Introduction to Kernel Methods for Sequences

Yoon Sang Cho

DMQA Open Seminar

2021-01-15

Introduction

- 발표자 소개

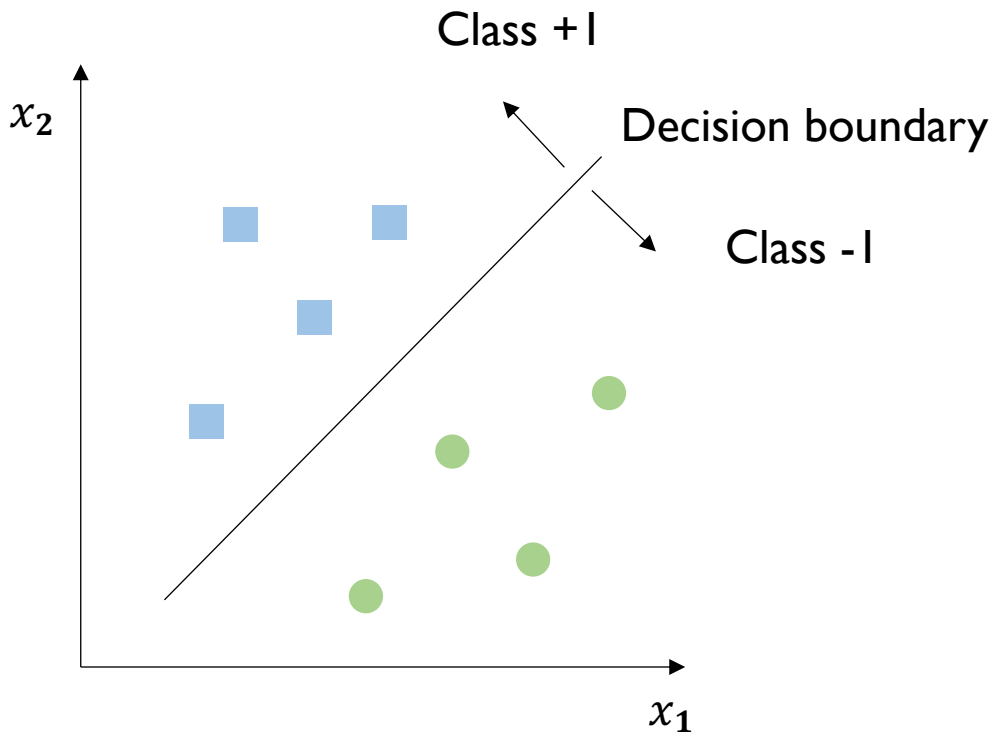
- **조윤상 (Yoon Sang Cho)**
  - 고려대학교 산업경영공학과 박사과정 (2017~)
    - Data Mining & Quality Analytics (DMQA) Lab.
- **연구 분야**
  - Explainable AI for multisensor signals
  - Predictive analysis for sequences

Contents

- Introduction
- Kernel Methods
- Sequence Data
- Kernels for Sequences
- Conclusions

