Sequence Classification

with emphasis on Hidden Markov Models and Sequence Kernels

Andrew M. White

September 29, 2009

THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL

THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

THE UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Examples

Biological Sequence Analysis

- genes
- proteins

Examples

Biological Sequence Analysis

- genes
- proteins

Temporal Pattern Recognition

- speech
- gestures

Examples

Biological Sequence Analysis

- genes
- proteins

Temporal Pattern Recognition

- speech
- gestures

Semantic Analysis

- handwriting
- part-of-speech detection

THE UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Sequential Data

Characteristics

- an example of *structured data*
- exhibit *sequential correlation*, i.e., nearby values are likely to be related

Why not just use earlier techniques?

- difficult to find appopriate features
- structural information is important

THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Example Framework

Speech Recognition

- goal: identify individual phonemes (the building blocks of speech, sounds like “ch” and “t”)

Example Framework

Speech Recognition

- goal: identify individual phonemes (the building blocks of speech, sounds like “ch” and “t”)
- source data:
  - quantized speech waveforms
  - tagged phonemes as sequence of values

THE UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Example Framework

Speech Recognition

- goal: identify individual phonemes (the building blocks of speech, sounds like “ch” and “t”)
- source data:
  - quantized speech waveforms
  - tagged phonemes as sequence of values
- multiple classes, each:
  - has hundreds to thousands of sequences
  - sequences vary in length

Sequential Data

Methods

Hidden Markov Models

- Evaluation: The Forward Algorithm
- Decoding: The Viterbi Algorithm
- Learning: The Baum-Welch Algorithm
- Profile HMMs

Kernels for Sequences

- Fixed-Length Subsequence Kernels
- All-Subsequences Kernel
- Variations

THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL

Methods for Sequence Classification

Generative Models

- Hidden Markov Models
- Stochastic Context-Free Grammars
- Conditional Random Fields

Methods for Sequence Classification

Generative Models

- Hidden Markov Models
- Stochastic Context-Free Grammars
- Conditional Random Fields

Discriminative Methods

- Kernel Methods (incl. SVMs)
- Max-margin Markov Networks

Methods for Sequence Classification

Generative Models

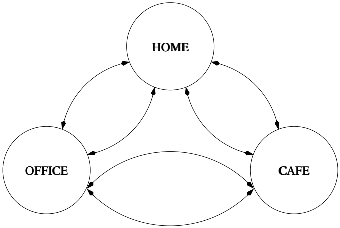
- Hidden Markov Models
- Stochastic Context-Free Grammars
- Conditional Random Fields

Discriminative Methods

- Kernel Methods (incl. SVMs)
- Max-margin Markov Networks

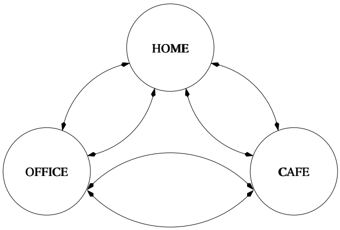
THE UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

First Order Markov Models



THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

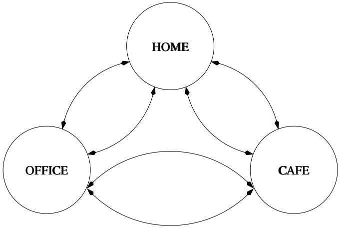
First Order Markov Models



Transition Probabilities

	home	office	cafe
home	0.2	0.6	0.2
office	0.5	0.2	0.3
cafe	0.2	0.8	0.0

First Order Markov Models

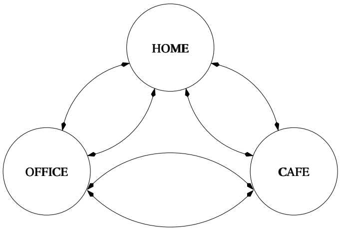


Transition Probabilities

	home	office	cafe
home	0.2	0.6	0.2
office	0.5	0.2	0.3
cafe	0.2	0.8	0.0

Assuming current state depends ONLY on previous, can easily determine probability of any path, e.g., *home, cafe, office, home*:

