

CSCI E-89B 자연어 처리 (NLP)

제8강: 구조적 토픽 모델링 (STM)

Dmitry Kurochkin
(Fall 2025)

- 강의명: CSCI E-89B: 자연어 처리 입문
- 주차: Lecture 08
- 교수명: Dmitry Kurochkin
- 목적: Lecture 08의 핵심 개념 학습

Abstract

본 문서는 자연어 처리(NLP) 8강의 핵심 내용을 정리합니다. 주요 주제는 기존 LDA의 한계를 극복하는 **구조적 토픽 모델링(Structural Topic Modeling, STM)**입니다. STM이 어떻게 문서의 **메타데이터(metadata)**를 모델에 통합하여 더 풍부하고 정확한 분석을 가능하게 하는지 수학적 원리와 R 실습을 통해 배웁니다. 또한, 과제 수행 시 발생하는 텍스트 오토인코더의 어려움과 올바른 제출 가이드라인을 다루며, LDA와 NMF에 대한 퀴즈 복습을 포함합니다.

Contents

1 주요 공지 및 과제 가이드라인	2
1.1 제출물 요구사항	2
2 과제 Q&A: 텍스트 오토인코더(Autoencoder)	2
2.1 Q: 텍스트 오토인코더로 원본 문장을 완벽하게 복원하기 매우 어렵습니다.	2
2.1.1 왜 텍스트 복원이 더 어려운가?	3
2.2 개선 시도 및 접근법	3
3 퀴즈 7 복습: LDA 및 NMF	4
3.1 퀴즈 문항 및 정답	4
3.2 LDA와 NMF 심층 비교	4
3.3 LDA의 비결정론적(Stochastic) 특성	5

4 구조적 토픽 모델링 (STM) 소개	6
4.1 LDA의 근본적인 한계: 메타데이터의 부재	6
4.2 STM (Structural Topic Modeling)의 정의	6
5 STM의 수학적 원리	6
5.0.1 1. LDA의 토픽 유병률 (Topic Prevalence)	6
5.0.2 2. STM의 토픽 유병률 (Topic Prevalence)	6
5.0.3 STM의 주요 파라미터	7
6 STM의 장점 및 활용	7
7 R 실습: stm 패키지	8
7.1 실습 1: ”고양이 vs. 강아지” (저자 분석)	8
7.2 실습 2: ”Kaggle 뉴스 헤드라인” (시계열 분석)	9
7.2.1 데이터 전처리: 날짜(Date) 공변량 다루기	9
7.2.2 전처리 시 중요사항: 빈 문서 자동 제거	9
7.2.3 시계열 효과 추정 및 시각화	10
8 최적의 토픽 수 (K) 찾기	11
8.1 방법: searchK() 함수	11
8.2 평가 지표: 일관성(Cohherence) vs. 배타성(Exclusivity)	11
9 토픽 해석하기: ”토픽 1”은 무엇인가?	12
9.1 방법 1: 상위 단어 확인 (summary() 또는 labelTopics())	12
9.2 방법 2: 대표 문서 확인 (findThoughts()) - 권장	12
10 용어 정리	13
11 FAQ (주요 질문 및 답변)	14

1 주요 공지 및 과제 가이드라인

본격적인 강의 내용에 앞서, 과제 제출과 관련된 주요 가이드라인을 명확히 합니다.

1.1 제출물 요구사항

과제 제출 시 다음 두 가지 구성 요소를 모두 제출해야 합니다.

1. 보고서 (Report): 단일 PDF 파일

- 최종 보고서는 하나의(single) PDF 파일이어야 합니다.
- 각 문제 풀이를 별도의 Jupyter Notebook에서 수행했더라도, 각 노트북에서 생성된 PDF 파일들을 하나로 병합하여 제출해야 합니다.
- Jupyter Notebook에서 직접 PDF를 생성하거나, 다른 워드 프로세서(예: MS Word)를 사용해 수동으로 보고서를 작성한 후 PDF로 변환할 수 있습니다.

2. 원본 코드 (Source Code)

- 보고서를 생성하는 데 사용된 모든 원본 코드를 제출해야 합니다.
- (예: .ipynb Jupyter Notebook 파일, .py Python 스크립트 파일 등)
- 여러 개의 코드 파일이 있는 경우, 하나의 .zip 파일로 압축하여 제출하는 것을 강력히 권장 합니다. (필수는 아니지만 관리에 용이함)

▣ 핵심 요약

제출 요약:

- 파일 1 (필수): 모든 결과가 포함된 하나의 PDF 보고서.
- 파일 2 (필수): 모든 소스 코드를 담은 하나의 .zip 파일(또는 개별 코드 파일들).

