KIPS Trans. Softw. and Data Eng.
Vol.8, No.1 pp.19~26 pISSN: 2287-5905

https://doi.org/10.3745/KTSDE.2019.8.1.19

Performance Comparison of Machine Learning Algorithms for TAB Digit Recognition

Jaehyeok Heo† · Hyunjung Lee†† · Doosung Hwang†††

ABSTRACT

In this paper, the classification performance of learning algorithms is compared for TAB digit recognition. The TAB digits that are segmented from TAB musical notes contain TAB lines and musical symbols. The labeling method and non-linear filter are designed and applied to extract fret digits only. The shift operation of the 4 directions is applied to generate more data. The selected models are Bayesian classifier, support vector machine, prototype based learning, multi-layer perceptron, and convolutional neural network. The result shows that the mean accuracy of the Bayesian classifier is about 85.0% while that of the others reaches more than 99.0%. In addition, the convolutional neural network outperforms the others in terms of generalization and the step of the data preprocessing.

Keywords : Digit Recognition, Machine Learning, Prototype Selection, Image Processing, Cross-Validation

타브 숫자 인식을 위한 기계 학습 알고리즘의 성능 비교

허 재 혁† · 이 현 종†† · 황 두 성†††

요 약

본 논문에서는 기타 타브 악보에서 추출한 프렛 번호를 대상으로 학습 알고리즘의 분류 성능을 비교한다. 타브 악보로부터 세그먼트를 통해 추출된 타브 숫자 데이터는 타브 선과 악보 기호가 포함하기 때문에 레이블링 기법과 비선형 필터를 이용하여 프렛 숫자를 추출한다. 추가적인 데이터 확보를 위해 전처리가 수행된 데이터에 대해 4 방향으로 이동 연산을 수행한다. 선택된 학습 모델은 베이지안 분류기, 지지벡터기기, 프로토타입 기반 학습, 다층 신경망 그리고 합성곱 신경망 모델 등이다. 실험 결과 베이지안 분류기는 85.0% 평균 정확도를 보였고 나머지 분류기는 99.0% 이상의 평균 정확도를 보였다. 일반화 성능과 전처리 단계를 고려 시 합성곱 신경망이 다른 학습 모델들보다 우수하다.

키워드 : 숫자 인식, 기계학습, 프로토타입 선택, 영상처리, 교차평가

1. 서 론

기계 학습 알고리즘을 이용한 숫자 인식의 경우 크게 필기체 인식과 인쇄체 인식으로 나눌 수 있다. 필기체 인식의 대표적인 응용으로 우편 번호 인식이 수행되었고 인쇄체 인식의 대표적인 응용으로 자동차 번호판 인식, 가옥 번호(house number) 인식 시스템 등이 수행되었다. 타브 악보는 기타 연주를 위해 고안된 악보로서 기존의 음표로 표시되었던 악보와 달리 기타 줄의 누르는 위치를 의미하는 프렛(fret) 숫자가 작성되어 있는 악보이며 인쇄체 분류 문제 중 하나이다.

본 논문에서는 수집된 기타 타브 악보로부터 추출한 타브 숫자로부터 숫자 분류를 위한 전처리 과정을 제안하고 기계 학습 알고리즘 실험 결과를 분석한다. 기타 타브 악보로부터 추출된 타브 숫자 데이터를 대상으로 전처리 과정을 수행해 다중 학습 문제를 도출하는 과정을 기술한다. 준비된 다중 분류 문제를 평가하는 기계학습 모델로는 베이지안(bayesian), 지지벡터기기(support vector machine), 프로토타입 기반 학습(prototype-based learning), 다층 신경망(multilayer neural network), 합성곱 신경망(convolution neural network) 등이 선택되었다.

합성곱 신경망의 평가에는 전처리 전과 후의 데이터를 학습한 모델의 일반화 성능을 비교한다. 실험 결과의 데이터 분석에서 합성곱 신경망의 특징추출 강건성(robustness)을 평가한다. 또한 다층 신경망과 합성곱 신경망은 학습 시간을 분석하기 위해 GPU와 CPU에서 학습을 비교한다.

† 준 회 원 : 단국대학교 컴퓨터공학과 석사과정
†† 준 회 원 : 단국대학교 소프트웨어학과 석사과정
††† 종신회원 : 단국대학교 소프트웨어학과 교수
Manuscript Received : June 5, 2018
First Revision : July 17, 2018
Accepted : July 25, 2018
* Corresponding Author : Doosung Hwang(dshwang@dankook.ac.kr)

2절에서는 기계학습을 이용한 숫자 인식과 관련된 연구들에 대해 서술한다. 3절에서는 타브 숫자 데이터 특징과 전처리 과정을 보이며, 4절에서는 프로토타입 기반 학습(PBL), 다층 신경망(MLP) 그리고 합성곱 신경망(CNN) 알고리즘을 기술한다. 5절에서는 학습 모델의 성능 비교와 신경망 알고리즘의 CPU와 GPU에서 학습 시간을 분석한다. 마지막으로 제안하는 방법의 문제점과 개선방안을 토의한다.

