Journal of Korea Multimedia Society Vol. 21, No. 3, March 2018(pp. 391-399)
https://doi.org/10.9717/kmms.2018.21.3.391

인공신경망 기반의 기타 코드 분류 시스템 성능 비교

박선배[†], 유도식[††]

Performance Comparison of Guitar Chords Classification Systems Based on Artificial Neural Network

Sun Bae Park[†], Do-Sik Yoo[††]

ABSTRACT

In this paper, we construct and compare various guitar chord classification systems using perceptron neural network and convolutional neural network without pre-processing other than Fourier transform to identify the optimal chord classification system. Conventional guitar chord classification schemes use, for better feature extraction, computationally demanding pre-processing techniques such as stochastic analysis employing a hidden markov model or an acoustic data filtering and hence are burdensome for real-time chord classifications. For this reason, we construct various perceptron neural networks and convolutional neural networks that use only Fourier tranform for data pre-processing and compare them with dataset obtained by playing an electric guitar. According to our comparison, convolutional neural networks provide optimal performance considering both chord classification acurracy and fast processing time. In particular, convolutional neural networks exhibit robust performance even when only small fraction of low frequency components of the data are used.

Key words: Audio Signal Processing, Artificial Neural Network, Convolutional Neural Network, Pattern Recognition

1. 서 론

기타는 6개의 현으로 이루어진 악기로 각 현마다 다른 음을 내기가 쉬워 코드(chord)기반의 연주가 주가 되는 악기이다. 코드란 음을 수직으로 쌓아 동시에 연주하여 연주를 풍성하게 만드는 음악적 요소로 기본적으로 1도, 3도, 5도 음으로 구성되어 있으며, 각 음의 음정에 따라 다른 분위기와 느낌을 만들어낸다. 추가적으로 4번째 음을 쌓아서 코드의 느낌을 더욱 다양하게 만들 수 있다.

시스템이 자동적으로 각 코드들을 분류하는 방법에는 전통적으로 주파수를 분석하여 어느 음을 사용했는지 직접 판별하는 방법이 있다. 하지만 기타는 줄의 상태, 운지 위치, 운지하는 손가락의 힘 등에 의해 음의 주파수가 수 % 차이가 발생하며 저주파 성분에서는 각 음마다 주파수의 차이가 크지 않기

※ Corresponding Author : Do-Sik Yoo, Address: (04066) Wausan-Ro 94, Mapo-Gu, Seoul, Korea, TEL : +82-2-320-3048, FAX : +82-2-320-1193, E-mail : yoodosik@hongik.ac.kr
Receipt date : Dec. 8, 2017, Revision date : Jan. 23, 2018
Approval date : Feb. 10, 2018
[†] Dept. of Electronic and Electrical Engineering., Graduate School, Hongik University
(E-mail : sun882125@gmail.com)
[††] Dept. of Electronic and Electrical Engineering., Graduate School, Hongik University
※ This research was supported by Basic Science Research Program through the National Research Foundation of Korea funded by the Ministry of Education under grant NRF-2015R1D1A1A01060234.

때문에 이를 구분하기는 더 어려워 이러한 방법은 좋은 성능이 나오기 힘들다.

따라서 현재의 기타 코드 분류 시스템에서는 연주음을 은닉 마르코프 모델(hidden markov model) 등을 이용한 확률적 분석 기반의 특징 추출 기법 [1-3], 12개의 음의 하모닉스 성분들을 한데 모아 특징들을 간략하게 압축하는 방법[4,5] 등으로 주어진 신호를 전처리하여 분석하는 방법을 사용한다. 그러나 이러한 방법은 매우 복잡한 전처리과정을 거친 후 분류를 해야하는 단점이 있다. 특히 최근 각광받고 있는 기계학습을 적용하여 분류성능을 높이기 위해선 수많은 데이터가 필요한데 이러한 복잡한 전처리과정을 거쳐야 한다면 전체 처리시간이 증가하게 된다.

본 논문에서는 전처리과정을 최소화하고 기타 코드음 분류에 적합한 몇 가지 인공신경망 알고리듬을 이용한 분류 시스템을 구성하고, 직접 구성한 데이터셋을 이용하여 분류한 결과를 비교, 분석하여 가장 적절한 분류 시스템을 제안하려 한다.

2. 관련연구

2.1 기존의 특징 추출 기법

기존의 기타 코드 분류 시스템이나 기기들은 서론에서 언급했듯이 연주음의 주파수를 분석하여 어떤 주파수를 사용하였는지 분석하여 코드를 분류하는 것인데, 기타는 지판을 누르는 손의 힘이나 누르는 위치 등에 따라 수%의 주파수 차이가 날 수 있다. 각 음은 기본 주파수의 정수배의 주파수로 이루어져 있는데, 고주파 음파에서는 각 음별 주파수가 상당히 차이가 나지만 저주파 부분에서는 각 음별 주파수가 크게 차이가 나지 않아 분류하는데 어려움이 있다 [5]. 따라서 이러한 신호를 시스템이 스스로 분석하여 코드를 분류하는 연구들이 진행되어왔다. Barbancho[1]는 신호의 기본주파수를 구하여 특징을 추출하고, 은닉 마르코프 모델을 통해 학습하여 연주음이 특정 코드일 확률을 추정하는 방법을 제안하였으며, Sheh[3]는 은닉 마르코프 모델에 EM 알고리듬을 적용하여 코드를 분류하는 방법을 제안하였다. 이러한 은닉 마르코프 모델 기반의 분류 시스템은 전처리가 상대적으로 간단하고 음악 데이터에서 코드의 진행을 기반으로 상태천이 확률과, 음악데이터를 입력받아 확률적으로 코드를 분류하는 것이며, 학습과정이 복잡하다는 단점이 있다.