First Order Markov Models



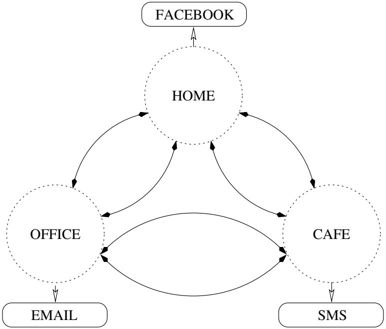
Transition Probabilities

	home	office	cafe
home	0.2	0.6	0.2
office	0.5	0.2	0.3
cafe	0.2	0.8	0.0

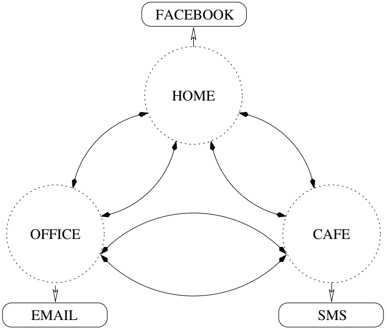
Assuming current state depends ONLY on previous, can easily determine probability of any path, e.g., *home, cafe, office, home*:

$$P(HCOH) = P(C|H)P(O|C)P(H|O) = (0.2)(0.8)(0.5) = 0.08$$

First Order Hidden Markov Models



First Order Hidden Markov Models

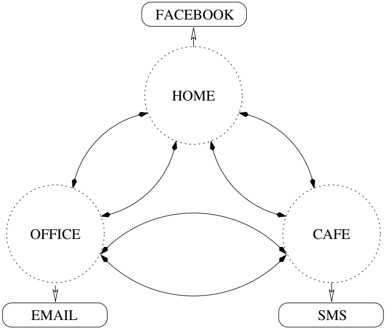


Transition Probabilities

	home	office	cafe
home	0.2	0.6	0.2
office	0.5	0.2	0.3
cafe	0.2	0.8	0.0

Can't directly observe the states – only the emissions. Does this change anything?

First Order Hidden Markov Models



Transition Probabilities

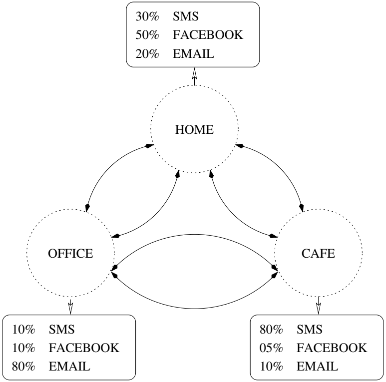
	home	office	cafe
home	0.2	0.6	0.2
office	0.5	0.2	0.3
cafe	0.2	0.8	0.0

Can't directly observe the states – only the emissions. Does this change anything?

In this case, no.

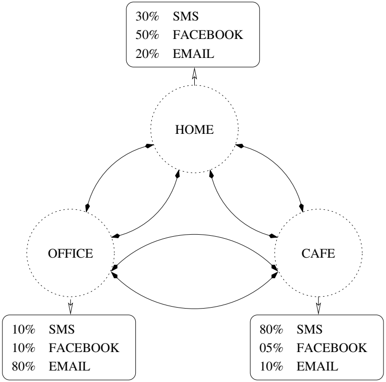
$$P(FSEF) = P(HCOH) = P(C|H)P(O|C)P(H|O) = (0.2)(0.8)(0.5) = 0.08$$

First Order Hidden Markov Models (cont.)



What if the emissions aren't tied to individual states?

First Order Hidden Markov Models (cont.)



What if the emissions aren't tied to individual states?

First Order Hidden Markov Models (cont.)

Transition Probabilities

	home	office	cafe
home	0.2	0.6	0.2
office	0.5	0.2	0.3
cafe	0.2	0.8	0.0

Emission Probabilities

	sms	facebook	email
home	0.3	0.5	0.2
office	0.1	0.1	0.8
cafe	0.8	0.1	0.1

Now we have to look at all possible state sequences which could have generated the given observation sequence.