Jupyter Notebook을 반드시 사용해야 하는 것은 아니지만, 코드와 리포트를 효율적으로 관리하는 데 유용할 수 있습니다.

2 과제 Q&A: 텍스트 오토인코더(Autoencoder)

많은 학생이 텍스트(예: 200단어 시퀀스)를 처리하는 오토인코더 과제에서 어려움을 겪고 있습니다.

2.1 Q: 텍스트 오토인코더로 원본 문장을 완벽하게 복원하기 매우 어렵습니다.

A: 네, 그것은 매우 예상된(expected) 결과입니다.

완벽하게 복원되지 않는다고 해서 모델이 잘못된 것은 아닙니다. 이 과제의 목적은 텍스트 오토인코더가 이미지 오토인코더보다 훨씬 더 어려운 과제임을 직접 경험하는 것입니다.

2.1.1 왜 텍스트 복원이 더 어려운가?

1. 극심한 병목 현상 (Severe Bottleneck)

- 표준적인 시퀀스-투-시퀀스(Seq2Seq) 오토인코더는 인코더(Encoder)가 전체 텍스트 시퀀스(예: 200단어)를 하나의 단일 벡터(single vector) (예: 인코더 RNN의 마지막 은닉 상태)로 압축합니다.
- 이 단일 벡터가 전체 문장의 문맥, 순서, 의미를 모두 담아야 합니다.
- 디코더(Decoder)는 오직 이 단일 벡터 정보에만 의존하여 원본 시퀀스를 복원해야 하므로, 정보 손실이 막대하게 발생합니다.

2. 이미지 vs. 텍스트

- 이미지 (쉬움):** 컨볼루셔널 오토인코더(Convolutional Autoencoder)는 이미지를 압축할 때 공간적 구조(spatial structure)를 유지하며 점진적으로 특징을 추출합니다. 복원 시에도 이 구조를 역으로 따라가면 되므로 상대적으로 복원력이 뛰어납니다.
- 텍스트 (어려움):** 텍스트는 순서와 문맥이 매우 중요한 데이터입니다. 이를 단일 벡터로 '뭉개버리면' 원본의 복잡한 순차 정보를 복원하기가 극도로 어려워집니다.

▣ 핵심 요약

과제의 핵심은 ”완벽한 복원”이 아니라, ”텍스트 시퀀스를 단일 벡터로 압축하고 복원하는 것이 왜 어려운지, 그 한계가 무엇인지”를 이해하는 것입니다.

2.2 개선 시도 및 접근법

- 학생의 접근 (Smart Approach):** 디코더의 softmax 출력에서 확률이 가장 높은 단어(top-1)만 선택하는 대신, **온도 함수(temperature function)**를 사용하여 확률 분포를 조절하고, 확률이 높은 *주변*의 단어들을 샘플링(sampling)하는 방식입니다. → 교수의 평가: 매우 똑똑한 접근입니다. 원본과 정확히 같지는 않더라도 문법적으로나 의미적으로 일관된 영어 문장을 생성해낼 수 있습니다.
- 데이터 증강 (Data Augmentation):** 모델의 성능이 낮은 이유 중 하나는 훈련 데이터가 부족하기 때문일 수 있습니다. 이 경우 데이터 증강 기법을 사용할 수 있습니다.
 - 유의어(Synonym) 대체:** 문장 내 일부 단어를 유의어로 교체합니다.
 - 단어 삽입/삭제:** 문장의 일부 단어를 무작위로 삭제(drop)하거나 다른 단어를 삽입합니다.
 - 문장 순서 변경:** 문맥에 큰 영향이 없는 선에서 문장 순서를 바꿉니다.

▣ 예제: title

놀랍게도, 이 ’시퀀스를 단일 벡터로 압축하는’ 오토인코더 구조는 초기 신경망 기계 번역(NMT)에서 실제로 시도되었던 방식입니다. (예: 영어 문장을 단일 벡터로 압축 → 프랑스어 문장으로 복원)

하지만 이 역시 동일한 병목 현상(bottleneck) 문제로 인해 긴 문장에서 성능이 급격히 저하되었고, 이후 ’어텐션(Attention) 메커니즘’이 등장하면서 이 접근법은 사장되었습니다. 여러분이 겪은 어려움은 과거 연구자들도 동일하게 겪었던 문제입니다.

3 퀴즈 7 복습: LDA 및 NMF

퀴즈 7의 주요 개념인 LDA(잠재 디리클레 할당)와 NMF(음수 미포함 행렬 분해)를 복습합니다.

