Spoiler Detector

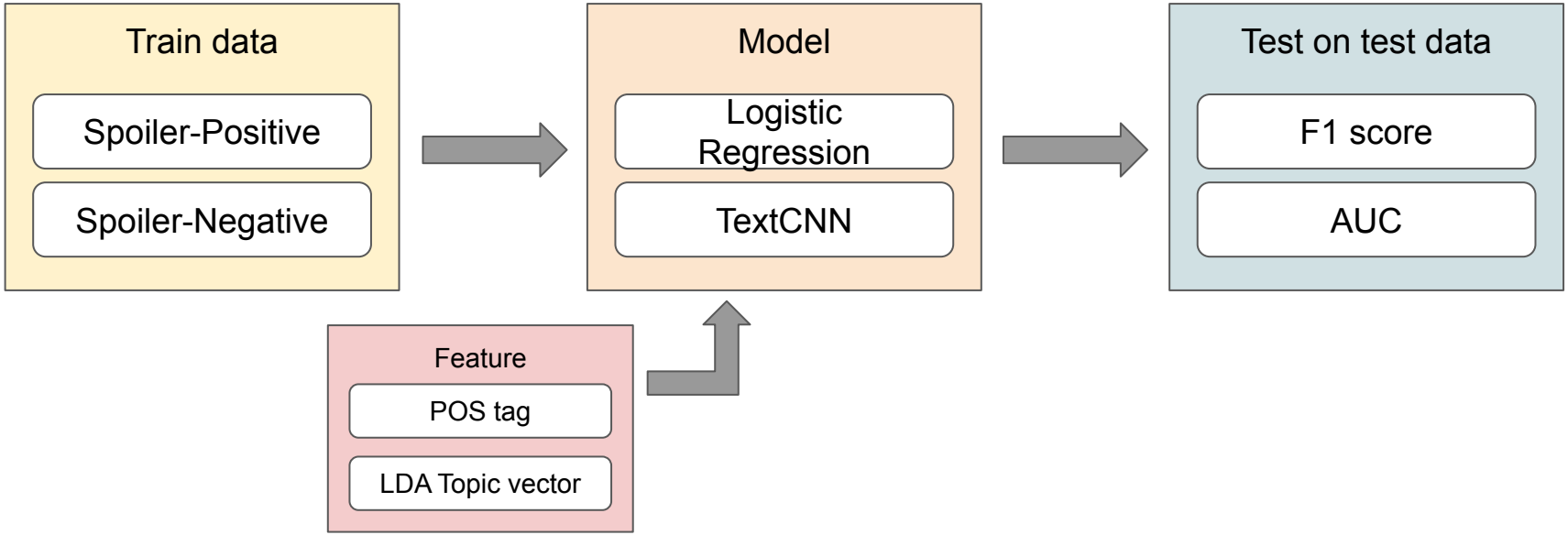
Team 2442

목차

- 개요
- 관련 연구
- 접근 방법
  - Logistic Regression
  - Topic LDA
  - textCNN
- 실험 및 결과
- 결론

개요

- 영화 리뷰에서 스포일러의 포함 여부를 분류 하여 스포일러가 포함된 리뷰를 탐지
  - Classify spoiler-positive / spoiler-negative
- Spoiler-positive : 스포일러가 포함 된 영화 리뷰 + 영화의 시놉시스(줄거리)
- Spoiler-negative : 스포일러가 포함되지 않은 순수 감상 리뷰



관련 연구(1/2)

- Simple way
  - Keyword-matching method[1]
    - TV 프로그램(드라마, 스포츠 경기,...)에 대한 스포일러를 탐지
    - 드라마의 배우 이름이나 스포츠 경기의 경기 결과와 같은 단어들이 들어간 문장들을 스포일러로 간주함
    - 단순한 방법이지만 precision이 매우 낮음

관련 연구(2/2)

- Using Machine Learning method
  - Based on SVM model[2]
    - TV프로그램에 대한 스포일러를 관련된 트위터에서 탐지
    - SVM 모델을 사용하여 스포일러 포함 여부를 분류
    - 트위터 데이터에 대한 특징(이모티콘, short-size sentences, 해시태그)들을 반영
  - Based on LDA model[3]
    - Topic 모델의 일종인 LDA 모델을 사용
    - 영화의 시놉시스와 리뷰 데이터를 학습하여 얻은 topic들의 predictive perplexity를 비교하여 유사도를 측정
    - dependency 정보를 모델 학습의 feature로 사용
    - 본 프로젝트에서는 LDA모델을 통해 나온 topic vector들을 학습의 feature로 사용