The Three Canonical Problems of Hidden Markov Models

Evaluation

Given:	parameters, observation sequence
Find:	P(observation sequence | parameters)

The Three Canonical Problems of Hidden Markov Models

Evaluation

Given: parameters, observation sequence
Find: P(observation sequence | parameters)

Decoding

Given: parameters, observation sequence
Find: most likely *state* sequence

The Three Canonical Problems of Hidden Markov Models

Evaluation

Given: parameters, observation sequence
Find: P(observation sequence | parameters)

Decoding

Given: parameters, observation sequence
Find: most likely *state* sequence

Learning

Given: observation sequence(s)
Find: parameters

HMMs: Notation

For now, consider a single observation sequence

$$O = o_1 o_2 \ldots o_T$$

and an associated (unknown) state sequence

$$Q = q_1 q_2 \ldots q_T.$$

HMMs: Notation

For now, consider a single observation sequence

$$O = o_1 o_2 \ldots o_T$$

and an associated (unknown) state sequence

$$Q = q_1 q_2 \ldots q_T.$$

Denote the transition probabilities:

$$a_{ij} = \text{ P(} \textit{transition} \text{ from node } i \text{ to node } j \text{)}$$

THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL

HMMs: Notation

For now, consider a single observation sequence

$$O = o_1 o_2 \ldots o_T$$

and an associated (unknown) state sequence

$$Q = q_1 q_2 \ldots q_T.$$

Denote the transition probabilities:

$$a_{ij} = \text{ P(} \textit{transition} \text{ from node } i \text{ to node } j \text{)}$$

Similarly, the emission probabilities:

$$b_{jk} = \text{ P(} \textit{emission} \text{ of symbol } k \text{ from node } j \text{)}$$

THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

HMMs: Assumptions

We have two big assumptions to make:

THE UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

HMMs: Assumptions

We have two big assumptions to make:

Markov Assumption

Current state depends *only* on the previous state.

THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL

HMMs: Assumptions

We have two big assumptions to make:

Markov Assumption

Current state depends *only* on the previous state.

Independence Assumption

Current emission depends *only* on the current state.

THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

HMMs: Evaluation

Under the Markov assumption, for a state sequence Q, can calculate the probablity of this sequence given the parameters $(A, B) = \lambda$:

THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

HMMs: Evaluation

Under the Markov assumption, for a state sequence Q, can calculate the probablity of this sequence given the parameters $(A, B) = \lambda$:

$$P(Q|\lambda) = \prod_{t=2}^{T} P(q_t|q_{t-1}) = a_{q_1 q_2} a_{q_2 q_3} \ldots a_{q_{T-1} q_T}$$

THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

HMMs: Evaluation

Under the Markov assumption, for a state sequence Q, can calculate the probablity of this sequence given the parameters $(A, B) = \lambda$:

$$P(Q|\lambda) = \prod_{t=2}^{T} P(q_t|q_{t-1}) = a_{q_1 q_2} a_{q_2 q_3} \ldots a_{q_{T-1} q_T}$$

Under our independence assumption, can find the probability of an observation sequence O given a state sequence Q and the parameters $(A, B) = \lambda$:

THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

HMMs: Evaluation

Under the Markov assumption, for a state sequence Q, can calculate the probablity of this sequence given the parameters $(A, B) = \lambda$:

$$P(Q|\lambda) = \prod_{t=2}^{T} P(q_t|q_{t-1}) = a_{q_1 q_2} a_{q_2 q_3} \ldots a_{q_{T-1} q_T}$$

Under our independence assumption, can find the probability of an observation sequence O given a state sequence Q and the parameters $(A, B) = \lambda$:

$$P(O|Q,\lambda) = \prod_{t=1}^{T} P(o_t|q_t) = b_{q_1 o_1} b_{q_2 o_2} \ldots b_{q_{T-1} o_{T-1}} b_{q_T o_T}$$

HMMs: Evaluation (continued)

Now we can find the probability of an observation sequence given the model:

THE UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

HMMs: Evaluation (continued)

Now we can find the probability of an observation sequence given the model:

$$P(O|\lambda) = \sum_Q P(O|Q,\lambda)P(Q|\lambda) = \sum_Q b_{q_1 o_1} \prod_{t=2}^{T} a_{q_{t-1} q_t} b_{q_t o_t}$$

HMMs: Evaluation (continued)

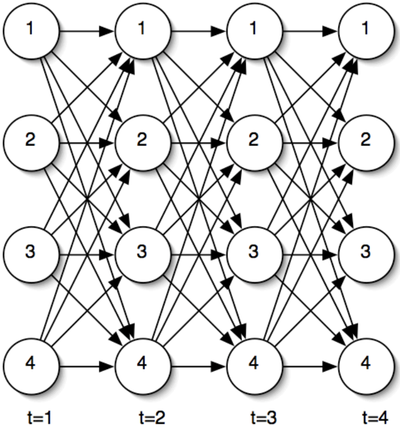
Now we can find the probability of an observation sequence given the model:

$$P(O|\lambda) = \sum_{Q} P(O|Q,\lambda) P(Q|\lambda) = \sum_{Q} b_{q_1 o_1} \prod_{t=2}^{T} a_{q_{t-1} q_t} b_{q_t o_t}$$

Note that this does a lot of redundant calculations.

