AiDA Lab.
Artificial Intelligence & Data Analysis Laboratory

AI 고급 : 자연어 처리 과정(정규)

10. 텍스트 분류 이론

Seokhwan Yang

텍스트 분류

- 텍스트 분류
  - 문장 또는 (문장들로 이루어진) 문서를 입력으로 받아

    1.사전에 정의된 클래스 중에 어디에 속하는지 분류(Classification)하거나

    2.각 데이터를 군집화(Clustering)하는 과정을 말함
  - 자연어 처리에서 가장 포괄적이면서 중요한 분야 중 하나
  - 텍스트를 입력으로 받아 불연속적인 값으로 출력해야 하는 문제는 대부분이 텍스트 분류 문제에 속한다고 볼 수 있음
    - 대부분의 일상과 관련된 자연어 처리 문제들은 분류 문제로 간주되거나 분류 문제로 적용할 수 있음

텍스트 분류

- **텍스트 분류의 응용 분야 (예시)**

문제	클래스(분류) 예
감성(감정) 분석	긍정, 중립, 부정
스팸 메일 탐지	정상, 스팸
사용자 의도 분류	명령, 질문, 잡담 등
주제 분류	각 주제
카테고리 분류	각 카테고리

감정 분석

- **감정 분석(Sentiment Analysis)은...**
  - **문장 또는 지문의 감정을 분석하는 것을 의미, 자연어 처리의 큰 분야 중 하나**
  - **텍스트에 나타난 사람들의 태도, 의견, 성향과 같은 데이터를 분석하는 기법**
  - **"오피니언 마이닝" 이라고 부르기도 함**
  - **자연어 처리 분야 중 상당히 어려운 분야로 손꼽히고 있음**
    - 자연어에 담긴 사람의 주관을 파악하기는 어렵다. (사람에게도 어렵다...)
    - 비슷해 보이는 문장이라도 작은 변화로 긍정, 부정 의견이 갈릴 수 있다.
    - 중의적인 문장의 경우 컴퓨터가 분석하기가 특히 난해하다
    - 주변의 상황적인 정보, 문맥에 따른 정보도 파악해야 한다

감정 분석

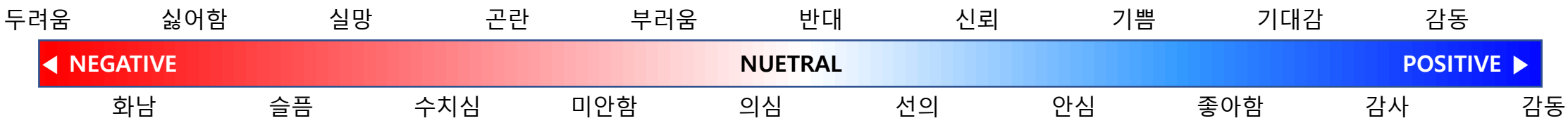
- 기존의 감성 분석 연구
  - 사전 구축을 통하여 긍정/부정에 대한 가중치를 계산하는 규칙 기반 모델
  - 문장에 긍정/부정이 포함되어 있는지 판별하는 모델

- 최근의 감성 분석 연구
  - 딥러닝 모델인 단어 임베딩, CNN, RNN 등을 이용한 감정 분석 모델

감정 분석

• 감정 분석 분류의 예시

해당 감정 중 1개를 컴퓨터가 분류하는 시스템
→ 감정분석 시스템



감성	포함 감성 예	감성	포함 감성 예
자신감	자신감	두려움	공포, 무서움, 걱정, 불안
감동	감탄, 존경, 경의	화남	격노, 분노, 억울함, 짜증
감사	감사	싫어함	미움, 증오, 혐오, 지루함, 심심함, 싫증
기대감	설렘, 희망, 기대됨	슬픔	연민, 애처로움, 쓸쓸함, 참담함, 비참함, 안타까움
좋아함	호감, 사랑	실망	체념, 아쉬움, 그리움, 허무, 어이없음, 황당함, 후회
기쁨	즐거움, 흥미, 행복함, 반가움, 만족	수치심	창피, 무안, 부끄러움, 민망
안심	위안, 안심, 안도	곤란	서먹, 어색, 거북함, 곤란, 난처
신뢰	신뢰, 믿음	미안함	미안함
선의	인사말, 축하, 응원, 격려	부러움	질투, 시기
의심	불신	반대	비 동의, 비 납득

인간의 감정을 20개로 분류해 놓은 예시(ETRI, 한국전자통신연구원)

감정 분석

- **감정 분석 분야에서 중요한 또 하나의 주제**
  - **감정 사전 구축**
    - 영어로 된 데이터는 많지만 한글 데이터는 그다지 많지 않음
    - 감정 분석 사전 종류의 예시
      - awkward

WordNet	SentiWordNet	Opinion Lexicon	MPQA
• Score : 0.0 • (Pos > 0, Neg < 0)	• Positive : 0.125 • Negative : 0.5	• Polarity : Negative	• Polarity : Negative • Strength : strongsubj

감정 분석

- 일상에서의 감정 분석의 예시
  - 영화: 이 리뷰는 긍정적인가 부정적인가?
  - 제품: 사람들은 새로운 모델에 대해 어떻게 생각하는가?
  - 민심: 소비자 신뢰도는 어떠한가? 절망감이 증가하고 있는가?
  - 정치: 사람들은 이 후보나 문제에 대해 어떻게 생각하는가?
  - 예측: 정서를 통해 선거 결과 또는 시장 동향을 예측하기

기타 텍스트 분류 예시

- 스팸 메일 탐지, 분류
  - 메일이 왔을 때 해당 내용을 보고 스팸 메일인지 아닌지 분류하는 것
  - 예시

텍스트(메일의 내용)	레이블(스팸 여부)
어제 보내 드린 보고서 확인 부탁드립니다.	정상 메일
마지막 혜택, 놓치지 마세요!	스팸 메일
답장 가능하세요?	정상 메일
(광고) 당신의 스타일을 찾아드립니다.	스팸 메일

기타 텍스트 분류 예시

- **사용자 의도 분류**
  - **문장이나 지문의 내용에 대하여 그 의도를 분류하는 것**
    - 예시: 챗봇에게 "오늘 영업하는 부산에 있는 식당을 추천해줘"

      → 챗봇에게 이 말이 어떤 의도를 갖고 한 말인지 분류하게 함

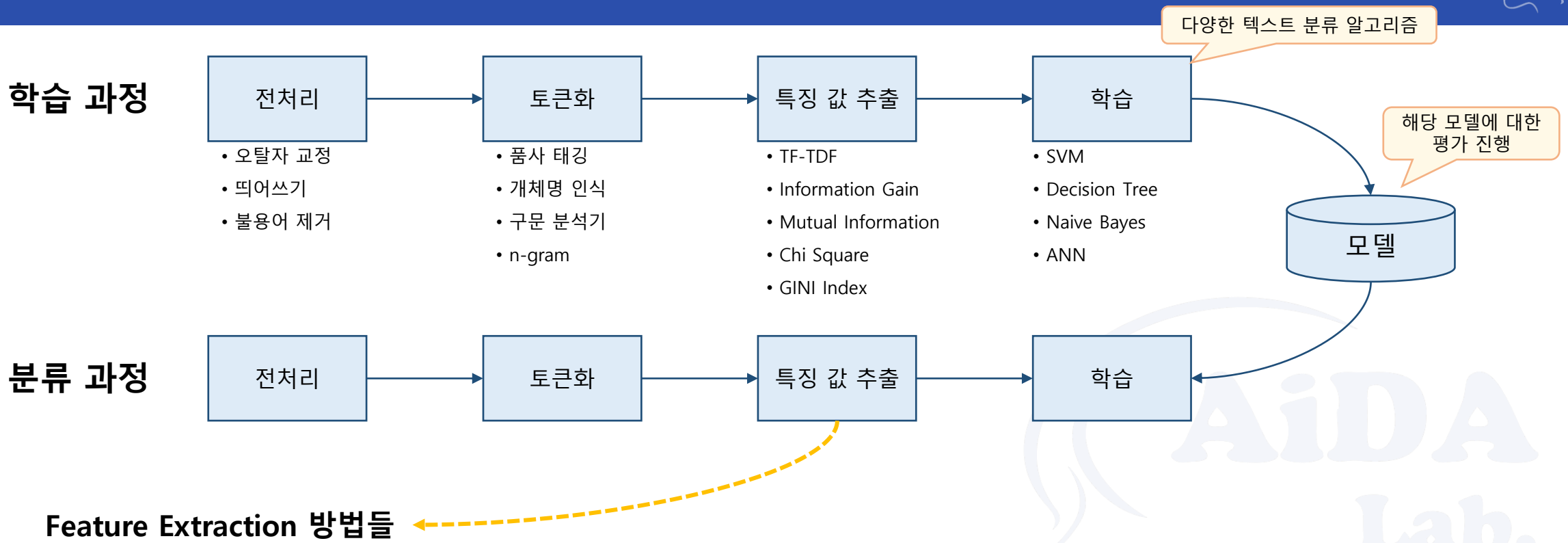
      → 의도: 식당 추천

      → 주요 정보: "오늘 영업", "부산에 있는" 의 두 구문을 잘 해석해서 대응
- **주제 분류**
  - **각각의 주제에 따라 내용을 분류하는 것**
    - 예: 이 블로그의 글은 스포츠와 관련된 글이다...

기타 텍스트 분류 예시

- **카테고리 분류** (도메인 분류라고도 함)
  - **문장이나 지문을 카테고리 별로 분류하는 것**
  - **주제 분류와 유사하게 처리되는 경우도 있음**
    - 예: 이 블로그의 글은 스포츠와 관련된 글이다...