3.1 퀴즈 문항 및 정답

- Q1 (LDA의 역할):** LDA가 무엇을 하는가? A: (C) 각 문서를 여러 토픽의 **혼합(mixture)**으로 간주합니다. (예: 이 문서는 정치 60)
- Q2 (LDA의 과제):** LDA 사용 시 흔히 겪는 어려움은? A: (B) 최적의 **토픽 수(K)**를 결정하는 것. (K는 하이퍼파라미터임)
- Q3 (디리클레 분포의 역할):** LDA에서 디리클레(Dirichlet) 분포의 역할은? A: (A) 각 문서 내의 **토픽 비율(proportions)**(θ)을 생성하기 위한 사전 확률 분포(prior distribution)입니다.
- Q4 (NMF의 역할):** NMF(음수 미포함 행렬 분해)에 대한 가장 좋은 설명은? A: (B) (음수가 아닌 항목을 가진) 행렬을 더 작은 행렬들로 분해합니다. (예: 문서-단어 행렬 $V \approx W \times H$)
- Q5 (LDA vs. NMF 차이):** 두 접근법의 차이는? A: (A) LDA는 가능도(likelihood)를 최대화하는 **확률적 모델**인 반면, NMF는 행렬을 분해하는 **행렬 대수(matrix algebra)** 기법입니다.

3.2 LDA와 NMF 심층 비교

퀴즈 5번 항목에서 혼동이 있을 수 있습니다. LDA와 NMF는 모두 토픽 모델링에 사용될 수 있지만, 그 철학이 다릅니다.

아래 표는 LDA와 NMF의 핵심적인 차이를 요약합니다.

특징	LDA (Latent Dirichlet Allocation)	NMF (Non-negative Matrix Factorization)
기본 원리	확률적 생성 모델(Probabilistic)	행렬 대수 분해(Linear Algebra)
목표	문서가 생성되는 확률적 과정을 모델링합니다. (최대 가능도 추정, MLE)	원본 행렬(V)을 두 개의 작은 음수 미포함 행렬(W, H)의 곱($V \approx WH$)으로 근사합니다.
결과물	1. 문서별 토픽 분포(θ): 문서 A는 토픽 1(40%), 토픽 2(60%) 2. 토픽별 단어 분포(β): 토픽 1은 '고양이'(30%), '강아지'(20%)...	1. 토픽-단어 행렬(W) 2. 문서-토픽 행렬(H)
가정	문서는 토픽의 혼합, 토픽은 단어의 분포라는 명확한 생성 가정이 있습니다. (디리클레 사전 분포)	모든 행렬의 성분이 0 이상이어야 한다는 제약 조건만 있습니다.
해석	결과가 '확률' 또는 '비율'로 나와 해석이 매우 직관적입니다.	분해된 행렬을 '토픽'으로 해석합니다. 이때 관계는 덧셈(additive)이 아닌 곱셈(multiplicative)입니다.

Table 1: LDA와 NMF의 핵심 원리 비교

3.3 LDA의 비결정론적(Stochastic) 특성

LDA는 선형 회귀처럼 단 하나의 정답이 나오는 결정론적(deterministic) 모델이 아닙니다. LDA는 **확률적(stochastic) 알고리즘**입니다.

- **실행 시마다 결과가 다름:** LDA를 실행할 때마다 (특히 초기값이 다를 경우) 결과가 미묘하게 달라질 수 있습니다.
 - 토픽 1과 토픽 2의 순서가 바뀔 수 있습니다.
 - 토픽을 구성하는 단어의 클러스터가 약간 달라질 수 있습니다.
- **이유: 유일한 해가 없음 (No Unique Solution)** LDA는 최대 가능성 추정(MLE)을 사용하는데, 이 가능성(likelihood)을 최대화하는 해가 유일하지 않을 수 있습니다.

□ 예제: title

최대 가능성 추정(MLE)은 ”관측된 데이터를 가장 잘 설명하는 원인(모델 파라미터)은 무엇인가?”를 찾는 과정입니다.

- **관측 (Data):** 고양이가 밖에서 젖은 채로 집에 돌아왔습니다.
- **가설 (Model):**
 1. 가설 A: 밖에 비가 왔다.
 2. 가설 B: 누군가 고양이에게 물을 뿐였다.
- **MLE 추론:** ”비가 왔을 때 고양이가 젖을 확률”과 ”누군가 물을 뿐였을 때 고양이가 젖을 확률”을 계산하여, 현재의 관측(젖은 고양이)을 가장 그럴듯하게 만드는 가설(예: 비가 왔다)을 선택합니다.

LDA도 마찬가지로, 우리가 관측한 ’문서들(단어들의 집합)’을 가장 그럴듯하게 생성했을 토픽 분포(θ)와 단어 분포(β)를 역으로 추정하는 것입니다.

LDA 실행 시 실전 팁

LDA는 확률적이며 안정적이지 않기 때문에, 실제 분석에서는 다음과 같은 방법을 사용합니다.

