유사도 기반 커널을 활용한 SVM 의 텍스트 분류 성능 개선

이상혁
성균관대학교 통계학과
leesanghyuk2000@gmail.com

1. 서 론

최근 인공지능 모델이 고도화됨에 따라 텍스트 분류(Text Classification) 분야는 Transformer, BERT 와 같은 복잡한 딥러닝(Deep Learning) 기반의 모델을 바탕으로 발전했다. 그러나 딥러닝의 경우 연산량이 많기 때문에 컴퓨팅 파워를 많이 소모한다. 이에 따라 전통적인 머신러닝 모델(ex. KNN, Logistic Regression, SVM 등)을 통해서 텍스트 분류 문제를 해결할 수 있다는 아이디어를 바탕으로 본 프로젝트를 진행하고자 한다. 이때 전통적인 머신러닝 모델은 자연어 처리(Natural Language Processing) 에 적합한 모델은 아니다. 따라서 자연어를 전통적인 머신러닝 모델의 입력 값으로 만들어주는 커널(Kernel)을 정의하고자 한다. 특히 커널을 기반으로 최적화를 거쳐 분류를 진행하는 모델인 Support Vector Machine 에 자연어의 특징을 반영할 수 있는 커널을 적용하여 텍스트 분류 성능을 개선해보고자 한다. 기존 연구에서는 문자열 커널(String Kernel) (Lodhi, H et al. 2000)을 활용하여 SVM 의 텍스트 분류 성능을 개선했다. 본 프로젝트에서는 문자열 커널을 소개한 논문의 아이디어에 착안하여 이와 유사한 커널을 정의하고, 실험을 통해 정의한 커널을 적용한 SVM 의 텍스트 분류 성능을 실험하고자 한다.

2. 방 법

일반적으로 전통적인 머신러닝 모델로 텍스트 분류를 수행하기 위해서는 자연어를 벡터로 바꿔주는 과정이 필요하다. 텍스트 임베딩(Text Embedding) 혹은 TF-iDF 와 같은 방식이 주로 사용된다. 이때 커널을 적용하여 자연어의 특성을 반영하면서 이를 벡터로 만들 수 있다. 먼저 Hu, Lodhi et al. (2000)에서 제안한 문자열 커널의 경우는 다음과 같이 정의된다.

$$K(s,t) = \sum_{u \in \sum^k} \phi_u(s) \cdot \phi_u(t)$$

[수식 1] 문자열 커널

문자열 커널은 문서를 연속된 문자 시퀀스로 보고, 특정 길이의 서브시퀀스를 비교하여 문서 간 유사성을 측정하는 방법이다. 따라서 문자열 커널을 통해 두 문자열 간의 공통 서브시퀀스를 찾고, 그 빈도와 연속성을 고려하여 가중치를 부여하는 방법이다.

본 프로젝트에서는 참고한 문헌에서 제안된 문자열 커널의 아이디어를 착안하여 TF-IDF 행렬을 활용하는 가중 자카드 유사도 커널(Weighted Jaccard Similarity Kernel)을 활용하여 SVM 의 텍스트 분류 성능을 개선하고자 한다. 가중 자카드 유사도 커널 다음과 같이 정의하고자 한다.

$$K_{WJ}(x,y) = \frac{\sum_{i=1}^{n} \min(x_i,\ x_y)}{\sum_{i=1}^{n} \max(x_i,\ x_y)}$$

[수식 2] 가중 자카드 유사도 커널

이때 x_i 와 y_i 는 s 각각 TF-IDF 로 벡터화된 두 문서의 단어 빈도이다. 또한 자카드 유사도를 측정하기 위해 분모에는 두 벡터에서 각 단어의 최대값이, 분자에는 두 벡터에서 각 단어의 최소값이 오게 된다. 즉 전체 서브 시퀀스 중에서 공통 단어에 집중하여 이에 가중치를 부여하는 커널 함수이다.

자연어를 자카드 유사도 행렬로 매핑하는 예시는 다음과 같다.

문서 1	*"I love machine learning"*
문서 2	*"machine learning is fun"*
문서 3	*"deep learning and AI"*
문서 4	*"I love AI"*

[표 1] 문서 예시

[표 1]과 같이 4 개의 문서가 있다고 했을 때, TF-IDF 행렬은 다음과 같다.

ai	and	deep	fun	is	learning	love	machine
0.0	0.0	0.0	0.0	0.0	0.4968	0.6136	0.6136
0.0	0.0	0.0	0.5745	0.5745	0.3667	0.0	0.4530
0.4530	0.5745	0.5745	0.0	0.0	0.3667	0.0	0.0
0.7071	0.0	0.0	0.0	0.0	0.0	0.7071	0.0

[표 2] 4 개 문서에 대한 TF-IDF 행렬

TF-IDF 를 통해 각 문서를 벡터화하고, 가중 자카드 유사도 커널을 적용하여 다음과 같이 매핑한다.

