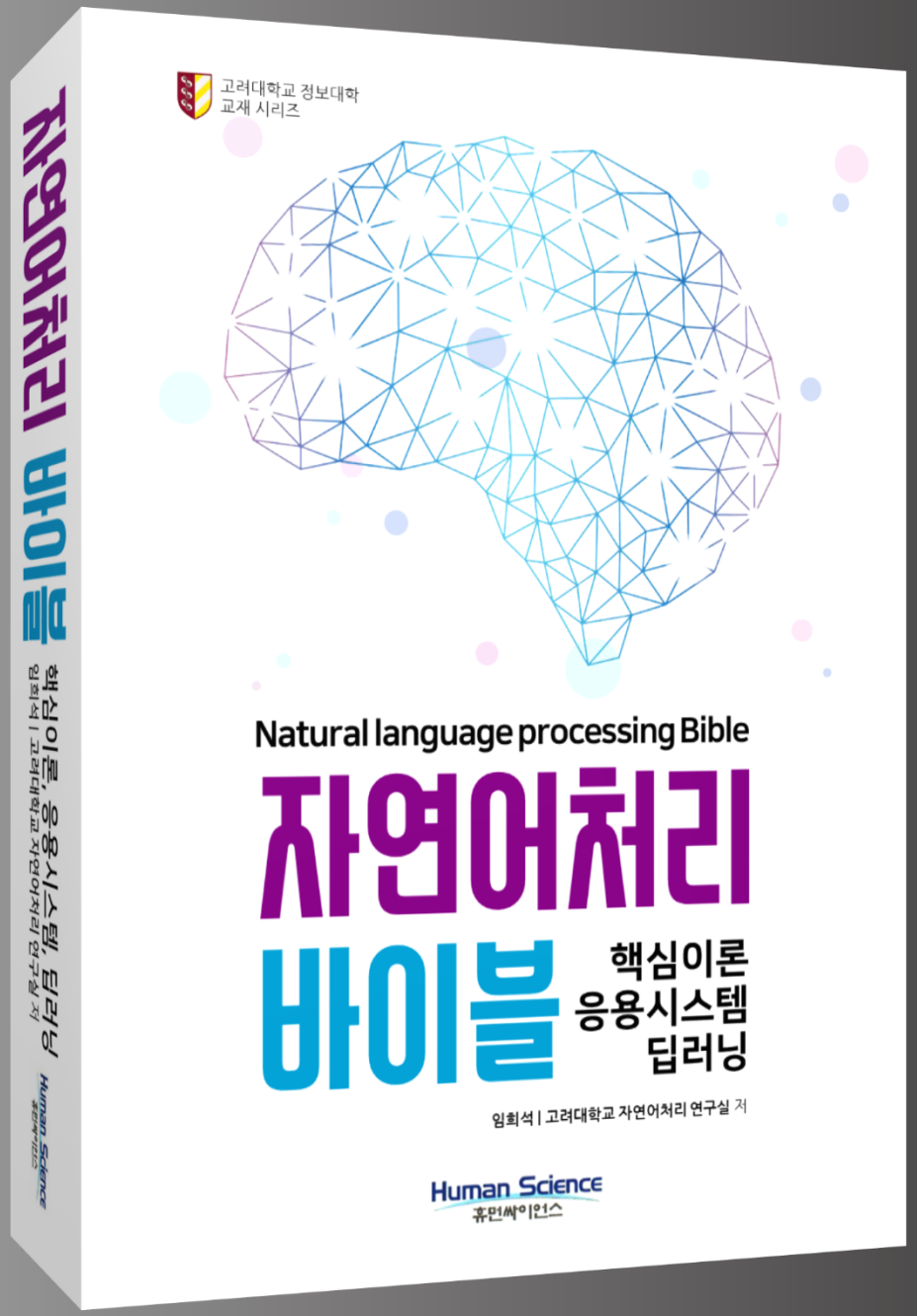
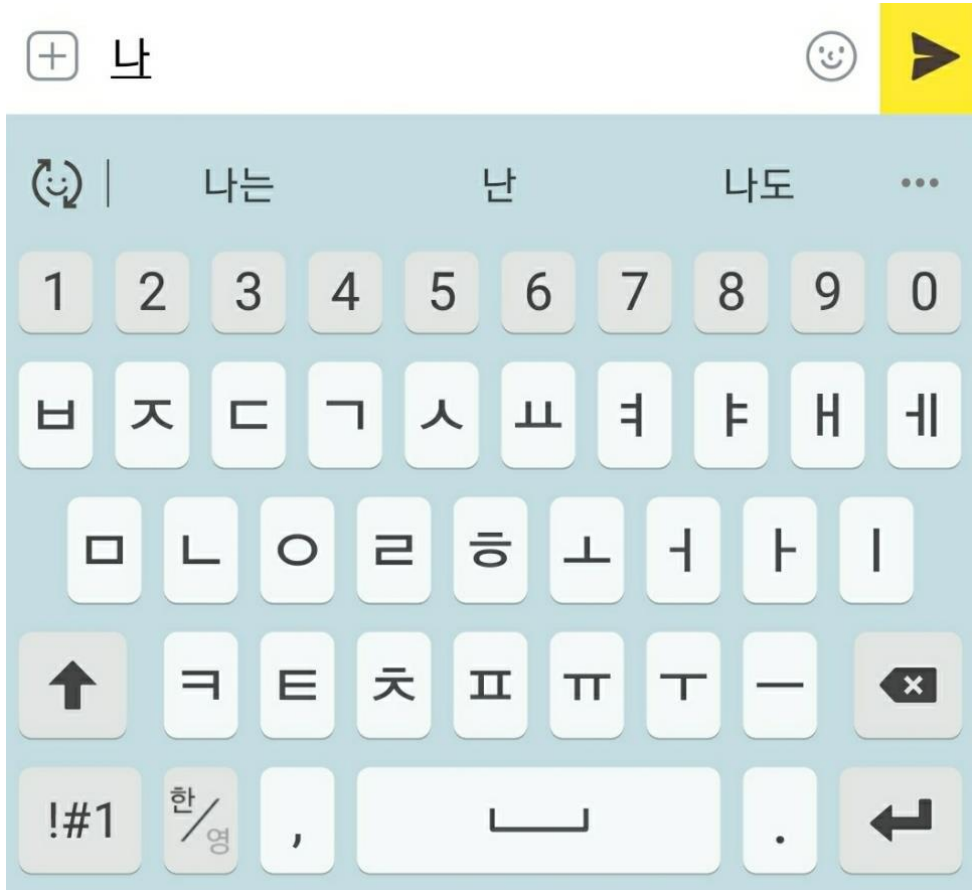