1. **여러 번 실행 (Multiple Runs):** 동일한 K(토픽 수)에 대해 모델을 여러 번 실행합니다.
2. **최적 모델 선택:** 실행된 모델들 중에서 가장 ”좋은” 모델을 선택합니다.
3. **평가 지표:** ”좋은” 모델을 판단하는 기준은 다음과 같습니다.
 - **의미론적 일관성 (Semantic Coherence):** 토픽 내의 상위 단어들이 의미적으로 얼마나 관련성이 높은가? (예: ’고양이’, ’강아지’, ’애완동물’ ... → 높음)
 - **배타성 (Exclusivity):** 토픽들이 서로 얼마나 겹치지 않고 고유한 단어들을 갖는가?

4 구조적 토픽 모델링 (STM) 소개

4.1 LDA의 근본적인 한계: 메타데이터의 부재

지금까지 배운 LDA는 문서를 '단어의 가방(Bag-of-Words)'으로만 취급합니다. 즉, 문서에 포함된 단어 외의 모든 맥락 정보를 무시합니다.

하지만 실제 세계의 문서는 풍부한 **메타데이터(Metadata)**와 함께 제공됩니다.

- **뉴스 기사:** 저자, 출판 날짜, 언론사 (예: New York Times vs. Financial Times)
- **학술 논문:** 저자, 출판 연도, 학회지 (Journal)
- **고객 리뷰:** 작성자, 평점(Rating), 작성 날짜
- **학생 강의평가:** 교수 성별, 개설 학과, 수강 시기 (예: 가을 vs. 봄 학기)

LDA는 이 중요한 메타데이터를 활용할 방법이 없습니다.

4.2 STM (Structural Topic Modeling)의 정의

구조적 토픽 모델링 (**Structural Topic Modeling, STM**)은 LDA를 확장하여, 이러한 문서 수준의 ****메타데이터****를 모델에 직접 통합하는 프레임워크입니다.

메타데이터는 통계학 용어로 ****공변량(Covariates)****이라고도 부릅니다.

▣ 핵심 요약

STM의 핵심 아이디어:

- **LDA:** 문서의 토픽 비율(θ)은 모든 문서에 동일한 디리클레 분포($\text{Dir}(\alpha)$)에서 나온다.
- **STM:** 문서의 토픽 비율(θ_d)은 해당 문서의 메타데이터(X_d)에 따라 달라진다.
(예: '교수 성별'이라는 메타데이터가 '돌봄(caring)' 토픽의 비율에 영향을 줄 수 있다.)

5 STM의 수학적 원리

STM이 LDA와 어떻게 다른지 수학적 공식을 통해 비교합니다. 가장 큰 차이는 문서별 토픽 비율(θ)을 생성하는 부분입니다.

5.0.1 1. LDA의 토픽 유병률 (Topic Prevalence)

문서 m 의 토픽 비율 θ_m 은 디리클레 분포에서 직접 샘플링됩니다.

$$\theta_m \sim \text{Dirichlet}(\alpha)$$

여기서 α 는 모든 문서에 동일하게 적용되는 하이퍼파라미터입니다.

5.0.2 2. STM의 토픽 유병률 (Topic Prevalence)

문서 d 의 토픽 비율 θ_d 는 해당 문서의 메타데이터 X_d 에 의존하는 **로지스틱 정규 분포(Logistic-Normal Distribution)**을 따릅니다.

$$\theta_d | X_d, \gamma, \Sigma \sim \text{LogisticNormal}(\mu = X_d \gamma, \Sigma)$$

이 공식이 생소해 보일 수 있지만, 두 단계로 나누어 생각하면 쉽습니다.

1단계 (회귀 분석): 먼저, 문서 d 의 메타데이터(X_d)를 사용하여 토픽의 평균적인 방향(μ)을 계산합니다.

$$\mu = X_d\gamma$$

- X_d : 문서 d 의 메타데이터 벡터 (예: '[1, date, author, gender, ...]')
- γ (감마): 메타데이터가 각 토픽에 미치는 영향을 나타내는 **회귀 계수** (모델이 학습해야 할 파라미터)
- 이는 선형 회귀 ($y = X\beta$)와 매우 유사한 형태입니다.

2단계 (Softmax): 1단계에서 계산된 μ 를 평균으로, Σ 를 공분산으로 하는 다변량 정규 분포 (Multivariate Normal)에서 벡터를 하나 샘플링합니다. 이 벡터를 소프트맥스(Softmax) 함수에 통과시켜 합이 1이 되는 확률 벡터(즉, 토픽 비율 θ_d)를 생성합니다. (이 과정을 합쳐 '로지스틱 정규 분포'라고 부릅니다.)

