

2021 Spring

Artificial Intelligence & Deep Learning

Prof. Minsuk Koo

Department of Computer Science &
Engineering
Incheon National University

시험범위 = 전범위



6.6.2 독립 성분 분석 = ICA

■ 블라인드 원음 분리 문제

- 실제 세계에서는 여러 신호가 섞여 나타남([그림 6-21]은 음악과 대화가 섞이는 예)

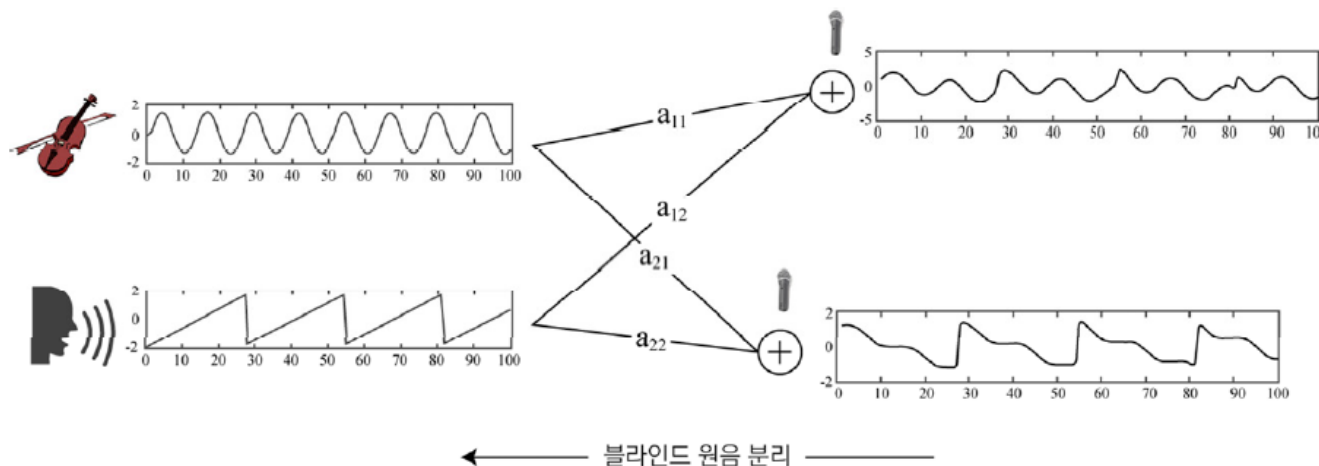


그림 6-21 블라인드 원음 분리 문제

- 마이크로 측정한 혼합 신호로부터 원음(음악과 목소리)을 복원할 수 있나? → 블라인드 원음 분리 문제라 부르며 독립 성분 분석 기법으로 해결 가능
- 아주 많은 예, 뇌파와 다른 장기 신호가 섞인 EEG, 장면과 잡음이 섞인 영상, ...

6.6.2 독립 성분 분석

■ 문제 정의

■ 표기

- 원래 신호를 $z_1(t)$ 와 $z_2(t)$, 측정된 혼합 신호를 $x_1(t)$ 와 $x_2(t)$ 로 표기
- t 순간에 획득된 $\mathbf{x}_t = (x_1(t), x_2(t))^T$ 를 훈련 샘플로 취함. 따라서 훈련집합은 $\mathbb{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$ → 1순간에 획득된 혼합 신호
- 블라인드 원음 분리 문제는 $\mathbb{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$ 로부터 $\mathbb{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_n\}$ 를 찾는 문제