CHAPTER 9
언어 모델
(Language Model)

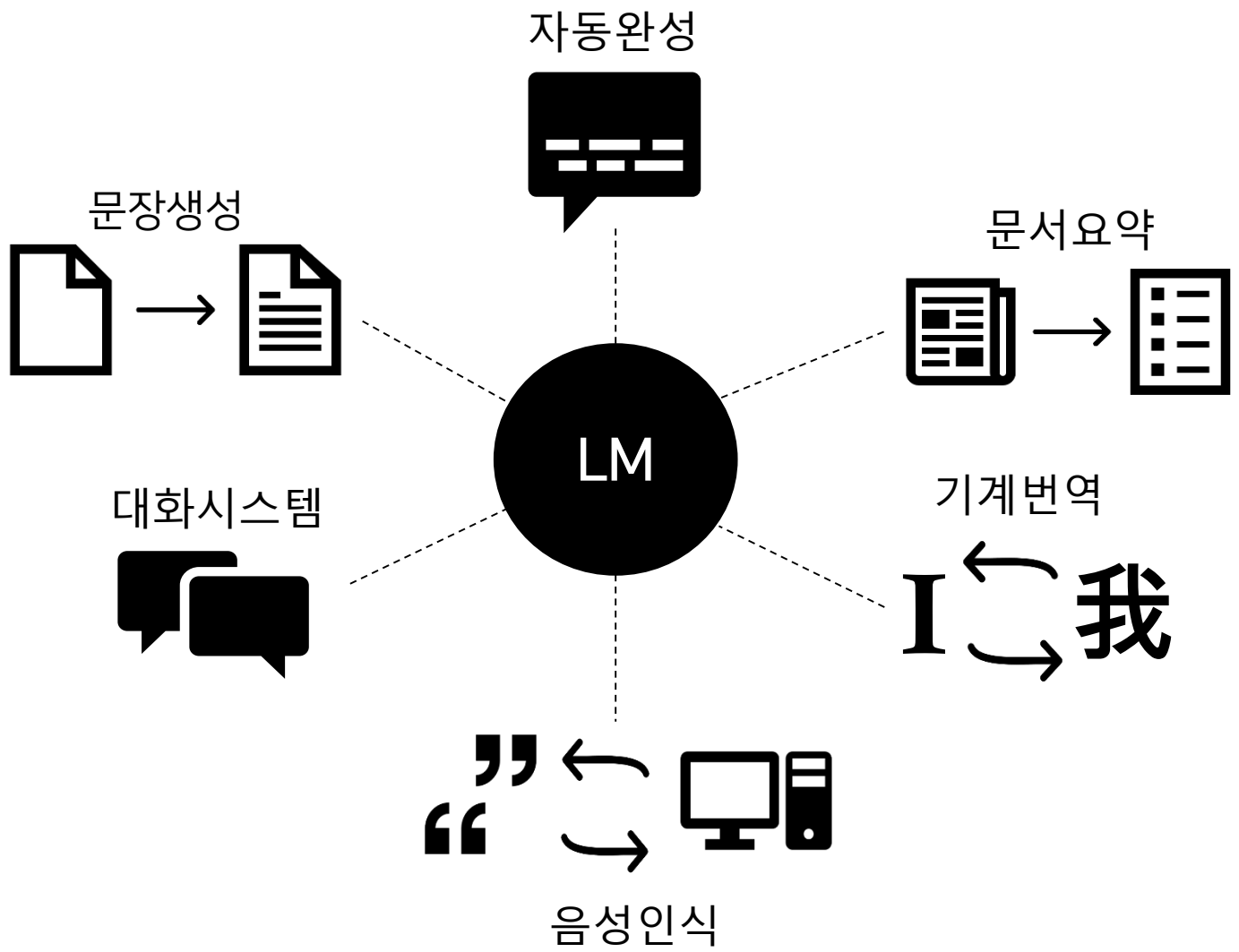
언어 모델이란?