접근 방법(1/4)

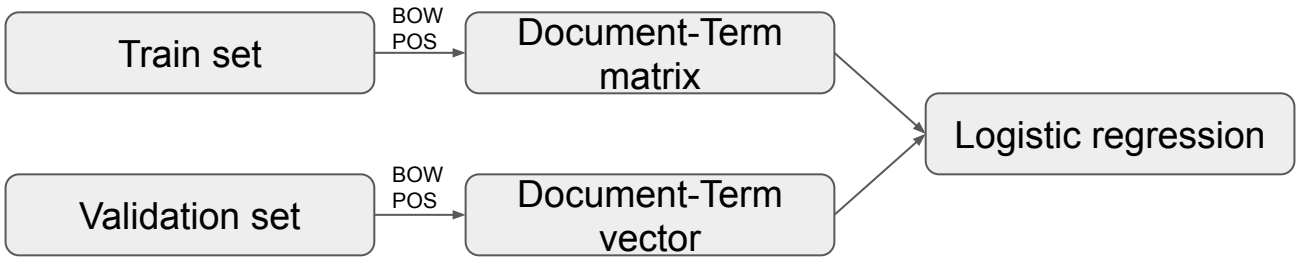
- Logistic regression with BOW/POS
- Topic vector with LDA(Latent Dirichlet Allocation)
- TextCNN

접근 방법(2/4)

Logistic Regression with BOW/POS

Bag of word를 사용하여 document-term matrix를 만들고, 이를 통해 word count를 가지고 review의 spoiler 여부를 판단하였다.

Bag of word에 POS tag를 붙여서 별도의 실험을 진행하였다.



접근 방법(2/4)