2.2 Uncertainty Measurement of Frequency Bin (UMFB) 알고리듬

UMFB 알고리듬은 Guerrero-Turrubiates[4]가 2015년에 제안한 알고리듬으로 필터링 등으로 특징을 추출하고, 인공신경망을 이용하여 분류를 하는 방법으로 분류에 필요한 특징만을 효과적으로 선택하여 학습 효율을 높일 수 있다. 그러나 모든 학습 데이터들에 대해 특징을 추출한 후 특징을 선택하기 때문에 특징 선택을 위한 전처리 시간이 많이 소요된다. 또한 학습 데이터가 새로 추가된 경우 추가된 데이터를 포함하여 모든 데이터들에 대해 다시 특징을 선택하기 위한 과정을 수행해야 하는 단점이 있다. UMFB 알고리듬의 처리 과정은 아래 Fig. 1과 같다.

Fig. 1과 같이 녹음한 시간축 데이터 0을 덧붙이는 zero-padding 과정을 거쳐 두 배의 성분 갯수를 가지는 벡터로 만든 후, Hamming window 함수를 곱하고 다시 이를 퓨리에 변환한다. 이후 주파수 축에서 평준화를 해주는데, 아래의 식 (1)과 같은 중심 주파수를 가지는 삼각형 모양의 필터들을 30개 생성하여 필터 뱅크를 구축하고 각각의 필터에 통과를 시켜준다.

$$c_b = 229(10^{(b+1)/21.4} - 1), \quad b = 1, 2, \cdots 30 \tag{1}$$

여기서 삼각형 모양의 필터는 중심주파수와 양쪽

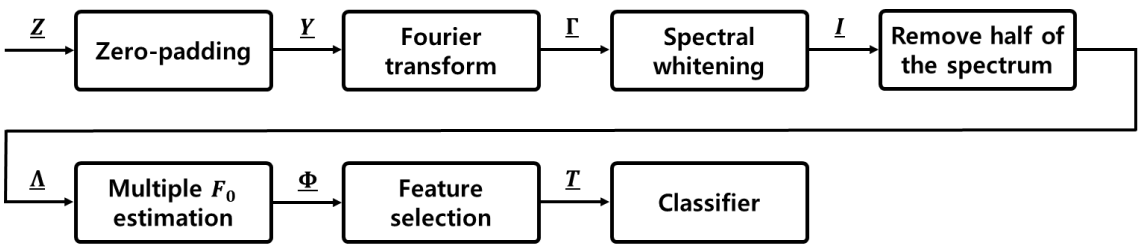


Fig. 1. Processing Diagram of UMFB algorithm.

인접 주파수를 꼭지점으로 가진다. 그리고 필터 뱅크의 각각의 필터 출력들에 대해 표준편차를 구하는 방식과 유사한 식 (2)와 같이 σ_b를 구한다.

$$\sigma_b = \left(\frac{1}{K} \sum_k H_b(k) |\Gamma(k)|^2 \right)^{1/2} \tag{2}$$

여기서 K는 퓨리에 변환을 완료한 데이터의 성분 수이며, $H_b(k)$는 필터 뱅크의 b번째 필터의 k번째 성분이다. 이후 $\gamma_b = \sigma_b^{\nu-1}$로 평준화계수를 구하는데, ν는 0.33으로 설정한다. 이러한 평준화 계수를 전체 주파수 축에 대해 선형보정(linear interpolation)을 해준 필터를 이용하여 아래 식 (3) 같이 평준화를 해준다.

$$I(k) = \gamma(k)\Gamma(k) \tag{3}$$

이후에는 평준화된 벡터의 절반의 성분만을 취하여 Λ를 얻고, 기본주파수를 증폭하여 보다 유용한 특징을 찾는 과정을 거치는데, 각 주파수의 배음들을 결합하는 다음의 식 (4)와 같은 과정을 거친다.

$$\Phi(k) = d^k \sum_{h_r} g(\tau, h_r) |\Lambda(h_r k)| \tag{4}$$

여기서 망각인자 d는 0.99로 설정하며, h_r은 배음 계수이다. $g(\cdot)$는 아래의 식 (5)와 같이 정의되는 함수이다.

$$g(\tau, h_r) = \frac{f_s/\tau_{\max} + \alpha}{h_r f_s/\tau + \beta} \tag{5}$$

이때 $f_s/\tau_{\max}$는 기본 주파수중 가장 낮은 주파수를 의미하며, f_s/τ는 현재 주파수축 데이터 성분의 주파수를 의미한다. 또한 이 기법에서는 $\alpha = 52, \beta = 320$으로 설정하였다고 한다. 이러한 특징추출을 모든 학습 데이터에 대해 수행한 후, 다음 식 (6) 같이 Φ행렬을 구성한다.