2. 관련 연구

우편번호 숫자 분류 문제인 MNIST는 손으로 작성된 필기체 데이터에 영상처리를 이용해 준비되었으며 벤치마크 문제로 기계 학습 모델의 성능을 평가하고 심층 학습 모델을 이용한 성능을 개선시키는 연구에 많이 사용되고 있다[1].

LeCun의 연구[2]에서는 오류 역전파 알고리즘(error backpropagation)을 적용하여 MNIST 분류를 평가하였다. 4개의 은닉층을 갖는 완전 연결 신경망(fully connected neural network)을 학습하여 96.6%의 분류 정확률로 실험 평가되었다[2].

Simard의 연구[2]에서는 MNIST 데이터에 대해서 다층 신경망과 합성곱 신경망 알고리즘의 성능을 비교했다. 은닉층의 노드 수를 800개인 3층 신경망을 구성하고 데이터를 학습해 약 98.4% 이상의 분류율로 평가되었으며, 합성곱 신경망 모델은 99.6% 이상으로 평가되어 다층 신경망 보다 개선된 성능을 보였다[3].

Dan의 연구[2]에서는 안드로이드 운영체제 기반의 스마트폰 에서 주성분 분석(principal component analysis)과 1개의 은닉층을 가진 역전파(back propagation) 신경망 알고리즘을 이용해 MNIST 데이터를 학습시킨 모델을 통한 필기체 인식 시스템을 제안했다. 설계된 시스템은 각 숫자에 대해 평균 91.2%의 인식률을 보였다[4].

Zang의 연구[2]에서는 교통 영상에서 가시적 주의 모델(visual attention model)을 통한 데이터 준비와 심층 학습을 이용한 자동차 번호판 인식 시스템을 제안했다. 설계된 시스템은 중국어 인식에서 98.3%의 재현율(recall rate)을 보였고, 숫자와 알파벳에서 99.1%의 재현율을 보였다[5].

Goodfellow의 연구[2]에서는 합성곱 신경망 모델을 통해 거리 영상에서 숫자를 인식하는 시스템을 제안했고 가옥 번호 데이터 셋인 SVHN(street view house number)에 대해서 97.8%의 정확도를 보였다[6].

신경망 알고리즘은 각 네트워크 층의 노드 수가 많아질수록 학습 파라미터 수가 증가하여 학습 시간이 증가하는 경향이 있다. 이런 단점을 극복하기 위해 GPU의 병렬 계산 능력을 활용해 학습 시간을 단축하는 연구가 활발히 진행되고 있고 GPU 계산을 쉽게 활용할 수 있게 도구가 개발되었다[7-9]. 대표적으로 분자 구조 분석, 암호 해독, 기상 변화 예측 분야 등에서 사용되었으며 CPU 만으로 연산할 때에 비해 최대 몇 십 배까지 연산 능력을 증가시켰다[10-12].

3. 학습데이터 준비

ByungHyun Baek이 제안한 타브 숫자 데이터는 6선인 타브 악보에서 템플릿 매칭 기법과 가상 블록 탐지 기법을사용하여 프렛 번호를 추출했다[13]. 타브 악보는 전체 0~25의 숫자를 사용하지만 수집된 기타 악보 53개에서 나타나는0~15까지의 숫자를 대상으로 데이터가 준비되었다.

기타 악보를 구성하는 타브 숫자 또는 프렛의 추출 과정에서 악보의 타브 선을 제거하지 않고 프렛을 추출했기 때문에 추출된 타브 숫자 데이터는 타브 선과 악보 기호를 포함한다. 또한 기타 악보에 따라 카메라 입력에 따른 프렛들의 영상 크기가 다르게 수집되었다. 기존의 프렛 번호 추출 과정에 본 논문에서 제안하는 타브 선 분리, 필터링 과정, 이동 연산(shift operation)을 이용하여 이러한 문제점을 해결한다(Fig. 1). 기타 악보(TAB chord image)로부터 이진화(binarization), 전경 추출(foreground extraction), 기울기 보정(ajustment of chord image)을 수행하고 악보 영역(chord segmentation)과 프렛 숫자 영역만을 추출(fret extraction)한다. 추출된 프렛 번호 영상에서 타브 선과 번호 영역을 분리(TAB line separation)하여 학습 데이터(TAB digit data)를 준비한다.

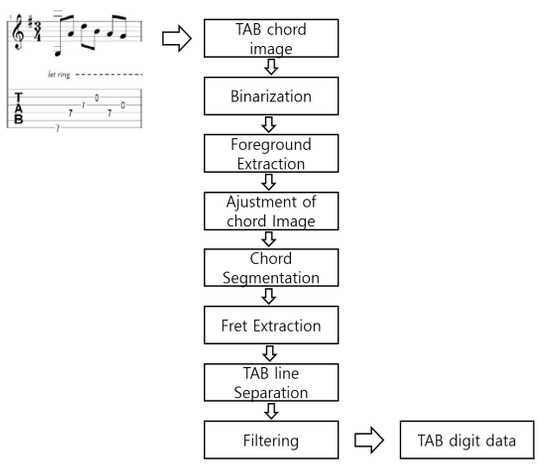


Fig. 1. The Process for Extracting Fret Digits

Table 1은 프렛 세그먼트의 크기에 따른 데이터의 수이다. 수집된 타브 숫자 데이터의 크기는 4 종류이며, 각 세그먼트의 크기가 상이하다. 추출된 프렛 번호 세그먼트 크기가 다르면 데이터마다 구성되는 특징 벡터의 길이가 다르며 학습 모델을 올바르게 생성할 수 없다. 동일한 크기의 타브 숫자 데이터를 확보하기 위해 패딩 단계를 도입하여 12×12 크기를 갖도록 상단과 하단에 여백 공간을 추가한다. 10×9 크기를 갖는 데이터의 경우 상단과 하단뿐만 아니라 좌측, 우측에도 여백을 추가한다.