5.0.3 STM의 주요 파라미터

- γ (감마): 메타데이터(공변량)가 토픽 유병률에 얼마나 영향을 미치는지 나타내는 **회귀 계수**입니다. 이 값을 분석하는 것이 STM의 핵심입니다.
- Σ (시그마): 토픽 간의 **공분산 행렬**입니다. LDA와 달리, STM은 토픽들이 서로 상관관계를 가질 수 있다고 가정합니다. (예: '정치' 토픽과 '경제' 토픽은 자주 함께 등장 → 상관관계가 높음). 이 Σ 역시 모델이 데이터로부터 학습합니다.

6 STM의 장점 및 활용

- **메타데이터 통합:** 문맥 정보를 활용하여 더 정확하고 의미 있는 토픽을 추출합니다.
- **가설 검증 (Hypothesis Testing):** STM의 진정한 강점입니다. 모델이 추정한 γ 계수를 분석하여 사회과학적, 경영학적 가설을 검증할 수 있습니다.
 - 예1: "시간이 지남에 따라(date) '기후 변화' 토픽의 언급이 증가했는가?"
 - 예2: "여성 교수(gender)가 남성 교수보다 '학생 지원' 관련 토픽을 더 많이 언급받는가?"
 - 예3: "보수 언론(source)이 진보 언론보다 '세금 감면' 토픽을 더 많이 다루는가?"
- **활용 분야:** 미디어 프레이밍 분석, 여론 조사, 소셜 미디어 트렌드 추적, 고객 피드백 분석, 학술 연구 동향 파악 등 메타데이터가 존재하는 모든 텍스트 분석에 활용됩니다.

7 R 실습: `stm` 패키지

STM은 Python보다 R의 `stm` 패키지가 가장 표준적이고 강력한 구현체로 인정받고 있습니다. 본 강의에서는 R을 사용하여 STM을 실습합니다.

7.1 실습 1: ”고양이 vs. 강아지” (저자 분석)

두 명의 저자가 각각 다른 주제(고양이, 강아지)에 대해서만 글을 쓴 간단한 예제입니다.

- **데이터:** 저자 1 (고양이 텍스트 8개), 저자 2 (강아지 텍스트 8개)
- **메타데이터 (X_d):** `author` (범주형 변수: "Author1", "Author2")

```

1 # 1. 라이브러리로드
2 library(stm)
3
4 # 2. 데이터및메타데이터준비생략      ()
5 # documents <- c("Cats purr gently...", "Dogs bark loudly...", ...)
6 # metadata <- data.frame(author = rep(c("Author1", "Author2"), each = 8)
7   )
8
9 # 3. 텍스트전처리
10 # 는 textProcessor 문서와메타데이터를함께처리하여
11 # 문서가삭제될경우메타데이터도함께동기화시킵니다 .
12 processed <- textProcessor(documents = documents, metadata = metadata)
13 out <- prepDocuments(processed$documents, processed$vocab, processed$meta)
14
15 # 4. STM 모델피팅
16 # K=2 토픽(개2), prevalence = ~ author 저자( 변수를공변량으로사용 )
17 stm_model <- stm(documents = out$documents,
18                   vocab = out$vocab,
19                   K = 2,
20                   prevalence = ~ author, # 핵심: 메타데이터지정
21                   data = out$meta,
22                   max.em.its = 100,
23                   init.type = "Spectral")
24
25 # 5. 결과요약
26 summary(stm_model)
27 # Topic 1 Top Words: dog, energet, bark, enjoy, chase...
28 # Topic 2 Top Words: cat, love, climb, purr, spot...
29
30 # 6. 메타데이터효과추정
31 effects <- estimateEffect(1:2 ~ author,
32                           stmobj = stm_model,
33                           metadata = out$meta,
34                           uncertainty = "Global")
35 # 7. 효과시각화

```

```

36 plot(effects,
37   covariate = "author",
38   method = "difference",
39   cov.value1 = "Author1",
40   cov.value2 = "Author2",
41   main = "Effect of Author Across Topics")

```

Listing 1: R 코드: 간단한 STM 모델 퍼팅 (저자 분석)

- 결과 해석: `summary` 결과, 토픽 1은 '강아지' 관련, 토픽 2는 '고양이' 관련 단어로 명확히 분리 됩니다. `plot(effects, ...)` 결과는 "Author1이 Author2에 비해 토픽 2(고양이)를 더 많이 사용하고, 토픽 1(강아지)을 덜 사용한다"는 것을 시각적으로 보여줍니다.

7.2 실습 2: "Kaggle 뉴스 헤드라인" (시계열 분석)

100만 개 이상의 호주 뉴스 헤드라인 데이터를 사용하여 시간에 따른 토픽 트렌드를 분석합니다.