텍스트 분류 프로세스



Feature Extraction 방법들

구분	내용
DictVectorizer	각 단어의 수를 세어 놓은 사전에서 BOW 벡터를 만든다.
CountVectorizer	문서 집합에서 단어 토큰을 생성하고 각 단어의 수를 세어 BOW 인코딩 한 벡터를 만든다.
TfidfVectorizer	CountVectorizer와 비슷하지만 TF-IDF 방식으로 단어의 가중치를 조정한 BOW 벡터를 만든다.
HashingVectorizer	해시 함수(Hash Function)를 사용하여 적은 메모리와 빠른 속도로 BOW 벡터를 만든다.

텍스트 분류 알고리즘

- 대표적인 분류 알고리즘 (딥러닝 이전의 경우 + 딥러닝)
  - 딥러닝 이전의 경우
    - **나이브 베이즈(Naive Bayes) 알고리즘**
    - SVM(Support Vector Machine)
    - KNN(K-Nearest Neighbors)
    - Decision Tree
    - Stochastic Gradient Descent
    - Random Forest 등 다양한 방법 존재

나이브 베이즈(Naïve Bayes)

- 나이브 베이즈 분류
  - 딥러닝 이전의 방식 중에서는 가장 간단한 분류 방식
  - 간단하지만 매우 강력한 결과를 도출함
  - 단어를 불연속적인 심볼로 다루기때문에 여전히 불완전한 부분이 남아있음

나이브 베이즈(Naïve Bayes)

- 베이즈 정리(Baye's Theorem)
  - Bayes(1763)가 증명한 확률에 관한 정리
  - 확률변수의 조건부 확률분포와 주변부 확률분포를 연관 짓는 확률이론
  - 어떤 사건이 서로 배반하는 원인 둘에 의해 일어난다고 할 때, 실제 사건이 일어났을 때 이것이 두 원인 중 하나일 확률을 구하는 정리
  - 1975년 Murphy와 Chase에 의해 유전상담에 대한 응용이 보고되었고, 그 이후로 위험률 추정 등에 이용되기 시작함
  - 새로운 증거에 기반하여 과거의 정보를 향상시키거나 개선시키는 것이 목적

나이브 베이즈(Naïve Bayes)

- **베이즈 정리의 활용 맥락**
  - **역확률(Inverse Probability) 문제 해결**
    - 베이즈 정리는 원리 역확률 문제를 해결하기 위한 방법이었음
    - 조건부 확률 P(B|A)를 알고 있을 때, 전제와 관심 사건의 관계가 정반대인 조건부 확률 P(A|B) 을 구하는 방법
    - 예시
      - 병 A를 앓고 있는지 판정하는 양성판정 정확도가 90%인 검사기가 있고
      - 어떤 사람이 이 검사기로 검사를 시행해서 양성 판정이 나왔다면
      - 이 사람은 90%의 확률로 병에 걸려 있다고 이야기 할 수 있을까?

나이브 베이즈(Naïve Bayes)

- **말할 수 없음**
- 검사가 알려주는 확률과 우리가 알고 싶은 확률은 조건부 확률의 의미에서 **정반대**이므로.

- 검사의 양성판정 정확도 '90%'는 검사가 병을 가진 사람을 정확하게 포착할 확률, 즉 병을 가지고 있다는 전제 하에 검사 결과가 양성일 확률이 90%임을 의미
- 우리가 알고 싶은 것은 검사 결과가 양성이라는 전제 하에 병을 앓고 있을 확률

	전제	관심 사건	수학적 표현
검사의 정확도	병을 앓고 있다	검사 결과: 양성	P(검사 결과: 양성|병을 앓고 있다) = 0.90
우리의 관심사	검사 결과: 양성	병을 앓고 있다	P(병을 앓고 있다|검사 결과: 양성) = ...?

나이브 베이즈(Naïve Bayes)

- 원래대로라면 검사의 정확도만을 가지고 우리의 관심사인 '(양성인 사람이) 병을 앓고 있을 확률'을 알 수 없음
- 그러나 검사 대상인 질병의 유병률을 알고 있다면, 베이즈 정리를 통해 역확률 계산 가능
- 예를 들어 전세계 인구 중 1%가 병 A를 앓는다고 알려져 있다고 가정한다면...
- 그리고 음성판정 정확도(병 A가 걸리지 않은 사람이 실제로 테스트 결과 음성으로 나올 확률)도 양성판정 정확도와 마찬가지로 90%라고 가정한다면...
  → 검사 결과가 양성으로 나온 사람이 실제로 병 A를 앓고 있을 확률은 약 **8.3%**
- 양성판정 정확도가 90%이지만 전체 인구 중 유병자가 1%밖에 되지 않으므로 8.3%에 그침

병을 앓고 있지 않는 99% 인구 중에서
병이 있다고 오진을 받은 경우가
검사기가 병을 앓는 사람을
제대로 진단한 경우를 압도하게 됨

$$
\begin{aligned}
P(\text{병}|\text{양성}) &= \frac{P(\text{양성}|\text{병})P(\text{병})}{P(\text{양성})} \\
&= \frac{P(\text{양성}|\text{병})P(\text{병})}{P(\text{양성}|\text{병})P(\text{병}) + P(\text{양성}|\text{무병})P(\text{무병})} \\
&= \frac{0.9 \times 0.01}{0.9 \times 0.01 + (1-0.9) \times 0.99} \approx 8.3\%
\end{aligned}
$$

나이브 베이즈(Naïve Bayes)

- **데이터를 이용한 사후 확률의 추정**
  - 베이즈 정리를 이전의 경험과 현재의 증거를 토대로 어떤 사건의 확률을 추론하는 알고리즘으로 보고 관심을 가지는 사람들
  - 어떤 사건이 일어날 확률에 대한 임의의 가정 $P(A)$ 에 실제로 발견된 자료나 증거 B 를 반영해서, 자료를 기반으로 어떤 사건이 일어날 확률 $P(A|B)$을 구하는 것이 관심의 대상
  - 대표적인 알고리즘: 나이브 베이즈 (=나이브 베이지안) 알고리즘

나이브 베이즈(Naïve Bayes)

- 예시
  - 어떤 사람이 병 A에 걸렸을 확률을 계산하는 문제에서...
  - 처음에는 어떤 사람이 병 A에 걸려있을 확률에 대해 아는 것이 없어, 전 세계 인구 일반이 해당 질병에 걸릴 확률인 1%의 유병률을 가정
  - 그런데 정확도가 90%인 검사를 받았더니 양성 판정을 받음
  - 이 사람이 검사에서 양성 판정을 받았다는 새로운 사실을 토대로 이 사람이 실제로 병에 걸려 있을 확률을 알 수 있지 않을까?

  - 베이즈 정리 → 어떤 사건 A가 일어났다고 가정할 때 B라는 자료를 얻게 될 확률'에 대한 정보만 알고 있다면, 자료에 근거해서 어떤 사건이 일어날 확률을 새로 계산할 수 있다...

나이브 베이즈(Naïve Bayes)

	(사건 A의) 사후 확률	∝	가능도	×	(사건 A의) 사전 확률
수학적 표현	P(A|B)		P(B|A)		P(A)
예(진단검사)	P(병을 앓고 있다|양성 판정)		P(양성 판정|병을 앓고 있다)		P(병을 앓고 있다)

- 새로운 자료가 없는 상태에서 어떤 사건이 일어날 확률에 대한 가정이 필요함 →사전 확률 (prior probability)
- 사건이 일어났다는 가정 하에서 새로이 가지게 된 자료가 관측될 확률 → 가능도(likelihood)
- 사전확률과 가능도를 이용해서 새롭게 계산한 '어떤 사건이 일어날 확률' → 사후 확률 (posterior probability)

$$P(A|B) \propto P(B|A) \times P(A)$$

∝ : 비례

나이브 베이즈(Naïve Bayes)