Fig. 2는 타브 숫자 세그먼트에서 나타나는 타브 선과 타브 숫자, 악보 기호의 예이다. 타브 선과 악보 기호는 학습 과

Table 1. The Number of Data Per Segment Size

Segment size	No. data
10×9	3
10×12	9,857
8×12	72
9×12	992
Total	10,714

Fig. 2. Examples of TAB Digit Segmentation

Fig. 3. Removing TAB Lines

정에서 불필요한 정보이기 때문에 제거한다. Fig. 3은 레이블링 기법을 이용한 타브 선 제거 과정을 보여준다. 타브 선 제거를 위한 레이블링 기법으로 Grassfire 알고리즘을 사용한다. Grassfire 알고리즘은 시작점과 같거나 유사한 지점의 값을 가진 픽셀을 재귀적인 방법을 사용해 찾아낸다[14].

Fig. 3과 Fig. 4에서 왼쪽 영상은 원본 타브 숫자 데이터이다. 중간 사진은 레이블링 결과이며, 각 그룹의 영역은 회색의 명암으로 구분한다. 프렛 번호 0~9인 경우 하나의 가장 큰 영역을 제외한 나머지 영역이 타브 선의 영역이며, 10~15의 경우 두 자리 수를 의미하는 두 개의 큰 영역을 제외한 나머지를 타브 선 영역으로 분리한다. 오른쪽 영상은 원본 영상에 레이블링이 수행된 결과이다. 타브 선이 프렛 번호와 인접해 있는 경우에는 레이블링 기법만으로 제거할 수 없다. Fig. 4는 타브 숫자 15가 타브 선과 인접하여 레이블링 방법으로 선과 숫자 영역을 분리하지 못하는 것을 보여준다.

Fig. 4. The Limitation of Labeling Method

레이블링 후 잔재하는 타브 선을 제거하기 위해 1차원이고 길이가 12인 비선형 필터를 사용했다.

$$G(x,j)=\frac{1}{255}\sum_{i=0}^{n-1}x_{ij} \qquad (1)$$

$$x_j=\begin{cases}0, & \text{if } G(x,j)=1\\ x_j, & \text{otherwise}\end{cases}\ (j=0,\ldots,n-1) \qquad (2)$$

Equation (1)의 n은 입력 영상의 행 크기이며 x_{ij}는 이진 영상의 i행, j열의 값이다. 함수 G는 필터링을 수행하는 함수로 j열 부분 영상에 255의 역수를 곱한 값을 반환한다. Equation (2)는 한 열에 대한 필터링 연산 과정의 수식이다. x_j는 원본 영상의 j열 부분 영상을 나타낸다. 필터 연산 값이 1이면 타브 선이며 원본 영상의 j열 부분 영상의 값을 0으로 채우고 그렇지 않으면 픽셀 값을 유지한다. 필터링 적용 후 남아있는 잡음과 불완전한 데이터는 크기를 재설정한 필터와 레이블링을 적용해 제거한다. Fig. 5는 필터를 타브 숫자 데이터 12에 적용시키는 예이다.

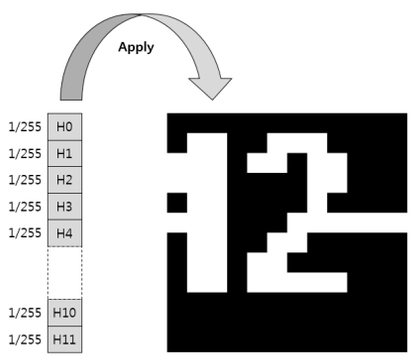


Fig. 5. The Example of Non-linear Filtering

Fig. 6은 기존 타브 숫자 데이터와 전처리가 완료된 데이터를 클래스 별 10개씩 출력한 결과이며 데이터의 추가 확보를 위해 상, 하, 좌, 우 4방향에 대해 이동 연산을 수행하여 기존 데이터 개수의 5배인 53,570개의 데이터가 준비되었다.