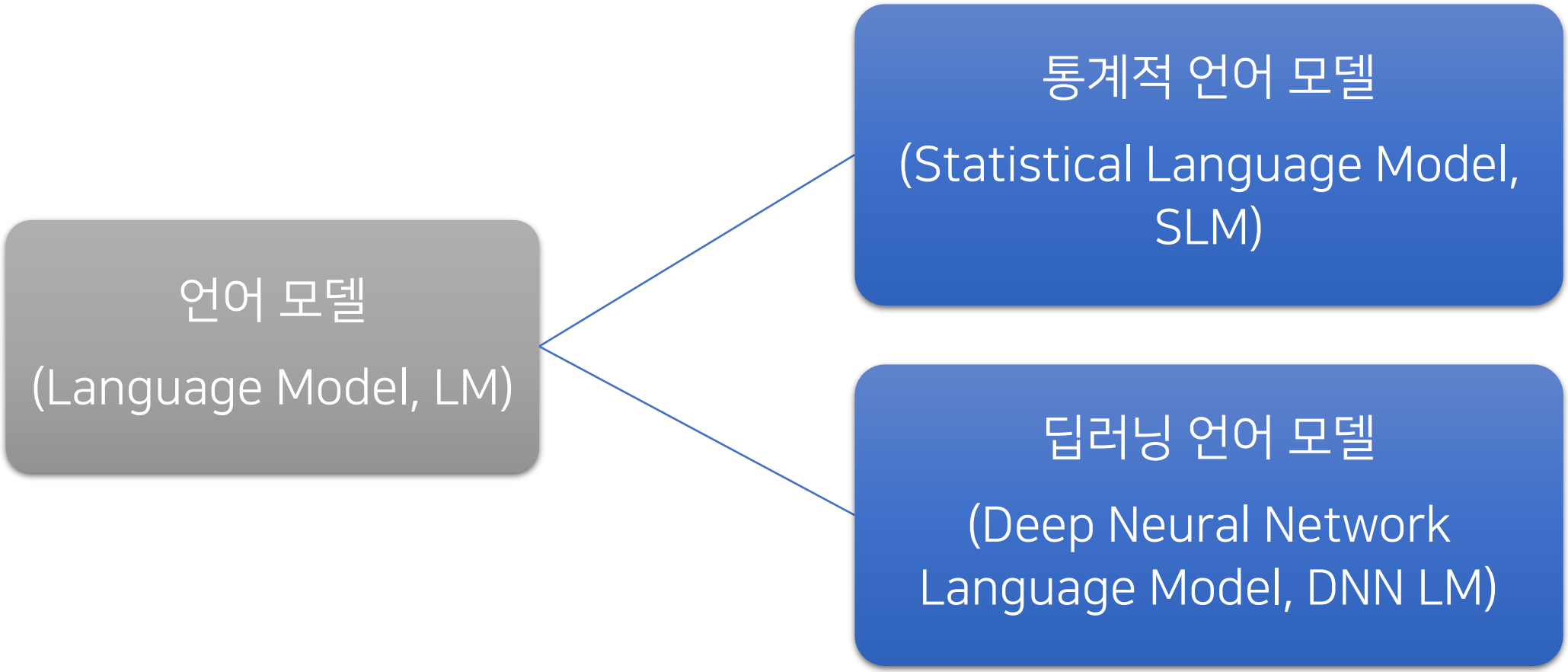
언어 모델(Language Model, LM)

: 언어를 이루는 구성 요소(글자, 형태소, 단어, 단어열 혹은 문장, 문단 등)를 문맥으로 하여 이를 바탕으로 다음 구성 요소를 예측하거나 생성하는 모델

언어 모델이란?



언어 모델이란?



통계적 언어 모델

- **주어진 문서(코퍼스) 내 단어열(혹은 문장)의 등장 확률**을 기반으로 각 단어의 조합을 예측하는 전통적인 언어 모델
- 실제로 많이 사용하는 단어열(문장)의 확률 분포를 정확하게 근사하는 것이 모델의 목표

통계적 언어 모델

- 조건부 확률(Conditional probabilities)과 언어 모델
  - 조건부 확률

조건부 확률 P(B|A): 사건 A가 일어났을 때 사건 B가 일어날 확률

언어 모델

단어 A가 등장했을 때 바로 다음에 단어 B가 등장할 확률

통계적 언어 모델

- 조건부 확률(Conditional probabilities)과 언어 모델
  - 조건부 확률과 결합확률

$$P(B|A) = \frac{P(A, B)}{P(A)}$$

$$P(A)P(B|A) = P(A, B)$$

$$P(A, B) = P(A)P(B|A)$$

통계적 언어 모델

- 조건부 확률(Conditional probabilities)과 언어 모델
  - 연쇄 법칙

연쇄 법칙(Chain rule)

$$P(A, B, C, D) = P(A)P(B|A)P(C|A, B)P(D|A, B, C)$$

언어 모델

$$P(\text{나는}, \text{사과를}, \text{먹는다}) = P(\text{나는})P(\text{사과를} \mid \text{나는})P(\text{먹는다} \mid \text{나는}, \text{사과를})$$

통계적 언어 모델

- 조건부 확률(Conditional probabilities)과 언어 모델
  - 연쇄 법칙

연쇄 법칙(Chain rule)

$$P(A, B, C, D) = P(A)P(B|A)P(C|A, B)P(D|A, B, C)$$

n개 단어(w)의 결합 확률

$$P(w_1, w_2, w_3, \cdots, w_n) = P(w_1)P(w_2|w_1)P(w_3|w_1, w_2)\cdots P(w_n|w_1, \cdots, w_{n-1})$$

$$= \prod_{n=1}^{n} P(w_n|w_1, \cdots, w_{n-1})$$

통계적 언어 모델

- 조건부 확률(Conditional probabilities)과 언어 모델
  - 카운트 기반 계산

: 코퍼스(corpus) 내에서 각 단어들의 조합이 나오는 횟수를 **카운트(count)**한 후 이에 기반하여 확률을 계산

$$P(\text{먹는다} \mid \text{나는, 사과를}) = \frac{count(\text{나는, 사과를, 먹는다})}{count(\text{나는, 사과를})}$$

→ 모든 단어 조합의 경우의 수를 다 세어야 함

→ 계산 복잡도가 높아질 뿐만 아니라 무한한 크기의 코퍼스 필요

→ 어려움, 비현실적

통계적 언어 모델

- 조건부 확률(Conditional probabilities)과 언어 모델
  - 마르코프 가정(Markov assumption)