$$\Phi = \begin{pmatrix} \underline{a_1} \\ \underline{a_2} \\ \vdots \\ \underline{a_f} \end{pmatrix} \tag{6}$$

여기서 $\underline{a_f}$는 학습 데이터의 개수만큼의 성분을 가진 행벡터이다. 이 행벡터들의 각 성분의 분산을 구하고, 분산이 어느 정도 크기 이상인 경우의 인덱스만 특징으로 사용한다. 여기서 분산 크기의 기준은 각 분산벡터 중 최대값의 0.01326배보다 크면 특징으로 사용하도록 설정하였다고 한다. 이런 과정을 거치면 각 유용한 특징들만을 소수만 추출하여 분류에 이용할 수 있다. 이후에는 이러한 특징들을 단층 퍼셉트론 신경망에 입력하여 최종 분류를 한다.

이러한 UMFB 알고리듬은 보다 유용한 특징들을 소수만 취하여 분류에 이용할 수 있기 때문에 효율적으로 좋은 성능을 보일 수 있지만, 모든 학습데이터들에 대해 특징을 우선 추출하고 통계값을 구하여 진행해야 하기 때문에 학습 데이터의 수가 추가된다면 그 과정을 다시 수행해야 하는 단점이 있다.

2.3 퍼셉트론 신경망

퍼셉트론 신경망은 가장 최초로 등장한 인공신경망으로 제안 초기에는 학습을 효율적으로 수행하는 방법이 개발되지 않아 다층으로 확장하기가 어려워 한때 연구가 이루어지지 않았다[6,7]. 하지만 끊임없는 연구로 인해 내리막 경사법을 이용한 오류역전파 알고리듬을 이용하여 효율적으로 학습을 진행할 수 있으며 다층으로도 쉽게 확장할 수 있게 되었고[8] 매우 많은 분야에서 연구가 이루어지고 있다.

우선 단층 퍼셉트론 신경망의 입출력 관계식을 보면 아래 식 (7)과 같다.

$$\underline{y} = f(W\underline{x} + \underline{b}) \tag{7}$$

여기서 W는 가중치 행렬, $\underline{b}$는 바이어스 벡터이고 $f(\cdot)$는 활성함수이다. 활성함수는 시그모이드 함수나 ReLU(Rectifer Linear Unit)등 다양한 함수를 이용할 수 있다.

단층 퍼셉트론 신경망을 확장한 2층 퍼셉트론 신경망은 이름 그대로 퍼셉트론 신경망이 두 개의 층으로 구성되어 있는 것이다. 첫 번째 층의 입출력 관계식은 아래의 식 (8)과 같이 나타낼 수 있다.

$$\underline{h} = f_1(W_1\underline{x} + \underline{b}_1) \tag{8}$$

보다시피, 단층 퍼셉트론 신경망의 입출력과 동일하며, 여기서 출력값 $\underline{h}$를 다시 2층 퍼셉트론에 입력한다. 이 과정은 식 (9)와 같이 나타낼 수 있고, 식 (8)과 (9)를 결합하여 식 (10)과 같이 전체 시스템의 입출력 관계식을 나타낼 수 있다.

$$\underline{y} = f_2(W_2\underline{h} + \underline{b}_2) \tag{9}$$

$$\underline{y} = f_2(f_1(\underline{x}W_1 + \underline{b}_1)W_2 + \underline{b}_2) \tag{10}$$

위 식을 보면, 1층 퍼셉트론 신경망의 결과를 그대

로 2층 퍼셉트론에 입력한 것을 볼 수 있다. 이러한 방법으로 퍼셉트론 신경망의 층을 얼마든지 확장할 수 있다. 이러한 입출력 관계식은 모두 미분이 가능하다는 특징이 있으며 출력식과 참값의 차이, 즉, 오차를 미분하여 가중치 행렬과 바이어스 벡터의 성분들을 오차를 줄여주는 방향으로 변화시켜주는 내리막 경사법 기반의 오류역전파 알고리듬으로 쉽게 학습할 수 있다.

2.4 합성곱 신경망

기존의 패턴인식 분야, 특히 이미지처리 분야에서는 특징을 추출하기 위하여 커널 이미지를 설정해두고, 이 커널들을 특징을 추출할 이미지에 합성곱을 실시하여 특정 형태의 특징이 있는지 여부를 파악하여 특징으로 사용하였다. 합성곱 신경망에서는 여기서 더 나아가 각 커널 이미지들의 성분들을 학습시켜, 주어진 상황에 맞는 최적의 특징 추출 기법을 설정할 수 있게 해준다[9-13].

