stolfo. ``Modeling System Calls for Intrusion Detection with Dynamic Window Sizes.'' *Proceedings of DISCEX II*. June 2001.
- IDS http://www.cs.columbia.edu/ids/publications/
- T. Jaakkola, M. Diekhans, and D. Haussler. A discriminative framework for detecting remote protein homologies. *Journal of Computational Biology*, 2000.
- D. Haussler. Convolution kernels on discrete structure. Technical report, UC Santa Cruz, 1999.
- Wenke Lee and Sal Stolfo. ``Data Mining Approaches for Intrusion Detection'' *In Proceedings of the Seventh USENIX Security Symposium (SECURITY '98)*, San Antonio, TX, January 1998
- A. G. Murzin, S. E. Brenner, T. Hubbard, and C. Chothia. SCOP: A structural classification of proteins database for the investigation of sequences and structures. *Journal of Molecular Biology*, 247:536–540, 1995.
- Rabiner, Lawrence R., 1990. *A tutorial on hidden Markov models and selected applications in speech recognition.* In Alex Weibel and Kay-Fu Lee (eds.), Readings in Speech Recognition. Los Altos, CA: Morgan Kaufmann, pages 267--296.
- D. Ron, Y. Singer and N. Tishby. The power of amnesia: learning probabilistic automata with variable memory length. Machine Learning, 25:117-- 149, 1996
- Warrender, Christina, Stephanie Forrest, and Barak Pearlmutter. Detecting Intrusions Using System Calls: Alternative Data Models. To appear, 1999 IEEE Symposium on Security and Privacy. 1999