✓ 기존 연쇄 법칙의 복잡성을 해결하고 간소화하기 위함

✓ **미래 사건에 대한 조건부가 과거에 대해서는 독립이며 현재의 사건에만 영향을 받는 다는 가정**을 전제로 연쇄법칙 설명

✓ 단어w_n이 나타날 확률은 그 앞의 단어 w_{n-1}이 나타날 확률하고만 연관이 있다고 봄

$$P(w_1, w_2, \cdots, w_n) \approx P(w_1)P(w_2|w_1)\cdots P(w_n|w_{n-1})$$

통계적 언어 모델

- 조건부 확률(Conditional probabilities)과 언어 모델
  - 마르코프 가정(Markov assumption)

$$P(w_1, w_2, \cdots, w_n) \approx P(w_1)P(w_2|w_1)\cdots P(w_n|w_{n-1})$$

$$P(\text{나는},\ \text{사과를},\ \text{먹는다}) \approx P(\text{나는})P(\text{사과를}\,|\,\text{나는})P(\text{먹는다}\,|\,\text{사과를})$$

통계적 언어 모델

- N-gram 언어 모델(N-gram Language Model)
  - 마르코프 가정(Markov assumption)의 한계

✓ 어떤 단어의 등장 확률이 바로 앞의 단어하고만 연관이 있다는 마르코프 가정

✓ 언어 현상에 적용하기에는 **지나친 단순화**

✓ 언어의 **장기 의존성** 간과됨

✓ (예: The **computer** which I had just put into the machine room on the fifth floor **crashed**.)

✓ 예측 정확도 낮아질 수 있음

통계적 언어 모델

- N-gram 언어 모델(N-gram Language Model)
  - N-gram

✓ 문장 내 단어는 **주변의 여러 단어**와 연관된다고 가정

✓ N: 주변 몇 개의 단어를 볼 것인지 정하는 임의의 숫자

✓ N-gram: N개의 단어열

통계적 언어 모델

- N-gram 언어 모델(N-gram Language Model)

  ✓ 1-gram(unigram): The, boy, is, looking, at, a, pretty, girl

  ✓ 2-gram(bigram): The boy, boy is, is looking, looking at, at a, a pretty, pretty girl

  ✓ 3-gram(trigram): The boy is, boy is looking, is looking at, looking at a, at a pretty, a pretty girl

  ✓ 4-gram: The boy is looking, boy is looking at, is looking at a, looking at a pretty, at a pretty girl

통계적 언어 모델

- N-gram 언어 모델(N-gram Language Model)
  - N-1차 마르코프 가정

: 특정 단어가 등장하는 확률을 계산할 때에 이전 N-1개의 단어가 등장하는 확률만을 고려한다는 가정

통계적 언어 모델

- N-gram 언어 모델(N-gram Language Model)
  - N-1차 마르코프 가정

$$\text{1-gram(유니그램, unigram):}\quad P(w_1, w_2, \cdots, w_n) \approx \prod_{i=1}^{n} P(w_i)$$

$$\text{2-gram(바이그램, bigram):}\quad P(w_1, w_2, \cdots, w_n) \approx \prod_{i=1}^{n} P(w_i | w_{i-1})$$

$$\cdots$$

$$\text{N-gram}\ :\ P(w_1, w_2, \cdots, w_n) \approx \prod_{i=1}^{n} P(w_i | w_{i-N}, \cdots, w_{i-1})$$

통계적 언어 모델

- N-gram 언어 모델(N-gram Language Model)
  - N-1차 마르코프 가정

$$P(\text{먹는다} \mid \text{나는, 사과를}) \approx P(\text{먹는다} \mid \text{사과를})$$

$$\frac{count(\text{나는, 사과를, 먹는다})}{count(\text{나는, 사과를})} \approx \frac{count(\text{사과를, 먹는다})}{count(\text{사과를})}$$

통계적 언어 모델

- N-gram 언어 모델(N-gram Language Model)
  - 예제: 3개 문장과 2-gram 모델로 단어열의 등장 확률 계산

코퍼스 예시

<s>I eat an apple</s>

<s>an apple I eat</s>

<s>I like cheese cake</s>

통계적 언어 모델

- N-gram 언어 모델(N-gram Language Model)
  - 예제: 3개 문장과 2-gram 모델로 단어열의 등장 확률 계산

$$P(I|<s>) = \frac{count(<s>, I)}{count(<s>)} = \frac{2}{3} = 0.6667$$

$$P(an|<s>) = \frac{count(<s>, an)}{count(<s>)} = \frac{1}{3} = 0.3333$$

$$P(eat|I) = \frac{count(I, eat)}{count(I)} = \frac{2}{3} = 0.6667$$

$$P(</s>|apple) = \frac{count(apple, </s>)}{count(apple)} = \frac{1}{2} = 0.5$$

$$P(like|I) = \frac{count(I, like)}{count(I)} = \frac{1}{3} = 0.3333$$

$$P(cake|cheese) = \frac{count(cheese, cake)}{count(cheese)} = \frac{1}{2} = 0.5$$

통계적 언어 모델

- N-gram 언어 모델(N-gram Language Model)
  - 예제: 셰익스피어 작품 기반 N-gram 문장 생성

Unigram

✓ To him swallowed confess hear both. Which. Of save on trail for are ay device and rote life have

✓ Hill he late speaks; or! A more to leg less first you enter

→ 1-gram 모델은 단어의 열(sequence) 고려X, 모든 단어가 독립적이라고 봄

→ 서로 무관한 단어들이 생성됨

통계적 언어 모델

- N-gram 언어 모델(N-gram Language Model)
  - 예제: 셰익스피어 작품 기반 N-gram 문장 생성

Bigram

✓ Why dost stand forth thy canopy, forsooth; he is this palpable hit the King Henry. Live king. Follow.

✓ What means, sir. I confess she? Then all sorts, he is trim, captain.