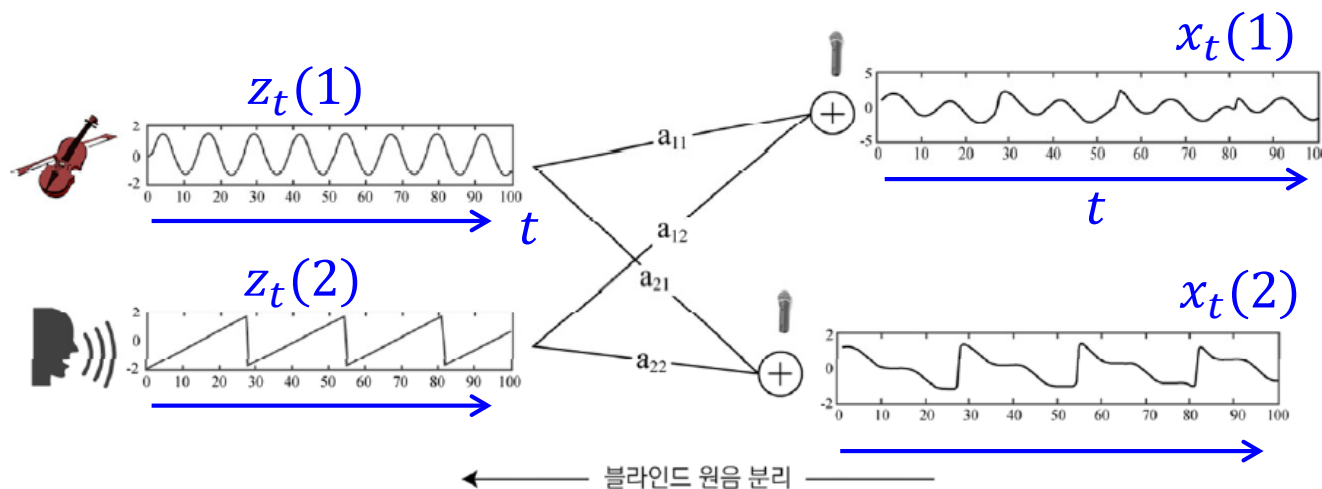


그림 6-21 블라인드 원음 분리 문제

6.6.2 독립 성분 분석

■ 문제 공식화

- 혼합 신호 \mathbf{x} 를 원래 신호 \mathbf{z} 의 선형 결합으로 표현 가능 ($z_1(t)$ 와 $z_2(t)$ 가 독립이라는 가정)

$$\begin{cases} x_1 = a_{11}z_1 + a_{12}z_2 \\ x_2 = a_{21}z_1 + a_{22}z_2 \end{cases} \quad (6.24)$$

- 행렬 표기로 쓰면,

$$\mathbf{x} = \mathbf{A}\mathbf{z} \quad (6.25)$$

- 블라인드 원음 분리 문제란 \mathbf{A} 를 구하는 것. \mathbf{A} 를 알면, 식 (6.26)으로 원음 복원

$$\tilde{\mathbf{z}} = \mathbf{W}\mathbf{x}, \quad \text{이때 } \mathbf{W} = \mathbf{A}^{-1} \quad (6.26)$$

■ 식 (6.25)는 과소 조건 문제

- 정수 하나를 주고 어떤 두 수의 곱인지 알아내라는 문제와 비슷함 (예를 들어, 32는 1×32 , 2×16 , 4×8 등 여러 답이 가능) ← 추가 조건을 주면 유일 해가 가능
- 문제도 과소 적합
- 추가 조건을 이용하여 식 (6.25)의 해를 찾음 → 독립성 가정과 비가우시안 가정

6.6.2 독립 성분 분석

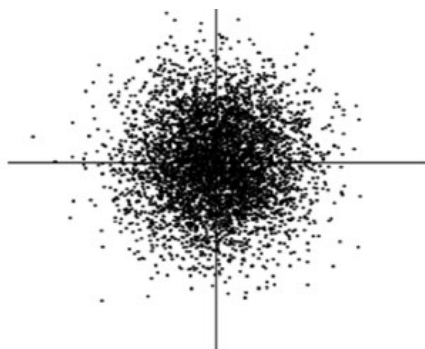
■ 독립성 가정

- 원래 신호가 서로 독립이라는 가정(예, 음악과 대화는 서로 무관하게 발생함)

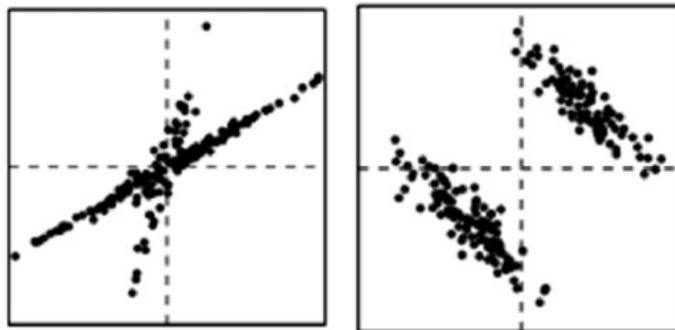
$$P(\mathbf{z}) = P(z_1, z_2, \dots, z_d) = \prod_{j=1}^d P(z_j) \quad (6.27)$$

■ 비가우시안 가정

- 원래 신호가 가우시안이라면 혼합 신호도 ([그림 6-22(a)]처럼) 가우시안이 되므로 분리할 실마리 없음. 비가우시안이면 ([그림 6-22(b)]처럼) 실마리가 있음



(a) 확률변수가 가우시안일 때



(b) 확률변수가 비가우시안일 때

구분이 가능하다.