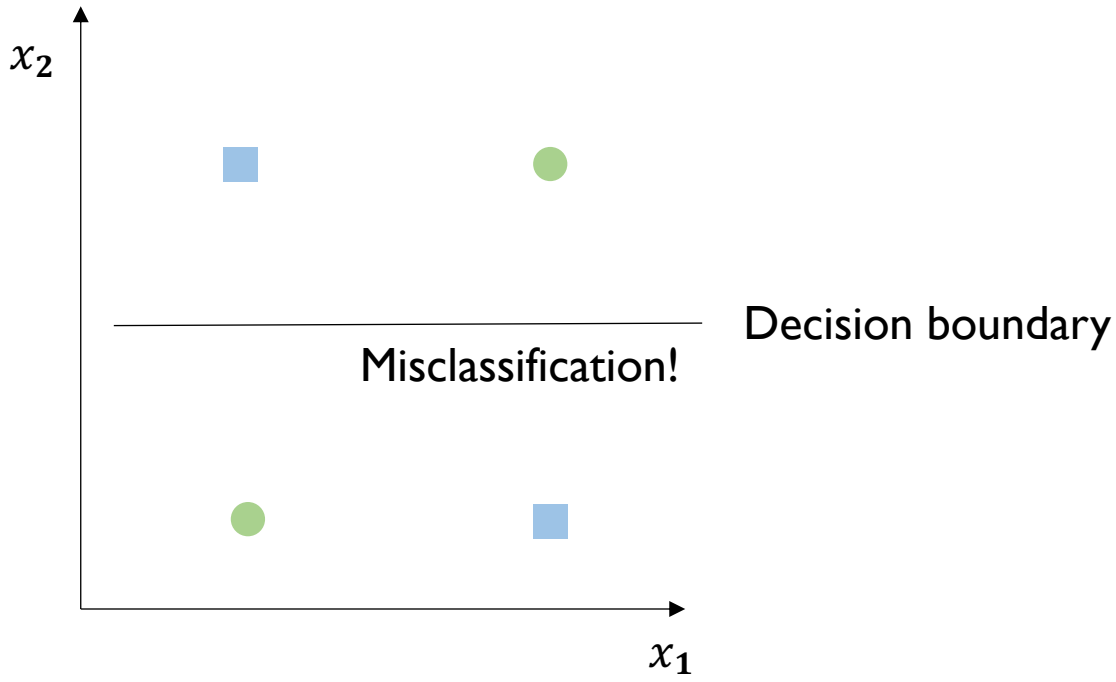
Introduction

- Machine learning algorithms
  - linear classifiers



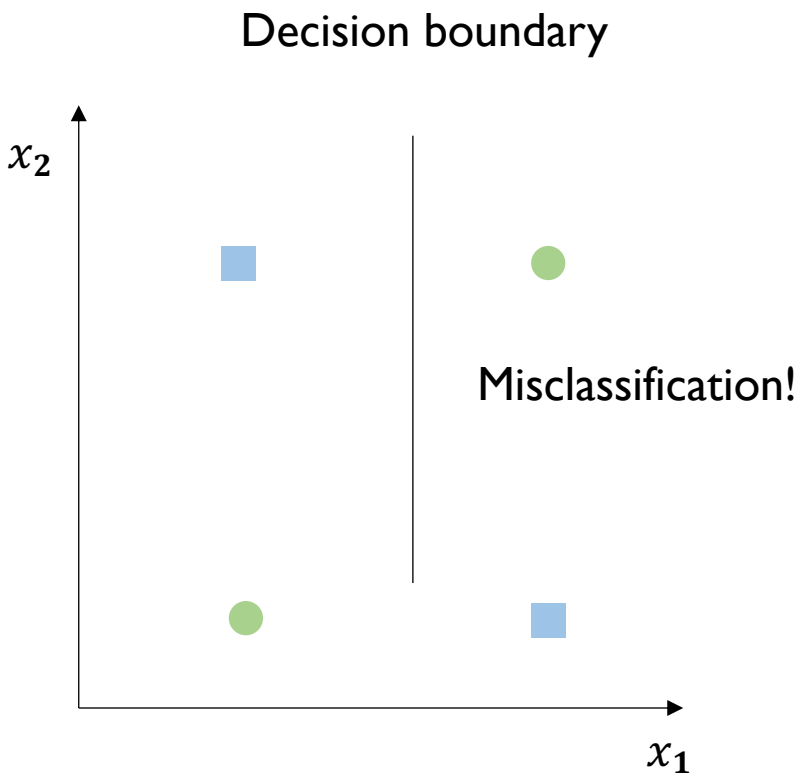
Introduction

- Machine learning algorithms
  - linear classifiers



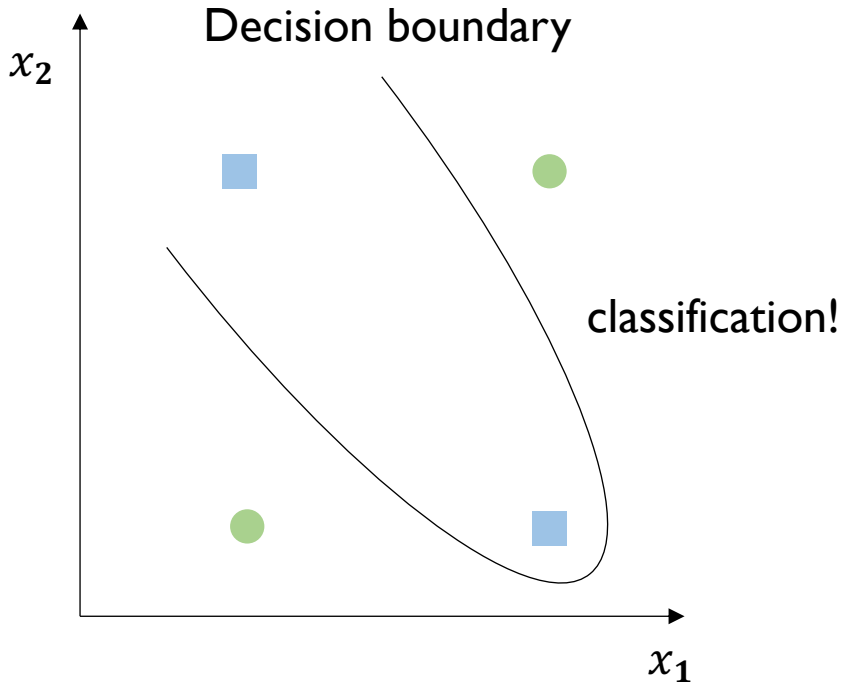
Introduction

- Machine learning algorithms
  - linear classifiers



Introduction

- Machine learning algorithms
  - linear classifiers

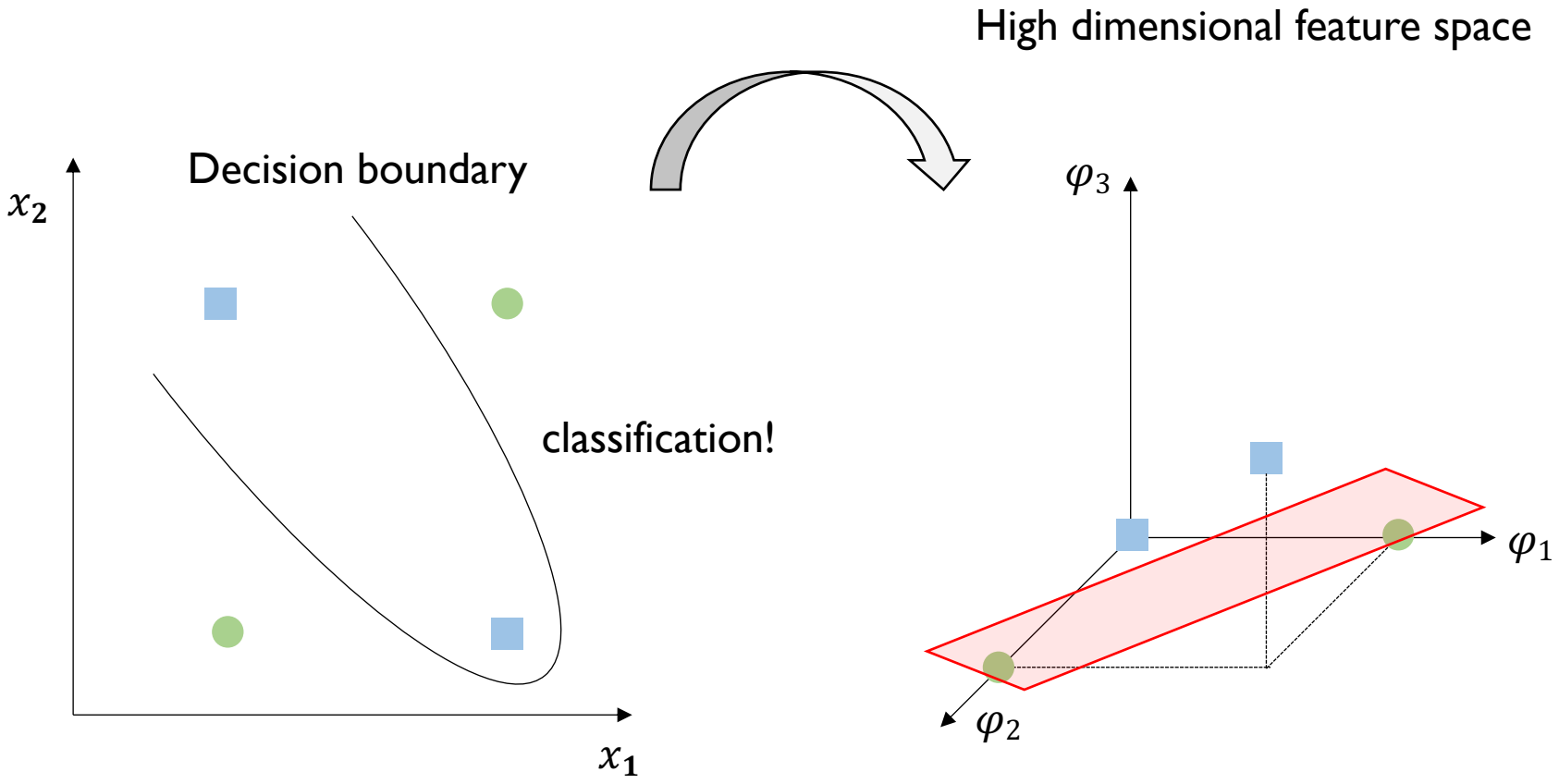


→ The kernel method enables a linear classifier to classify such data!