→ 2-gram 모델은 바로 앞 1개의 단어를 고려

→ 1-gram 보다 자연스럽지만 전체적으로는 여전히 매우 부자연스러움

통계적 언어 모델

- N-gram 언어 모델(N-gram Language Model)
  - 예제: 셰익스피어 작품 기반 N-gram 문장 생성

Trigram

✓ King Henry. What! I will go seek the traitor Gloucester. Exeunt some of the watch. A great banquet serv'd in;

✓ It cannot be but so.

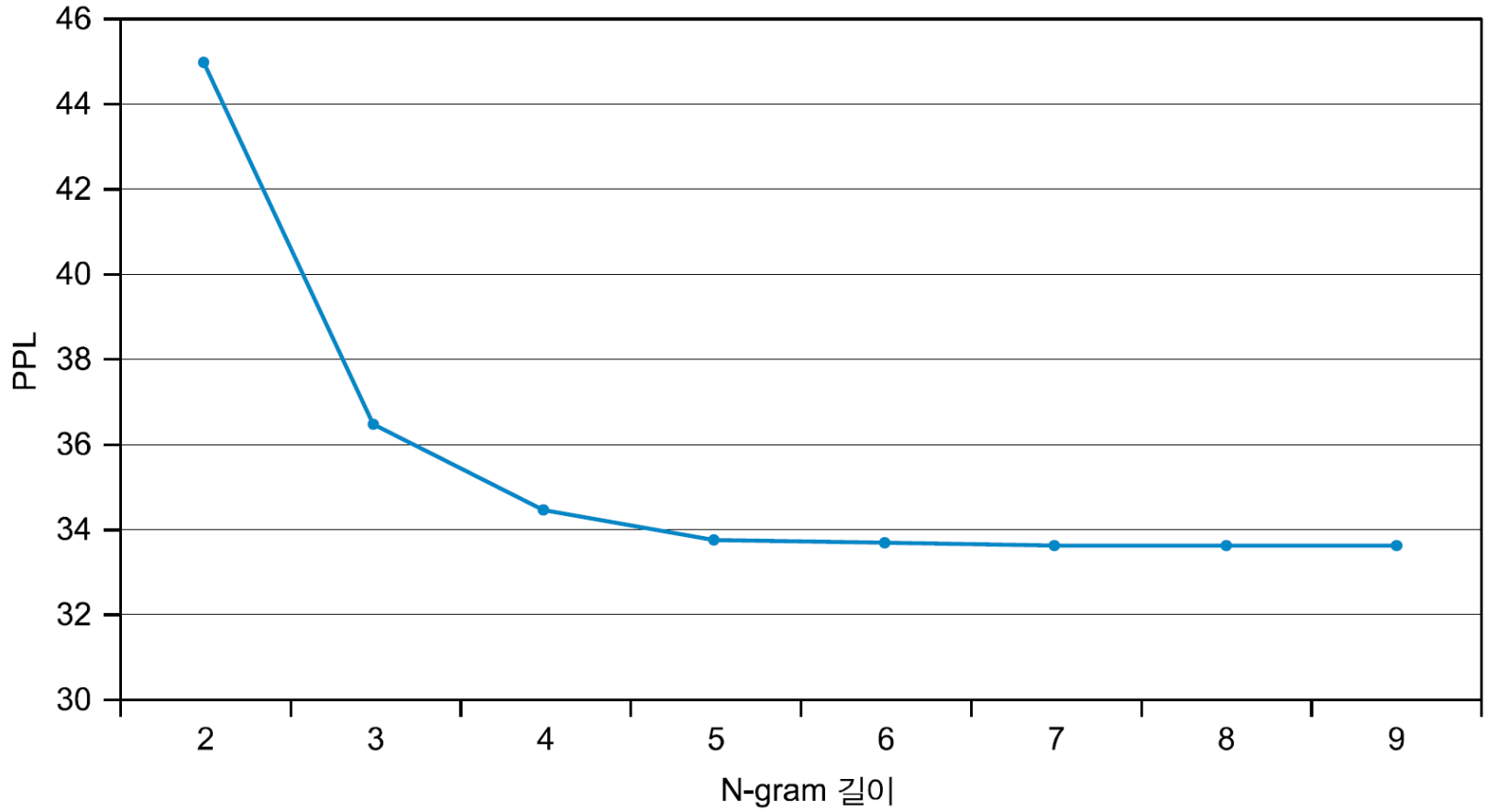
→ 1-gram 및 2-gram 모델의 문장보다 자연스러움

→ 코퍼스에 존재하는 텍스트에 가깝게 생성

→ 새로운 문장이라고 보기 어려움(low diversity)

통계적 언어 모델

- N-gram 언어 모델(N-gram Language Model)
  - N-gram에 따른 성능 비교



통계적 언어 모델

- N-gram 언어 모델(N-gram Language Model)
  - 한계

✓ 생성된 문장이 지나치게 부자연스럽거나 기존 코퍼스와 지나치게 유사함

✓ 단어열의 확률값이 코퍼스에 따라 크게 달라짐

✓ **방대한 양의 코퍼스 필요**

✓ **희소성 문제**(코퍼스에 등장하지 않는 단어열의 확률값은 모두 0)

✓ 예측의 정확도를 떨어트리는 요인

✓ 교착어인 한국어에서 희소성 문제가 크게 발생

✓ (예: 한국어에서 형태소 분석 등 전처리 진행하지 않으면 '사과가', '사과를', '사과도', '사과에'는 모두 다른 단어로 처리되어 '사과'가 포함된 단어열의 확률값이 0에 가까워짐)

통계적 언어 모델

- N-gram 언어 모델(N-gram Language Model)
  - 로그 확률(Log probabilities)