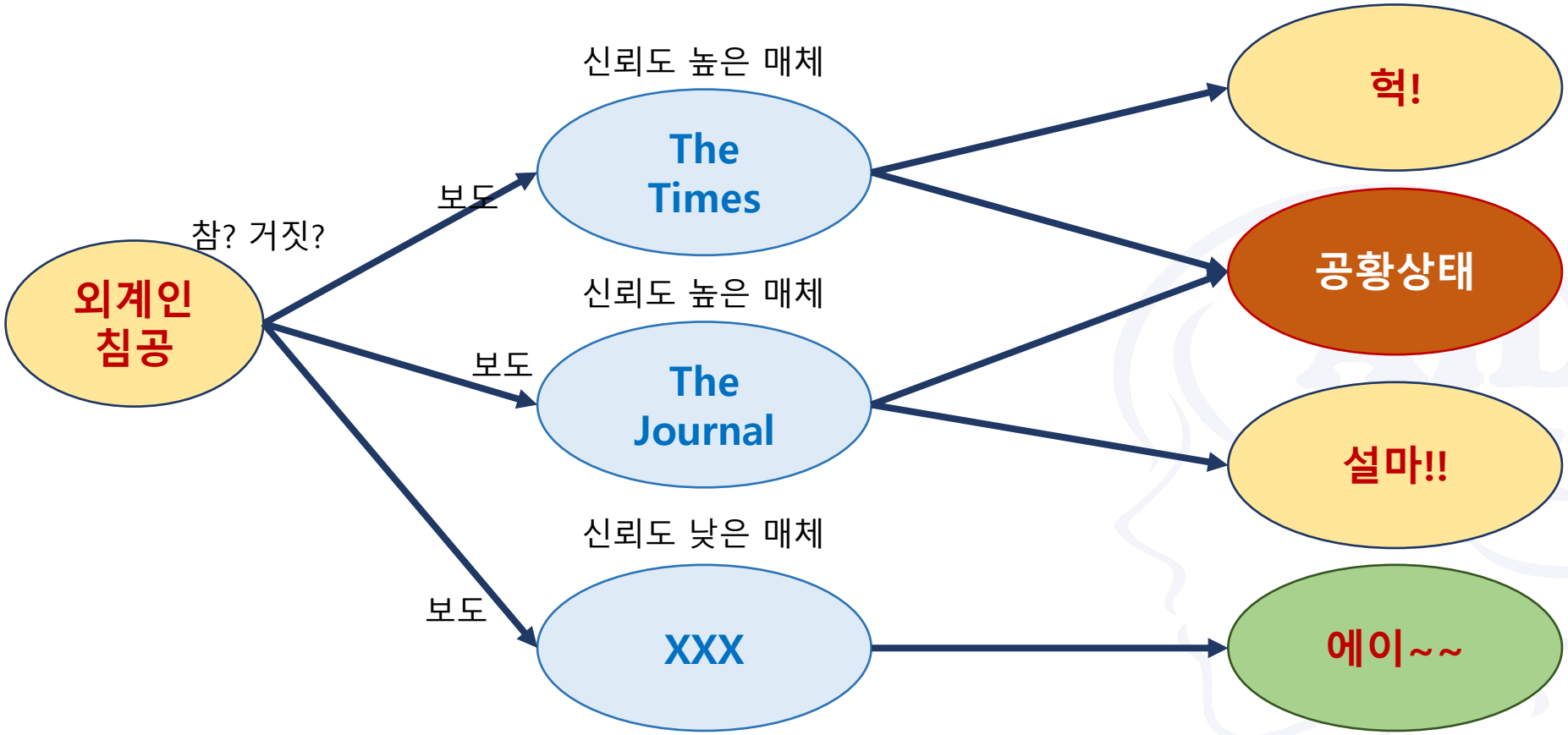
- 베이즈 정리를 좀 쉽게 정리한다면...
  - 어떤 원인에서 어떤 결과가 일어날 가능성이 더 높을 수록

    → 그 결과가 나타났을 때 그것이 원인일 가능성이 더 높다 (확률적인 기반)
  - 원인과 결과, 즉 인과관계에 대한 추론을 기반으로 학습, 예측을 진행함

- 그런데... 인간도 언어 추리가 연관될 경우, 베이즈 추론(베이즈 정리를 기반으로 하는 추론)을 매우 잘 하는 것은 아니다. 인간은 원인의 사전 확률을 무시하는 경향이 있다.

나이브 베이즈(Naïve Bayes)

- 베이즈 추론의 쉬운 예시



나이브 베이즈(Naïve Bayes)

- **MAP (Maximum a Posterior, 사후 확률 최대화)**
  - **베이즈 정리를 활용하면 조건부 확률은 다음과 같다.**

$$P(c|D) = \frac{P(D|c)P(c)}{P(D)} = \frac{P(D|c)P(c)}{\sum_{i=1}^{|C|} P(D|c_i)P(c_i)}$$

수식	영어 명칭	한글 명칭
$P(c\|D)$	posterior	사후 확률
$P(D\|c)$	likelihood	가능도(우도)
$P(c)$	prior	사전 확률
$P(D)$	evidence	증거

나이브 베이즈(Naïve Bayes)

- 풀고자 하는 대부분의 문제에서 $P(D)$는 구하기 어려우므로 $P(A|B) \propto P(B|A) \times P(A)$ 를 통해 접근하기도 함

$P(c|D)$ 자체는 클래스 c에 관한 함수

- 위의 성질을 이용하여 주어진 데이터 D를 만족하며 확률을 최대로 하는 클래스 c를 구할 수 있음

- 이처럼 사후 확률을 최대화하는 클래스 c를 구하는 방법을 **사후확률 최대화 (Maximum a Posterior, MAP)** 라고 함

$$\hat{c}_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(C = c|D)$$

수식의 내용:
D(데이터)가 주어졌을때 가능한 클래스의 집합 c 중에서 사후 확률을 최대로 하는 클래스 D를 선택하기

나이브 베이즈(Naïve Bayes)

- **MLE (Maximum Likelihood Estimation, 최대우도(가능도)추정)**

$$\hat{c}_{MLE} = \underset{c \in C}{\operatorname{argmax}} P(D|C = c)$$

  - 데이터 D 가 나타날 가능도(우도)를 최대로 하는 클래스 D 를 선택하는 것

  - MLE는 주어진 데이터 D와 클래스 레이블 c가 있을 때, 확률 분포를 근사하기 위한 함수 파라미터 θ를 훈련하는 방법으로도 사용됨

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}} P(C|D, \theta)$$

나이브 베이즈(Naïve Bayes)

- **MLE vs MAP**
  - **MAP는 사전 확률이 이미 반영되어 있기 때문에 경우에 따라 MLE보다 더 정확할 수 있다.**
  - **예시**
    - 범죄 현장에서 범인의 것으로 의심되는 발자국을 발견하고 신발 크기(데이터, x)를 측정한 결과, 235로 나타났다. 범인의 성별(클래스, y)을 예측해보자.
      - 성별 클래스의 집합: $Y = \{male, female\}$
      - 신발 크기의 집합: $\mathrm{X} = \{\cdots, 225, 230, 235, 240, \cdots\}$

나이브 베이즈(Naïve Bayes)

- 신발크기 235는 남자신발 크기 치고는 작은 편 → 범인을 여자라고 특정해 볼 수 있음
  - 남자일 때 신발크기 235일 확률 $P(X = 235|y = male\}$은 여자일 때 신발크기 235일 확률 $P(X = 235|y = female\}$보다 낮다.
  - 그러나 보통 남녀 비율은 0.5로 같기 때문에 사실상 큰 관련이 없는 예측이 된다.
  - 그런데 만약 범죄 현장이 대부분 남자들로 구성된 군부대라면? 사전확률
    → 남녀의 성비는 $P(y = male) > P(y = female)$로 매우 불균형한 값이 된다.
  - 이때, 이미 가지고 있는 가능도에 사전 확률을 곱해주면 사후 확률을 최대화하는 클래스를 더 정확하게 예측할 수 있다.

$$P(y = male|x = 235) > P(y = female|x = 235),$$

$$if\ P(x = 235|y = male)P(y = male) > P(x = 235|y = female)P(y = female)$$

- 신발 크기가 235라고해도 사전확률 자체가 매우 불균형하다면 여성이 범인이라고 예측할 수 없다.

나이브 베이즈(Naïve Bayes)

- 나이브 베이즈
  - 나이브 베이즈는 MAP를 기반으로 동작함
  - 대부분의 경우, 사후 확률을 바로 구하기 어렵기 때문에 가능도와 사전 확률의 곱을 통해 클래스 y를 예측함

사후 확률

  - n개의 단어 $w_1, w_2, \cdots, w_n$가 주어졌을때, 문장이 c 클래스에 속할 확률값

$$P(y = c | x = w_1, w_2, \cdots, w_n)$$

나이브 베이즈(Naïve Bayes)

- 만약 x가 복잡한 특징으로 이루어진 데이터라면

  → 훈련 데이터에서 매우 희소할 것이다

  → 사후 확률뿐만 아니라 가능도 $P(x = w_1, w_2, \cdots, w_n | y = c)$를 구하는 것도 어려울 것이다

- 일반적으로 문장에서 확률은 코퍼스의 출현빈도를 통해 추정

  → 특징이 복잡할수록(문장이 복잡하거나 길어질수록) 가능도 도는 사후 확률을 만족하는 경우가 매우 드물 것

- 그렇다고 해서 코퍼스에 없는 특징의 조합이라는 이유로 확률값을 0으로 추정하는 것도 지나친 가정이 됨

나이브 베이즈(Naïve Bayes)

- **이때, 나이브 베이즈를 적용한다면?**
  - 각 특징이 모두 독립적이라고 가정 →

    각 특징의 결합 확률을 각 독립된 확률의 곱으로 근사할 수 있음

$$P(y = c|x = w_1, w_2, \cdots, w_n) \propto P(x = w_1, w_2, \cdots, w_n|y = c)P(y = c)$$

$$\approx P(w_1|c)P(w_2|c) \cdots P(w_n|c)P(c)$$

$$= \prod_{i=1}^{n} P(w_i|c)P(c)$$

  - MAP을 활용한 클래스는 다음과 같은 사후 확률을 최대화하는 클래스가 됨

    → 나이브 베이즈의 가정에 따라 각 특징들의 확률의 곱에 사전 확률을 곱한 값을 최대화하는 클래스와 같을 것

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} P(y = c|x = w_1, w_2, \cdots, w_n) \approx \underset{c \in C}{\operatorname{argmax}} \prod_{i=1}^{n} P(w_i|c)P(c)$$

나이브 베이즈(Naïve Bayes)

- 이때 사용되는 사전 확률은 실제 데이터(코퍼스)에서 출현한 빈도를 통해 추정할 수 있음

$$P(y = c) = \frac{Count(c_i)}{\sum_{i=1}^{|C|} Count(c_i)}$$

- 특징 별 가능도 확률도 데이터에서 바로 구할 수 있음
- 만약 모든 특징의 조합이 데이터에 실제로 나타난 횟수를 통해 확률을 구하려 했다면 희소성 문제때문에 구할 수 없었을 것이다.
- 그러나 각 특징이 독립적이라는 나이브 베이즈의 가정을 통해서 데이터 또는 코퍼스에서의 출현 빈도를 쉽게 활용할 수 있다.