Introduction to kernel methods for sequence data

Introduction to kernel methods for sequence data

Kernel Methods



A mapping function $\emptyset(\mathbf{x})$

Kernel Methods

- A mapping function $\emptyset$:
  - $\mathcal{L}$(raw data space) $\mapsto$ $\mathcal{H}$(hilbert space: high−dimensional feature space)

$$\emptyset(\mathbf{x}) = \left(x_1{}^2, x_2{}^2, \sqrt{2}x_1x_2\right)$$

$\mathcal{L}$ $\mathcal{H}$

$$\mathbf{x} = \begin{pmatrix}\mathbf{0}\\ \mathbf{0}\end{pmatrix}, \mathbf{z} = \begin{pmatrix}\mathbf{1}\\ \mathbf{0}\end{pmatrix} \qquad \phi(\mathbf{x}) = \begin{pmatrix}\mathbf{0}\\ \mathbf{0}\\ \mathbf{0}\end{pmatrix}, \phi(\boldsymbol{z}) = \begin{pmatrix}\mathbf{1}\\ \mathbf{0}\\ \mathbf{0}\end{pmatrix}$$

Support vector machine (SVM) with a mapping fuction

Kernel Methods

- SVM's objective function: Maximize $\tilde{\mathcal{L}}(\boldsymbol{\alpha})$
  - s.t. $\sum_{i=1}^{N} \alpha_i t_i = 0$ and $0 \leq \alpha_i \leq C$
  - t: class $+1$ or -1, α_i: Lagrange multipliers, C: hyperparameters

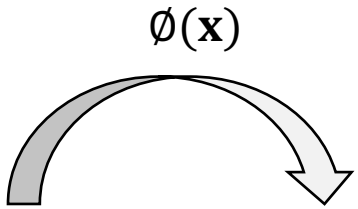
$$\text{Maximize } \tilde{\mathcal{L}}(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - 1/2 \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j t_i t_j \mathbf{x}_i{}^{\mathbf{T}} \mathbf{x}_j$$

Mapping input data to high dimensional feature space

$$\text{Maximize } \tilde{\mathcal{L}}(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - 1/2 \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j t_i t_j \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$$

Kernel Methods

- $\emptyset(\mathbf{x})$: A mapping function
- $P \leq K$



$\mathcal{L}$ $\mathcal{H}$

	X_1	X_2	⋯	X_{P-1}	X_P
O_1			⋯		
O_2			⋯		
⋯	⋯	⋯	⋯	⋯	⋯
O_{N-1}			⋯		
O_N			⋯		

	φ_1	⋯	φ_P	⋯	φ_K
O_1		⋯		⋯	
O_2		⋯		⋯	
⋯	⋯	⋯	⋯	⋯	⋯
O_{N-1}		⋯		⋯	
O_N		⋯		⋯	

Mapping all points is very difficult or impossible. → Kernel function!

Kernel Methods

- Kernel function

$$\emptyset(\mathbf{x}) = \left(x_1{}^2, x_2{}^2, \sqrt{2}x_1x_2\right)$$

$\mathcal{L}$ → $\mathcal{H}$

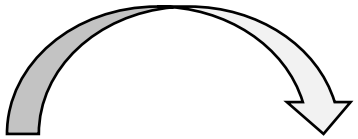
$$\mathbf{x} = \begin{pmatrix}\mathbf{0}\\\mathbf{0}\end{pmatrix}, \mathbf{z} = \begin{pmatrix}\mathbf{1}\\\mathbf{0}\end{pmatrix} \qquad \phi(\mathbf{x}) = \begin{pmatrix}\mathbf{0}\\\mathbf{0}\\\mathbf{0}\end{pmatrix}, \phi(\boldsymbol{z}) = \begin{pmatrix}\mathbf{1}\\\mathbf{0}\\\mathbf{0}\end{pmatrix}$$

$$\begin{aligned} K(\mathbf{x}\cdot\mathbf{z}) &= (\mathbf{x}\cdot\mathbf{z})^{\mathbf{2}} \\ &= \left(x_1{}^2, x_2{}^2, \sqrt{2}x_1x_2\right)\cdot\left(z_1{}^2, z_2{}^2, \sqrt{2}z_1z_2\right) \\ &= x_1{}^2z_1{}^2 + 2x_1z_1x_2z_2 + x_2{}^2z_2{}^2 \\ &= \phi(\mathbf{x})\cdot\phi(\boldsymbol{z}) \end{aligned}$$

Kernel Methods

- Kernel function

$$\mathcal{L} \qquad \emptyset(\mathbf{x}) = \left(x_1{}^2, x_2{}^2, \sqrt{2}x_1x_2\right) \qquad \mathcal{H}$$



$$\mathbf{x} = \begin{pmatrix}\mathbf{0}\\\mathbf{0}\end{pmatrix}, \mathbf{z} = \begin{pmatrix}\mathbf{1}\\\mathbf{0}\end{pmatrix} \qquad \phi(\mathbf{x}) = \begin{pmatrix}\mathbf{0}\\\mathbf{0}\\\mathbf{0}\end{pmatrix}, \phi(\boldsymbol{z}) = \begin{pmatrix}\mathbf{1}\\\mathbf{0}\\\mathbf{0}\end{pmatrix}$$

$$K(\mathbf{x}\cdot\mathbf{z}) = (\mathbf{x}\cdot\mathbf{z})^{\mathbf{2}} = 0 \qquad \phi(\mathbf{x})\cdot\phi(\boldsymbol{z}) = 0$$

→ Kernel function: Mapping function의 내적으로 표현 가능한 함수(Mercer's theorem)

Kernel Methods

- SVM's objective function: Maximize $\tilde{\mathcal{L}}(\boldsymbol{\alpha})$
  - s.t. $\sum_{i=1}^{N} \alpha_i t_i = 0$ and $0 \leq \alpha_i \leq C$
  - t: class $+1$ or -1, α_i: Lagrange multipliers, C: hyperparameters

$$\text{Maximize } \tilde{\mathcal{L}}(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - 1/2 \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j t_i t_j \mathbf{x}_i{}^{\mathrm{T}} \mathbf{x}_j$$

Mapping input data to high dimensional feature space

$$\text{Maximize } \tilde{\mathcal{L}}(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - 1/2 \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j t_i t_j \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$$

Kernel function (Mercer's theorem)

$$\text{Maximize } \tilde{\mathcal{L}}(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - 1/2 \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j t_i t_j K(\mathbf{x}_i, \mathbf{x}_j)$$

"어떤 수식이 벡터의 내적을 포함할 때, 커널 함수로 대치하여 계산 ➔ Kernel substitution (or kernel trick)"

Kernel Methods

- Kernel trick을 적용할 수 있는 Algorithms
  - Kernel-SVM, Kernel-PCA, Kernel-FDA
- Mercer's theorem을 만족하는 사실이 증명된 kernel functions
  - Polynomial kernel $K(\mathbf{x} \cdot \mathbf{z})$: $(\mathbf{x} \cdot \mathbf{z} + 1)^p$
  - RBF kernel $K(\mathbf{x} \cdot \mathbf{z})$: $\exp\left(\frac{\|\mathbf{x}-\mathbf{z}\|_2^2}{2\sigma^2}\right)$
  - Hyperbolic tangent kernel: $K(\mathbf{x} \cdot \mathbf{z})$: $\tanh(\alpha\mathbf{x} \cdot \mathbf{z} + \beta)$

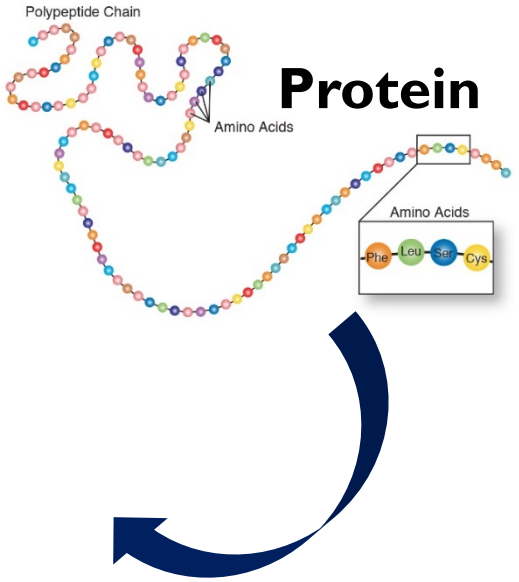
Introduction to kernel methods for sequence data