합성곱 신경망은 커널(kernel)이라 불리는 벡터를 데이터와 합성곱을 실시하여 특징을 추출한다. 이를 수식으로 나타내면 아래 식 (11)과 같다.

$$\underline{h}_i = f(\underline{x} * \underline{k}_i + b_i I) \tag{11}$$

여기서 $\underline{h}_i$는 i번째 커널 $\underline{k}_i$를 이용하여 특징을 추출한 결과이며, b_i는 바이어스 값이고 I는 단위행렬이다. 또한 $f(\cdot)$는 활성함수이며, 합성곱 신경망에서는 보통 ReLU(Rectifier Linear Unit)함수를 이용한다. 만일 n개의 커널을 이용하여 합성곱을 실시할 경우 결과 또한 n개가 존재하게 되며, 이때 출력 결과의 성분 수가 매우 증가하게 된다. 따라서 이를 최대값 풀링[10]을 통해 인접한 두 개의 성분들 중 큰값만을 취하여 총 성분의 수를 절반으로 줄인다. 층의 개수를 늘리기 위해서는, 커널을 이용한 합성곱과 최대값 풀링을 반복하면 된다.

이렇게 합성곱과 최대값 풀링을 마친 출력값을 최종 분류 라벨로 만들어주기 위해선 한 층의 퍼셉트론 신경망을 이용하여 마무리한다. 학습할 변수인 커널 벡터들과 가중치 행렬, 바이어스 벡터 등은 모두 입력 데이터와 선형결합 후 미분가능한 활성함수에 통과시키기 때문에 내리막 경사법 기반의 오류 역전파 알고리듬으로 학습이 가능하다.

3. 제안하는 방법

최근 화두가 되고 있는 인공신경망, 그중에서도 합성곱 신경망은 시스템이 스스로 특징을 추출하여 분류까지 해내는 것으로 영상처리 분야를 중심으로 수많은 연구가 이루어지고 있는 상황이다[9-12]. 기타 코드음 분류에 인공신경망을 사용하는 연구는 소수 존재하나, 합성곱 신경망을 이용하여 시스템이 스스로 특징을 추출하여 분류하는 시스템을 구현한 연구는 없다. 따라서 본 논문에서는 전처리 과정을 최소화하고도 좋은 성능을 보일 수 있는 다양한 인공신경망을 이용하여 기타 코드음을 분류하는 시스템을 구성하려 하며, 각 인공신경망 시스템의 성능과 최근 제안된 UMFB 알고리듬의 성능을 비교 검증하려 한다.

본 논문에서 제안하는 처리과정은 아래의 Fig. 2와 같이 녹음한 기타 코드음의 시간 축 데이터 $\underline{Z}$의 성분와 동일한 수만큼 0을 덧붙이는 zero- padding을 수행하고 퓨리에 변환을 한 뒤, 중복되는 부분을 제거하기 위하여 절반의 성분만을 취한다. 그 후 평준화를 해주는데 이때에는 각 벡터의 모든 성분의 합으로 나눠주어 각 성분들의 합이 1이 되도록 하였다. 이렇게 얻은 특징벡터를 인공신경망 분류시스템에 입력한다. 사용하는 인공신경망 시스템은 총 4가지 종류로, 각각 1층 퍼셉트론 신경망, 2층 퍼셉트론 신경망, 1층 합성곱 신경망, 2층 합성곱 신경망이다.

1층 퍼셉트론 신경망은 입력벡터인 n차원 벡터에 n×48차원 가중치 행렬을 곱하고, 48차원 바이어스 벡터를 더한 후, 소프트맥스 활성함수를 이용하여 분류하는 시스템을 구현하였다. 이때 학습을 위한 변수는 가중치 행렬의 모든 성분과 바이어스 벡터의 모든 성분의 개수인 n×48+48개이다.

2층 퍼셉트론 신경망은 입력벡터인 n차원 벡터에 n×2048 행렬을 곱하고, 2048차원 바이어스 벡터를

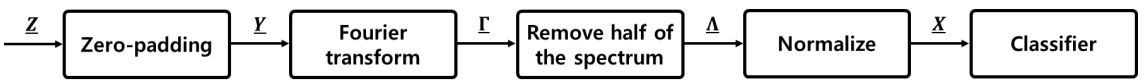


Fig. 2. Processing Diagram of Proposed algorithm.

더한 후, 시그모이드 활성함수를 이용하여 1층을 구성하고, 1층의 출력 2048차원 출력벡터에 다시 2048×48 행렬을 곱한 뒤, 48차원 바이어스 벡터를 더하여 소프트맥스 활성함수를 이용하여 분류하는 시스템이다. 이때의 학습 변수의 개수는 두 층의 가중치 행렬과 바이어스 벡터의 성분 수인 n×2048+2048+2048×48+48개가 되어 1층 퍼셉트론 신경망에 비해 크게 증가한다.

1층 합성곱 신경망의 경우는 n차원 입력벡터를 k_1차원 커널 벡터 8개를 이용하여 합성곱을 실시한 후, ReLU(Rectifier Linear Unit) 활성화함수를 통과시켜 특징을 추출하고, 추출한 특징 벡터들을 최대값 풀링(max pooling)을 수행함으로써 그 차원을 절반으로 낮춰주어 간략화 된 특징벡터들로 만들어준다. 그 후 간략화 된 특징벡터들을 1열로 나열하여 1×(n/2×8) 벡터로 만들어 준 후, 여기에 (n/2×8)×48 가중치 행렬을 곱한 다음 48차원 바이어스 벡터를 더하여 얻은 벡터를 소프트맥스 활성함수에 입력하여 얻은 결과물을 이용하여 최종 분류를 해 낸다. 학습에 필요한 변수들은 합성곱 커널과 가중치 행렬, 바이어스 벡터의 모든 성분들이 있으며 그 갯수는 총 k_1×8+8+(n/2×8)×48+48개이다. 사용한 커널벡터의 차원 k_1은 입력벡터가 4096 또는 512차원일 때는 256이라 두었으며, 입력벡터가 256과 128차원 일때는 각각 128과 64으로 설정하였다.