$$P(w|c) = \frac{Count(w, c)}{\sum_{j=1}^{|V|} Count(w_j, c)}$$

간단한 가정으로 데이터의 희소성 문제를
해결하는 쉬우면서도 강력한 방법을 통해
사후 확률을 최대화하는 클래스를 예측할 수 있음

나이브 베이즈(Naïve Bayes)

- 감성 분석
  - 감성 분석은 가장 많이 활용되는 텍스트 분류 기법
  - 사용자의 댓글이나 리뷰 등을 긍정 또는 부정으로 분류하여 마케팅이나 서비스 향상에 활용하는 방법

$$C = \{pos, neg\} \quad , \qquad D = \{d_1, d_2, \cdots\}$$

  - 긍정(pos)과 부정(neg)으로 클래스의 집합 C가 구성되어 있고, 문서 d로 구성된 데이터 D가 있음

나이브 베이즈(Naïve Bayes)

- "I am happy to see this movie!"라는 문장이 주어진다면 이 문장은 긍정인가? 부정인가?
  - $P(pos|I, am, happy, to, see, this, movie, !)$

$$= \frac{P(I, am, happy, to, see, this, movie, !|pos)P(pos)}{P(I, am, happy, to, see, this, movie, !)}$$

$$= \frac{P(I|pos)Pam|pos)P(happy|pos) \cdots P(!|pos)P(pos)}{P(I, am, happy, to, see, this, movie, !)}$$

나이브 베이즈(Naïve Bayes)

- 나이브 베이즈를 활용하여 단어의 조합에 대한 확률을 각각 분해할 수 있다

  → 각 단어의 출현 확률을 독립적이라고 가정한 후에 결합 가능도 확률을 모두 각각의 가능도 확률로 분해함

$$P(happy|pos) = \frac{Count(happy, pos)}{\sum_{j=1}^{|V|} Count(w_j, pos)}$$

$$P(pos) \approx \frac{Count(pos)}{|D|}$$

나이브 베이즈(Naïve Bayes)

- 부정의 감성도 동일한 방법으로…
  - $P(neg|I, am, happy, to, see, this, movie, !)$

$$= \frac{P(I, am, happy, to, see, this, movie, !|neg)P(neg)}{P(I, am, happy, to, see, this, movie, !)}$$

$$= \frac{P(I|neg)Pam|neg)P(happy|neg)\cdots P(!|neg)P(neg)}{P(I, am, happy, to, see, this, movie, !)}$$

- $P(happy|neg) = \frac{Count(happy,neg)}{\sum_{j=1}^{|V|} Count(w_j,neg)}$
- $P(neg) \approx \frac{Count(neg)}{|D|}$

코퍼스에서 각 단어의 클래스 당 출현 빈도를 계산하는 것만으로 간단하게 감성 분석을 수행할 수 있음

나이브 베이즈(Naïve Bayes)

- 문제점
  - 나이브 베이즈 가정을 통해서 각 단어가 출현할 확률을 독립으로 만들어 코퍼스에서 출현 횟수를 적극적으로 활용할 수 있게 되었음
  - 그러나 훈련 데이터에서 $Count(happy, neg)$가 0이었다면
  - $P(happy, neg) = 0$ 이 되어 출현확률을 0으로 추정하는 문제가 발생
  - 각 출현 횟수에 1을 더해주어 문제 해결 가능(add-one Smoothing)

$$\hat{P}(w|c) = \frac{Count(w, c) + 1}{\sum_{j=1}^{|V|} Count(w_j, c) + |V|}$$

나이브 베이즈(Naïve Bayes)

- 장점과 한계
  - ‘I am not happy to see this movie!”라는 문장이 주어진다면?
  - ‘not’ 하나가 추가되었을 뿐이지만 문장의 뜻은 정반대가 됨
    - $P(pos|I, am, npt, happy, to, see, this, movie, !)$
    - $P(neg|I, am, not, happy, to, see, this, movie, !)$
  - ‘not’은 ‘happy’를 수식하므로 두 단어를 독립적으로 보는 것은 옳지 않음
    - $P(not, happy) \neq P(not)P(happy)$
  - 문장은 단어들이 순서대로 나타나서 의미를 이루는 것이므로, 단어간의 순서로 인해 생기는 관계와 정보를 무시할 수 없음

나이브 베이즈(Naïve Bayes)

- 나이브 베이즈의 가정은 이러한 언어의 특징을 단순화하여 접근하므로 한계가 발생함
- 그러나 딥러닝을 활용하기에 레이블 당 문장 수가 매우 적은 경우라면

  → 복잡한 딥러닝보다 나이브 베이즈와 같은 간단한 방법을 사용하는 것이 좋은 대안이 될 수 있음

RNN을 활용한 텍스트 분류

- 문장은 단어들의 시퀀스로 이루어진 시퀀셜 데이터
  - 각 위치(또는 time-step)의 단어들은 다른 위치의 단어들과 서로 영향을 주고 받음
  - RNN은 이러한 문장의 특징을 가장 잘 활용할 수 있는 신경망 구조

RNN을 활용한 텍스트 분류

- RNN 모델은
  - 각 time-step의 단어를 입력으로 받아 자신의 은닉 상태를 업데이트

$$h_t = f_\theta(x_t, h_{t-1})$$

  - n개의 단어로 이루어진 문장 x → RNN → n개의 은닉상태 확보 가능
  - 가장 마지막 은닉 상태를 활용하여 텍스트의 클래스 분류

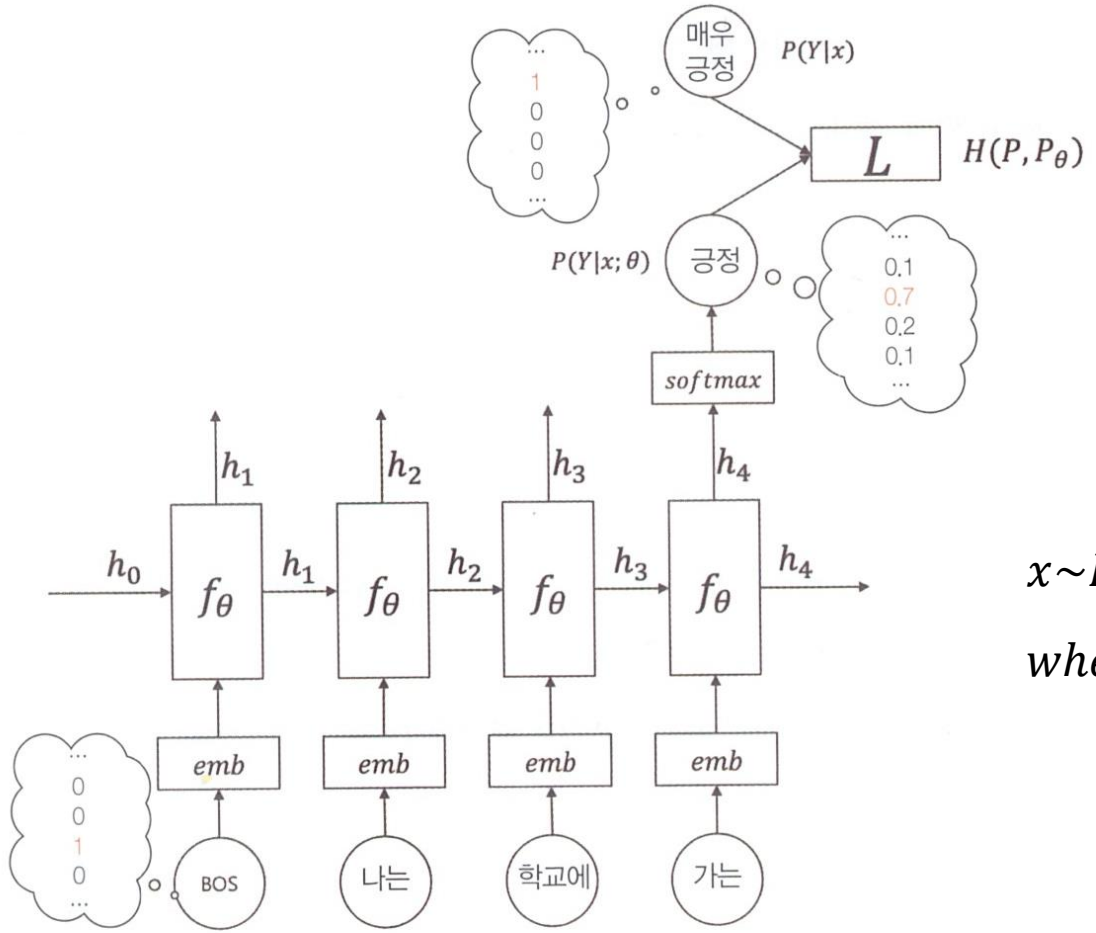
$$\hat{y} = \underset{y \in Y}{\boldsymbol{argmax}}\, P(y|x;\theta)$$

$$where\ P(y|x;\theta) = h_n = f_\theta(x_n, h_{n-1})\ and\ x = \{w_1, w_2, \cdots, w_n\}$$

RNN은 입력으로 주어진 문장을 분류 문제에 맞게 인코딩 → RNN의 출력값은 문장 임베딩 벡터

RNN을 활용한 텍스트 분류

- **RNN을 통한 텍스트 분류 구조**



▶ RNN의 마지막 time-step의 출력에 softmax를 씌워 분류를 수행합니다.