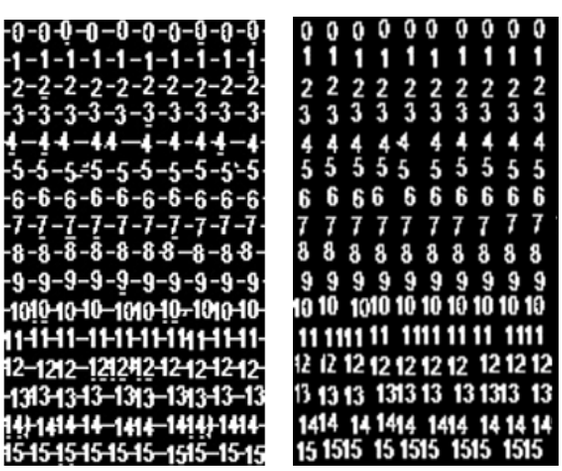


Fig. 6. Examples of Segmented Data and Preprocessed Data

4. 기계 학습 모델

4.1 다층 신경망(Multi-Layer Perceptron)

다층 신경망은 완전 연결층(fully-connected layer)으로 구성되며 비선형 분리 문제에서 학습이 가능하도록 하나 이상의 은닉층이 추가된다[15]. 다층 신경망의 층은 입력층, 은닉층, 출력층으로 구성되며 각 층 안에는 여러 뉴런(neuron)들로 구성된다. 각 층의 뉴런과 뉴런들은 완전 연결되어 있으며 가중치를 부여한다. 입력층을 제외한 층 안의 뉴런은 연결된 이전 뉴런의 출력 값과 가중치를 곱하여 더한 값에 활성 함수(activation function)를 적용시켜 해당 뉴런의 출력을 낸다. 뉴런의 활성 함수로는 선형(linear), 시그모이드(sigmoid), ReLU (retified linear unit), 소프트맥스(softmax) 등이 주로 선택된다[16].

출력층의 계산 결과는 입력 벡터에 대한 예측 분류이다. 뉴런과 뉴런을 연결하는 가중치는 초기에는 임의 작은 값으로 초기화시킨다. 예측 분류 결과와 입력 데이터의 클래스 정보의 평균 제곱 오차 값의 손실함수(loss function)에 대한 오류 역전파 학습을 이용한 기울기 하강 방법으로 학습을 수행한다. 오류 값의 차이를 최소화시킬 수 있도록 각 층 사이의 가중치를 변화시키는 단계를 반복한다.

다층 신경망 알고리즘은 많은 기 연구에서 우수한 분류 성능을 보이지만 학습 파라미터 수가 많아 최적의 네트워크 구조를 정의하는 과정이 어려우며 층이 깊어질수록 모델 복잡도가 증가하기 때문에 학습 시간이 오래 걸리며 과적합 현상이 발생할 수 있다.

4.2 지지벡터기기(Support Vector Machine)

지지벡터기기는 클래스 분류 경계면에 근접한 학습 데이터 또는 지지벡터들 간의 최대 마진(margin)을 이용하는 선형 분류 학습 알고리즘이다[17]. 이진 분류에 적합한 지지벡터기기는 실험에서 이진 분류 문제에서 기존의 분류기들보다 우수한 성능을 보였다. 최대 마진을 갖는 지지벡터의 선택은 2차 계획법(quadratic programming) 최적화 문제의 해로 결정된다. 선형 분류 학습에 적합한 지지벡터기기는 비선형 문제를 효과적으로 해결하기 위해 훈련 데이터를 커널(kernel)을 통해 선형분류가 가능한 고차원으로 변환시킨 후 선형 분류 학습을 수행한다. 고차원 안에서 데이터를 최적으로 분리하는 지지벡터를 통해 획득한 초평면(hyperplane)을 이용해 입력 공간에서 비선형 분류가 가능한 클래스 분류 함수를 구성시킨다. 고차원으로 매핑 시키는데 다항식(polynomial), RBF(radial basis function), 쌍곡탄젠트(hyperbolic tangent) 등 비선형 커널들이 주로 사용된다.

오류의 최적값을 찾아 학습하기 때문에 상대적으로 우수한 분류 성능을 보이며 많은 응용 분야에서 사용되고 있다. 하지만 학습을 위해 조정해야 할 파라미터 수가 많으며 데이터가 많아질수록 학습 시간이 지수 증가한다.

4.3 합성곱 신경망(Convolutional Neural Network)

합성곱 신경망은 영상처리에서 사용되는 필터 방법과 신경망을 병합하여 다차원 데이터 학습에 최적화시킨 알고리즘이다[18]. 합성곱 신경망은 입력 데이터를 합성곱 층, 최대 풀링(max pooling) 층 그리고 완전 연결 층을 거치고 오류 역전파 알고리즘을 통해 학습을 진행한다. 합성곱 층에서는 필터를 영상에 적용시키는 합성곱 과정을 거쳐 특징 지도(feature map)를 생성한다.

최대 풀링 층은 생성된 특징 지도에 최대 필터(maximum filter)를 적용시켜 크기를 축소하며 부분 샘플링 층(subsampling layer)이라고도 한다. 합성곱과 풀링 단계를 거치면서 입력 데이터의 구조적 형상이 유지되며 공간 정보를 학습에 이용하게 된다. 합성곱 층과 최대 풀링 층의 반복을 통해 특징 추출이 완료되면 특징 지도를 1 차원 벡터로 변환 후 완전 연결층으로 입력된다. 다층 신경망과 마찬가지로 역전파 알고리즘을 통해 필터의 값과 완전 연결층의 가중치를 학습한다[15]. 합성곱 신경망은 영상, 음성 분야 모두에서 우수한 성능이 보고되고 있으며, 다층 신경망 알고리즘보다 적은 수의 매개변수를 사용해도 성능이 떨어지지 않고 빠른 학습이 가능하다는 장점이 있다.