2층 합성곱 신경망의 경우는 n차원 입력벡터를 k_1차원 커널 벡터 8개를 이용하여 합성곱을 실시한 후, ReLU 활성화함수를 통과시켜 특징을 추출하고, 추출한 특징 벡터들을 최대값 풀링을 수행하여 절반의 차원으로 낮춰주어 간략화 된 특징벡터들로 만들어 준 1층에 추가적으로, 각각의 특징벡터들에 다시 k_2차원 커널 벡터를 이용하여 추가적으로 특징을 추출한다. 그 후, ReLU 활성화함수를 통과시키고 최대값 풀링을 수행하여 다시 차원을 절반으로 낮춰준다. 이렇게 하면 8개의 특징벡터들은 각각 n/4 차원 벡터가 된다. 그 후 1층 합성곱 신경망과 마찬가지로 간략화 된 특징벡터들을 1열로 나열하여 1×(n/4×8) 벡터로 만들어 준 후, 여기에 (n/4×8)×48 가중치 행렬과 48차원 바이어스 벡터, 소프트맥스 활성함수를 이용하여 최종 분류를 한다. 학습에 필요한 변수들의 갯수는 두 번의 풀링을 거쳐 특징벡터의 차원이 많이 감소하기 때문에 1층 합성곱 신경맘보다 절반 가량인 k_1×8+8+k_2×8+8+(n/4×8)×48+48개가 된다. 이에 따라 커널 벡터의 차원 k_2의 값은 k_1의 값의 1/2로 선택하였다.

4. 시뮬레이션 및 결과 분석

4.1 데이터셋

시뮬레이션에 사용한 데이터셋은 전기기타를 PC에 연결하여 직접 코드 음을 연주한 후 이를 녹음하여 구성하였다. 데이터셋의 코드 음은 총 12음의 근음을 가지며, 각각의 근음마다 4종류의 코드로 구성되어 있다. 예를 들면 C(도)를 근음으로 갖는 코드는 각각 C, Cm, C7, Cm7의 4종류의 코드가 있다. 따라서 분류시스템의 총 부류 개수는 48개로 구성되어 있다. 데이터의 총 개수는 학습 데이터 62000개, 검증 데이터 6200개로 구성되어 있다.

녹음 시 샘플링 주기, 데이터 취합 개수 등의 선택과 관련하여서는 기타 코드 분류 시스템에 대한 최근 연구를 참조하였다[4]. 샘플링 주기는 CD 음질과 같은 44100 Hz로 하였으며, 각 샘플의 크기는 16bit로 설정하였다. 이러한 음성 데이터를 약 92ms동안의 샘플들 4096개 씩 잘라 벡터로 구성하였다. 이렇게 구성한 데이터는 시간 축에서 나타낸 데이터이며, 여기서 추가적으로 이산 푸리에 변환을 이용하여 주파수 축 데이터로 변환한 것을 사용하였다. 주파수 데이터로 변환 시에는 4096차원 벡터에 4096개의 0 성분을 추가한 제로 패딩을 하여 9192차원 벡터로 만든 후, 이것을 이산 푸리에 변환하였으며, 대칭성분을 제거하기 위하여 다시 4096개로 자른 뒤 절대값을 취하였다. 이러한 4096 벡터를 필요에 따라 설정한 개수의 저주파 성분만을 취하여 학습에 이용하였다.

각각의 데이터 샘플 별로 라벨을 부여할 때에는 총 48종류의 데이터를 48차원 라벨 벡터에 대응하였다. 라벨 벡터의 하나의 성분만 1로 두고, 나머지는 0으로 둔 one-hot encoding으로 부여해두었는데, 이는 기계학습 시스템에서 라벨을 표현하기 위해 많은 장점을 가지고 있으며[14], 학습 시스템의 수렴 속도가 빠른 교차 엔트로피 비용함수에 최적화된 라벨 표현 방식이다.

4.2 시스템 구성

시뮬레이션에 사용하여 분류시스템을 구성한 신

경망 시스템은 4가지 종류로, 각각 1층 퍼셉트론 신경망, 2층 퍼셉트론 신경망, 1층 합성곱 신경망, 2층 합성곱 신경망이다. 이들 4가지 시스템에 대하여 전체 데이터 중 각각 저주파 성분 4096, 512, 256, 128개의 부분 데이터를 입력벡터로 대입하여 시뮬레이션을 수행한 후 비교 및 분석하였다. 여기에 2장에서 소개한 UMFB 알고리듬을 구현하여 대조군으로 활용하였다. 알고리듬 구현시 각 변수들은 해당 논문[4]에 나오는 것을 그대로 활용하였고, 최종적으로 67개의 특징들을 사용하여 48 종류의 코드들을 분류하는 단층 퍼셉트론 신경망을 구성하였다.