Introduction to kernel methods for sequence data

Introduction to kernel methods for sequence data

: a series of discrete things

Sequence Data



- **The sequence data: a series of discrete things**
  - Biological data : DNA (protein) sequences
  - 20개 symbol (amino acids)에 대한 sequence data

Symbols

Amino Acids

Ala: Alanine	Gln: Glutamine
Arg: Arginine	Glu: Glutamic acid
Asn: Asparagine	Gly: Glycine
Asp:Aspartic acid	His: Histidine
Cys:Cysteine	Ile: Isoleucine
Leu: Leucine	Ser: Serine
Lys: Lysine	Thr: Threonine
Met: Methionine	Trp: Tryptophane
Phe: Phenylalanine	Tyr: Tyrosisne
Pro: Proline	Val: Valine

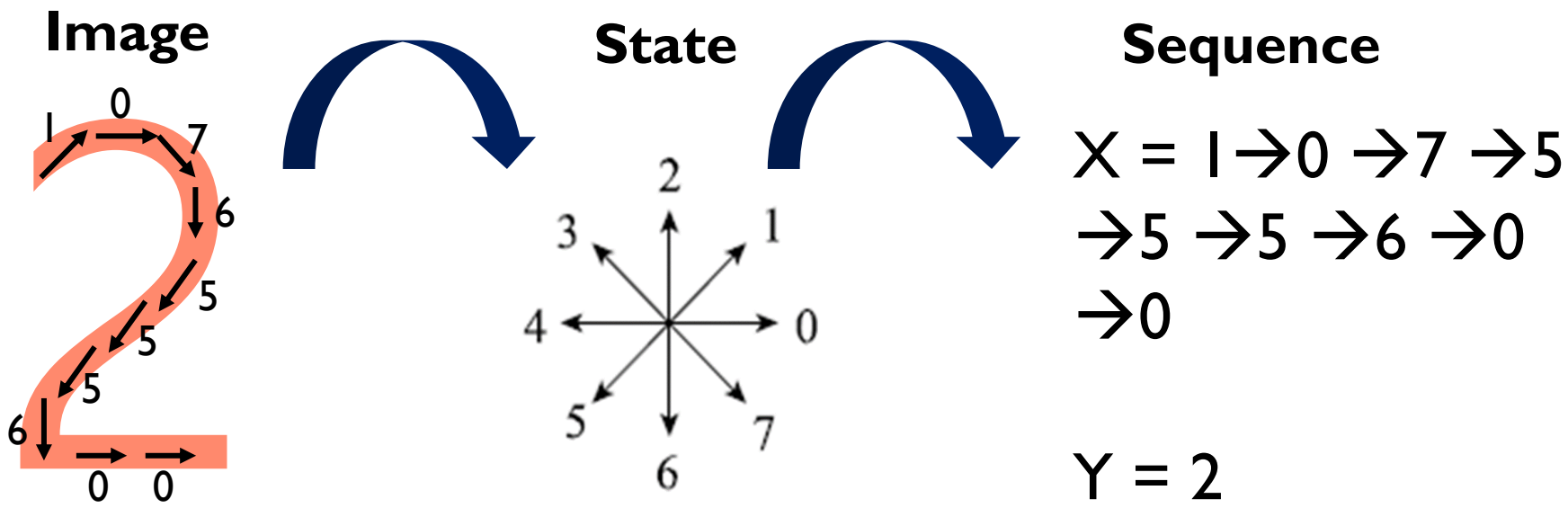
Sequence

	X	Y
1	SEIKLLHAQVNPHFLFNALNTLSAVI...	Family A
2	KRTFDLIGSLLLLTLLSPLLLTLSLA...	Family B
3	VSLFITFFEIGLFGFGGGYGMLSLIQ...	Family C
…	…	…
19078	ILDYGCGSGEITLDLATIVGPDGEV...	Family A
19079	AELRAVHYQINPHLLFNTLNSIQW...	Family B
19080	ILELGSGGGRDAVELARSRVGIDFV...	Family C

https://www.genome.gov/genetics-glossary/Amino-Acids

Sequence Data

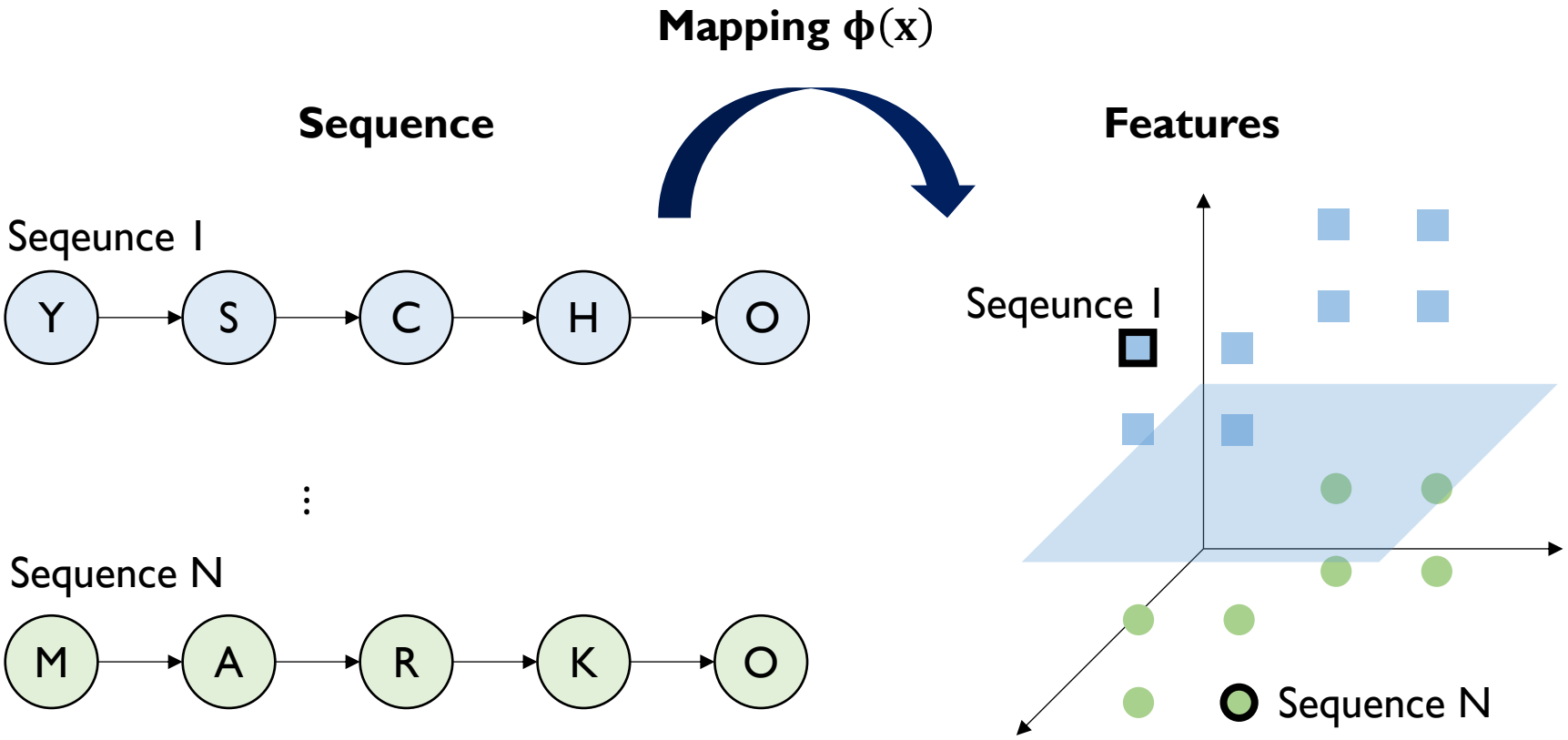
- **The sequence data: a series of discrete things**
  - Image data: MNIST Sequence
  - 8개 state를 갖는 sequence data