합성곱 층을 거쳐 학습을 위한 데이터 패턴을 추출해 학습하기 때문에 별도의 전처리 과정이 요구되진 않지만 우수한 분류 성능을 내기 위해 많은 데이터 수가 요구되며 다층 신경망 알고리즘과 비슷하게 최적의 네트워크 구조를 정의하기 어려움이 있다.

4.4 프로토타입 기반 학습(Prototype Based Learning)

프로토타입 기반 학습에서 프로토타입 선택은 주어진 분류 문제의 데이터의 집합으로부터 각 클래스 내 데이터를 대표할 수 있는 적은 수를 갖는 프로토타입 집합을 선택한다. 프로토타입은 동일 클래스의 부분 영역을 커버하는 데이터이며, 프로토타입으로 구성되는 새로운 학습 데이터는 새로운 분류 문제로써 지도학습을 수행 시 적은 수의 데이터로 인한 빠른 학습이 가능하며, 중복 또는 잡음 데이터를 제거하는 장점을 제공한다[19].

프로토타입 집합은 각 훈련 데이터가 자신의 클래스 영역을 대표하는 초월구(hypersphere)를 구성시킨 후 훈련 클래스 영역 내 모든 데이터를 포함하는 최소의 클래스 데이터 집합이다. 초월구의 반지름은 동일 클래스 데이터의 최대 거리와 다른 클래스 데이터의 최소 거리의 중간 값으로 정한다. 클래스 프로토타입은 가능한 많은 동일 클래스 데이터를 대표하는 데이터를 우선으로 선택한다.

선택된 UCI 벤치마크 분류 문제들의 일반화 성능 비교에서 PBL을 적용 후 지지벡터기기, 최근접 이웃 학습, 베이지안 분류 등의 일반화 성능은 본래의 성능과 유사하며, 학습데이터의 축소와 더불어 빠른 학습이 진행되었으나, 프로토타입 선택은 계산 비용이 필요하다[19].

5. 실 험

타브 숫자 인식의 성능 평가 실험을 위한 하드웨어 환경은 다음과 같다. CPU는 Intel Core i7-3770@3.40GHz 8개, RAM 8GB, GPU는 GeForce GTX 580을 사용했으며 운영체제는 Linux Mint 17 Qiana를 사용했다. 제안하는 타브 데이터 전처리와 이동 연산을 적용하여 준비된 클래스 별 데이터 수는 Table 2와 같다.

Table 2. The Number of Fret Digits Per Class

Fret	Number	Fret	Number
0	10,245	8	2,230
1	3,410	9	2,805
2	6,955	10	1,480
3	6,290	11	1,065
4	3,030	12	950
5	4,875	13	1,100
6	1,790	14	755
7	5,745	15	845
Total		53,570	

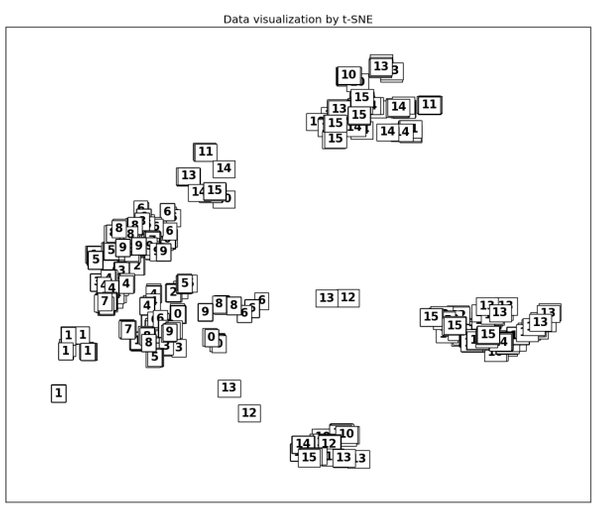


Fig. 7. A Visualization of TAB Digits Using the t-SNE Method

정확도(accuracy)는 알고리즘의 성능을 평가하는데 일반적으로 사용된다. 타브 숫자 분류의 클래스 별 데이터 수가 불균형적인 경우에 정확도는 데이터 수가 많은 클래스에 민감한 평가 척도이다. 클래스 분포에 덜 민감한 균형 정확도(balanced accuracy)를 평가 척도로 선택하였다. Equation (3)은 균형 정확도 *bacc*를 구하는 식이다.

$$bacc = \frac{1}{C}\sum_{l=1}^{C}\frac{1}{n_l}\sum_{i=1}^{n_l}1\left(h\left(x_i\right)= y_i\right) \tag{3}$$

Equation (3)의 $h\left(x_i\right)$는 입력 데이터 x_i의 분류 결과이며 $y\left(x_i\right)$는 x_i의 클래스 레이블이다. n_l은 l클래스의 데이터 수, C는 전체 클래스 수이며 균형 정확도는 각 클래스 정확도의 평균이다.