- 단어는 불연속적인 값 → 문장도 결국 불연속
- 즉, 이산 확률 분포에서 문장을 샘플링한 것
- 입력: 원핫벡터들이 여러 time-step으로 주어짐
- 미니배치를 고려하면 → 입력은 3차원 텐서

  → 크기는 $n \times m \times |V|$

  (n : 미니배치 크기, m :문장의 길이)

$$x \sim P(x)$$

$$where\ x = \{w_1, w_2, \cdots, w_m\}\ and\ w_i \in \{0,1\}^{|V|}\ and\ |w_i| = |V|$$

$$Thus, |x_{1:n}| = (n, m, |V|)$$

$$where\ x_{1:n} = [x_1, x_2, \cdots, x_n]\ and\ n = batch\_size$$

RNN을 활용한 텍스트 분류

- 원핫 벡터는 주어진 $|V|$차원에 단 하나의 1과 $|V|-1$개의 0으로 구성됨

  → 효율성을 위하여 각 벡터 별 1의 위치 인덱스만 기억하고 나머지는 버림

$$|x_{1:n}| = (n, m, 1) = (n, m)$$

- 입력으로 주어진 원핫 인코딩된 $n \times m$텐서를 임베딩 계층에 통과시키면 단어 임베딩 텐서를 얻을 수 있음

$$\tilde{x}_{1:n} = emb_\theta(x_{1:n})$$

$$|\tilde{x}_{1:n}| = (n, m, d) where\ d = word\_vec\_dim$$

단어 임베딩 텐서의 크기

RNN을 활용한 텍스트 분류

- 다음으로 단어 임베딩 텐서를 RNN에 통과시키면

$$h_t = RNN_\theta(x_t, h_{t-1})$$

$$where\ |x_t| = (n, 1, d), |h_t| = (n, 1, h)\ and\ \ h = hidden\_size$$

- RNN에 대하여 각 time-step 별, 계층별로 구분하여 단어 임베딩 텐서 또는 은닉 상태를 넣어줄 필요는 없음

  → 초기 은닉상태인 h_0와 전체 입력(단어 임베딩 텐서 $x_{1:n}$)을 RNN에 넣어주면 RNN 모듈에서 모든 time-step에 대한 출력과 마지막 은닉상태를 반환함

RNN을 활용한 텍스트 분류

$$H = RNN_\theta(x_{1:n}, h_0)$$

$$where\ H = [h_1; h_2; \cdots; h_m]\ and\ |H| = (n, m, h)$$

- RNN을 통해 얻은 모든 time-step에 대한 RNN의 출력값 중에서 제일 마지막 time-step만 선택하여 softmax 계층을 통과시켜 이산 확률 분포 $P(y|x;\theta)$로 나타냄

$$h_m = H[:, -1]$$

$$\hat{y} = softmax(h_m \cdot W + b)$$

$$where\ |\hat{y}| = (n, |C|), |h_m| = (n, 1, h), W \in \mathbb{R}^{h \times |C|}\ and\ b \in \mathbb{R}^{|C|}$$

RNN을 활용한 텍스트 분류

- 계산된 $\hat{y}$는 데이터 x와 확률 분포 함수 파라미터 θ가 주어졌을 때, 클래스를 나타내는 확률 변수 y의 확률 분포를 나타냄
- 정답 y와 예측값 $\hat{y}$의 차이에 대한 손실 값을 구하고, 이를 최소화 하도록 경사하강법을 통한 최적화를 수행하면 신경망 θ를 훈련시킬 수 있음

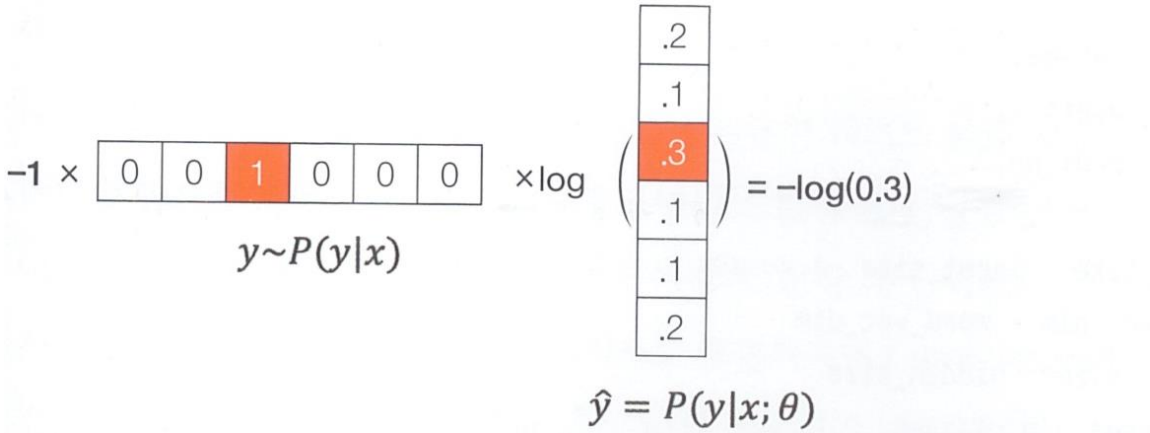
$$\mathcal{L}(\hat{y}, y) = -\frac{1}{n}\sum_{i=1}^{n} y_i \, log \hat{y}_i$$