https://edwin-de-jong.github.io/blog/mnist-sequence-data/

Kernels for Sequences

- Objective of kernels for sequences



Kernels for Sequences

- **Sequence data 를 위한 kernel function을 설계하기 위해?**

① Character (categorical variable) 순차 정보를 feature space에 표현하고,

$\phi(\mathbf{x}_1)$ → Mapping function $\mathcal{L}$  $\mathcal{H}$

② Mapping function의 내적으로 표현하여 kernel trick을 적용할 수 있어야 함.

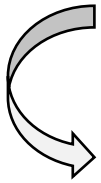
$\phi(\mathbf{x}_1) \cdot \phi(\mathbf{x}_2)$→ Kernel function

(1) Spectrum kernel

Kernels for Sequences

- Mapping function for Sequence data
  - $|\Sigma|=\{A, B, C\}$, k길이를 가질 수 있는 문자열의 빈도수 표현해보자

Sequence 1: AAABBBCCCAAACCCBBB



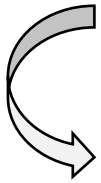
If $k=3$, all possible sets:

AAA AAB AAC ⋯ CCA CCB CCC

Kernels for Sequences

- Mapping function for Sequence data
  - $|\Sigma|=\{A, B, C\}$, k길이를 가질 수 있는 문자열의 빈도수 표현해보자

Sequence 1: AAABBBCCCAAACCCBBB



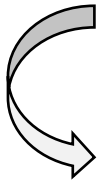
If $k=3$, all possible sets:

AAA AAB AAC ⋯ CCA CCB CCC
1

Kernels for Sequences

- Mapping function for Sequence data
  - $|\Sigma|=\{A, B, C\}$, k길이를 가질 수 있는 문자열의 빈도수 표현해보자

Sequence 1: AAABBBCCCAAACCCBBB



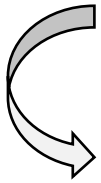
If $k=3$, all possible sets:

AAA	AAB	AAC	…	CCA	CCB	CCC
1	1					

Kernels for Sequences

- Mapping function for Sequence data
  - $|\Sigma|=\{A, B, C\}$, k길이를 가질 수 있는 문자열의 빈도수 표현해보자

Sequence 1: AAABBBCCCAAACCCBBB



If $k=3$, all possible sets:

AAA	AAB	AAC	⋯	CCA	CCB	CCC
2	1	1		1	1	2

k-mer set (u), mapping function $\phi_u(\text{seq1})$

→ 길이에 관계 없이 fixed-length 차원으로 표현

Kernels for Sequences

- Mapping function for Sequence data
  - Approach 1: 순차적으로 k개 문자열을 표현해보자

Sequence 1: AAABBBCCCAAACCCBBB

$\phi_u(\mathbf{x}), u \in \mathcal{A}^{k=3}$

AAA	AAB	AAC	…	CCA	CCB	CCC
2	1	1		1	1	2

Sequence 2: BBBAAAAAACCCCCCBBB

$\phi_u(\mathbf{x}), u \in \mathcal{A}^{k=3}$

AAA	AAB	AAC	…	CCA	CCB	CCC
4	0	1		0	1	4

$$K(\text{seq1},\text{seq2}) = \sum_{u \in \mathcal{A}^k} \phi_u(\text{seq1})\phi_u(\text{seq2})$$

Kernels for Sequences

- Mapping function for Sequence data
  - Approach 1: 순차적으로 k개 문자열을 표현해보자

$$K(\text{seq1},\text{seq2}) = \sum_{u \in \mathcal{A}^k} \phi_u(\text{seq1})\phi_u(\text{seq2})$$

	AAA	AAB	AAC	…	CCA	CCB	CCC
Sequence 1	2	1	1		1	1	2
Sequence 2	4	0	1		0	1	4

$$\sum_{u \in \mathcal{A}^k} \phi_u(\text{seq1})\phi_u(\text{seq2}) = 8 + 0 + 1 + 1 + 8 = K(\text{seq1},\text{seq2})$$

The spectrum kernel (Leslie et al., 2002)

동일한 k-mer를 많이 공유할 경우 큰 커널 값을 갖는 sequence 유사성 지표

(2) Mismatch kernel

Kernels for Sequences

- The mismatch kernel (Leslie et al., 2004)
  - Spectrum kernel with mismatch

Spectrum kernel (2-mer)

Sequence **x**: ABBA

$|\Sigma|=\{A, B, C\}$

k-mer: k=2

u	count
AA	0
AB	1
AC	0
BA	1
BB	1
BC	0
CA	0
CB	0
CC	0

Kernels for Sequences

- The mismatch kernel (Leslie et al., 2004)
  - Spectrum kernel with mismatch

Sequence **x**: ABBA

$|\Sigma|=\{A, B, C\}$

k-mer: k=2

α

β	AA	AB	AC	BA	BB	BC	CA	CB	CC
AA									
AB									
AC									
BA									
BB									
BC									
CA									
CB									
CC									

Kernels for Sequences

- The mismatch kernel (Leslie et al., 2004)
  - Spectrum kernel with mismatch: M 길이만큼 일치하지 않는 경우도 고려하자

Sequence **x**: ABBA

|Σ|={A, B, C}

k-mer: k=2

Mismatch (m)=1

α

β	AA	AB	AC	BA	BB	BC	CA	CB	CC
AA									
AB									
AC									
BA									
BB									
BC									
CA									
CB									
CC									

Kernels for Sequences

- The mismatch kernel (Leslie et al., 2004)
  - Spectrum kernel with mismatch: M 길이만큼 일치하지 않는 경우도 고려하자

Sequence **x**: ABBA

$|\Sigma|=\{A, B, C\}$

k-mer: k=2

Mismatch (m)=1

β 가 α 에 m길이만큼 속하면 1
그렇지 않으면 0

α

β	AA	AB	AC	BA	BB	BC	CA	CB	CC
AA									
AB									
AC									
BA									
BB									
BC									
CA									
CB									
CC									

Kernels for Sequences

- The mismatch kernel (Leslie et al., 2004)
  - Spectrum kernel with mismatch: M 길이만큼 일치하지 않는 경우도 고려하자

Sequence **x**: ABBA

|Σ|={A, B, C}

k-mer: k=2

Mismatch (m)=1

β 가 α 에 m길이만큼 속하면 1
그렇지 않으면 0

α

β	AA	AB	AC	BA	BB	BC	CA	CB	CC
AA	1								
AB									
AC									
BA									
BB									
BC									
CA									
CB									
CC									