학습 알고리즘 성능 평가는 10-식 교차 검증(10-way cross validation)을 수행하였다. Table 3은 타브 숫자 학습데이터에 대한 균형 정확도 실험 결과이다. 3절에서 논의한 기계학습 알고리즘과 베이지안 학습의 결과가 비교된다. 프로토타입 기반 학습은 프로토타입을 선택 후 최근접 이웃 알고리즘으로 분류 예측을 수행한다.

Fig. 8은 합성곱 신경망의 학습 종료 후 클래스 별 임의의 입력 영상 2개를 합성곱 층과 최대 풀링 층을 거쳐 생성된 중간 결과를 가시화한 것이다. Conv는 합성곱 층을 거쳐 나온 변환된 특징 지도의 예이며, Pooling은 최대 풀링 층을 거쳐 나온 결과 변환된 특징 지도의 예이다. 합성곱 층의 결과 영상은 합성곱 층 내 특징 지도 10개 중 2개를 선택해 구성했다. 동일 클래스인 데이터의 최대 풀링 결과 영상은 유사하게 나타나고 타 클래스 데이터와 영상의 형태 차이를 보이기 때문에 높은 일반화 성능을 기대할 수 있다고 분석된다.

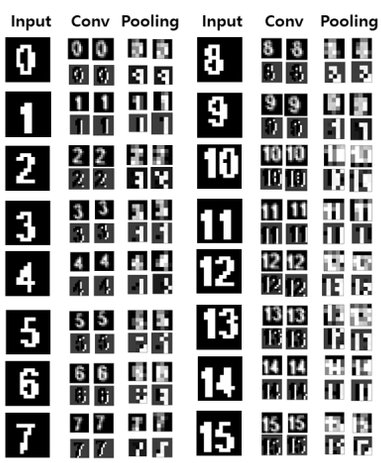


Fig. 8. Feature Map Examples

실험을 위해 파이썬과 기계학습 라이브러리인 scikit-learn [21]과 keras[22]를 이용한다. 지지벡터기기의 실험은 RBF 커널을 선택하며, *C=0.5*와 $\gamma = 3$로 설정하였다. 다층 신경망 모델과 합성곱 신경망 모델의 네트워크 구조는 노드 수 비율에 따른 분류 성능을 10-식 교차 검증으로 평가해 탐욕적 탐색법(greedy search)으로 공간 복잡도와 시간 복잡도를 고려해 가장 합리적인 결과를 보인 구조를 선택했다.

다층 신경망 모델의 네트워크 구조는 Table 4에 명시되어 있다. 배치 크기는 200, 학습 반복수는 20으로 설정했으며 Adam 최적화[23]를 사용하고 손실 함수로 MSE(mean squared error)를 사용했다. CNN-PRE 네트워크 구조는 Table 5와 같으며 배치 크기와 최적화 알고리즘은 다층 신경망과 동일하다. 학습 반복수는 10으로 설정했으며 손실 함수는 교차 엔트로피(cross entropy)를 선택했다.

Table 3. The Performance Comparison of TAB Digit Recognition

Fold	Bayes	SVM	PBL	MLP	CNN-PRE	CNN-ORG
1	85.60	100.00(636.0)	99.99(116.0)	99.98	100.0	100.0
2	84.60	99.99(634.0)	99.98(117.0)	99.98	100.0	99.98
3	85.30	100.00(634.0)	99.99(116.0)	99.71	100.0	99.97
4	86.40	99.99(631.0)	99.99(113.0)	99.96	100.0	99.99
5	85.70	99.99(633.0)	99.99(115.0)	99.95	100.0	100.0
6	83.80	100.00(638.0)	99.99(116.0)	99.71	100.0	100.0
7	84.50	99.99(634.0)	99.99(115.0)	99.84	100.0	99.99
8	85.40	100.00(636.0)	99.98(116.0)	99.96	100.0	99.97
9	84.60	100.00(639.0)	100.00(116.0)	99.92	100.0	99.97
10	86.60	99.99(630.0)	99.99(115.0)	99.99	100.0	100.0
AVG.	85.00	100.00(634.5)	99.99(115.5)	99.90	100.0	99.99
Mean time (sec)	0.7	147.4	32.7	87.8	175.9	431.9

Table 4. The Structure of a MLP Network

Layer	No. of neurons	Activation
Input	144	
Hidden	72	ReLU
Output	16	Sigmoid

Table 5. The Structure of a CNN-PRE Network

Layer	No. of neurons	Activation
Input	12 × 12	
Conv2D	10 × (2 × 2)	ReLU
MaxPooling	2 × 2	
Dropout	20.0%	
Hidden	28	ReLU
Output	16	Softmax

Table 6. The Structure of a CNN-ORG Network

Layer	No. of neurons	Activation
Input	12 × 12	
Conv2D	10 × (2 × 2)	Sigmoid
MaxPooling	2 × 2	
Dropout	20.0%	
Hidden	28	Sigmoid
Output	16	Softmax

합성곱 신경망 테스트에서는 전처리한 데이터를 학습한 합성곱 신경망 모델(CNN-PRE)과 전처리를 하지 않은 데이터를 학습한 합성곱 신경망 모델(CNN-ORG)의 성능을 비교했다. 추가적으로 합성곱 신경망과 다른 학습 모델의 전처리 전, 후 성능 차이를 비교했다.