- 데이터: 뉴스 헤드라인 텍스트
- 메타데이터 (X_d): `publish_date` (날짜, 예: 20030219)

7.2.1 데이터 전처리: 날짜(Date) 공변량 다루기

날짜 데이터를 STM에서 공변량으로 사용하는 방법은 두 가지가 있습니다.

1. 범주형(Categorical) 변수: 모든 날짜(예: '2003-02-19', '2003-02-20'...)를 별개의 범주로 취급 합니다. (단점: 변수의 수가 너무 많아져(수천 수만 개) 계산 비용이 매우 비싸지고(expensive) 분석이 어려워짐.)
 2. 연속형(Continuous) 수치 변수: 날짜를 하나의 숫자로 변환합니다. (예: '2003.0 + (month-1)/12') (장점: 변수가 하나이므로 계산이 효율적임.) (가정: 이 방식을 사용하면, 토픽의 유병률이 시간에 따라 선형적(linear)으로 증가하거나 감소한다고 가정하는 것입니다.)
- 본 실습에서는 2번 (연속형 수치 변수) 방식을 사용합니다.

7.2.2 전처리 시 중요사항: 빈 문서 자동 제거

`textProcessor`와 `prepDocuments` 함수는 텍스트를 정리(불용어, 숫자, 구두점 제거)합니다.

데이터 정합성(Consistency) 유지

- 텍스트 정리 과정에서, 어떤 문서는 내용이 완전히 비게(empty) 될 수 있습니다 (실습 예제에서 50개 문서).
- `prepDocuments`는 이 빈 문서들을 자동으로 제거합니다.
- 치명적 오류 방지: 만약 원본 메타데이터를 그대로 `stm` 모델에 전달하면, 문서 수(줄어듦)와 메타데이터 행 수(그대로)가 일치하지 않아 오류가 발생합니다.
- 해결책: `prepDocuments`가 반환한 `out$meta`를 새로운 메타데이터로 사용해야 합니다. `out$meta`는 빈 문서에 해당하는 행이 이미 제거된, 정제된 메타데이터입니다.

잘못된 사용

```

meta <- original_metadata
out <- prepDocuments(...)
stm_model <- stm(..., data = meta) # 오류! 행개수가 안맞음

```

```
# 올바른 사용
out <- prepDocuments(..., metadata = original_metadata)
meta_synced <- out$meta # <-- 반드시 out$를 meta 사용
stm_model <- stm(..., data = meta_synced) # 정상작동
```

7.2.3 시계열 효과 추정 및 시각화

```

1 # 1. STM 모델피팅 (K=5, 날짜를 공변량으로 )
2 # (date_는 numeric 2003.083... 같이 변환된 수치형 날짜 )
3 stm_model_news <- stm(...,
4                         K = 5,
5                         prevalence = ~ date_numeric,
6                         data = meta_synced)
7
8 # 2. 효과추정
9 # 는 estimateEffect 모델의 후방분포 (posterior distribution)에서
10 # 샘플링하여 date_에 numeric 따른다를 prevalence 회귀분석합니다 .
11 effects_news <- estimateEffect(1:5 ~ date_numeric,
12                                   stmobj = stm_model_news,
13                                   metadata = meta_synced,
14                                   uncertainty = "Global")
15
16 # 3. 시계열 트렌드 시각화
17 plot(effects_news,
18       covariate = "date_numeric",
19       method = "continuous", # <-- 연속형 변수 지정
20       topics = 1:5,
21       xlab = "Year",
22       main = "Topic Prevalence Over Time")
```

Listing 2: R 코드: STM 시계열 효과 추정

- **결과 해석:** 생성된 플롯은 시간에 따른 각 토픽의 유병률 추세(trend)와 신뢰 구간(confidence interval)을 보여줍니다.
 - (예: 토픽 1 (Council, Govt)은 시간이 지남에 따라 유병률이 증가하는 선형 추세를 보임.)
 - (예: 토픽 3 (Politics)은 시간이 지남에 따라 유병률이 감소하는 선형 추세를 보임.)

8 최적의 토픽 수 (K) 찾기

지금까지 K(토픽 수)를 임의로 ($K=2$, $K=5$) 지정했습니다. 하지만 LDA와 마찬가지로 STM에서 도 최적의 K를 찾는 것은 매우 중요한 하이퍼파라미터 튜닝 과정입니다.

8.1 방법: searchK() 함수

stm 패키지는 searchK() 함수를 제공하여, 여러 K 값에 대한 모델 성능을 한 번에 비교할 수 있게 합니다.

```

1 # K부터 =2 까지 10 모델을 모두 실행하고 성능을 비교
2 k_search_results <- searchK(documents = out$documents,
3                               vocab = out$vocab,
4                               K = c(2, 3, 4, 5, 6, 7, 8, 9, 10),
5                               prevalence = ~ date_numeric,
6                               data = meta_synced)
7
8 # 결과 시각화
9 plot(k_search_results)

```

Listing 3: R 코드: 최적의 K 탐색

주의사항

searchK()는 지정된 모든 K 값에 대해 STM 모델을 (여러 번) 실행하고 평가합니다. 데이터가 크면 매우 오랜 시간 (수 시간 수 일)이 소요될 수 있습니다.