4.3 결과 분석

시뮬레이션 환경은 intel Core i7-7700K CPU, NVIDIA GTX 1060 GPU, 16GB RAM을 탑재한 우분투 16.04 운영체제에서 수행하였으며, 특징 추출 등의 처리과정은 python 3.6버전을 이용하여 구현하였으며, 신경망 시스템은 구글에서 공개한 오픈소스 기계학습 라이브러리인 텐서플로우[15] GPU 버전을 이용하여 구현하였다.

우선 UMFB 알고리듬의 경우 특징 추출을 수행하기 위하여 모든 학습데이터의 특징들을 비교해야하는 과정이 있는데, 본 논문에서 사용하는 학습 데이터 62,000개에 대하여 모든 특징을 추출하고 특징들의 통계값을 이용하여 유효한 특징들을 추출하는데 걸리는 시간은 약 2282.38초로 상당한 시간이 소요된다. 제안한 최소한의 전처리과정 알고리듬에서는 이러한 과정이 필요치 않다. 또한 각각의 입력 데이터를 특징 벡터로 변화시키는 시간에도 큰 차이가 났는데, 이들 시간에 관한 결과를 정리한 것은 아래의 Table 1과 같다.

Table 1에서 데이터 전처리 시간은 본 논문에서 구현한 시스템에서 사용하는 데이터셋의 62,000개의 학습 데이터에 대한 전처리 시간이며, 특징 추출 시간은 데이터 한 개를 특징벡터로 변화시킬 때 걸리는 시간을 통계적으로 측정한 결과이다.

Table 1. Data processing time

System	data preprocessing time(s)	feature extraction time(ms)
UMFB	2282.38	36.23
Proposed	-	0.14

3장에서 설명한 네 가지의 기타 코드음 분류 시스템 각각에 대하여, 총 62,000개의 학습용 데이터 중 임의로 1,000개를 뽑아 1회의 학습을 하는 미니 배치(mini batch[16]) 학습을 진행하였으며, 변수들을 학습할 때에는 내리막 경사법을 기반으로 한 Adam Optimizer[17]를 이용하였으며 학습률로는 0.001을 선택하였다. 또한 UMFB 알고리듬에서 이용한 1층 퍼셉트론의 경우는 미니 배치 학습은 동일하게 적용하였으며, 내리막 경사법의 학습률은 0.01로 설정하였다. 왜냐하면 UMFB 알고리듬은 사용하는 특징벡터의 성분이 상대적으로 크기 때문에 큰 학습률이 필요하기 때문이다. 이러한 학습을 각 시스템별로 1,000회 진행하였고, 각각의 학습 결과를 100회 획득하여 평균한 시뮬레이션 결과는 Table 2에 나타낸 것과 같다.

Table 2를 보면 UMFB 알고리듬의 경우 원래의 녹음데이터 벡터의 성분 4096개를 처리하여 고작 67개로 줄였는데도 82%대의 성능을 보이는 점에서 특징추출 능력은 상당하다는 것을 볼 수 있다. 또한 67개의 특징성분만 사용하기 때문에 학습 시간이 매우 작은 것을 볼 수 있다. 이 알고리듬이 제안된 논문[4]에서는 93%대의 분류 정확도를 보인다고 하여 본 논문에서의 정확도와 차이가 있지만, 이는 사용한 데이터셋이 다르고 학습 횟수를 1000개 샘플의 미니 배치학습 1000회로 고정하였기 때문인 것으로 판단된다. 제안하는 알고리듬에서는 원래의 데이터인 4096 차원 벡터를 그대로 이용한 경우 합성곱 신경망의 성능이 우수한 것을 볼 수 있으나, 학습 시간이 상당히 오래 걸리는 것을 볼 수 있다. 합성곱 신경망의 경우 학습해야 할 변수 자체는 많지 않으나, 합성곱 연산 등 복잡한 연산을 수반하기 때문이다.

입력 데이터 벡터를 저주파 성분 512개만 사용한 경우 전체적으로 분류 정확도가 소폭 상승하였는데, 이는 분류에 기여하지 않는 다수의 성분들을 제거하고 분류에 큰 기여를 하는 저주파 성분들만을 학습에 이용하였기 때문이다. 특징 성분이 더 많을 경우 더 많은 학습 횟수가 필요하기 때문에 유효한 특징을 소수 사용하는 것이 더 효율적인 학습을 할 수 있다. 또한 1층 합성곱 신경망의 분류 성능이 오히려 2층 합성곱 신경망보다 뛰어난 것을 볼 수 있는데, 2층 합성곱 신경망의 경우 2번의 풀링으로 데이터가 지나치게 압축이 되어 이런 결과가 나온 것으로 보인다.