THE UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

HMMs: Evaluation (continued)



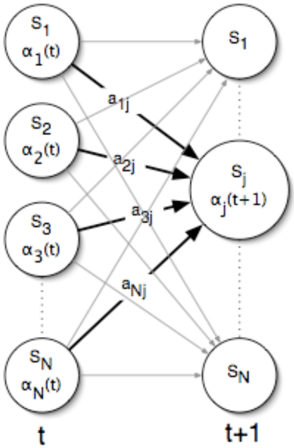
A trellis algorithm.

We can use dynamic programming to cache the redundant calculations by thinking in terms of partial observation sequences:

$$\alpha_j(t) = P(o_1 o_2 ... o_t, q_t = s_j | \lambda)$$

We'll refer to these as the forward probabilities for the observation sequence.

HMMs: the Forward algorithm

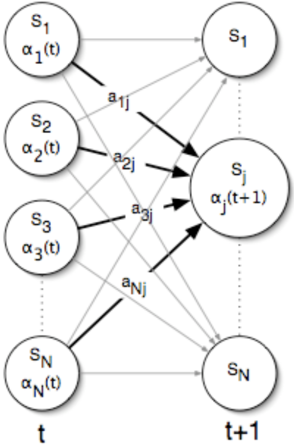


The forward trellis.

For the first time step we have:

$$\alpha_j(1) = P(o_1 | q_1 = s_j) = b_{jo_1}$$

HMMs: the Forward algorithm



The forward trellis.

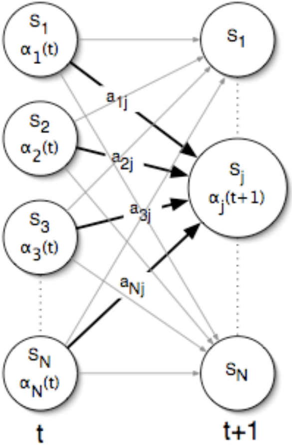
For the first time step we have:

$$\alpha_j(1) = P(o_1 | q_1 = s_j) = b_{jo_1}$$

Then we can calculate the forward probabilities from the trellis:

$$\begin{aligned}\alpha_j(t) &= P(o_1 o_2 \ldots o_t, q_t = s_j | \lambda) \\ &= b_{jo_t} \sum_{i=1}^{N} a_{ij} \alpha_i(t-1)\end{aligned}$$

HMMs: the Forward algorithm (continued)



The forward trellis.

Finally, the probability of the full observation sequence is the sum of the forward probabilities at the last time-step:

$$P(O|\lambda) = \sum_{j=1}^{N} \alpha_j(T)$$

THE UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

HMMs: the Viterbi Algorithm

Kevin Snow will present details of the Viterbi algorithm in the next class.

Sequential Data

Methods

Hidden Markov Models
- Evaluation: The Forward Algorithm
- Decoding: The Viterbi Algorithm
- Learning: The Baum-Welch Algorithm
- Profile HMMs

Kernels for Sequences
- Fixed-Length Subsequence Kernels
- All-Subsequences Kernel
- Variations

THE UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

HMMs: the Baum-Welch algorithm

Given an observation sequence, how do we estimate the parameters of the HMM?

HMMs: the Baum-Welch algorithm

Given an observation sequence, how do we estimate the parameters of the HMM?

- transition probabilities:

$$\bar{a}_{ij} = \frac{\text{expected number of transitions from state } i \text{ to state } j}{\text{expected number of transitions from state } i}$$

HMMs: the Baum-Welch algorithm

Given an observation sequence, how do we estimate the parameters of the HMM?

- transition probabilities:

$$\bar{a}_{ij} = \frac{\text{expected number of transitions from state } i \text{ to state } j}{\text{expected number of transitions from state } i}$$

- emission probabilities:

$$\bar{b}_{jk} = \frac{\text{expected number of emissions of symbol } k \text{ from state } j}{\text{expected number of emissions from state } j}$$

THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL

HMMs: the Baum-Welch algorithm

Given an observation sequence, how do we estimate the parameters of the HMM?