8.2 평가 지표: 일관성(Cohherence) vs. 배타성(Exclusivity)

searchK()는 여러 지표를 보여주지만, 토픽의 품질을 평가하는 데 가장 중요한 두 가지 지표는 다음과 같습니다.

의미론적 일관성 (Semantic Coherence) • 의미: 토픽 내의 상위 단어들이 의미적으로 얼마나 관련성이 높은가?

- 예시: '고양이', '강아지', '애완동물', '먹이' ... → 일관성 높음
- (높을수록 좋음)

배타성 (Exclusivity) • 의미: 토픽들이 서로 얼마나 겹치지 않고 고유한(unique) 단어들을 갖는가?

- 예시: 토픽 A: '고양이', '야옹', '집사' / 토픽 B: '강아지', '멍멍', '산책' → 배타성 높음
- (높을수록 좋음)

최적의 K 선택 전략

- 트레이드오프(Trade-off):** 일반적으로 K가 너무 작으면 일관성은 높지만 배타성이 낮고, K가 너무 크면 배타성은 높지만 일관성이 낮아지는 경향이 있습니다.
- 시각화 및 선택:** X축을 '배타성', Y축을 '의미론적 일관성'으로 하는 2D 플롯을 그립니다. 가장 오른쪽 위 (Top-Right)에 위치하는, 즉 두 지표가 모두 가장 높은 K 값을 선택하는

것이 이상적입니다.

3. 정규화(Rescaling) 팁: 두 지표는 스케일이 매우 다를 수 있습니다 (예: 일관성 9 vs. 배타성 -260). 따라서 각 지표를 [0, 1] 범위로 정규화(rescale)한 뒤, 두 정규화된 값의 평균을 최대화하는 K를 선택하는 것이 합리적인 방법입니다.

9 토픽 해석하기: "토픽 1"은 무엇인가?

모델을 훈련하고 최적의 K를 찾아도, "토픽 1", "토픽 2" 등은 그저 숫자에 불과합니다. 이 토픽들이 실제 어떤 의미를 갖는지 해석하는 과정이 필요합니다.

9.1 방법 1: 상위 단어 확인 (summary() 또는 labelTopics())

- 내용: 각 토픽에서 등장 확률이 가장 높은 단어들을 봅니다.
- 예시: (토픽 1: cat, purr, climb...), (토픽 2: dog, bark, fetch...)
- 단점: (실습 2의 경우) 'australia', 'council', 'govt' 등 모든 토픽에 공통적으로 등장하거나 의미 파악에 도움이 안 되는 단어들이 상위를 차지할 수 있습니다.

9.2 방법 2: 대표 문서 확인 (findThoughts()) - 권장

가장 권장되는 방법입니다. findThoughts() 함수는 각 토픽에 대해 가장 높은 유병률(θ)을 갖는 원본 문서(들)을 직접 보여줍니다.

```

1 # 토픽에 3 대해 가장 대표적인 문서 헤드라인 ( ) 개를 2 보여줌
2 findThoughts(stm_model_news,
3             texts = original_headlines, # 원본 텍스트
4             n = 2,
5             topics = 3)

```

Listing 4: R 코드: 대표 문서로 토픽 해석

• 해석 과정:

- findThoughts()를 실행하여 토픽 3의 대표 헤드라인을 읽습니다.
- (예: "Police investigate crash on highway", "Man charged over stabbing incident")
- 이 문서들을 읽어본 분석가는 "아, 토픽 3은 '사건/사고 및 범죄'에 관한 토픽이구나"라고 수동으로 레이블(label)을 붙일 수 있습니다.

□ 예제: title

findThoughts는 모델의 theta (θ) 행렬을 기반으로 작동합니다.

- theta는 (문서 수 \times 토픽 수) 크기의 행렬입니다.
- theta[d, k] 값은 d번째 문서에서 k번째 토픽이 차지하는 비율(유병률)을 의미합니다.
- findThoughts(..., topics=3)는 theta 행렬의 3번째 열(토픽 3)에서 값이 가장 큰 행(문서)을 찾아, 그 문서의 원본 텍스트를 보여주는 것입니다.