	문서 1	문서 2	문서 3	문서 4
문서 1	1.0	0.2852	0.1102	0.2430
문서 2	0.2852	1.0	0.1026	0.0
문서 3	0.1102	0.1026	1.0	0.1546
문서 4	0.2430	0.0	0.1546	1.0

[수식 3] 가중 자카드 유사도 행렬 예시

이를 통해 단어의 중요도와 문서 사이의 관계를 반영하여 SVM 이 레이블과 문서의 관계를 학습하도록 하여 텍스트 분류 성능을 최적화하고자 한다.

3. 실험 계획

3-1. 실험 1

첫번째 실험은 가중 자카드 유사도 커널의 적용한 SVM 의 다중 분류 성능을 검증한다. "scikit-learn"의 datasets API 에서 제공하는 "fetch_20newsgroups" 데이터셋을 활용한다. 해당 데이터셋은 약 18,000 개의 샘플로 이뤄져 있고, 20 개의 카테고리로 나뉘어져 있다. datasets API 공식 문서에서는 해당 데이터셋을 분류 문제에 적용할 경우, 몇가지 메타데이터를 제거하고 진행할 것을 권장하기 때문에 이 또한 반영한다. 단 컴퓨팅 자원의 한계로 학습 데이터는 클래스 비율을 고려하여 10,000 개를 랜덤샘플링하여 진행하고자 한다.

모델의 성능을 검증하기 위해 다음의 모델의 성능과 비교하고자 한다.

Target 모델	SVM with weighted jaccard similarity kernel
비교 모델	SVM with linear kernel
	Logistic Regression
	Multinomial Naive Bayes
	SGD Classifier
	K-Nearest Neighbors (K=3, 5)
	Decision Tree
	Random Forest

[표 3] 실험 1 에 대한 타겟&비교 모델

이때 비교 모델군의 경우 가중 자카드 유사도 커널의 효과를 검증해야 하기 때문에 텍스트 데이터에 TF-IDF 만 적용하여 벡터화를 진행한다.

각 모델 간의 성능을 비교하기 위해 "Accuracy", "Precision", "Recall", "F1 Score", "Training Time(s)", "Prediction Time(s)"를 지표로 활용한다.

3-2. 실험 2

두번째 실험은 가중 자카드 유사도 커널의 적용한 SVM 의 이진 분류 성능을 검증한다 "keras"의 "imdb" 데이터셋을 활용한다. 해당 데이터셋은 영화 리뷰에 대한 긍/부정을 예측하는 데 활용되며 50,000 개의 데이터로 구성되어 있다. 또한 가장 빈도수가 높은 10,000 개의 단어로 학습 데이터를 구성한다. 단 컴퓨팅 자원의 한계로 빈도수와 학습 데이터는 클래스 비율을 고려하여 10,000 개를 랜덤샘플링하여 진행하고자 한다.

비교 모델과 성능 지표는 [실험 1]과 동일하다.

3-3. 실험 3

세번째 실험은 imdb 데이터셋의 num_words 와 데이터 개수를 변경하며 가중 자카드 유사도 커널의 성능이 가장 높을 때의 유사도 행렬의 형상, 데이터셋의 크기 등을 분석하고자 한다.

성능 지표는 [실험 1]과 동일하다.

4. 자유 논의

일반적으로 TF-IDF 와 같은 방식으로 텍스트를 벡터화하면 희소성(Sparsity)으로 인해 선형적인 분리가 쉬워진다. 이때 선형 SVM 과 로지스틱 회귀와 같은 모델보다 가중 자카드 유사도 커널을 적용한 SVM 의 성능에 유의미한 개선이 있는지 검토하는 것이 본 프로젝트의 핵심이라고 생각한다. 이를 통해 문서 사이의 관계와 단어의 관계를 SVM 이 적합하게 학습할 수 있는지 확인한다. 또한 컴퓨팅 자원의 한계로 각각 18,000 개, 50,000 개로 이루어진 데이터셋을 모두 활용하기는 어려울 것으로 예상된다. 따라서 랜덤 샘플링을 통해 같은 크기의 데이터셋을 각각 10,000 개씩 사용하고자 한다.

참고문헌

[1] Lodhi, H., Shawe-Taylor, J., Cristianini, N., & Watkins, C. (2000). Text classification using string kernels. Advances in neural information processing systems, 13.