차원 축소를 활용한 SVM 딥페이크 음성 분류 모델 연구

팀명: hongchoi04

홍승혁*, 최수환**
인천대학교 *컴퓨터공학부, **컴퓨터공학부
sdbrandon04@inu.ac.kr. ghks3691@inu.ac.kr

요 약

본 연구는 DFCC(DeepFake speech Classification Challenge)를 위한 SVM 모델에 관한 연구로, 딥페이크 음성을 정확하게 분류하는 목적을 가지고 있다. 효과적인 음성 데이터 피쳐 추출을 위해 MFCC, CQCC를 사용했고, 전처리 과정에서 StandardScaler, KernelPCA, LDA로 차원 데이터를 축소하고 데이터를 정규화했다. RBF 커널을 사용한 SVC를 이용해 SVM를 설계하여 96.2%의 정확도를 기록하였다.

1. 서론

본 연구는 딥페이크 음성과 실제 음성을 효과적으로 분류하는 SVM 기반 모델에 관한 연구이다. 진행 과정은 효과적인 데이터 전처리, 특징 추출, 모델링 및 평가의 단계로 진행되었다. 최종적으로 모델에 적용한 기법들 이외에도 LFCC, CQT 등의 다양한 음성 데이터 특징 추출 방법과 PCA, 학습 프로세스 과정에서의 Real, Fake 데이터를 분리하여 각각 최적의 파이프라인을 구축하여 학습시키는 등의 시행착오를 거쳐 후술할 모델을 구축하였다.

2. 방법론

2-1. 특징 추출

모델 학습에 사용할 특징을 추출하기 위해서 librosa 라이브러리로 오디오 신호를 로딩하고, 음성의 주요 특징을 효과적으로 추출하기 위해 MFCC와 CQCC의 두 가지 피쳐를 함께 사용했다.

MFCC

입력 데이터의 스펙트럼에 Mel-scale filter bank를 적용하고, log를 씌워 DCT(Discrete Cosine Transform)을 수행한 피쳐로 [1], 음성 인식을 위해 사용된다. 여러 시도를 통한 최적화 과정을 통해 가장 효과를 보인 MFCC 계수를 30으로 설정하고, 시간적 변화를 반영하기 위해 delta, delta-delta 계수를 추가해 각 피쳐의 평균과 표준편차를 사용했다. 또한 스펙트럼을 부드럽게 하고 노이즈에 대한 민감도를 낮추는 liftering을 적용한 MFCC도 활용하였다.

CQCC

딥페이크와 같은 음성 변조 탐지에 이용되는 피쳐로, 입력 신호에 CQT(Constant-Q Transform)를 적용하여 스펙트럼을 얻고, 로그를 취한 뒤 DCT를 적용한다. CQCC의 경우 librosa에 구현되어 있지 않아 추출 과정을 참고하여 직접 함수로 구현해 사용했다 [2]. 1차 및 2차 미분계수를 계산해 사용했다. 각 특징의 평균과 표준편차를 최종 피쳐 벡터에 포함했다.

```python
# CQCC feature extraction function
def compute_cqcc(y, sr, n_cqcc=15, hop_length=512):
    cqt_spec = np.abs(librosa.cqt(y=y, sr=sr, n_bins=84,
                                  bins_per_octave=12, hop_length=hop_length))
    log_cqt = np.log(cqt_spec + 1e-6)
    N_bins, _ = log_cqt.shape
    n = np.arange(N_bins)
    k = np.arange(n_cqcc)[:, None]
    dct_basis_cqcc = np.cos(np.pi * k * (2*n + 1) / (2 * N_bins))
    cqcc = dct_basis_cqcc.dot(log_cqt)
    return cqcc
```

그림 1. CQCC 구현 함수(model.py)

2-2. 데이터 전처리

특징 추출 단계를 거쳐 추출된 데이터는 고차원의 특징 벡터로 표현되기에 이를 직접적으로 활용하는 것이 어려워 모델이 효과적으로 학습할 수 있도록 다음의 3단계의 과정을 통해 데이터를 전처리하였다.

StandardScaler

각 특징을 평균 0, 표준편차 1로 변환해 스케일하는 과정으로 [3], 각 특징의 단위를 표준화하여 모든 특징이 동등한 중요도로 학습에 반영되도록 유도했다.

KernelPCA

RBF 커널을 사용하여 비선형 데이터를 고차원 공간으로 매핑한 후 PCA를 적용한다 [4]. PCA는 음성 데이터의 비선형적인 특성을 잘 반영하지 못하는 경향을 보여 KernelPCA를 적용했다.

LDA(Linear Discriminant Analysis)

클래스 외부의 분산은 최대화, 클래스 내부의 분산은 최소화하도록 유도한다. [5]. 분류 작업에 있어서 가장 중요한 정보만 남기면서 차원을 축소했다.

2-3. SVM 기반 분류 모델

SVM을 효과적으로 이용하기 위해 비선형 데이터 분류의 강점을 보이는 RBF 커널을 SVC의 커널로 적용했다 [6].

초기 모델에서 보인 Fake 데이터에 대한 분류 성능이 떨어지는 불균형 문제를 해결하기 위해 클래스별 가중치를 조절했다. Fake 클래스에 9배 높은 가중치를 부여해 모델이 Fake 데이터를 오분류할 경우 더

큰 페널티를 받도록 설계했고, 그 결과 분류 성능의 불균형 문제를 해결하였다.