Kernels for Sequences

- The mismatch kernel (Leslie et al., 2004)
  - Spectrum kernel with mismatch: M 길이만큼 일치하지 않는 경우도 고려하자

Sequence **x**: ABBA

|Σ|={A, B, C}

k-mer: k=2

Mismatch (m)=1

α

β

	AA	AB	AC	BA	BB	BC	CA	CB	CC
AA	1	1							
AB									
AC									
BA									
BB									
BC									
CA									
CB									
CC									

β 가 α 에 m길이만큼 속하면 1
그렇지 않으면 0

Kernels for Sequences

- The mismatch kernel (Leslie et al., 2004)
  - Spectrum kernel with mismatch: M 길이만큼 일치하지 않는 경우도 고려하자

Sequence **x**: ABBA

$|\Sigma|=\{A, B, C\}$

k-mer: k=2

Mismatch (m)=1

β 가 α 에 m길이만큼 속하면 1
그렇지 않으면 0

α

β	AA	AB	AC	BA	BB	BC	CA	CB	CC
AA	1	1	1						
AB									
AC									
BA									
BB									
BC									
CA									
CB									
CC									

Kernels for Sequences

- The mismatch kernel (Leslie et al., 2004)
  - Spectrum kernel with mismatch: M 길이만큼 일치하지 않는 경우도 고려하자

Sequence **x**: ABBA

$|\Sigma|$={A, B, C}

k-mer: k=2

Mismatch (m)=1

β 가 α 에 m길이만큼 속하면 1
그렇지 않으면 0

α

β	AA	AB	AC	BA	BB	BC	CA	CB	CC
AA	1	1	1	1	0	0	1	0	0
AB									
AC									
BA									
BB									
BC									
CA									
CB									
CC									

Kernels for Sequences

- The mismatch kernel (Leslie et al., 2004)
  - Spectrum kernel with mismatch: M 길이만큼 일치하지 않는 경우도 고려하자

Sequence **x**: ABBA

$|\Sigma|$={A, B, C}

k-mer: k=2

Mismatch (m)=1

α

β	AA	AB	AC	BA	BB	BC	CA	CB	CC
AA	1	1	1	1	0	0	1	0	0
AB	1	1	1	0	1	0	0	1	0
AC	1	1	1	0	0	1	0	0	1
BA	1	0	0	1	1	1	1	0	0
BB	0	1	0	1	1	1	0	1	0
BC	0	0	1	1	1	1	0	0	1
CA	1	0	0	1	0	0	1	1	1
CB	0	1	0	0	1	0	1	1	1
CC	0	0	1	0	0	1	1	1	1

β 가 α 에 m길이만큼 속하면 1
그렇지 않으면 0

Kernels for Sequences

- The mismatch kernel (Leslie et al., 2004)
  - Spectrum kernel with mismatch: M 길이만큼 일치하지 않는 경우도 고려하자

Sequence **x**: ABBA

|Σ|={A, B, C}

k-mer: k=2

Mismatch (m)=1

β 가 α 에 m길이만큼 속하면 1
그렇지 않으면 0

α

β	AA	AB	AC	BA	BB	BC	CA	CB	CC
AA	1	1	1	1	0	0	1	0	0
AB	1	1	1	0	1	0	0	1	0
AC	1	1	1	0	0	1	0	0	1
BA	1	0	0	1	1	1	1	0	0
BB	0	1	0	1	1	1	0	1	0
BC	0	0	1	1	1	1	0	0	1
CA	1	0	0	1	0	0	1	1	1
CB	0	1	0	0	1	0	1	1	1
CC	0	0	1	0	0	1	1	1	1

Kernels for Sequences

- The mismatch kernel (Leslie et al., 2004)
  - Spectrum kernel with mismatch: M 길이만큼 일치하지 않는 경우도 고려하자
  - Sequence **x**: ABBA, |Σ|={A, B, C}, k-mer: k=2, Mismatch (m)=1

α

β	AA	AB	AC	BA	BB	BC	CA	CB	CC
AA	1	1	1	1	0	0	1	0	0
AB	1	1	1	0	1	0	0	1	0
AC	1	1	1	0	0	1	0	0	1
BA	1	0	0	1	1	1	1	0	0
BB	0	1	0	1	1	1	0	1	0
BC	0	0	1	1	1	1	0	0	1
CA	1	0	0	1	0	0	1	1	1
CB	0	1	0	0	1	0	1	1	1
CC	0	0	1	0	0	1	1	1	1

sum →

$\phi_{k:2,m:1}(\mathbf{x})$
2
2
1
2
3
2
1
2
0

Kernels for Sequences

- The mismatch kernel (Leslie et al., 2004)
  - Sequence **x**: ABBA, |Σ|={A, B, C}, k-mer: k=2, Mismatch (m)=1

Spectrum kernel : $\phi_k(\mathbf{x})$

$k = 2$	$\phi(\mathbf{x})$
AA	0
AB	1
AC	0
BA	1
BB	1
BC	0
CA	0
CB	0
CC	0

Spectrum kernel with mismatch: $\phi_{k,m}(\mathbf{x})$

$k = 2, m = 1$	$\phi(\mathbf{x})$
AA	2
AB	2
AC	1
BA	2
BB	3
BC	2
CA	1
CB	2
CC	0

Kernels for Sequences

- The mismatch kernel (Leslie et al., 2004)
  - Sequence **x**: ABBA, |Σ|={A, B, C}, k-mer: k=2, Mismatch (m)=1

Spectrum kernel:

$$K_k\,(\text{seq1},\text{seq2}) = \sum_{u \in \mathcal{A}^k} \phi_k(\text{seq1})\phi_k(\text{seq2})$$

Spectrum kernel with mismatch:

$$K_{k,m}(\text{seq1},\text{seq2}) = \sum_{a,\beta\ \in \mathcal{A}^k} \phi_{k,m}(\text{seq1})\phi_{k,m}(\text{seq2})$$

저자-"biologically important notion"을 반영하기 위함.

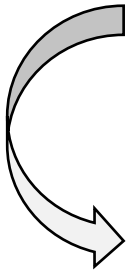
분류 모델 정확도 높고, 두 sequence 간 특정 시점에 문자열이 일치하지 않더라도 범위를 넓혀 유사성 측정

(3) Substring kernel

Kernels for Sequences

- The substring kernel (Lodhi et al., 2004)

Sequence (string) I = ABRACADABRA



특정 자리에 있는 문자열

Subsequence (substring) I = RADAR

Kernels for Sequences

- The substring kernel (Lodhi et al., 2004)

Sequence (string) = ABRACADABRA

Number of Indices k=5

Kernels for Sequences

- The substring kernel (Lodhi et al., 2004)

i_1 i_k

1 2 3 … 10 11

Sequence (string) = ABRACADABRA

Number of Indices k=5

Indices **i** = (3, 4, 7, 8, 10)

Kernels for Sequences

- The substring kernel (Lodhi et al., 2004)

i_1 i_k

1 2 3 … 10 11

Sequence (string) = ABRACADABRA

Number of Indices k=5

Indices $\mathbf{i}$ = (3, 4, 7, 8, 10)