CNN-ORG 네트워크 구조는 Table 6과 같으며 기존 타브 데이터 학습 과정에서 ReLU 함수의 기울기 소실 현상이 발생해 활성화 함수를 시그모이드 함수로 변경했으며 학습 반복수는 50으로 설정했다.

프로토타입 기반 학습의 프로토타입 수(psize)와 지지벡터기기의 지지벡터 수(svsize)가 비교되었다. 지지벡터기기와 합성곱 신경망 모델의 경우 모든 프렛 숫자의 분류는 약 100.0%로 평가된다. 합성곱 신경망의 평가에서 전처리가 수행되지 않은 기존의 타브 숫자 데이터를 적용했을 때보다 정확도가 개선되었다. 베이지안의 경우 정확도가 85.0%으로 가장 낮게 나타났다. 프로토타입 기반 학습의 경우 약 100.0%의 정확도를 보였다. 클래스 별 프로토타입 수는 평균 155.5개로 원본 데이터의 0.21%가 선택되었으며 지지벡터기기 가에서 선택된 지지벡터 수의 약 20.0%이다. 프로토타입 선택에는 평균 1792.6초(약 30분)가 소요된다. 프로토타입으로 추출한 학습 데이터의 학습 및 분류 시간은 3.27초로 전체 데이터를 통해 실험한 다른 학습 모델보다 빠르다.

전처리를 수행하지 않은 데이터에 대해서 베이지안, 지지벡터기기, 프로토타입 기반 학습, 다층 신경망, 합성곱 신경망 모델을 학습시켜 전처리가 수행된 데이터로 학습시킨 모델과 비교했으며 베이지안을 제외한 나머지 모델에서 성능 개선이 있거나 비슷했다. 지지벡터기기와 프로토타입 기반 학습 알고리즘의 성능은 비슷했으며 다층 신경망의 경우 11.4%가 개선되었으며 합성곱 신경망의 경우 큰 차이가 없었다.

CNN-PRE 모델과 CNN-ORG 모델의 일반화 성능과 학습 시간을 비교했을 때 CNN-ORG 모델의 학습 반복 수 증가에 따라서 학습 시간이 CNN-PRE 모델보다 오래 소요되었으나 분류 성능은 비슷했다.

다층 신경망과 합성곱 신경망 모델(CNN-PRE)의 학습 시간 단축을 위해 GPU를 이용했다. CPU를 사용했을 때와 GPU를 사용했을 때의 균형정확도는 유사하게 나타났으며 학습 시간에서 차이가 있었다. 다층 신경망 알고리즘의 경우 CPU에서는 49.2초의 시간이 소요되고 GPU에서는 48.2초가 소요되었다. 합성곱 신경망 알고리즘의 경우 CPU에서는 24.58초가 소요되고 GPU에서는 19.15초가 소요되었다.

6. 결 과

본 연구에서는 기존 타브 숫자 데이터에 레이블링 기법과 필터링 과정을 거쳐 타브 선과 기호와 같은 잡음을 제거했다. 타브 숫자 데이터에 대해서 다층 신경망과 합성곱 신경망 모델의 네트워크 구조를 제안하고 베이지안 학습기, 지지벡터기기, 프로토타입 기반 학습과 일반화 성능을 비교 평가했다.

전처리 전 데이터의 분류 성능 실험 결과와 비교했을 때 전체적으로 성능 개선이 있었다. 신경망 모델의 경우 CPU와 GPU 기반일 때 학습 시간과 분류 성능을 비교했다. 신경망 네트워크 층 수가 충분히 깊지 않아 학습 속도 개선이 미미했지만 분류 성능은 우수했다. 전처리 전과 후 데이터에 대해 합성곱 신경망 모델의 비교에서 분류 성능은 비슷했으며, 각 데이터에 대한 특징 지도 가시화 영상을 비교했을 때 합성곱 층에서 숫자 고유의 패턴을 추출해 전처리 과정과 비슷한 결과를 도출하는 것으로 보이며 합성곱 신경망 모델이 데이터 잡음의 영향을 피할 수 있다고 분석된다. 또한 프렛 숫자의 가변 영역에 따른 동일 크기로 패딩한 효과는 일반화 성능에 영향이 없다고 분석된다.

본 논문의 학습 데이터는 빈번히 나타나는 프렛 숫자 0~15까지 데이터가 수집되어 사용되었다. 타브 악보에서 사용되는 16~25까지 숫자 데이터가 확보되면 기계학습 기반 타브 숫자 데이터에 대한 실질적 평가가 진행될 수 있다.

References

[1] LeCun, Yann, Cortes Corinna, and Christopher JC Burges, “The MNIST database of handwritten digits.” 2009.

[2] LeCun, Yann, et al., “Handwritten digit recognition with a back-propagation network,” Advances in Neural Information Processing Systems. 1990.