Table 2. Simulation result

System	# of Feature	Structure	Accuracy(%)	Mean learning time per single batch(ms)
UMFB	67	1 layer perceptron	82.062	2.2
Proposed	4096	1 layer perceptron	44.365	19.6
		2 layer perceptron	95.478	33.8
		1 layer CNN	98.706	220.7
		2 layer CNN	99.478	332.5
	512	1 layer perceptron	44.491	4.6
		2 layer perceptron	95.513	9.1
		1 layer CNN	99.586	32.6
		2 layer CNN	99.487	44.5
	256	1 layer perceptron	44.481	4.2
		2 layer perceptron	95.665	7.3
		1 layer CNN	99.424	12.0
		2 layer CNN	99.166	18.1
	128	1 layer perceptron	43.473	3.6
		2 layer perceptron	93.881	6.4
		1 layer CNN	98.734	7.7
		2 layer CNN	98.129	11.8

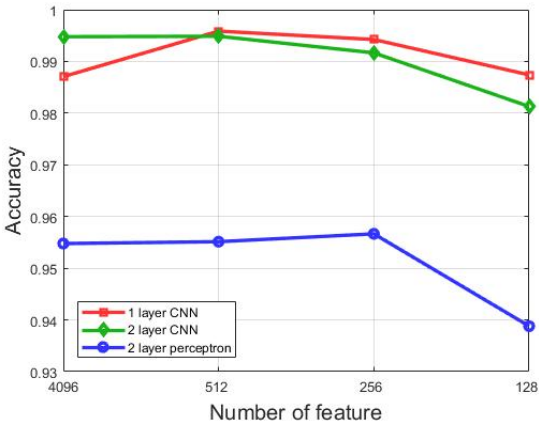
128개의 저주파 성분을 사용하는 시스템에서는 유효한 특징들의 일부만을 사용하였으므로 분류 정확도가 다소 감소했지만, 합성곱 신경망의 경우는 감소하는 정도가 작았으며, 특히 1층 합성곱 신경망의 경우는 2층 퍼셉트론 신경망과 학습시간 차이도 적은 것을 볼 수 있다.

전체적으로 기존의 UMFB 알고리듬에 비해 본 논문에서 제안한 최소한의 전처리 과정만을 수행한 후 합성곱 신경망을 이용하여 분류하는 시스템이 처리속도나 학습 정확도 측면에서 우수한 것을 볼 수 있다. UMFB 알고리듬의 경우 67차원의 특징벡터를 사용하므로 학습 시간 자체는 짧으나, 전처리시간과 특징 추출시간까지 합산한 경우 합성곱 신경망 시스템의 분류 소요시간보다 상당히 오래걸리며, 분류 정확도 또한 82%대로 합성곱 신경망의 98-99%대의 정확도에 비해 낮았다.

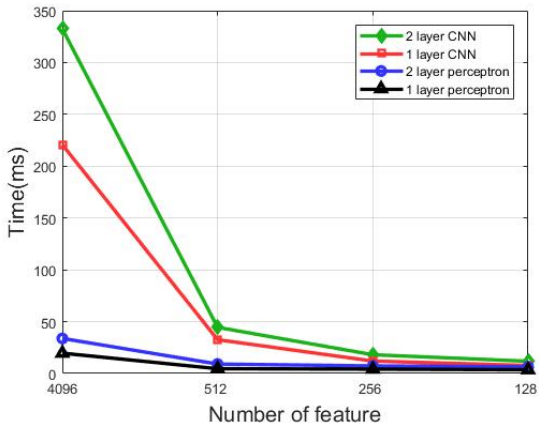
제안한 신경망 알고리듬의 학습 양상은 Fig. 3에 그래프로 나타내었는데, Fig. 3의 (a)를 보면 1층 합성곱 신경망이 특징벡터의 성분 수가 감소하여도 학습 정확도가 크게 감소하지 않는 것을 볼 수 있으며, (b)를 보면 특징벡터의 수가 128개까지 감소한 경우 다른 분류 시스템과의 학습시간 차이가 크지 않은 것을 볼 수 있다. 결론적으로 1층 합성곱 신경망을 원래 신호의 저주파 성분을 일부 추출하여 분류하는 것이 학습시간이나 성능 면에서 효율적이라고 판단된다.

5. 결 론

본 논문에서는 기타 코드 음을 분류하는 시스템을 다양한 인공신경망 시스템으로 구성하고, 직접 구성한 데이터셋을 활용하여 분류 성능이나 처리시간 등을 비교하여 어떤 분류 시스템이 가장 효율적인지를 서술하였다. 최근 제안된 UMFB 알고리듬의 경우 적은 수의 특징 성분들을 이용하여 학습 시간 자체는 짧지만, 이러한 특징을 추출하기 위한 전처리 과정의 소요시간과 특징 추출시간이 길어서 전체 시스템 수행시간이 길다는 단점이 있다. 특히, 모든 학습데이터들의 특징 통계값을 바탕으로 어떤 특징을 선택할지를 결정하기 때문에 데이터가 추가된 경우 다시 통계값을 구해야 하는 단점이 있고, 분류 정확도 또한 제안하는 시스템에 비해 10% 이상 낮았다. 제안



(a)



(b)

Fig. 3. Classification result by the number of feature using on their accuracy and leaning time, (a) Classification accuracy, (b) Mean learning time per single batch.