Subsequence (substring) $\mathbf{x}(\mathbf{i})$=RADAR

Kernels for Sequences

- The substring kernel (Lodhi et al., 2004)

i_1 i_k

1 2 3 ⋯ 10 11

Sequence (string) = ABRACADABRA

Length of substring 8

Number of Indices k=5

Indices $\mathbf{i}$ = (3, 4, 7, 8, 10)

Subsequence (substring) $\mathbf{x}(\mathbf{i})$=RADAR

Length of subsequence $l(\mathbf{i})$ = 10 − 3 + 1 = 8 (gap 반영)

Kernels for Sequences

- The substring kernel (Lodhi et al., 2004)

i_1 i_k

1 2 3 ⋯ 10 11

Sequence (string) = ABRACADABRA

Length of substring 8

Number of Indices k=5

Indices $\mathbf{i}$ = (3, 4, 7, 8, 10)

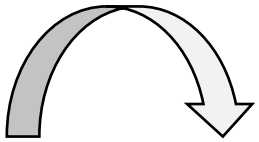
Subsequence (substring) $\mathbf{x}(\mathbf{i})$=RADAR

Length of subsequence $l(\mathbf{i})$ = 10 − 3 + 1 = 8 (gap 반영)

k가 정해졌을 때 substring set(RADAR) → $\mathbf{u}$로 표기

Kernels for Sequences

- The substring kernel (Lodhi et al., 2004)

k=2  $\boldsymbol{u}$

Sequence 1 = CAT	C-A	C-T	A-T
Sequence 2 = CAR	C-A	C-R	A-R
Sequence 3 = BAT	B-A	B-T	A-T
Sequence 4 = BAR	B-A	B-R	A-R

$$\phi_{\boldsymbol{u}}(\mathbf{x}): \sum_{i \in I(k,n):\ \mathbf{x}(\mathbf{i})=\mathbf{u}} \lambda^{l(\mathbf{i})}$$

Kernels for Sequences

- The substring kernel (Lodhi et al., 2004)

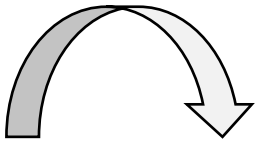
$$\phi_{\boldsymbol{u}}(\mathbf{x}): \sum_{i \in I(k,n):\ \mathbf{x}(\mathbf{i})=\mathbf{u}} \lambda^{l(\mathbf{i})}$$

- Hyperparameter λ: 0~1
- $\lambda^{l(\mathbf{i})}$: substring (gap을 반영한) 길이가 증가할 수록 감소하는 (weight decay) 변수
- 즉, 같은 substring 인데 gap이 크면 유사하다고 보기 어렵다는 점을 반영

Kernels for Sequences

- The substring kernel (Lodhi et al., 2004)

$\boldsymbol{u}$ $\phi_{\boldsymbol{u}}(\mathbf{x})$ $\sum_{i \in I(k,n):\ \mathbf{x}(\mathbf{i})=\mathbf{u}} \lambda^{l(\mathbf{i})}$



C-A C-T A-T $\phi_{\boldsymbol{u}}(\mathbf{CAT}): \lambda^2$

C-A C-R A-R

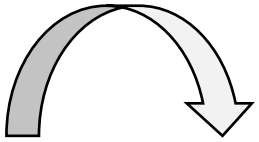
B-A B-T A-T

B-A B-R A-R

Kernels for Sequences

- The substring kernel (Lodhi et al., 2004)

$$\boldsymbol{u} \qquad \phi_{\boldsymbol{u}}(\mathbf{x}) \qquad \sum_{i \in I(k,n):\ \mathbf{x}(\mathbf{i})=\mathbf{u}} \lambda^{l(\mathbf{i})}$$



C-A C-T A-T $\qquad \phi_{\boldsymbol{u}}(\mathbf{CAT}): \lambda^2$

C-A C-R A-R

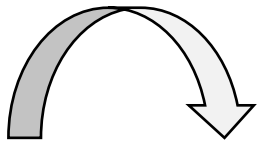
B-A B-T A-T

B-A B-R A-R

Kernels for Sequences

- The substring kernel (Lodhi et al., 2004)

$\boldsymbol{u}$ $\phi_u(\mathbf{x})$ $\sum_{i \in I(k,n):\ \mathbf{x}(\mathbf{i})=\mathbf{u}} \lambda^{l(\mathbf{i})}$



C-A C-T A-T $\phi_u(\mathbf{CAT}): \lambda^2, \lambda^3$

C-A C-R A-R

B-A B-T A-T

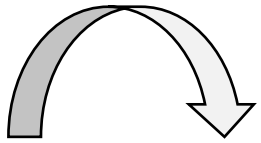
B-A B-R A-R

Kernels for Sequences

- The substring kernel (Lodhi et al., 2004)

$$\phi_u(\mathbf{x}) \qquad \sum_{i \in I(k,n):\ \mathbf{x}(\mathbf{i})=\mathbf{u}} \lambda^{l(\mathbf{i})}$$

$\boldsymbol{u}$



C-A C-T A-T $\quad \phi_u(\mathbf{CAT}): \lambda^2, \lambda^3, \lambda^2$

C-A C-R A-R

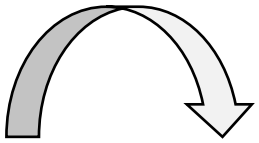
B-A B-T A-T

B-A B-R A-R

Kernels for Sequences

- The substring kernel (Lodhi et al., 2004)

$$\boldsymbol{u} \qquad \phi_{\boldsymbol{u}}(\mathbf{x}) \qquad \sum_{i \in I(k,n):\ \mathbf{x}(\mathbf{i})=\mathbf{u}} \lambda^{l(\mathbf{i})}$$



C-A C-T A-T $\quad \phi_{\boldsymbol{u}}(\mathbf{CAT}): \lambda^2, \lambda^3, \lambda^2$

C-A C-R A-R

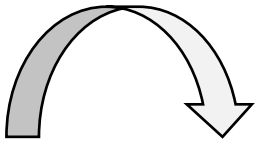
B-A B-T A-T

B-A B-R A-R

Kernels for Sequences

- The substring kernel (Lodhi et al., 2004)

$$\boldsymbol{u} \qquad \phi_{\boldsymbol{u}}(\mathbf{x}) \qquad \sum_{i \in I(k,n):\ \mathbf{x}(\mathbf{i})=\mathbf{u}} \lambda^{l(\mathbf{i})}$$



C-A C-T A-T $\quad \phi_{\boldsymbol{u}}(\mathbf{CAT}): \lambda^2, \lambda^3, \lambda^2$

C-A C-R A-R $\quad \phi_{\boldsymbol{u}}(\mathbf{CAR}): \lambda^2, \lambda^3, \lambda^2$

B-A B-T A-T $\quad \phi_{\boldsymbol{u}}(\mathbf{BAT}): \lambda^2, \lambda^2, \lambda^3$

B-A B-R A-R $\quad \phi_{\boldsymbol{u}}(\mathbf{BAR}): \lambda^2, \lambda^2, \lambda^3$

Kernels for Sequences

- The substring kernel (Lodhi et al., 2004)

$$\phi_{\boldsymbol{u}}(\mathbf{x}) = \sum_{i \in I(k,n):\ \mathbf{x}(\mathbf{i})=\mathbf{u}} \lambda^{l(\mathbf{i})}$$

$\boldsymbol{u}$

C-A C-T A-T $\quad \phi_{\boldsymbol{u}}(\mathbf{CAT}): \lambda^2, \lambda^3, \lambda^2$

C-A C-R A-R $\quad \phi_{\boldsymbol{u}}(\mathbf{CAR}): \lambda^2, \lambda^3, \lambda^2$

B-A B-T A-T $\quad \phi_{\boldsymbol{u}}(\mathbf{BAT}): \lambda^2, \lambda^2, \lambda^3$

B-A B-R A-R $\quad \phi_{\boldsymbol{u}}(\mathbf{BAR}): \lambda^2, \lambda^2, \lambda^3$

→ **u** 중복 제거 후 표 형태로!