[3] Patrice Y. Simard, Dave Steinkraus, and John C. Platt, “Best Practice for Convolutional Neural Networks Applied to Visual Documnet Analysis,” *Proceedings of the Seventh International Conference on Document Analysis and Recognition*, IEEE Computer Society, 2003.

[4] Dan, Zhu, and Chen Xu, “The recognition of handwritten digits based on bp neural network and the implementation on android,” *Intelligent System Design and Engineering Applications (ISDEA), 2013 Third International Conference on*. IEEE, 2013.

[5] Zang, Di, et al. “Vehicle license plate recognition using visual attention model and deep learning,” *Journal of Electronic Imaging*, Vol.24, No.3, pp.033001-033001. 2015.

[6] Goodfellow, Ian J., et al., “Multi-digit number recognition from street view imagery using deep convolutional neural networks,” arXiv preprint arXiv:1312.6082, 2013.

[7] Nvidia, C. U. D. A. “Programming guide.” 2010.

[8] Stone, John E., David Gohara, and Guochun Shi, “OpenCL: A parallel programming standard for heterogeneous computing systems,” *Computing in Science & Engineering*, Vol.12, No.3, pp.66-73, 2010.

[9] Wienke, Sandra, et al., “OpenACC—first experiences with real-world applications,” European Conference on Parallel Processing. Springer, Berlin, Heidelberg, 2012.

[10] Chen, FeiGuo, Wei Ge, and JingHai Li. “Molecular dynamics simulation of complex multiphase flow on a computer cluster with GPUs,” *Science in China Series B: Chemistry*, Vol.52, No.3, pp.372-380, 2009.

[11] Shao, Fei, Zinan Chang, and Yi Zhang, “AES encryption algorithm based on the high performance computing of GPU,” *Communication Software and Networks, 2010. ICCSN'10. Second International Conference on*. IEEE, 2010.

[12] Michalakes, John, and Manish Vachharajani. “GPU acceleration of numerical weather prediction,” *Parallel Processing Letters*, Vol.18, No.4, pp.531-548, 2008.

[13] Baek, Byung-Hyun, Hyun-Jong Lee, and Doosung Hwang. “Guitar Tab Digit Recognition and Play using Prototype based Classification,” *The Korean Society of Computer and Information*, Vol.21, No.9, pp.19-25, 2016.

[14] R. C Gonzalez and R. E Woods, “Digital Image Processing,” Pearson Education, New Jersey, 2010.

[15] Richard O. Duda, Peter E. hart, and David G. Stork, Pattern Classification, 2nd, Willey-Interscience, 2000.

[16] Peemen, Maurice, Bart Mesman, and Henk Corporaal. “Efficiency Optimization of Trainable Feature Extractors for a Consumer Platform,” *ACIVS*, Vol.6915. 2011.

[17] B.E. Boser, I.M. Guyon, V.N. Vapnik, “A Training Algorithm for Optimal Margin Classifiers,” *Proc. Fifth Ann. Workshop Computational Learning Theory*, pp.144-152, 1992.

[18] Ward, Jonatan, et al. “Efficient mapping of the training of Convolutional Neural Networks to a CUDA-based cluster,” Eindhoven University of Technology, The Netherlands 12, 2011.

[19] S. Y. Shim, D. H. Hwang, “Prototype based Classification by Generating Multidimensional Spheres per Class Area,” *Journal of The Korea Society of Computer and Information*, Vol.20, No.2, Feb. 2015.

[20] L.J.P. van der Maaten and G.E. Hinton. “Visualizing High-Dimensional Data Using t-SNE,” *Journal of Machine Learning Research*, Vol.9, pp.2579-2605, Nov. 2008.

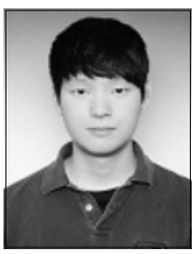
[21] Pedregosa, Fabian, et al. “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, Vol.12. Oct. pp.2825-2830, 2011.

[22] Chollet, François. “Keras: Deep learning library for theano and tensorflow,” https://keras.io 7.8, 2015.

[23] Kingma, Diederik, and Jimmy Ba. “Adam: A method for stochastic optimization,” arXiv preprint arXiv:1412.6980, 2014.

허 재 혁

https://orcid.org/0000-0001-6311-6674
e-mail : loplan98@gmail.com
2017년 단국대학교 컴퓨터과학과(학부)
2017년~현 재 단국대학교 컴퓨터공학과 석사과정
관심분야 : Machine Learning, Parallel Processing, Image Processing



이 현 종

https://orcid.org/0000-0002-2990-1545
e-mail : guswhd321@naver.com
2017년 단국대학교 컴퓨터과학과(학부)
2017년~현 재 단국대학교 소프트웨어학과 석사과정
관심분야 : Machine Learning, Parallel Processing, Image Processing

황 두 성

https://orcid.org/0000-0003-1840-9296
e-mail : dshwang@dankook.ac.kr
1986년 충남대학교 계산통계학과(학부)
1990년 충남대학교 전자계산학과(석사)
2003년 Wayne State Univ. 컴퓨터과학전공 인공지능(박사)
2003년~현 재 단국대학교 소프트웨어학과 교수
관심분야 : Machine Learning, Parallel Processing, Image Processing