✓ 언어 모델의 확률 계산시 원래 확률값(raw probabilities)에 **로그(log)**를 취하는 것이 보편적

✓ 이는 **언더플로(underflow)**를 피하기 위함

* 언더플로: 부동 소수점 연산에서 컴퓨터가 표현할 수 있는 것보다 작은 값이 발생하여 계산 결과를 표시할 수 없는 상태

통계적 언어 모델

- N-gram 언어 모델(N-gram Language Model)
  - 로그 확률(Log probabilities)

$$\log(p_1 \times p_2 \times p_3 \times p_4) = \log p_1 + \log p_2 + \log p_3 + \log p_4$$

✓ **곱셈 연산**은 덧셈 연산보다 **계산 리소스가 크고** 결과값이 **0에 가까운 작은 값으로 계산될 가능성**이 있음

✓ 전체 확률 계산을 **로그** 공간에서 계산할 시 **곱셈을 덧셈으로 환산**할 수 있음

✓ **계산이 간단**해지고 원래 확률 계산보다 **언더플로를 피할 수 있음**

통계적 언어 모델

- N-gram 언어 모델(N-gram Language Model)
  - 로그 확률(Log probabilities)

✓ **로그** 공간에서 n개의 단어로 이루어진 단어열의 확률을 계산하는 **N-gram 모델 식**

$$\log P(w_1, w_2, \cdots, w_n) \approx \sum_{i=1}^{n} \log P(w_i | w_{i-N}, \cdots, w_{i-1})$$

일반화(Generalization)

- ✓ **통계적 언어 모델**은 **제한된 양의 코퍼스**로 인해 이전에 보지 못한 단어열에 대해서는 제대로 예측하지 못하고 **정확도가 떨어짐**
- ✓ 이와 같은 **희소성 문제를 해결**하고 모델의 **일반화 능력을 향상**시키기 위해 다양한 일반화 기법들 제시됨 (Laplace smoothing, Good-Turing smoothing, Witten-Bell smoothing, Interpolation, Back off, Kneser-Ney discounting 등)

일반화(Generalization)

- 스무딩(Smoothing)

✓ 모델이 한번도 본 적 없는 단어 조합(unseen n-gram)에 특정 값(α)을 부여하여 확률 분포에 약간의 변화를 주는 방법 $(0 < \alpha \leq 1)$

✓ 코퍼스에 없는 단어열로 인해 전체 문장의 확률이 0이 되는 희소성 문제 방지

일반화(Generalization)

- 스무딩(Smoothing)

$$P(w_i|w_{<i}) \approx \frac{count(w_{<i}, w_i) + \alpha}{count(w_{<i}) + \alpha V}$$

$\mathbf{w} < \boldsymbol{i}$

: i번째 단어 이전에 등장하는 모든 단어

$\boldsymbol{V}$

: 어휘(vocabulary) 사이즈 (코퍼스에 등장하는 단일 단어 개수)

일반화(Generalization)

- 스무딩(Smoothing)
  - 라플라스 스무딩(Laplace smoothing)

$$P(w_i|w_{<i}) \approx \frac{count(w_{<i}, w_i) + \alpha}{count(w_{<i}) + \alpha V}$$

- ✓ α 값을 1로 지정하는 방법
- ✓ 한번도 등장하지 않은 단어열이 최소 한번은 등장했다고 가정

일반화(Generalization)

- 스무딩(Smoothing)
  - 라플라스 스무딩(Laplace smoothing) 예제

코퍼스 예시

"I eat a strawberry"

"I eat a blueberry"

"I eat a strawberry cake"

문제: 2-gram 모델로 "I eat a blueberry cake"의 확률 구하기

일반화(Generalization)

- 스무딩(Smoothing)
  - 라플라스 스무딩(Laplace smoothing) 예제

$$P(I, eat, a, blueberry, cake)$$

$$\approx P(eat|I)\,P(a|eat)\,P(blueberry|a)\,P(cake|blueberry)$$

기존 확률 계산식: $P(cake|blueberry)$ = 0, 전체 문장의 등장 확률 = 0

일반화(Generalization)

- 스무딩(Smoothing)
  - 라플라스 스무딩(Laplace smoothing) 예제

$$P(cake\,|\,blueberry) \approx \frac{count\,(blueberry,\,cake\,)+1}{count\,(blueberry\,)+\,V}$$

$$\approx \frac{0+1}{1+6} = \frac{1}{7} \fallingdotseq 0.143$$

라플라스 스무딩 적용시: $0 < P(cake|blueberry) \leq 1,\ 0 <$ 전체 문장의 등장 확률 ≤ 1

일반화(Generalization)

- 스무딩(Smoothing)
  - 라플라스 스무딩(Laplace smoothing) 한계

✓ 라플라스 스무딩은 **제로 데이터**(코퍼스에 등장하지 않는 단어열)가 **적은 경우 유용**

✓ 그러나, 계산을 거듭할수록 원래 **단어의 빈도수에서 크게 벗어나는 문제**를 야기

✓ 또한, 제로 데이터에 특정 값을 부여하여도 **N-gram 모델의 일반화 문제는 완전히 해소되지 않음**

일반화(Generalization)

- 보간법(Interpolation)

✓ 특정 N-gram의 확률을 **이전 N-gram의 확률과 섞는 방법**

✓ 3-gram 모델 예시: 2-gram, 1-gram 모델의 확률까지 모두 구한 후 **일정한 비율(λ)의 가중치**를 각각 곱한 후 모두 합하는 방식 ($0 < \lambda \leq 1, \sum_i \lambda_i = 1$)

$$\hat{P}(w_n|w_{n-1}, w_{n-2}) = \lambda_1 P(w_n|w_{n-1}, w_{n-2}) + \lambda_2 P(w_n|w_{n-1}) + \lambda_3 P(w_n)$$

일반화(Generalization)

- 보간법(Interpolation)