Kernels for Sequences

- The substring kernel (Lodhi et al., 2004)

$\phi_u(\mathbf{x})$

$\phi_u(\mathbf{CAT}): \lambda^2, \lambda^3, \lambda^2$

$\phi_u(\mathbf{CAR}): \lambda^2, \lambda^3, \lambda^2$

$\phi_u(\mathbf{BAT}): \lambda^2, \lambda^2, \lambda^3$

$\phi_u(\mathbf{BAR}): \lambda^2, \lambda^2, \lambda^3$

8-dimensional feature space

u	C-A	C-T	A-T	B-A	B-T	C-R	A-R	B-R
$\phi_u(\mathbf{CAT})$	λ^2	λ^3	λ^2	0	0	0	0	0
$\phi_u(\mathbf{CAR})$	λ^2	0	0	0	0	λ^3	λ^2	0
$\phi_u(\mathbf{BAT})$	0	0	λ^2	λ^2	λ^3	0	0	0
$\phi_u(\mathbf{BAR})$	0	0	0	λ^2	0	0	λ^2	λ^3

→ **kernel?** $K_{k,\lambda}(\mathbf{seq1}, \mathbf{seq2}) = \sum_{u \in \mathcal{A}^k} \phi_u(\mathbf{seq1}) \phi_u(\mathbf{seq2})$

Kernels for Sequences

- The substring kernel (Lodhi et al., 2004)

$$\phi_u(\mathbf{x}) \qquad K_{k,\lambda}(\mathbf{seq1}, \mathbf{seq2}) = \sum_{u \in \mathcal{A}^k} \phi_u(\mathbf{seq1})\phi_u(\mathbf{seq2})$$

u	C-A	C-T	A-T	B-A	B-T	C-R	A-R	B-R
$\phi_u(\mathbf{CAT})$	λ^2	λ^3	λ^2	0	0	0	0	0
$\phi_u(\mathbf{CAR})$	λ^2	0	0	0	0	λ^3	λ^2	0
$\phi_u(\mathbf{BAT})$	0	0	λ^2	λ^2	λ^3	0	0	0
$\phi_u(\mathbf{BAR})$	0	0	0	λ^2	0	0	λ^2	λ^3

$$e.g.\ K_{k=2,\lambda}(\mathbf{CAT}, \mathbf{CAR})$$
$$= \lambda^2 \cdot \lambda^2 + 0 + \cdots + 0$$
$$= \lambda^4$$

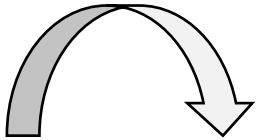
Kernels for Sequences

- The substring kernel (Lodhi et al., 2004)

$K(\text{seq1},\text{seq2})$

Kernel normalization

$$\frac{K(\text{seq1},\text{seq2})}{\sqrt{K(\text{seq1},\text{seq1})}\sqrt{K(\text{seq2},\text{seq2})}}$$



$e.g.\ K_{k=2,\lambda}(\text{CAT}, \text{CAR})$

$= \lambda^2 \cdot \lambda^2 + 0 + \cdots + 0$

$= \lambda^4$

Non-normalized kernel

$$\frac{K_{k=2,\lambda}(\text{CAT}, \text{CAR})}{\sqrt{K_{k=2,\lambda}(\text{CAT}, \text{CAT})}\sqrt{K_{k=2,\lambda}(\text{CAR}, \text{CAR})}}$$

$$= \frac{\lambda^4}{\sqrt{2\lambda^4 + \lambda^6}\sqrt{2\lambda^4 + \lambda^6}}$$

$$= \frac{1}{2 + \lambda^2}$$

→ **Normalized kernel !**

Kernels for Sequences

- The substring kernel (Lodhi et al., 2004)

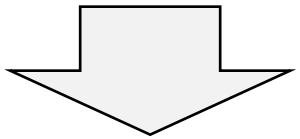
$$\frac{K(\mathbf{seq1},\mathbf{seq2})}{\sqrt{K(\mathbf{seq1},\mathbf{seq1})}\sqrt{K(\mathbf{seq2},\mathbf{seq2})}} = \frac{\sum_{u\in\mathcal{A}^k}\phi_u(\mathbf{seq1})\phi_u(\mathbf{seq2})}{\sqrt{\sum_{u\in\mathcal{A}^k}\phi_u(\mathbf{seq1})\phi_u(\mathbf{seq1})}\sqrt{\sum_{u\in\mathcal{A}^k}\phi_u(\mathbf{seq1})\phi_u(\mathbf{seq2})}}$$

$$\textbf{similarity} \rightarrow \quad \cos(\theta) = \frac{A\cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n}(A_i)^2} \times \sqrt{\sum_{i=1}^{n}(B_i)^2}}$$

Kernels for sequences = compute the similarity of two input sequences

Conclusions

- The spectrum kernel: K_k (seq1,seq2)= $\sum_{u \in \mathcal{A}^k} \phi_k(\text{seq1})\phi_k(\text{seq2})$
  - substring 의 빈도수 고려 (gap은 고려하지 않음)
- The mismatch kernel: $K_{k,m}$(seq1,seq2)= $\sum_{a,\beta \in \mathcal{A}^k} \phi_{k,m}(\text{seq1})\phi_{k,m}(\text{seq2})$
  - Mismatch 경우를 포함한 substring 의 빈도수 고려 (gap은 고려하지 않음)
- The substring kernel: $K_{k,\lambda}(\text{seq1}, \text{seq2}) = \sum_{u \in \mathcal{A}^k} \phi_u(\text{seq1})\phi_u(\text{seq2})$
  - substring 길이의 gap을 고려한 weight 값을 기반으로 유사도 정의



String kernels
(Mercer's theorem, similarity 표현)

Conclusions

- **Define a (possibly high-dimensional) feature space of interest**
  - Spectrum, mismatch, substring kernels → 가장 일반적으로 적용되는 kernels
  - Physico-chemical kernels
  - Pairwise, motif kernels
- **Derive a kernel from a generative model**
  - Fisher kernel
  - Mutual information kernel
  - Marginalized kernel
- **Derive a kernel from a similarity measure**
  - Local alignment kernel
  - Global alignment kernel

References

- [1] Leslie, C., Eskin, E., & Noble, W. S. (2001). The spectrum kernel: A string kernel for SVM protein classification. In Biocomputing 2002 (pp. 564-575).

- [2] Leslie, C. S., Eskin, E., Cohen, A., Weston, J., & Noble, W. S. (2004). Mismatch string kernels for discriminative protein classification. Bioinformatics, 20(4), 467-476.

- [3] Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., & Watkins, C. (2002). Text classification using string kernels. Journal of Machine Learning Research, 2(Feb), 419-444

Thank You !

EOD