그림 6-22 서로 독립인 두 확률변수의 결합 분포

6.6.2 독립 성분 분석

■ ICA의 문제 풀이 → 얼마나 멀리 떨어져있는지

- 원래 신호의 비가우시안인 정도를 최대화하는 가중치를 구하는 전략 사용
 - 원래 신호를 식으로 쓰면,

$$\left. \begin{array}{l} z_j = w_{j1}x_1 + w_{j2}x_2 \\ \text{행렬 형태로 쓰면 } z_j = \mathbf{w}_j \mathbf{x} \end{array} \right\} \quad (6.28)$$

- 비가우시안을 최대화하는 가중치를 구하는 식을 쓰면,

$$\hat{\mathbf{w}}_j = \underset{\mathbf{w}_j}{\operatorname{argmax}} \check{G}(z_j) \quad (6.29)$$

- \check{G} 는 비가우시안 정도를 측정하는 함수
- 주로 식 (6.31)의 첨도를 사용

$$\text{kurtosis}(z_j) = \frac{1}{n} \sum_{i=1}^n z_{ji}^4 - 3 \left(\frac{1}{n} \sum_{i=1}^n z_{ji}^2 \right)^2 \quad (6.31)$$

첨도를 최대화

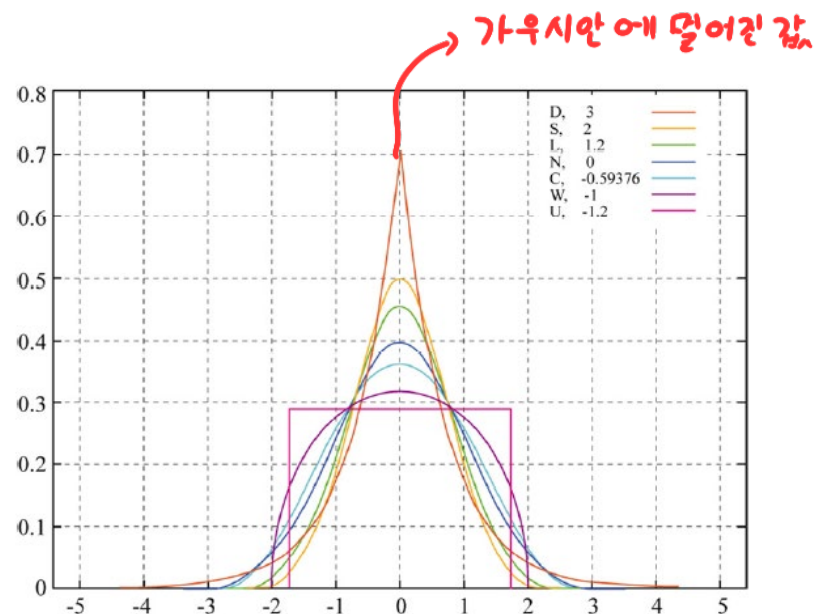


그림 6-23 여러 가지 분포의 첨도 측정

6.6.2 독립 성분 분석

■ ICA 학습

1. 전처리 수행

- 훈련집합 \mathbf{x} 의 평균이 0이 되도록 이동(식 (6.19) 적용)
- 식 (6.30)의 화이트닝 변환 적용

$$\mathbf{x}'_i = \left(\mathbf{D}^{-\frac{1}{2}} \mathbf{V}^T \right) \mathbf{x}_i, i = 1, 2, \dots, n$$



(6.30)

2. 식 (6.29)를 풀어 최적 가중치 구함

■ PCA와 ICA 비교

- ICA는 비가우시안과 독립성 가정, PCA는 가우시안과 비상관을 가정
- ICA는 4차 모멘트까지 사용, PCA는 2차 모멘트까지 사용
- ICA로 찾은 축은 수직 아님, PCA로 찾은 축은 서로 수직
- ICA는 주로 블라인드 원음 분리 문제를 푸는데, PCA는 차원 축소 문제를 품