- transition probabilities:

$$\bar{a}_{ij} = \frac{\text{expected number of transitions from state } i \text{ to state } j}{\text{expected number of transitions from state } i}$$

- emission probabilities:

$$\bar{b}_{jk} = \frac{\text{expected number of emissions of symbol } k \text{ from state } j}{\text{expected number of emissions from state } j}$$

This is another Expectation-Maximization algorithm.

THE UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

HMMs: Baum-Welch as Expectation-Maximization

Remember, the sequence of states for the observation sequence is our hidden variable.

HMMs: Baum-Welch as Expectation-Maximization

Remember, the sequence of states for the observation sequence is our hidden variable.

Expectation

Given

- an observation sequence O
- an estimate of the parameters λ

we can find the expectation of the log-likelihood for the observation sequence over the possible state sequences.

HMMs: Baum-Welch as Expectation-Maximization

Remember, the sequence of states for the observation sequence is our hidden variable.

Expectation

Given

- an observation sequence O
- an estimate of the parameters λ

we can find the expectation of the log-likelihood for the observation sequence over the possible state sequences.

Maximization

Then we maximize this expectation over all possible $\hat{\lambda}$.

Baum et al proved that this procedure converges to a local maximum.

THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Profile HMMs

Left-Right HMMs

- special type of HMMs
- links only go in one direction
- no circular routes involving more than one node
- specialized start and end nodes

Profile HMMs

- special type of Left-Right HMMs
- has special *delete* states which don't emit symbols
- consists of sets of *match*, *insert*, and *delete* states

THE UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Profile HMM (continued)

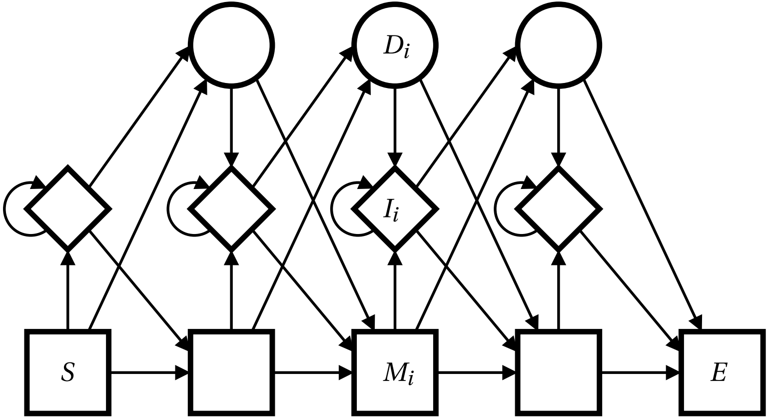


Figure: Topology of a Profile HMM

HMMs in our Example Framework

Recall the framework for classifying phonemes.

HMMs in our Example Framework

Recall the framework for classifying phonemes.

Generative HMM classifier for speech recognition

- for each phoneme, *train* a (profile) HMM using Baum-Welch
- for each test example:
  - *score* using the Forward algorithm for each HMM
  - *classify* according to whichever HMM scores highest

THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Generative vs. Discriminative

HMMs as Generative Models

- can treat an HMM as a generator for a distribution
- build an individual HMM for each class of interest
- can give probability of an example given the model

Generative vs. Discriminative

HMMs as Generative Models

- can treat an HMM as a generator for a distribution
- build an individual HMM for each class of interest
- can give probability of an example given the model

Kernel Methods as Discriminative Models

- model pairs of classes
- find discriminant functions

THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Kernels for Sequences

Fixed-length Subsequence Kernels

Based on counting common subsequences of a fixed length.

- *p*-spectrum kernels
- fixed-length subsequences kernel
- gap-weighted subsequences kernel

All-subsequences Kernel

Based on counting *all* common contiguous or non-contiguous subsequences.