효과적인 하이퍼파라미터 조합을 찾기 위해 sklearn의 GridSearchCV를 추가했다 [7]. 하이퍼파라미터 후보군 데이터를 분리 후 3회의 훈련-검증 절차를 반복하는 3겹 교차 검증을 수행하며 각 과정의 정확도를 측정해, 각 최고 정확도를 가진 모델을 최종 모델로 선택하도록 구성하였다. 최종 모델은 model.pkl 파일로 저장해 test 단계에서 train 과정을 반영할 수 있도록 설계했다.

3. 실험 및 분석

3-1. 실험 환경

아래와 같은 환경에서 실험이 진행되었다.
OS : Windows 11
하드웨어 : Intel Core Ultra 5 CPU, 16GB RAM
개발 도구 : Python 3.10, Jupyter Notebook
라이브러리 : librosa, scikit-learn, pickle, numpy, os
파라미터 개수 & 모델 사이즈: 51개, 11.4MB

3-2. 실험

학습과 테스트를 진행하는 train.ipynb, test.ipynb와 데이터 전처리와 모델 파이프라인을 담은 model.py, 학습을 진행한 모델 파일인 model.pkl로 모델을 구성했다. Train, Test 실험 과정은 다음과 같다.

1. train.ipynb에서 train 데이터에 대해 특징 추출, 전처리
2. GridSearchCV를 위한 하이퍼파라미터 그리드 정의 및 model.py에 구현한 모델 파이프라인에 GridSearchCV를 적용해 학습 진행
3. 가장 성능이 높은 하이퍼파라미터를 채택하여 model.pkl 생성
4. test.ipynb 파일에서 학습된 모델(model.pkl) 로드
5. test 데이터에 대해 학습 시와 동일한 전처리 수행
6. test 데이터에 대한 모델의 예측 진행

3-3. 실험 결과

전체 테스트 데이터에 대해 정확도 96.2%를 달성하며 분류 성능을 입증했다.

```
========== Results Analysis ==========
Test: ./choihong04_test_result.txt
True: ./2501ml_data/label/test_label.txt
Accuracy: 96.20%
Hit: 1924, Total: 2000
======================================
```

그림 2. 테스트 Accuracy

3-4. 제거 실험(Ablation)

모델의 성능에 영향을 준 요소들을 단계적으로 확인하기 위해 MFCC를 적용한 SVM baseline 모델로부터 최종 모델의 모든 구성 요소를 점진적으로 추가하며 각각의 효과를 검증하였다.

구성 요소	Accuracy(%)	F1-Score(%)
MFCC + SVM (Baseline)	59.80	52.76
+CQCC	59.30	51.22
+StandardScaler	87.30	87.10
+KPCA	86.50	86.25
+LDA	94.40	94.39
+SVM 가중치 적용 (Final)	**96.20**	**96.20**

표 1. Ablation study 결과

3-5. 학습률 곡선

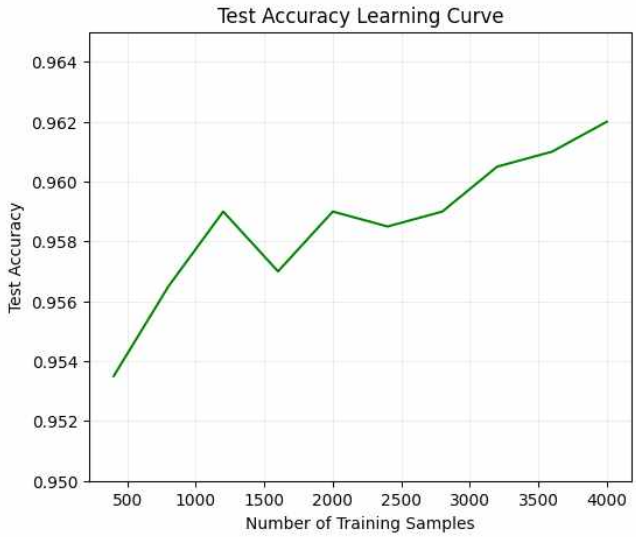

그림 3. SVM 기반 분류 모델이기에 train data 수에 따른 정확도에 대해 시각화하였다.

4. 결론

음성 데이터를 클래스별로 뚜렷한 분포와 경계를 가질 수 있도록 상술한 방법론을 사용했고, 그 결과로 96.2%의 정확도를 보였다. 성능을 향상함과 동시에 ablation을 진행하여 요소별 모델에 미치는 영향을 확인하였고, 학습률 곡선에서 적은 양의 데이터로 학습이 가능한 SVM의 특징을 확인하였다.

5. 참고문헌

[1] https://librosa.org/doc/0.11.0/generated/librosa.feature.mfcc.html
[2] M. Todisco, H. Delgado, and N. Evans, "Constant Q cepstral coefficients: A spoofing countermeasure for automatic speaker verification," Computer Speech & Language, vol. 64, 101035, 2020.
[3] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html
[4] https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.KernelPCA.html
[5] https://scikit-learn.org/stable/modules/generated/sklearn.discriminant_analysis.LinearDiscriminantAnalysis.html
[6] https://scikit-learn.org/stable/modules/svm.html
[7] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html
[8] ChatGPT