6.6.3 희소 코딩

■ 기저함수 또는 기저 벡터의 선형 결합으로 신호를 표현

- 푸리에 변환([그림 6-24(a)) 또는 웨이블릿 변환 등

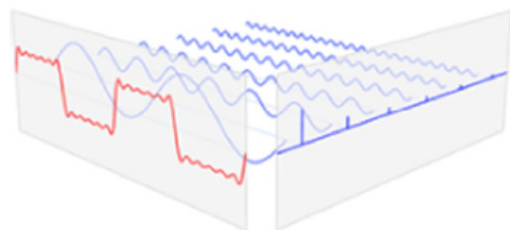
■ 희소 코딩

- 사전 \mathbf{D} 를 구성하는 기저 벡터(단어) $\mathbf{d}_1, \mathbf{d}_2, \dots, \mathbf{d}_m$ 의 선형 결합으로 신호(영상) \mathbf{x} 를 표현

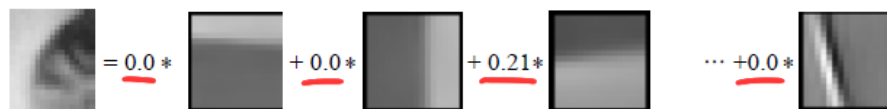
$$\mathbf{x} = \mathbf{D}\mathbf{a}$$

이때 $\mathbf{D} = (\mathbf{d}_1 \ \mathbf{d}_2 \ \dots \ \mathbf{d}_m)$

(6.32)



(a) 푸리에 변환


$$\mathbf{x} = 0.0 * \mathbf{d}_1 + 0.0 * \mathbf{d}_2 + 0.21 * \mathbf{d}_3 + \dots + 0.0 * \mathbf{d}_m$$

$$\text{사전 } \mathbf{D} = \left\{ \begin{bmatrix} \text{basis vector 1} \\ \text{basis vector 2} \\ \text{basis vector 3} \\ \vdots \\ \text{basis vector m} \end{bmatrix} \right\}$$

$$\text{희소 코드 } \mathbf{a} = (0.0, 0.0, 0.21, \dots, 0.0)$$

(b) 희소 코딩

그림 6-24 신호를 기저함수 또는 기저 벡터의 선형 결합으로 근사 표현

6.6.3 희소 코딩

■ 희소 코딩이 다른 변환 기법과 다른 점

- 비지도 학습이 사전(즉 기저 벡터)를 자동으로 알아냄(푸리에 변환은 삼각함수를 사용함)
→ 희소 코딩은 데이터에 맞는 기저 벡터를 사용하는 셈
- 사전의 크기를 과잉 완벽하게 책정($m > d$)
- 희소 코드 \mathbf{a} 를 구성하는 요소 대부분이 0 값을 가짐

■ 회소 코딩 구현

- 최적의 사전과 최적의 희소 코드를 알아내야 함
- ϕ 는 희소 코드의 희소성을 강제하는 규제항

$$\hat{\mathbf{D}}, \hat{\mathbf{A}} = \underset{\mathbf{D}, \mathbf{A}}{\operatorname{argmin}} \sum_{i=1}^n \|\mathbf{x}_i - \mathbf{D}\mathbf{a}_i\|_2^2 + \lambda \phi(\mathbf{a}_i) \quad (6.33)$$

6.7 오토인코더

- 6.7.1 규제 오토인코더
- 6.7.2 적층 오토인코더

6.7 오토인코더

■ 오토인코더

- 특징 벡터 \mathbf{x} 를 입력 받아 동일한 또는 유사한 벡터 \mathbf{x}' 를 출력하는 신경망

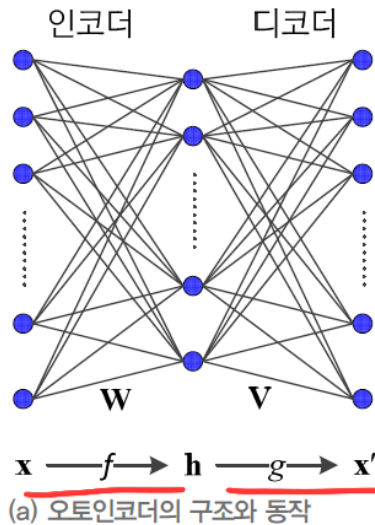
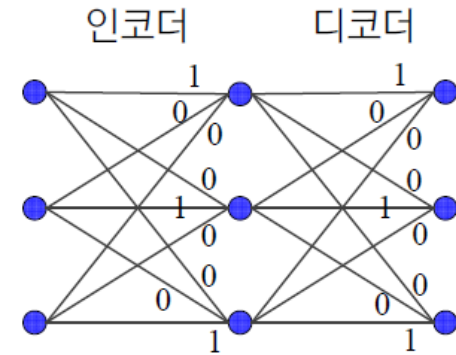
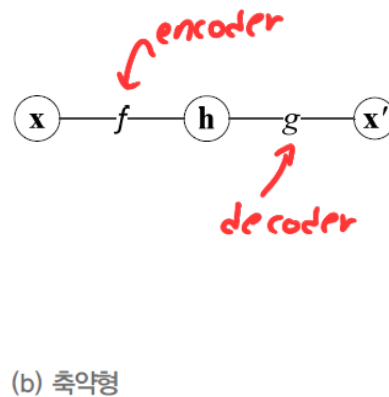