- 위와 같이 교차 엔트로피 수식을 통해 실제 확률 분포에 학습시킨 신경망 확률 분포 함수가 근사하도록 적용

RNN을 활용한 텍스트 분류

- y_i가 원핫벡터이므로 1의 인덱스의 로그 확률 값만 최대화하면 됨

  → softmax의 수식에 따라 다른 인덱스의 확률값이 작아짐



▶ 원핫 벡터로 구성된 정답 샘플과 신경망을 통해 얻은 이산 확률 분포 사이의 손실 함수 계산

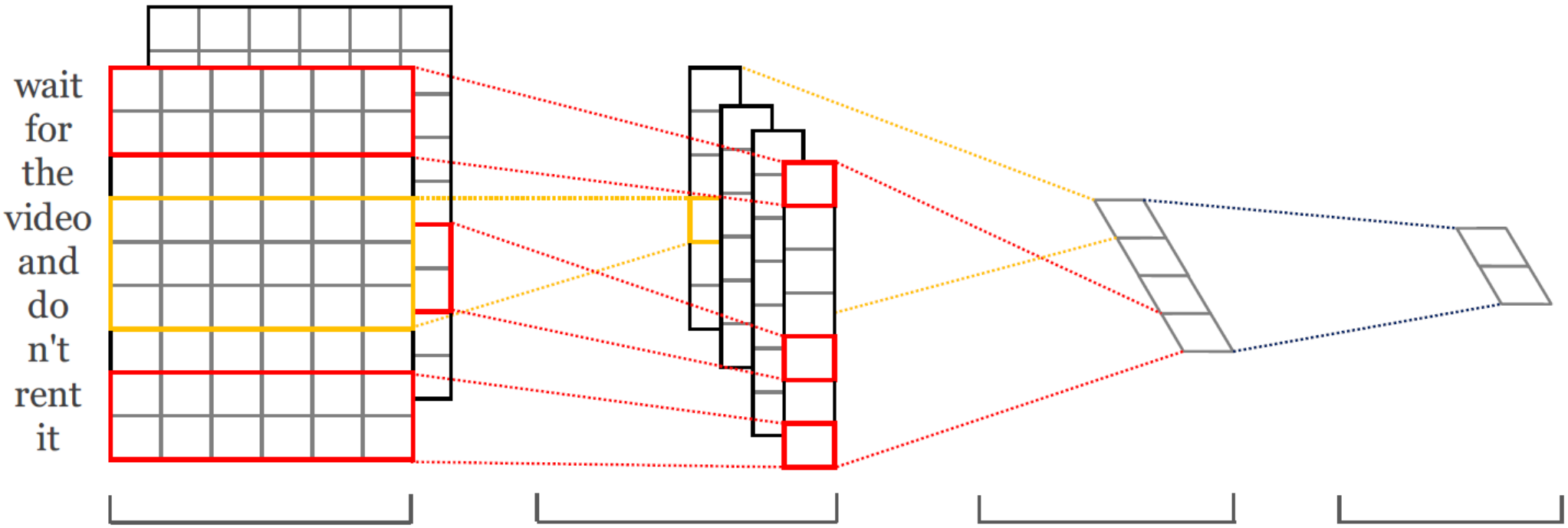
RNN을 활용한 텍스트 분류

- 계산된 손실 함수에 대해 확률 분포 함수 신경망 파라미터 θ로 미분하면 가능도를 최대화하는 θ를 업데이트 할 수 있음

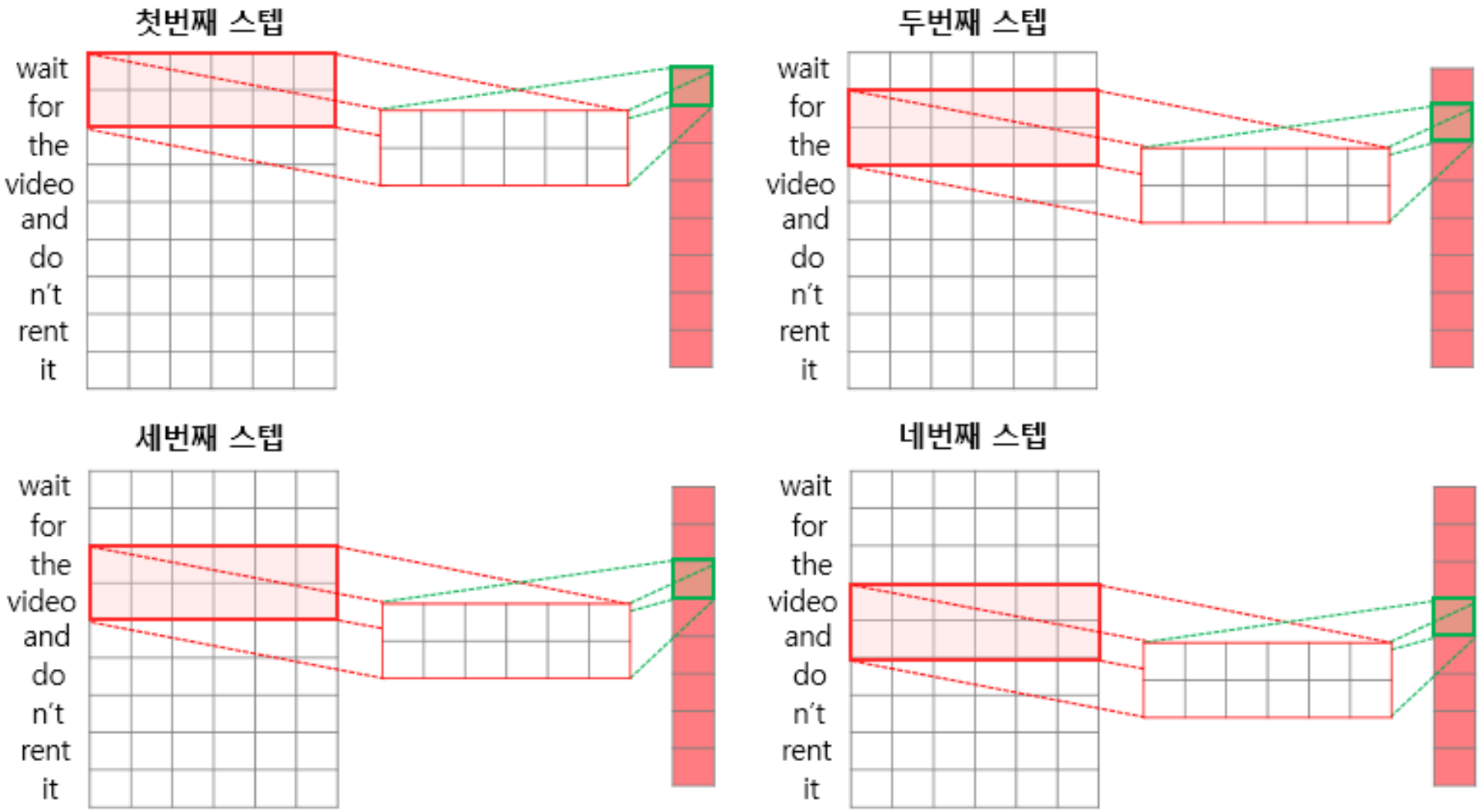
$$\theta \leftarrow \theta - \lambda \nabla_{\theta} \mathcal{L}(\hat{y}, y)$$

CNN을 활용한 텍스트 분류

- **CNN을 활용한 텍스트 분류 아키텍처**



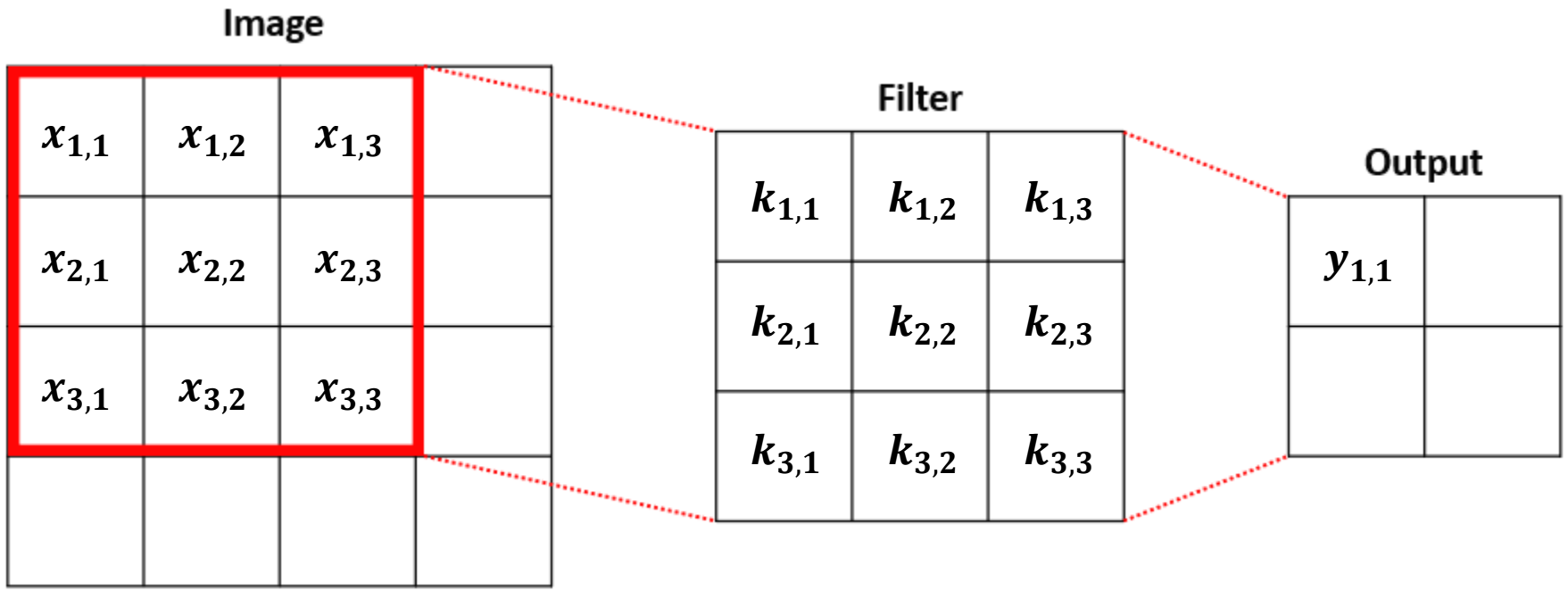
CNN을 활용한 텍스트 분류



CNN을 활용한 텍스트 분류

- 기존의 영상처리 기술
  - 사람이 직접 필터를 개발, 생성한 후 개발된 필터를 이용하여 윤곽선 검출 등의 전처리 과정을 수행한 뒤
  - 검출된 특징을 통해 객체 탐지 등의 동작을 직접 구현함
- 각 문제 별, 단계 별로 필요한 합성곱 필터를 자동으로 찾아준다면?
  - CNN의 역할
  - 역전파를 통해 더 나은 합성곱 필터 값 검색, MLE를 통한 최적화 수행 후
  - 해당 데이터셋의 특징을 잘 추출하는 여러 종류의 합성곱 필터 확보 가능

CNN을 활용한 텍스트 분류



- **합성곱 필터 연산의 피드포워드 연산**

$$y_{1,1} = Convolution(x_{1,1}, \cdots, x_{3,3}, \theta) \quad where\ \theta = \{k_{1,1}, \cdots, k_{3,3}\}$$
$$= x_{1,1} * k_{1,1} + \cdots + x_{3,3} * k_{3,3}$$
$$= \sum_{i=1}^{3} \sum_{j=1}^{3} x_{i,j} * k_{i,j}$$

필터가 주어진 이미지에서 차례로 합성곱 연산 수행

CNN을 활용한 텍스트 분류

- 합성곱 연산의 결과물은
  - 기본적으로 필터의 크기에 따라 입력보다 크기가 줄어든다
  - 입력과 같은 크기를 유지하고자 한다면 → 결과물의 바깥쪽에 패딩 추가

$$output_{size} = input_{size} - fileter_{size} + 1$$

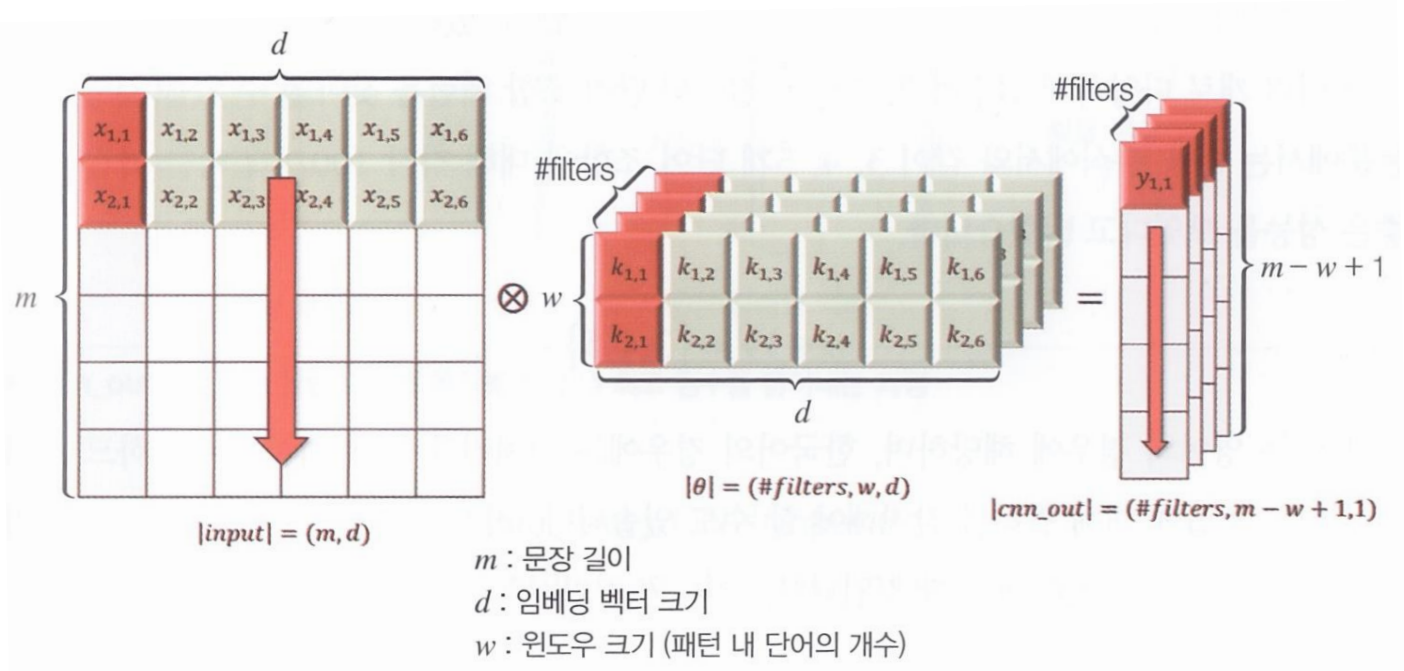
$$for\ example, \qquad if\ y = CNN(x, k),$$