10 용어 정리

용어 (원어)	쉬운 설명	비고 (관련 개념)
LDA (Latent Dirichlet Allocation)	문서가 여러 토픽의 혼합으로 이루어져 있다고 가정하는 확률적 토픽 모델.	메타데이터를 사용하지 못함.
STM (Structural Topic Modeling)	LDA를 확장하여, 문서의 메타데이터가 토픽 비율에 영향을 미친다고 보는 고급 토픽 모델.	R <code>stm</code> 패키지.
메타데이터 (Metadata)	텍스트 자체는 아니지만 텍스트에 대한 정보. (예: 저자, 날짜, 출처)	STM에서는 '공변량(Covariate)'이라고도 부름.
공변량 (Covariate)	분석 대상(토픽 비율)에 영향을 줄 수 있는 외부 변수. (예: X_d)	통계학 용어. 메타데이터와 거의 동일한 의미로 사용됨.
토픽 유병률 (Topic Prevalence)	특정 문서 또는 문서 집합에서 특정 토픽이 차지하는 비율 또는 중요도.	θ (세타) 값.
로지스틱 정규 분포 (Logistic-Normal)	다면량 정규 분포에서 샘플링한 벡터를 Softmax 함수에 통과시켜 합이 1인 확률 벡터를 얻는 분포.	STM에서 θ_d 를 생성하는 방식.
최대 가능성 추정 (MLE, Max. Likelihood Est.)	관측된 데이터를 가장 그럴듯하게 설명하는(생성할 확률이 가장 높은) 모델 파라미터를 찾는 통계적 방법.	"젖은 고양이" 비유.
확률적/비결정론적 (Stochastic)	무작위성을 포함하므로 실행할 때마다 결과가 달라질 수 있음.	LDA, STM 모두 해당.
의미론적 일관성 (Semantic Coherence)	토픽 내의 단어들이 의미적으로 얼마나 일관된지를 나타내는 지표.	(높을수록 좋음)
배타성 (Exclusivity)	토픽들이 서로 얼마나 겹치지 않고 고유한 단어들로 구성되었는지를 나타내는 지표.	(높을수록 좋음)
NMF (Non-negative Matrix Fact.)	음수 미포함 행렬(V)을 두 개의 작은 행렬($W \times H$)의 곱으로 분해하는 기법.	LDA와 달리 확률 모델이 아님.

Table 2: 주요 용어 정리

11 FAQ (주요 질문 및 답변)

Q: STM을 꼭 R로만 해야 하나요? Python은 없나요?

A: `stm` 패키지는 R에서 개발되었고, 가장 많은 기능과 안정성을 제공하며 학계에서도 표준으로 사용됩니다. Python에 일부 구현체가 존재하긴 하지만(GitHub 등), R 패키지만큼 신뢰할 수 있거나 기능이 풍부하지 않은 경우가 많습니다.

본 강의에서는 안정적인 분석과 `estimateEffect` 같은 강력한 효과 추정 기능을 위해 R 사용을 권장합니다.

Q: `searchK()`를 실행하는 데 시간이 너무 오래 걸립니다.

A: 정상입니다. `searchK()`는 (K 의 개수 \times 실행 횟수) 만큼 STM 모델을 처음부터 끝까지 훈련시킵니다.

실제 분석에서는 (1) 데이터의 일부를 샘플링하여 빠르게 K 의 범위를 좁히거나, (2) 밤새도록 또는 며칠간 실행할 것을 예상해야 합니다.

Q: 메타데이터로 날짜를 사용할 때, '연속형'과 '범주형' 중 무엇이 더 좋은가요?

A: 정답은 없습니다.

- **연속형 (예: `year + (m-1)/12`):** "토릭이 시간에 따라 선형적으로 증가/감소한다"는 강한 가정을 합니다. 트렌드를 부드럽게 볼 수 있지만, 계절성이나 특정 이벤트(예: 선거)로 인한 급격한 변화를 놓칠 수 있습니다.
- **범주형 (예: `as.factor(year)`):** "연도별로 토릭 비율이 자유롭게 다를 수 있다"고 가정합니다. 더 유연하지만, 해석이 복잡해지고 더 많은 데이터가 필요합니다.

분석의 목적에 맞게 선택해야 합니다. "장기적 트렌드"를 보려면 연속형, "특정 연도의 차이"를 보려면 범주형이 적합할 수 있습니다.

Q: 전처리 후 문서가 50개나 사라졌습니다. 괜찮은가요?

A: 네, 괜찮습니다. 뉴스 헤드라인처럼 매우 짧은 텍스트는 불용어, 숫자, 구두점 등을 제거하고 나면 내용이 완전히 비게 되는 경우가 흔합니다.

`textProcessor`와 `prepDocuments`는 이런 빈 문서들을 자동으로 제거해 주므로 편리합니다.

단, `out$meta`를 사용해야 한다는 점을 절대 잊지 마세요. (데이터 정합성)