그림 6-25 오토인코더



- 단순 복사하는 단위 행렬([그림 6-26])은 무의미

$$\mathbf{x} = g(f(\mathbf{x})) = \mathbf{VW}\mathbf{x} = \mathbf{I}\mathbf{x}$$

- 여러 가지 규제 기법 적용하여 유용한 신경망으로 활용

6.7 오토인코더

input node 보다 작은 hidden node 가짐

■ 병목 구조 오토인코더의 동작 원리

- $m < d$ 인 구조(예, 256*256 영상을 입력 받아 256*256 영상을 출력하는 경우 $d = 65536$ 인데 $m = 1024$ 로 설정)
- 은닉층의 h 는 훨씬 적은 메모리로 데이터 표현. 필요하면 디코더로 원래 데이터 복원
- h 는 x 의 핵심 정보를 표현 → 특징 추출, 영상 압축 등의 응용

■ 여러 형태의 오토인코더

- 은닉 노드의 개수에 따라 $m < d$, $m = d$, $m > d$ 구조
- 활성화함수에 따라 선형(식 (6.34))과 비선형(식 (6.35))

$$\left. \begin{aligned} \mathbf{h} &= f(\mathbf{x}) = \mathbf{W}\mathbf{x} \\ \mathbf{x} &= g(\mathbf{h}) = \mathbf{V}\mathbf{h} \end{aligned} \right\} \quad (6.34)$$

$$\left. \begin{aligned} \mathbf{h} &= f(\mathbf{x}) = \tau_{\text{encode}}(\mathbf{W}\mathbf{x}) \\ \mathbf{x} &= g(\mathbf{h}) = \tau_{\text{decode}}(\mathbf{V}\mathbf{h}) \end{aligned} \right\} \quad (6.35)$$

6.7 오토인코더

■ 오토인코더의 학습

- 주어진 데이터는 훈련집합 $\mathbb{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$, 알아내야 하는 매개변수는 f 와 g 라는 매핑 함수, 즉 가중치집합 $\Theta = \{\mathbf{W}, \mathbf{V}\}$
- 오토인코더 학습을 최적화 문제로 쓰면,

$$\hat{\Theta} = \underset{\Theta}{\operatorname{argmin}} \sum_{i=1}^n L(\mathbf{x}_i, g(f(\mathbf{x}_i))) \quad (6.36)$$

$$\underbrace{L(\mathbf{x}_i, g(f(\mathbf{x}_i)))}_{\text{Loss function}} = \underbrace{\|\mathbf{x}_i}_{\substack{\downarrow \\ \text{real} \\ \text{input}}} - \underbrace{g(f(\mathbf{x}_i))}_{\substack{\downarrow \\ \text{복원한} \\ \text{값의}}} \|_2^2 \quad \underbrace{\text{L2 norm}}_{\text{L2 norm}} \quad (6.37)$$

6.7.1 규제 오토인코더

■ 여러 규제 기법을 적용

- $m > d$ 인 상황에서도 단순 복사를 피할 수 있음 ← 충분히 큰 모델을 사용하되 적절한 규제 기법을 적용하는 현대 기계 학습 추세를 오토인코더로 따르는 셈

■ SAE(sparse autoencoder)

- 은닉 벡터 \mathbf{h}_i 가 희소하도록 강제화(0이 아닌 요소의 개수를 적게 유지)
- $\phi(\mathbf{h}_i)$ 는 벡터 \mathbf{h}_i 가 희소하도록 강제하는 규제항

$$\text{SAE: } \hat{\Theta} = \underset{\Theta}{\operatorname{argmin}} \sum_{i=1}^n L(\mathbf{x}_i, g(f(\mathbf{x}_i))) + \lambda \phi(\mathbf{h}_i) \quad (6.38)$$

↳ 은닉 노드들 많이 사용 할수록
 $\lambda \phi(\mathbf{h}_i)$ 올라감 (↑)

■ DAE(denoising autoencoder)