$$y \in \mathbb{R}^{4\times4} \quad where\ x \in \mathbb{R}^{6\times6} \quad and \quad k \in \mathbb{R}^{3\times3}$$

  - 이런 과정을 반복하면서 필터를 자동으로 최적화 함 → 영상, 음성 등에 적용

CNN을 활용한 텍스트 분류

- 텍스트 분류에 CNN 적용하기



1. 원핫벡터를 표현하는 인덱스 값을 단어 임베딩 벡터로 변환
   → 1차원 벡터

2. 문장 내의 모든 time-step의 단어 임베딩 벡터를 합치면
   → 2차원 행렬

3. 여기에 합성곱 연산 수행

CNN을 활용한 텍스트 분류

- 각 텐서별 크기에서 맨 앞에 미니배치를 위한 차원을 추가하면

  → 실제 구현에서의 텐서 크기

$$cnn_{out} = CNN(input, \theta)$$

$$|input| = (n, m, d) = (n, 1, m, d)$$
$$|\theta| = (\#filters, w, d)$$
$$|cnn_{out}| = \left(n, \#filters, m - w + 1, \underbrace{1}_{d-d+1}\right),$$
$$where\ n = batch\_size$$

- 이런 식으로 정해진 길이 w의 단어 조합 패턴 검사 가능
- w를 한 개가 아닌 여러 개로 다양하게 설정하면
  → 다양한 길이의 단어 조합 패턴 검출 가능

- 필터: 신경망의 가중치 파라미터가 되므로
  → θ로 표현
- input → 임베딩 계층을 거친 결과값
- n개의 문장이 각각 m개의 time-step을 가지고 있음
- 각 time-step은 문장 내 단어
  → d차원의 벡터로 표현
- 각 합성곱 연산을 위한 필터
  → 찾고자 하는 w개의 단어에 대한 패턴을 찾기 위한 d차원의 $w \times d$ 크기를 가짐

CNN을 활용한 텍스트 분류

- CNN을 활용한 텍스트 분류 논문에서는
  - 3, 4, 5개의 단어 조합에 대하여 각각 100개의 필터를 가질 경우 좋은 성능을 보였음을 발표 → 그러나 영문의 예시
  - 한국어의 경우, 전처리를 통해 접사를 분리하므로
    → 더 긴 길이의 조합에 대하여 필터를 감지해야 할 수도 있음
  - 최적의 조합은 주어진 문제에 따라 파라미터 튜닝 작업을 통해 찾아내야 함

CNN을 활용한 텍스트 분류

- **CNN 계층의 결과 값 → 필터별 점수**
- **필터가 각각의 특징을 나타내므로 →각 특징별 점수**

  → 문장 내에서 각 특징 또는 원하는 단어 조합 패턴의 등장을 확인해야 함

  ➔ 이전 단계에서 구한 cnn_out → Max Pooling

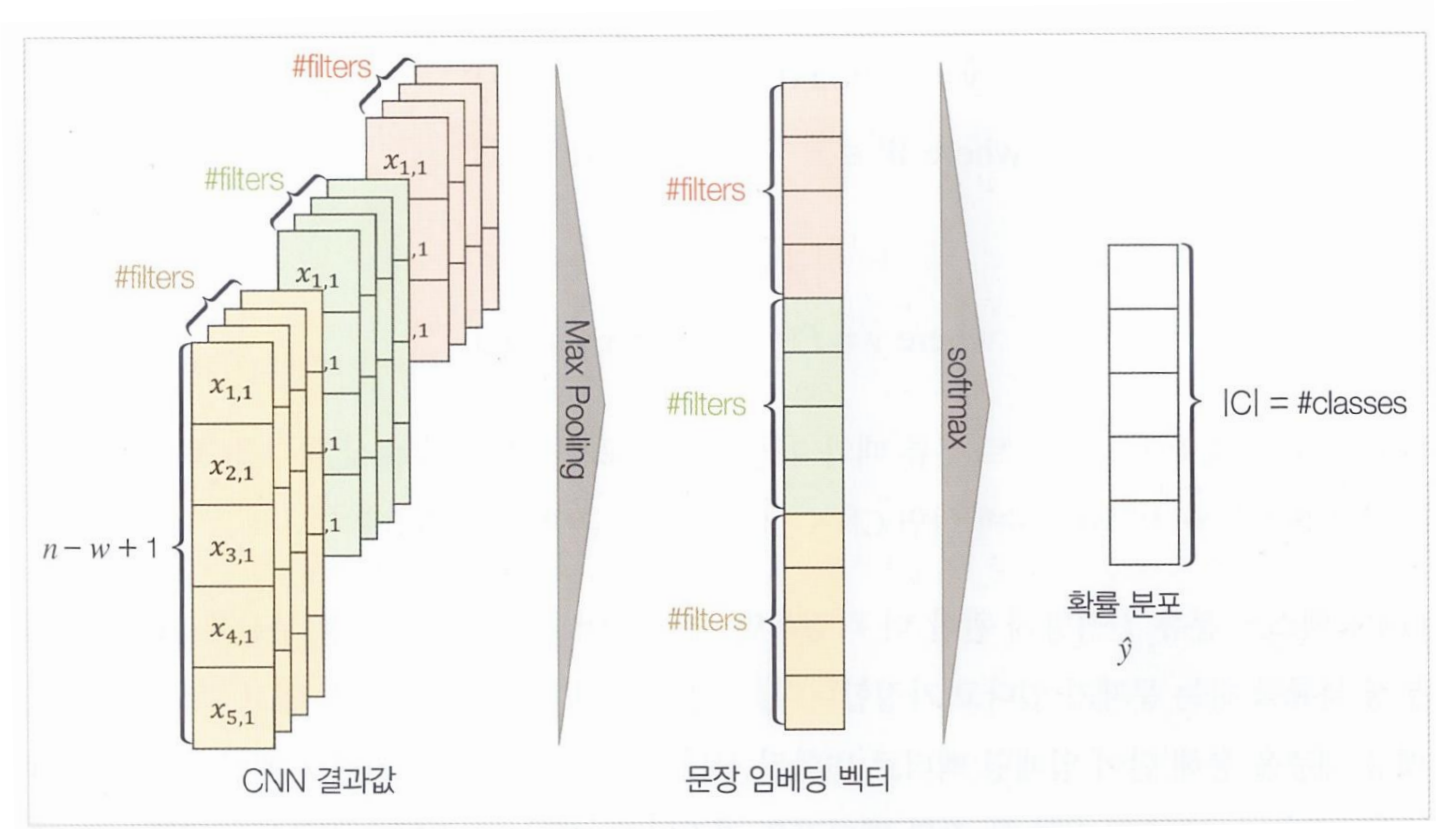
  → 문장당 각 특징에 대한 최고 점수 계산

  ➔ Max Pooling:

  - 특징별 최고 점수 계산
  - 이 과정에서는 가변 길이의 cnn_out을 고정 길이로 바꿔주는 역할도 수행

CNN을 활용한 텍스트 분류

- **Max Pooling 계층의 결과 → 문장의 임베딩 벡터**
  - **임베딩 벡터의 크기 = 특징의 개수**



- **cnn_out에서 Max Pooling을 통해 특징별 문장 내 최고 점수를 뽑아내는 과정**

CNN을 활용한 텍스트 분류

$$cnn_{out_i} = CNN(input, \theta)$$

$$where\ |\theta_i| = (\#filters, w_i, d)\ and\ w_i \in \{w_1, w_2, \cdots, w_h\}$$

$$pool_{out_i} = max_{pooling}(cnn_{out_i})$$

$$|pool_{out_i}| = (n, \#filters)$$

$$pool_{out} = [pool_{out_1}; pool_{out_2}; \cdots; pool_{out_h}]$$

$$|pool_{out}| = (n, h \times \#filters)$$

CNN을 활용한 텍스트 분류

- 이렇게 얻어진 문장별 임베딩 벡터

  → Softmax를 통해 클래스별 확률 값을 가진 불연속적인 확률 분포 반환

  → 분류를 위한 피드포워드 동작은 종료

$$\hat{y} = softmax(pool_{out} \cdot W + b)$$

$$where\ W \in \mathbb{R}^{(h \times \#filters) \times |C|}\ and\ b \in \mathbb{R}^{|C|}$$

$$|\hat{y}| = (n, |C|)$$

$$where\ \hat{y} = P(y|x)\ and\ x = input$$

- RNN을 활용한 텍스트 분류와 마찬가지로
- 교차 엔트로피 손실 함수를 적용

  → 이를 최소화하도록 최적화 수행

  → CNN 신경망도 훈련 적용됨

CNN을 활용한 텍스트 분류

- **CNN 텍스트 분류 신경망에서**
  - **특정 문장에 대해 긍정/부정 분류를 수행하는 문제를 가정**
    - 문장은 여러 단어로 이루어짐
    - 각 단어는 임베딩 계층을 통해 단어 임베딩 벡터로 변환
    - 각 단어의 임베딩 벡터는 비슷한 의미를 가진 단어일수록 비슷한 값의 벡터값을 가짐
    - 예: 'good'이라는 단어를 임베딩하면 → 그에 해당하는 적절한 임베딩 벡터로 구성