✓ 라플라스 스무딩은 모든 제로 데이터에 똑같은 빈도수를 부여하기 때문에 문제 발생

✓ **보간법** 사용시, **제로 데이터들의 N-gram 정보에 따라 서로 다른 빈도를 부여**할 수 있음

✓ 가중치(λ)는 검증 코퍼스에서 각 N-gram의 확률을 최대화하는 0보다 크고 1보다 작거나 같은 값으로 설정

일반화(Generalization)

- 백오프(Back off)

✓ 보간법과 유사, 여러 N-gram을 함께 고려

✓ **모든 N-gram의 확률을 합하지 않는다**는 점이 보간법과의 차이

✓ 3-gram 모델 예시: 3-gram, 2-gram, 1-gram 확률 중 빈도수가 0 이상이며 N의 차수가 높은 N-gram 확률을 사용

$$\hat{P}(w_i|w_{i-2},w_{i-1}) = \begin{cases} P(w_i|w_{i-2},w_{i-1}), \text{ if } count(w_{i-2},w_{i-1},w_i) > 0 \\ \alpha_1 P(w_i|w_{i-1}), \text{ if } count(w_{i-2},w_{i-1},w_i) = 0 \text{ and } count(w_{i-1},w_i) > 0 \\ \alpha_2 P(w_i), \ otherwise. \end{cases}$$

일반화(Generalization)

- 생각해 볼 문제: 사람의 일반화와 차이점

✓ 보간법과 백오프는 특정 N-gram보다 작은 N-gram의 단어열을 고려하여 보다 나은 일반화 가능

✓ 그러나 스무딩과 보간법, 백오프 등의 일반화 방법은 문장 내에서 유사한 **패턴**을 찾음으로써 새로운 정보를 학습하는 **사람의 일반화** 방식과는 사뭇 다름

일반화(Generalization)

- 생각해 볼 문제: 사람의 일반화와 차이점

✓ 사람의 경우 “I eat a strawberry”, “I eat a blueberry”, “I eat a strawberry cake”라는 문장을 학습하고 나면 비슷한 **패턴**의 “I eat a blueberry cake”라는 문장도 가능할 수 있음을 직관적으로 학습함

✓ **훌륭한 언어 모델**이라면 문장 내에 처음 보는 단어 혹은 단어열이 있다고 하더라도 문장 내 단어열 간의 **유사도**를 고려하여 **새로운 단어열**에 대해 **추론**할 수 있는 일반화가 가능해야 함 → 딥러닝 언어 모델에서의 주요 과제 중 하나

모델 평가와 퍼플렉서티(Perplexity)

- 언어 모델의 평가

✓ 일반적인 방법은 모델 간 비교. But 상당한 시간 소요

✓ **퍼플렉서티(Perplexity, PPL)**: 언어 모델의 성능을 자체적으로 평가하는 내부 평가(intrinsic evaluation) 척도

→ **간이 실험 등 짧은 시간 안에 간단히 모델을 평가할 때, 혹은 모델 간 비교시 평가척도로 사용**

모델 평가와 퍼플렉서티(Perplexity)

- 퍼플렉서티(Perplexity, PPL)

✓ 주어진 확률 모델이 샘플을 얼마나 잘 예측하는가에 대한 측정 지표

✓ **'perplexity'**는 **'헷갈리는 정도'**를 뜻함

✓ 모델이 테스트 데이터셋에 대하여 확률 분포를 얼마나 확실하게 (헷갈리지 않게) 예측할 수 있는지를 나타내는 지표

✓ **PPL 점수가 낮을수록 (헷갈리는 경우가 적을수록) 좋음**

모델 평가와 퍼플렉서티(Perplexity)

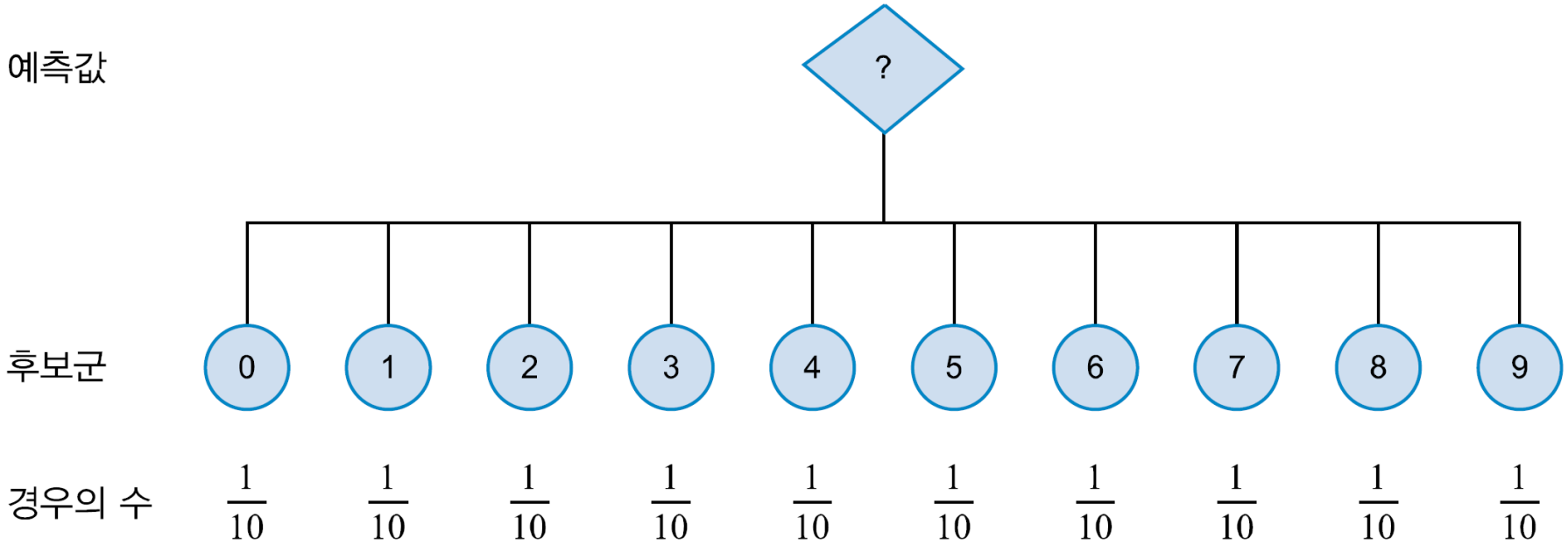
- 퍼플렉서티(Perplexity, PPL) 계산

✓ 퍼플렉서티는 모델이 선택할 수 있는 경우의 수를 의미하는 분기계수(branching factor)