- 잡음을 추가한 다음 원본을 복원하도록 학습하는 원리
- 특징 벡터 \mathbf{x}_i 에 적절한 양의 잡음을 추가한 $\tilde{\mathbf{x}}_i$ 를 입력으로 사용

$$\text{DAE: } \hat{\Theta} = \underset{\Theta}{\operatorname{argmin}} \sum_{i=1}^n L(\mathbf{x}_i, g(f(\tilde{\mathbf{x}}_i))) \quad (6.39)$$

6.7.1 규제 오토인코더

■ CAE(contractive autoencoder)

- 인코더함수 f 의 야코비안 행렬의 프로베니우스 놈을 작게 유지

$$\left. \begin{aligned} \text{CAE: } \hat{\Theta} = \underset{\Theta}{\operatorname{argmin}} \sum_{i=1}^n L(\mathbf{x}_i, g(f(\mathbf{x}_i))) + \lambda \phi(\mathbf{x}_i, \mathbf{h}_i) \\ \text{이때 } \phi(\mathbf{x}_i, \mathbf{h}_i) = \left\| \frac{\partial f}{\partial \mathbf{x}} \right\|_F^2 \end{aligned} \right\} \quad (6.40)$$

- CAE는 공간을 축소하는 효과

6.7.2 적층 오토인코더

■ 오토인코더는 얇은 신경망

- 은닉층이 하나뿐임 → 표현력에 한계
- 여러 층으로 쌓으면 용량이 커짐

■ 적층 오토인코더

- 층별 예비학습을 이용하여 [그림 6-29]처럼 깊은 신경망을 만들

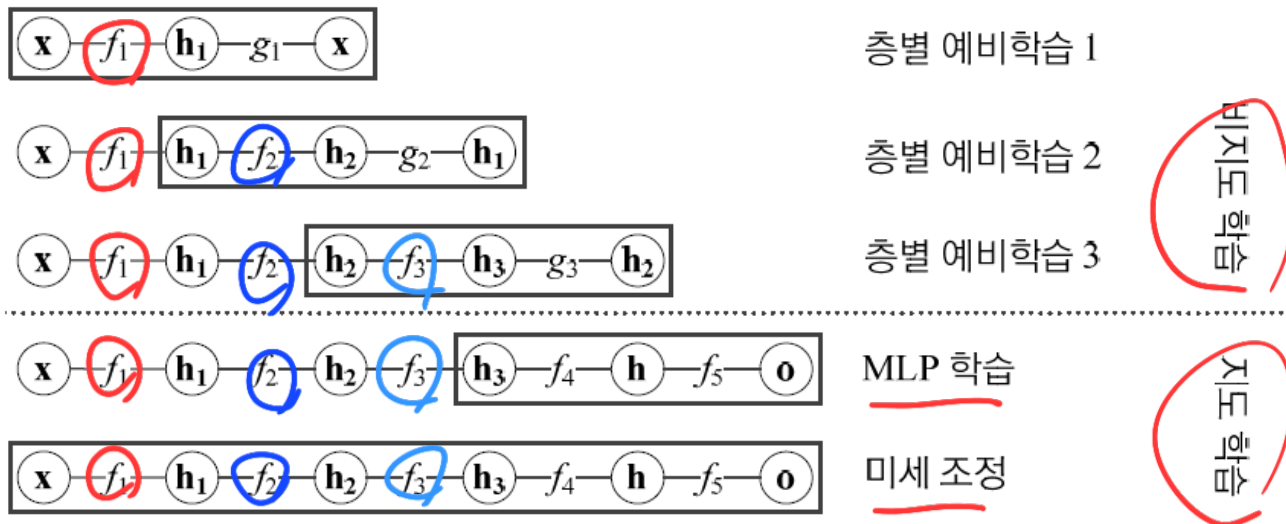


그림 6-29 적층 오토인코더의 학습 과정

6.7.2 적층 오토인코더

■ 적층 오토인코더를 지도 학습(분류)에 활용하는 경우의 학습 과정

1. 층별 예비학습을 필요한 만큼 수행한다. (\mathbb{X} 만 가지고 비지도 학습) // [그림6-29]는 3번 수행
2. 마지막 층의 출력을 입력으로 하여, MLP를 학습한다. (\mathbb{X} 와 \mathbb{Y} 를 가지고 지도 학습) // [그림6-29]는 \mathbf{h}_3 가 마지막 층의 출력임
3. 신경망 전체를 한꺼번에 추가로 학습한다. ← 미세 조정 단계