      → 'better', 'best', 'great' 등의 단어도 비슷한 벡터값을 가짐
      - 'good'은 긍정/부정 분류에서 긍정을 나타내는 중요한 신호로 작용

        → 'good'에 해당하는 임베딩 벡터의 패턴을 감지하는 필터를 가진다면

        → 'better' 등의 단어도 함께 감지 가능, 단어들의 조합 패턴을 감지하는 필터들도 학습 가능

멀티 레이블 분류

- **멀티 레이블 분류**
  - **기존의 소프트맥스 분류와 달리 여러 개의 클래스가 동시에 정답이 될 수 있는 분류 문제**
  - **상품 품질은 참 좋은데, 배송이 너무 느리네요 → 긍정? 부정?**
    - 품질에 대한 반응은 긍정적이지만 배송에 대한 반응은 부정적 → 중립?
    - 만약 단순한 사용자의 반응이 아니라 특정 항목에 대한 사용자의 반응이 궁금하다면?
    - 품질에 대한 감성 분석과 배송에 대한 감성 분석이 동시에 필요한 경우라면?

      → 레이블링 작업 시점에서부터 각 항목에 대한 레이블을 각각 달아주어야 함

문장	품질	배송
상품 품질은 참 좋은데, 배송이 너무 느리네요.	상	하

멀티 레이블 분류

- **이진 분류**
  - **앞의 문제를 여러 개의 이진 분류로 접근하는 경우**
    - 어떤 활성화 함수, 손실 함수를 사용해야 하나?
  - **기존의 Softmax 분류와 이진 분류의 차이**
    - 신경망의 마지막 계층
      - 2개의 노드에 softmax함수를 사용하는 대신
      - 1개의 노드에 sigmoid함수를 사용 → 2개의 노드를 사용하지 않는 이유는?

        → 2진 분류이므로 하나가 정해지면 나머지는 자동으로 결정됨. 따라서 1개의 노드 사용

$$P(y = 1|x) = 1 - P(y = 0|x)$$

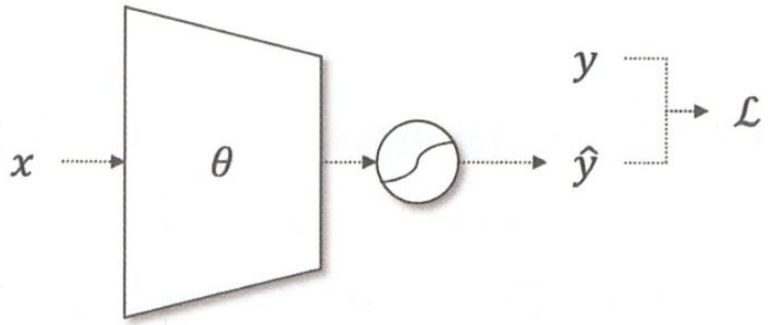
이진 분류 상황(또는 베르누이 분포)에서는
이 수식을 항상 만족함

멀티 레이블 분류

- **이진 분류에서는 이진 교차 엔트로피 손실(BCELoss) 함수를 사용할 수 있음**
  - BCELoss: 기존의 교차엔트로피 손실 함수 중 이진 분류에 특화된 버전

$$\boldsymbol{BCELoss}(\hat{\boldsymbol{y}}, \boldsymbol{y}) = -(\boldsymbol{y} \times \log \hat{\boldsymbol{y}} + (\boldsymbol{1} - \boldsymbol{y}) \times \log(\boldsymbol{1} - \hat{\boldsymbol{y}}))$$

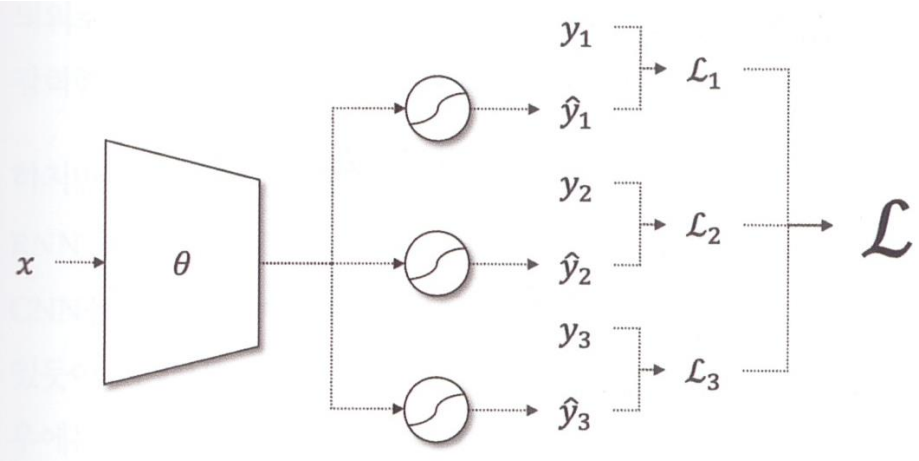
    - $\boldsymbol{y}$ 는 0 또는 1의 값을 가지는 불연속적인 값
    - $\hat{\boldsymbol{y}}$은 sigmoid 함수의 출력값 → 0~1 사이의 연속적인 실숫값



이진 분류 모델 신경망 구조의 예

멀티 레이블 분류

- 멀티 이진 분류
  - n개의 항목을 가지는 분류 문제에 대해서
  - 신경망의 마지막 계층에 n개의 노드를 주고
  - 모두 sigmoid 함수 적용



멀티 이진 분류 모델 신경망 구조의 예

  - 최종 손실함수

$$\mathcal{L} = \sum_{i=1}^{n} BCELoss\,(\hat{y}_i, y_i)$$

$$where\ BCELoss(\hat{y}, y) = -(y \times \log \hat{y}_i + (1 - y_i) \times \log(1 - \hat{y}_i))$$

멀티 레이블 분류

- 이진 분류가 아닌 경우
  - 예: 감성 분석을 '긍정', '부정', '중립' 과 같이 더 세밀하게 분류하는 경우

문장	품질	배송	가격
상품 품질은 참 좋은데, 배송이 너무 느리네요.	상	하	중
가격도 싸고, 품질도 너무 만족스럽네요!	상	중	상

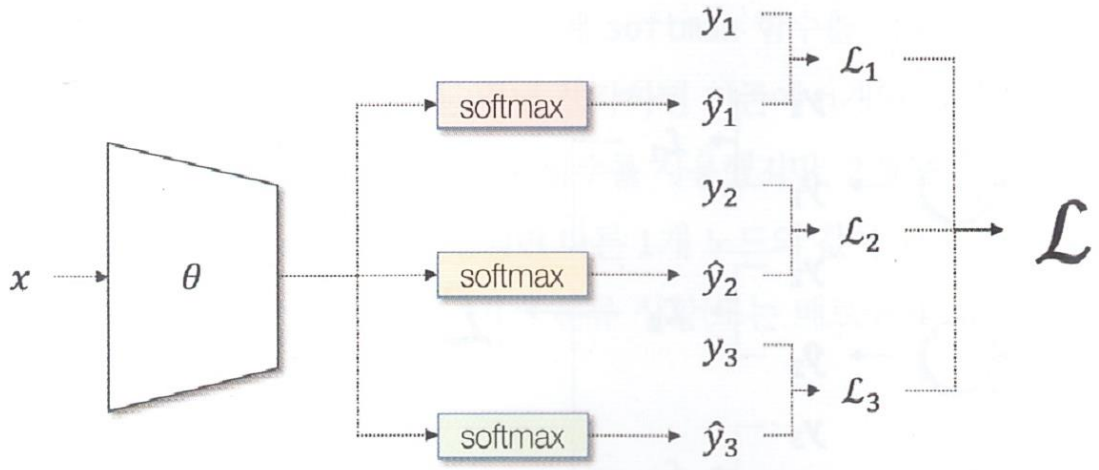
  - 여러 개의 softmax 계층이 요구됨
  - 최종적인 손실 함수

$$\mathcal{L} = \sum_{i=1}^{n} CrossEntropy\left(\hat{y}_i, y_i\right)$$

$$where\ \hat{y}_i \in \mathbb{R}^3, y_i \in \{0,1\}^3\ \ and\ \ |y_i| = 1$$

$$CrossEntropy(\hat{y}_i, y_i) = -y_i \times \log \hat{y}_i$$

$$= -\sum_{j=1}^{3} y_{i,j} \times \log \hat{y}_{i,j}$$



이진 분류가 아닌 경우에 대한 멀티 레이블 분류 신경망 구조의 예