모델 평가와 퍼플렉서티(Perplexity)

- 퍼플렉서티(Perplexity, PPL) 계산

10개 MNIST 데이터에 대한 분기계수 예시



모델 평가와 퍼플렉서티(Perplexity)

- 퍼플렉서티(Perplexity, PPL) 계산

✓ 퍼플렉서티는 모델이 선택할 수 있는 경우의 수를 의미하는 분기계수(branching factor)

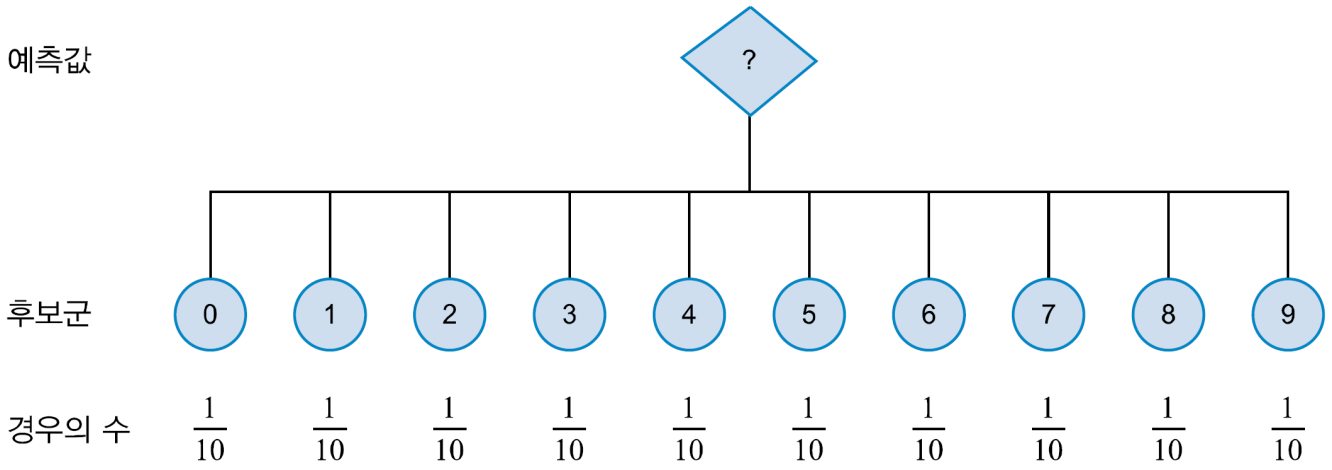
✓ 즉, 모델이 샘플의 확률을 예측하는 시점에서 **얼마나 많은 후보군을 두고 고민하는가**를 나타냄

→ PPL이 높다는 것은 모델이 더 많은 후보군을 두고 고민하는 것

→ 예측에 대한 확실성이 낮음을 뜻함

모델 평가와 퍼플렉서티(Perplexity)

- 퍼플렉서티(Perplexity, PPL) 계산



✓ PPL = 10

✓ 30,000개 단어들 중 다음 단어로 올 확률이 가장 높은 단어를 예측하는 언어모델이라면 PPL = 30,000

✓ 후보군에 대한 확률의 역수를 후보군의 개수로 정규화(normalization)하여 계산

모델 평가와 퍼플렉서티(Perplexity)

- 퍼플렉서티(Perplexity) 계산
  - 언어 모델의 PPL 계산 식

$$PPL(W) = P(w_1, w_2, \ldots, w_N)^{-\frac{1}{N}}$$

$$= \sqrt[N]{\frac{1}{P(w_1, w_2, \ldots, w_N)}}$$

$$= \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_1 | w_1, w_2, \ldots, w_{i-1})}}$$

✓ N개의 단어 w_N으로 이루어진 문장 W에 대하여 다음 단어를 예측하는 언어 모델의 PPL 계산 식

모델 평가와 퍼플렉서티(Perplexity)

- 퍼플렉서티(Perplexity) 계산
  - MNIST 예제 적용 예시

$$PPL(W) = P(w_1, w_2, \ldots, w_N)^{-\frac{1}{N}}$$

$$= \sqrt[N]{\frac{1}{P(w_1, w_2, \ldots, w_N)}}$$

$$= \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_1 | w_1, w_2, \ldots, w_{i-1})}}$$

독립가정

$$PPL(W) = P(w_1, w_2, \ldots, w_N)^{-\frac{1}{N}}$$

$$= \left(\frac{1}{10^N}\right)^{-\frac{1}{N}}$$

$$= \left(\frac{1}{10}\right)^{-1}$$

$$= 10$$

모델 평가와 퍼플렉서티(Perplexity)

- 퍼플렉서티(Perplexity) 계산
  - 2-gram 모델 PPL 계산식

✓ N-gram 모델에서는 'N-1차 마르코프 가정'에 따른 연쇄법칙이 적용됨

✓ 이를 반영한 2-gram 모델의 계산식

$$PPL(W) = \sqrt[N]{\frac{1}{\prod_{i=1}^{N} P(w_i|w_{i-1})}}$$

질의 응답