■ 층별 예비학습의 역사적 의미

- 깊은 구조의 MLP 학습이 번번이 실패하던 상황에서 2006년에 힌튼 교수가 층별 예비 학습 아이디어를 제안하고, 성공적인 성능을 입증
- 이후 기계 학습 연구자들의 관심 고조되고, 현재 딥러닝은 기계 학습을 주도

■ 여러 가지 기술 향상으로 현재는 층별 예비학습을 별로 사용하지 않음

6.8 매니폴드 학습

- 6.8.1 매니폴드란? \leadsto 고차원에 보이는 저차원
- 6.8.2 IsoMap
- 6.8.3 LLE
- 6.8.4 t-SNE
- 6.8.5 귀납적 학습 모델과 트랜스덕티브 학습 모델
"transductive"
- 오토인코더는 데이터 구조를 간접적으로 표현
- 반면 매니폴드 학습은 데이터의 비선형 구조를 직접적으로 반영

6.8.1 매니폴드란?

■ 매니폴드

- 매니폴드는 고차원 공간에 내재한 저차원 공간
- [그림 6-30]에서 도로가 매니폴드에 해당
- 자동차 위치를 데이터로 간주하면, $\mathbf{x} = (\text{위도}, \text{경도}, \text{고도})^T$
- 이 3차원(고차원) 공간 데이터를 $\mathbf{x} = (\text{기준점에서의 거리})^T$ 라는 1차원(저차원) 공간, 즉 매니폴드로 표현할 수 있음
- 보통 매니폴드는 비선형 공간이지만 지역적으로 살펴보면 선형 구조



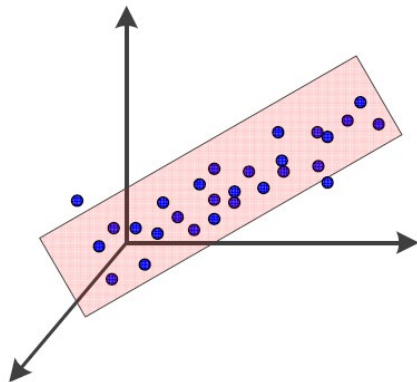
그림 6-30 매니폴드

6.8.1 매니폴드란?

■ 매니폴드 가정

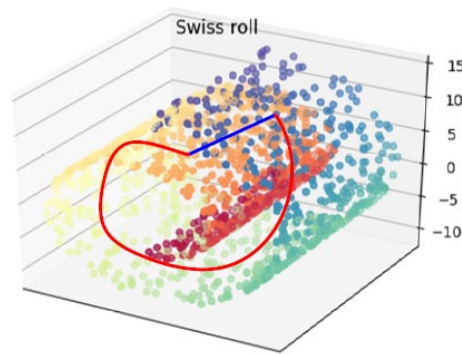
“real-world data presented in high-dimensional spaces are expected to concentrate in the vicinity of a manifold M of much lower dimensionality d_M , embedded in high-dimensional input space R^d . 고차원 공간에 주어진 실제 세계의 데이터는 고차원 입력 공간 R^d 에 내재한 훨씬 저차원인 d_M 차원 매니폴드의 인근에 집중되어 있다.”

■ 인위적인 상황 예시



(a) PCA로 찾은 매니폴드가 적절한 상황

그림 6-31 매니폴드 가정



(b) 비선형 매니폴드가 필요한 상황

■ 매니폴드를 어떻게 찾고, 어떻게 표현할 것인가?

6.8.5 귀납적 학습 모델과 트랜스덕티브 학습 모델

■ 트랜스덕티브 학습 모델

- 훈련집합 이외의 새로운 샘플을 처리할 능력이 없는 모델
- IsoMap, LLE, t-SNE는 모두 트랜스덕티브 모델
- 데이터 가시화라는 목적에 관한 한 PCA나 오토인코더와 같은 귀납적 모델보다 성능이 뛰어나

“If you possess a restricted amount of information for solving some problem, try to solve the problem directly and never solve a more general problem as an intermediate problem. 어떤 문제를 풀 때 데이터가 제한되어 있으면, 그 문제를 직접 풀어야 한다. 중간 문제로서 좀 더 일반적인 문제를 풀 필요가 전혀 없다.”