Fixed-Length Subsequence Kernels (continued)

p-Spectrum Kernels

- counts number of common (contiguous) subsequences of length p
- useful where contiguity is important structurally

THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Fixed-Length Subsequence Kernels (continued)

p-Spectrum Kernels

- counts number of common (contiguous) subsequences of length p
- useful where contiguity is important structurally

ϕ	ar	at	ba	ca
bar	1	0	1	0
bat	0	1	1	0
car	1	0	0	1
cat	0	1	0	1

Fixed-Length Subsequence Kernels (continued)

p-Spectrum Kernels

- counts number of common (contiguous) subsequences of length p
- useful where contiguity is important structurally

ϕ	ar	at	ba	ca
bar	1	0	1	0
bat	0	1	1	0
car	1	0	0	1
cat	0	1	0	1

K	bar	bat	car	cat
bar	2	1	1	0
bat	1	2	0	1
car	1	0	2	1
cat	0	1	1	2

Fixed-Length Subsequence Kernels (continued)

Fixed-Length Spectrum Kernels

- counts common (non-contiguous) subsequences of a given length
- allows insertions/deletions with no penalties

Fixed-Length Subsequence Kernels (continued)

Fixed-Length Spectrum Kernels

- counts common (non-contiguous) subsequences of a given length
- allows insertions/deletions with no penalties

ϕ	aa	ar	at	ba	br	bt
baa	1	0	0	2	0	0
bar	0	1	0	1	1	0
bat	0	0	1	1	0	1

Fixed-Length Subsequence Kernels (continued)

Fixed-Length Spectrum Kernels

- counts common (non-contiguous) subsequences of a given length
- allows insertions/deletions with no penalties

ϕ	aa	ar	at	ba	br	bt
baa	1	0	0	2	0	0
bar	0	1	0	1	1	0
bat	0	0	1	1	0	1

K	baa	bar	bat
baa	3	2	2
bar	2	3	1
bat	2	1	3

Fixed-Length Subsequence Kernels (continued)

Gap-Weighted Subsequences Kernel

- interpolates between fixed-length and *p*-spectrum kernels
- allows weighting of the importance of contiguity

ϕ	aa	ar	at	ba	br	bt
baa	λ^2	0	0	$\lambda^2+\lambda^3$	0	0
bar	0	λ^2	0	λ^2	λ^3	0
bat	0	0	λ^2	λ^2	0	λ^3

THE UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

All-Subsequences Kernel

All-subsequences Kernel

- counts number of common subsequences of any length
- contiguous and non-contiguous subsequences considered

Variations of Sequences for Kernels

Character Weights

different *inserted* characters result in different values

THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Variations of Sequences for Kernels

Character Weights

different *inserted* characters result in different values

Soft Matching

two different characters can match with penalty

Variations of Sequences for Kernels

Character Weights

different *inserted* characters result in different values

Soft Matching

two different characters can match with penalty

Gap-number Weighting

weight *number* of gaps instead of gap *lengths*

Variations of Sequences for Kernels

Character Weights

different *inserted* characters result in different values

Soft Matching

two different characters can match with penalty

Gap-number Weighting

weight *number* of gaps instead of gap *lengths*

For details, see *Shawe-Taylor and Christianini, 2004.*

THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Sequence Kernels in our Example Framework

Recall the framework for classifying phonemes.

Sequence Kernels in our Example Framework

Recall the framework for classifying phonemes.

Discriminative SVM classifier for speech recognition

- for each pair of phonemes, *train* a binary classifier using a sequence kernel
- for each test example:
  - each binary classifier *votes* for a label
  - *classify* according to whichever label receives the most votes

THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Sequential Data

- can be treated differently from feature-vector data
- provides structurally important information

Sequential Data

- can be treated differently from feature-vector data
- provides structurally important information

Hidden Markov Models

- generative models of sequences
- assume:
  - state t depends only on state $t-1$
  - emission t depends only on state t

Sequential Data

- can be treated differently from feature-vector data
- provides structurally important information

Hidden Markov Models

- generative models of sequences
- assume:
  - state t depends only on state $t-1$
  - emission t depends only on state t

Kernels for Sequences

- discriminative approach
- many different kernels

THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

References

Phil Blunsom.
Hidden markov models, 2004.

Lawrence R. Rabiner.
A tutorial on hidden markov models and selected applications in speech recognition.
In *Proceedings of the IEEE*, pages 257–286, 1989.

John Shawe-Taylor and Nello Cristianini.
Kernel Methods for Pattern Analysis.
Cambridge University Press, Cambridge, United Kingdom, 2004.