한 합성곱 신경망 알고리듬은 복잡한 전처리과정이 없이도 커널 벡터를 내리막 경사법을 이용하여 스스로 최적의 특징을 추출하여 UMFB 알고리듬에 비해 전처리 과정 소요시간이나 정확도 측면에서 좋은 성능을 보였다. 추가적으로 합성곱 신경망의 경우 특징 선택을 위해 모든 학습데이터의 특징을 구한 후 통계값을 도출하는 과정이 필요치 않으므로 실시간 분류 시스템을 구성하는 데에도 유리하다고 판단된다.

향후 우리는 데이터셋을 더 확장하여 더 다양한 코드를 분류하는 시스템을 구성하려 하며, 특히 실제 음악을 연주하는 데이터를 추가하여 더욱 실제적인 환경에서 코드 음을 분류하는 시스템을 구성할 수 있을 것으로 기대한다. 또한 합성곱 신경망은 학습한 커널들을 다른 분류시스템에 바로 적용하는 '전이학습'에도 활용할 수 있어 학습시간을 크게 줄이는 연구도 수행할 계획이다.

REFERENCE

[1] A.M. Barbancho, A. Klapuri, L.J. Tardon, and I. Barbancho, "Automatic Transcription of Guitar Chords and Fingering from Audio," *IEEE Transactions on Audio, Speech, and Language Processing,* Vol. 20, No. 3, pp. 915-921, 2012.

[2] J. Bello and J. Pickens, "A Robust Mid-Level Representation for Harmonic Content in Music Signal," *Proceeding of 6th International Symposium on Music Information Retrieval*, pp. 304-311, 2005.

[3] A. Sheh and D. Ellis, "Chord Segmentation and Recognition Using EM-Trained Hidden Markov Models," *Proceedings of the 4th International Society for Music Information Retrieval Conference*, pp. 183-189, 2006.

[4] J. Guerrero-Turrubiates, S. Ledsema, S. Conzalez-Reyna, and G. Avina-Cervates, "Guitar Chords Classification Using Uncertainty Measurements of Frequency Bins," *Mathematical Problems in Engineering,* Vol. 2015, Article ID 205369, pp. 1-9, 2015.

[5] S. Arun and Y. Wang, "Key, Chord, and Rhythm Tracking of Popular Music Recordings," *Computer Music Journal,* Vol. 29, No. 3, pp. 75-86, 2005.

[6] F. Rosenblatt, "The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain," *Psychological Review,* Vol. 65, No. 6, pp. 386-408, 1958.

[7] M. Minsky, A.P. Seymour, and B. Léon, *Perceptrons: An Introduction to Computational Geometry*, MIT Press, Cambridge, Massachusetts, 2017.

[8] J.L. McClelland, E.R. David, and PDP Research Group, *Parallel Distributed Processing,* Vol.

2, MIT Press, Cambridge, Massachusetts, 1987.

[9] Y. LeCun, B.D. Boser, J.S. Denker, D. Henderson, and R.E. Howard, "Backpropagation Applied to Handwritten Zip Code Recognition," *Neural Computation,* Vol. 1, No. 4, pp. 541-551, 1989.

[10] Y. LeCun, K. Koray, and F. Clément, "Convolutional Networks and Applications in Vision," *Proceedings of IEEE International Symposium on Circuits and Systems*, pp. 253-256, 2010.

[11] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, et al., "Recent Advances in Convolutional Neural Networks," *arXiv Preprint arXiv:1512.07108,* 2015.

[12] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, et al., "Going Deeper with Convolutions," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* pp. 1-9, 2015.

[13] H.G. Kang, J.S. Park, J.K. Song, and B.W. Yoon, "CCTV Based Gender Classification Using a Convolutional Neural Networks," *Journal of Korea Multimedia Society*, Vol. 19, No. 12, pp. 1943-1950, 2016.

[14] A.C. Timothy, "Random Forests, Decision Trees, and Categorical Predictors: The" Absent Levels" Problem," *arXiv Preprint arXiv:1706.03492,* 2017.

[15] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, et al., "Tensorflow: Large Scale Machine Learning on Heterogeneous Distributed Systems," *arXiv Preprint arXiv: 1603.04467*, 2016.

[16] J. Ngiam, A. Coates, A. Lahiri, B. Prochnow, Q.V. Le, and A.Y. Ng, "On Optimization Methods for Deep Learning," *Proceedings of the 28th International Conference on Machine Learning*, pp. 265-272, 2011.

[17] P.K. Diederik and B. Jimmy, "Adam: A Method for Stochastic Optimization," *arXiv Preprint arXiv:1412.6980*, 2014.

박 선 배

2015년 2월 홍익대학교 전자전기공학부(공학사)
2017년 2월 홍익대학교 대학원 전자전기공학과(공학석사)
2017년 3월~현재 홍익대학교 대학원 전자전기공학과 박사과정
관심분야 : 신호처리, 패턴인식, 영상처리 및 필터링 이론, 머신러닝 등

유 도 식

2002년 2월 미시간대학교 전자컴퓨터공학과(공학박사)
2006년 9월~2011년 3월 홍익대학교 전자전기공학부 조교수
2011년 4월~2016년 8월 홍익대학교 전자전기공학부 부교수
2016년 9월~현재 홍익대학교 전자전기공학부 교수
관심분야 : 통신 및 신호처리, 정보이론, 어레이신호처리, 영상처리 및 필터링, 신호처리, 변환광학, 머신러닝 등