■ 귀납적 모델

- 훈련집합 이외의 새로운 샘플을 처리할 능력이 있는 모델
- IsoMap, LLE, t-SNE를 제외한 지금까지 공부한 모든 모델

■ 주어진 문제에 따라 둘 중 적절한 것을 선택하는 지혜 필요

6장 연습문제

연습문제

1 [예제 6-1]에서 초기 군집 중심을 $z_1 = x_2, z_2 = x_3, z_3 = x_7$ 로 다시 설정했을 때 k -평균의 동작을 예제처럼 보이시오. 다중 시작을 사용한다면 둘 중 어느 해를 취할지 판단하시오.

J라는 값은 동등해 판단

2 [예제 6-1]의 초기 군집 중심과 문제 1의 초기 군집 중심 각각에 k -medoids 알고리즘을 적용하시오.

3 k -평균과 k -medoids의 계산 시간을 분석하시오. 어느 것이 몇 배 정도 더 많은 시간을 쓸지 추정할 수 있는가?
 $O(n)$
대물점에서 n 에 따라 샘플에 따라 다른 샘플들 간의 거리와 합의 최소로 \Rightarrow 대표!

4 매개변수를 가지지 않은 비모수적 모델은 모두 트랜스덕티브한가? 아니라면 반례를 제시하시오.

5 친밀도 전파 알고리즘에 대해 답하시오.

- (1) 식 (6.3)을 이용하여 [그림 6-5]에 있는 데이터의 유사도 행렬 **S**를 구하시오. 대각선 요소인 s_{kk} 는 유사도의 최솟값을 사용하시오.
- (2) 한 번 반복한 후의 책임 행렬 **R**과 가용 행렬 **A**를 구하시오.
- (3) 한 번 더 반복한 후의 책임 행렬 **R**과 가용 행렬 **A**를 구하시오.

eigen vector

$$A = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix} \quad A \cdot \lambda I = \begin{pmatrix} 2-\lambda & 1 \\ 1 & 2-\lambda \end{pmatrix}$$

$$\lambda = 3 \text{ or } 1$$

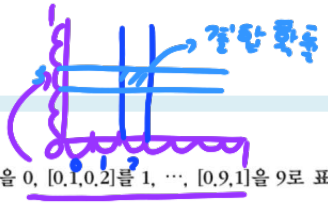
$$\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} = 0$$

$$u_1 - u_2 = 0$$

$$u_1 = u_2$$

$$\therefore \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

6 [그림 6-8]에는 80개의 샘플이 있다. $[0, 0.1]$ 구간을 0, $[0.1, 0.2]$ 를 1, ..., $[0.9, 1]$ 을 9로 표시하고, 히스토그램 방식을 사용한다.



- (1) 다음 확률을 구하시오.
 결합확률 $P(x_1 = 2, x_2 = 3)$, 조건부 확률 $P(x_1 = 2 | x_2 = 3)$, 주변 확률 $P(x_2 = 3)$
- (2) 다음 확률을 구하시오.
 결합확률 $P(x_1 = 3, x_2 = 5)$, 조건부 확률 $P(x_2 = 5 | x_1 = 3)$, 주변 확률 $P(x_1 = 3)$

7 가우시안 혼합을 위한 EM 알고리즘의 계산 시간을 분석하시오.

- (1) E 단계의 시간 복잡도는?
- (2) M 단계의 시간 복잡도는?
- (3) 알고리즘 전체의 시간 복잡도는?

8 EM 알고리즘은 욕심 알고리즘 greedy algorithm인가? 답에 대한 이유를 설명하시오.

9 혼련집합이 다음과 같다. PCA를 이용하여 2차원을 1차원으로 변환하는 행렬을 구하시오.
 Hint: Matlab 또는 Python을 사용하시오.

$$X = \left\{ \begin{pmatrix} 1 \\ 2 \end{pmatrix}, \begin{pmatrix} 2 \\ 2 \end{pmatrix}, \begin{pmatrix} 3 \\ 2 \end{pmatrix}, \begin{pmatrix} 3 \\ 3 \end{pmatrix}, \begin{pmatrix} 4 \\ 3 \end{pmatrix}, \begin{pmatrix} 4 \\ 4 \end{pmatrix}, \begin{pmatrix} 5 \\ 4 \end{pmatrix}, \begin{pmatrix} 6 \\ 4 \end{pmatrix} \right\} \quad \mu = \begin{bmatrix} 3.5 \\ 3 \end{bmatrix}$$

공분산 행렬, eigen vector, eigen value

$$\frac{1}{n} \sum (x - \mu)(x - \mu)^T$$

eigen value (기저) = $\sum u = \lambda u$

$$(\sum u - \lambda)u = 0$$

공분산 $\lambda = \text{eigen value}$