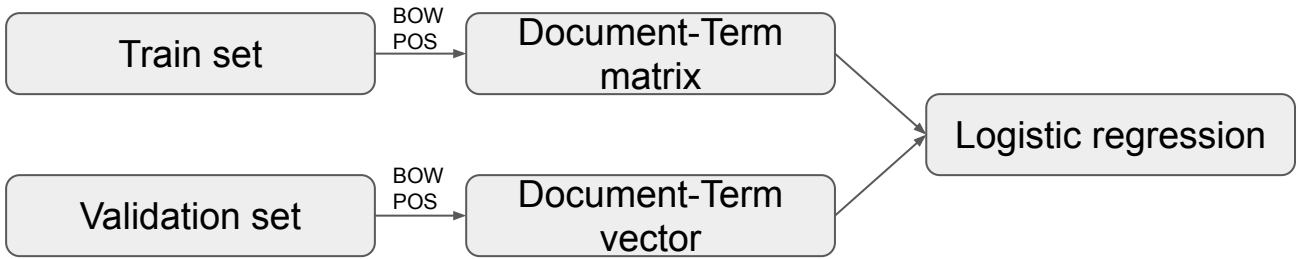
Logistic Regression with BOW/POS

Bag of word를 사용하여 각 단어를 word count vector로 인코딩함

LR모델의 입력으로 document-term matrix를 주고 2-class 분류를 하도록 학습

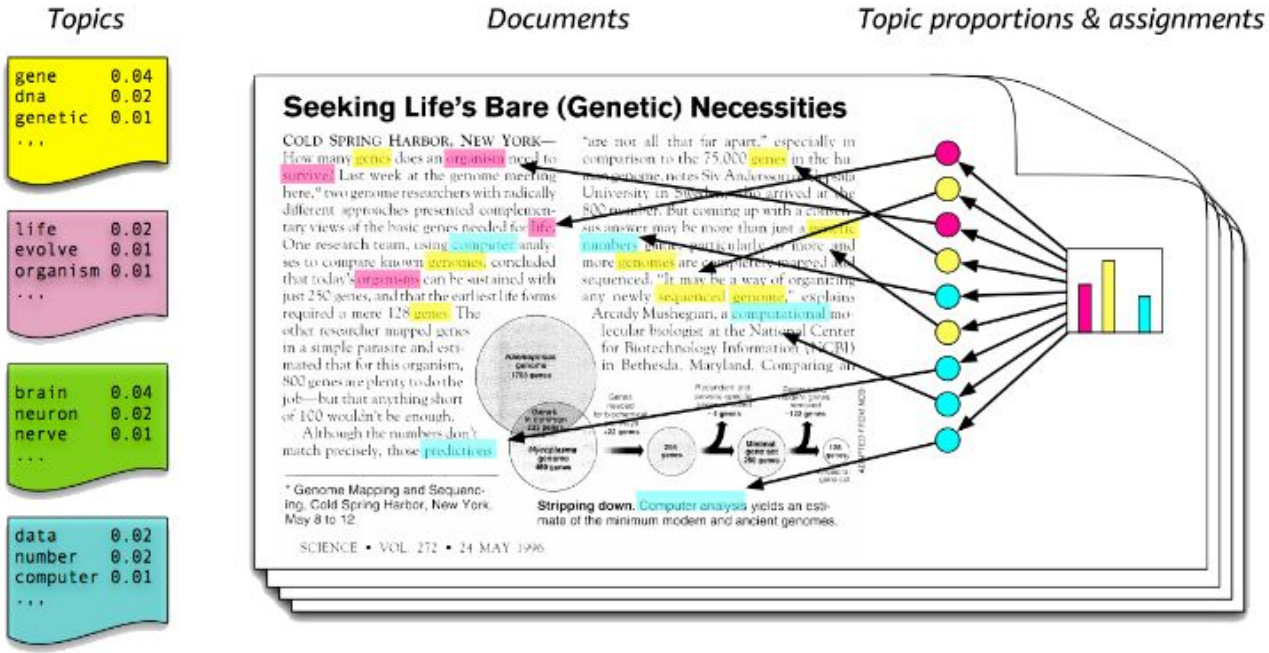
→ Spoiler-Positive / Spoiler-Negative 분류

단어에 POS tag를 붙여 학습 진행



접근 방법(3/4)

Topic modeling by LDA(Latent Dirichlet Allocation)



접근 방법(3/4)

Topic modeling by LDA(Latent Dirichlet Allocation)

$$p(z_{d,i} = j | z_{-i}, w) = \frac{n_{d,k} + \alpha_k}{\sum_{i=1}^{K}(n_{d,i} + \alpha_i)} \times \frac{v_{k,w_{d,n}} + \beta w_{d,n}}{\sum_{i=1}^{V} v_{k,j} + \beta_j}$$

말뭉치를 분석하여 Topic vector로 표현

1. 토픽의 개수를 선정(k)
2. 단어별로 토픽을 랜덤하게 할당 (1,...,k)
3. 문서의 토픽분포와 토픽의 단어분포를 결합하여 문서내 단어를 추정하도록 학습
4. 학습과정을 거쳐 각 단어는 크기가 k인 Topic vector로 표현되어짐

접근 방법(3/4)

Topic modeling by LDA(Latent Dirichlet Allocation)

LDA로 얻은 Topic vector를 활용하여 3가지 실험을 진행

- Cosine distance: 리뷰와 영화 시놉시스의 유사도 측정
- LR의 입력으로 Topic vector만 사용
- LR의 feature로 topic vector를 추가

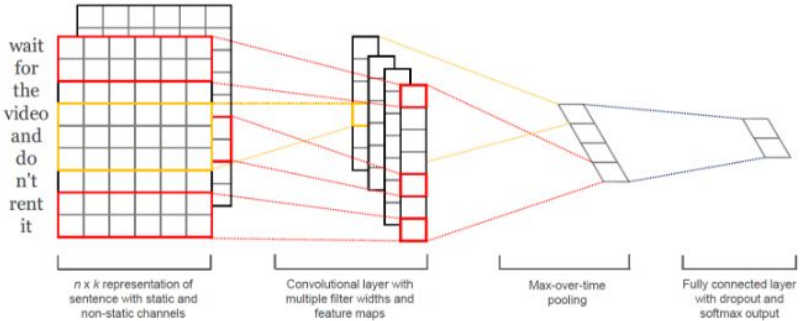
접근 방법(4/4)

TextCNN (Yoon Kim, 2014.)

Dataset에서 word embedding을 이용하여 Word2Vec으로 먼저 변환하였다.

이를 CNN 모델에 적용하여 학습하였는데, 모델의 hyper-parameter들은 training set의 development set을 따로 만들어 적용하여 최적의 모델을 만들도록 하였다.

실험 진행은 POS tag를 사용하지 않은 문장, 사용한 문장을 분리하여 총 2가지의 실험을 진행하였다.



실험

Dataset: 신과 함께, 테이큰, 설국열차, 인터스텔라 의 리뷰, 시놉시스 데이터

- 각각의 영화에 3000문장 내외로 spoiler tagging을 수행하여 golden set 제작

Movie Name	Total Sentences	Spoilers
Along With the God	2999	206
Interstella	5189	1224
Snow Piercer	3287	1197
Taken	3186	956

Table 1: Golden set.

실험

Dataset: 신과 함께, 테이큰, 설국열차, 인터스텔라 의 리뷰, 시놉시스 데이터

- 각각의 영화에 3000문장 내외로 spoiler tagging을 수행하여 golden set 제작

Evaluation: 9:1 비율로 train set과 validation set을 나누어 실험을 진행하였고, 각각의 실험마다 10번의 평균 score를 구하였다.

실험 결과

Movie	#. of topics	BOW	BOWPOS	Cosine	Topic	BOW&Topic	CNN	CNNPOS
A	2	0.682	0.894	0.236	0.229	0.868	0.567	0.722
	10			0.186	0.255	0.886		
	50			0.195	0.352	0.904		
	100			0.148	0.391	**0.915**		
T	2	0.455	0.839	0.446	0.545	0.830	0.641	0.741
	10			0.482	0.655	0.837		
	50			0.478	0.661	0.849		
	100			0.515	0.677	**0.864**		
S	2	0.552	0.841	0.482	0.578	0.833	0.747	0.764
	10			0.346	0.615	0.841		
	50			0.434	0.648	0.851		
	100			0.443	0.670	**0.865**		
I	2	0.494	0.832	0.203	0.438	0.826	0.725	0.779
	10			0.313	0.555	0.836		
	50			0.163	0.593	0.846		
	100			0.175	0.642	**0.849**		

Table 2: Experiment results in F1 score. (A=Along with the gods, T=Taken, S=Snow piercer, I=Interstella)

실험 결과

Movie	#. of topics	BOW	BOWPOS	Cosine	Topic	BOW&Topic	CNN	CNNPOS
A	2	0.759	0.915	0.617	0.571	0.902	0.715	0.802
	10			0.591	0.573	0.912		
	50			0.586	0.611	0.924		
	100			0.546	0.628	**0.934**		
T	2	0.642	0.874	0.495	0.695	0.870	0.739	0.806
	10			0.558	0.746	0.872		
	50			0.586	0.751	0.883		
	100			0.623	0.760	**0.894**		
S	2	0.674	0.867	0.573	0.668	0.863	0.795	0.808
	10			0.516	0.699	0.868		
	50			0.565	0.725	0.876		
	100			0.578	0.740	**0.886**		
I	2	0.663	0.871	0.522	0.647	0.871	0.792	0.837
	10			0.580	0.694	0.875		
	50			0.515	0.715	0.881		
	100			0.530	0.744	**0.886**		

Table 3: Experiment results in AUROC score. (A=Along with the gods, T=Taken, S=Snow piercer, I=Interstella)

결론

POS tagging은 아주 강력한 feature이다.

LDA topic vector를 feature로 추가했을 때 좋은 성능을 보여 주었다.

실험 dataset이 크지 않아 CNN이 큰 힘을 내기 어려웠고, 비교적 단순한 모델인 logistic regression이 좋은 성능을 보여 주었다.

References

[1] S. Nakamura and K. Tanaka, “Temporal filtering system to reduce the risk of spoiling a user’s enjoyment,” *in Proceedings of the 12th International Conference on Intelligent User Interfaces*, ser. IUI ’07. New York, NY, USA: ACM, 2007, pp. 345–348.

[2] S. Jeon, S. Kim, and H. Yu, “Spoiler detection in tv program tweets,” *Information Sciences*, vol. 329, pp. 220 – 235, 2016, special issue on Discovery Science.

[3] S. Guo and N. Ramakrishnan, “Finding the storyteller: Automatic spoiler tagging using linguistic cues,” *in Proceedings of the 23rd International Conference on Computational Linguistics*, ser. COLING ’10. Stroudsburg, PA, USA: Association for Computational Linguistics, 2010, pp. 412–420

[4] Y. Kim, “Convolutional neural networks for sentence classification,” CoRR, vol. abs/1408.5882 2014.

End