Journal of Korea Multimedia Society Vol. 22, No. 6, June 2019(pp. 665-675)
https://doi.org/10.9717/kmms.2019.22.6.665

앙상블 학습 알고리즘을 이용한 컨벌루션 신경망의 분류 성능 분석에 관한 연구

박성욱[†], 김종찬[††], 김도연[†††]

A Study on Classification Performance Analysis of Convolutional Neural Network using Ensemble Learning Algorithm

Sung-Wook Park[†], Jong-Chan Kim[††], Do-Yeon Kim[†††]

ABSTRACT

In this paper, we compare and analyze the classification performance of deep learning algorithm Convolutional Neural Network(CNN) ac cording to ensemble generation and combining techniques. We used several CNN models(VGG16, VGG19, DenseNet121, DenseNet169, DenseNet201, ResNet18, ResNet34, ResNet50, ResNet101, ResNet152, GoogLeNet) to create 10 ensemble generation combinations and applied 6 combine techniques(average, weighted average, maximum, minimum, median, product) to the optimal combination. Experimental results, DenseNet169-VGG16-GoogLeNet combination in ensemble generation, and the product rule in ensemble combination showed the best performance. Based on this, it was concluded that ensemble in different models of high benchmarking scores is another way to get good results.

Key words: Deep Learning, Computer Vision, CNN, Ensemble Learning Algorithm

1. 서 론

컴퓨터 비전(Computer Vision) 분야에서 이미지 인식(Image Recognition) 문제는 지난 몇 년 사이 큰 개선을 이뤘다. 이 개선에는 '컴퓨팅 파워의 증가', 'GPGPU(General-Purpose computing on Graphics Processing Units)', '인터넷 기술의 발전'등의 요소가 크게 기여했다. 컴퓨팅 파워의 증가는 오류 역전파 (Back-Propagation) 기법의 개발과 함께 딥러닝(Deep Learning)[1] 알고리즘의 진화를 촉진시켰고 GPGPU와 같은 GPU(Graphics Processing Unit) 범용 프로그래밍 방법의 개발은 신경망(Neural Network)을 병렬로 학습할 수 있게 하여 기존 대비 작업 속도를 10배 이상 향상시켰다. 인터넷 기술의 발전은 대규모 학습 데이터세트(Dataset)를 손쉽게 확보할 수 있게 했다. 딥러닝은 이러한 요소들을 통합해 정밀한 이미지 인식을 할 수 있다. 즉, 딥러닝이 이미지 인식 성능 향상의 가장 큰 견인차가 됐다.

이미지 인식 작업에 특화된 컨벌루션 신경망(Convolutional Neural Network, CNN)[2]은 동물의 시각 피질을 인공적으로 흉내 내어 필기체 숫자나 얼굴과 같은 복잡한 이미지를 효율적으로 인식할 수 있다. CNN은 크게 컨벌루션층에서 이미지를 변환하고 풀링(Pooling)층에서 차원을 축소하며 분류(Classifi-

※ Corresponding Author : Do-Yeon Kim, Address: (57922) Jungang-ro 255, Sunchon, Jeonnam, Korea, TEL : +82-61-750-3628, FAX : +82-61-750-3620, E-mail : dykim@sunchon.ac.kr
Receipt date : May 2, 2019, Revision date : June 10, 2019
Approval date : June 13, 2019

[†] Dept. of Computer Engineering, Sunchon National University (E-mail : park7231654@naver.com)
[††] Dept. of Computer Engineering, Sunchon National University (E-mail : seaghost@sunchon.ac.kr
[†††] Dept. of Computer Engineering, Sunchon National University

cation)층에서 클래스(Class)를 분류한다. CNN은 선구적인 연구를 계속해 왔기 때문에 특정 분야의 이미지 인식에서는 사람보다 높은 정밀도를 얻을 수 있지만 아직 완벽한 성능은 내지 못한 실정이다[3].

이미지 인식 분야에서는 이전보다 분류 성능을 높이기 위해 앙상블 방법을 주로 사용한다. 앙상블이란 여러 전문가로부터 얻은 다수의 의견을 가장 합리적이면서 효율적인 방법으로 결합하여 의사결정하는 방법이다. 딥러닝에서 앙상블 방법은 여러 모델(Model)이 출력한 예측값(Prediction Value)을 적절한 방법으로 결합하여 더 높은 정확도(Accuracy)를 출력한다[4]. 앙상블 방법은 모델을 여러 개 제작하는 앙상블 생성 단계와 복수의 예측값을 결합하는 앙상블 결합 단계로 이뤄진다. 앙상블을 사용하는 이유는 가장 우수한 모델 하나보다 앙상블의 성능이 더 우월하기 때문이다[5-6]. 하지만 어떤 모델들을 조합해서 생성하고 어떤 규칙들을 사용해야 최적인지 참조할만한 연구 사례가 아직 부족한 실정이다.

본 논문에서는 앙상블 생성 및 결합 규칙에 따라 CNN의 분류 성능이 어떻게 변화하는지 비교 분석했다. 학습에 사용된 CNN 모델은 Simonyan et al[7]의 VGGNet, Szegedy et al[8]의 GoogLeNet, He et al[3]의 ResNet, Huang et al[9]의 DenseNet이고 각 모델 출력층의 클래스 수를 변형시켜 사용했다. 딥러닝 라이브러리로 텐서플로(Tensorflow)[10], 상위 툴로 케라스(Keras)[11]를 사용했다. 텐서플로는 복잡한 신경망을 구성하여 딥러닝 문제를 효과적으로 해결하는 공개소스 소프트웨어 라이브러리다. 케라스는 딥러닝 모델을 만들기 위한 고수준의 구성 요소를 제공하는 모델 수준의 라이브러리다. 실험 결과, 앙상블 생성에서는 DenseNet169-VGG16-GoogLeNet 조합이, 앙상블 결합에서는 곱 규칙(Product Rule)이 가장 우수한 성능을 보였다.

2. 앙상블 방법

일련의 모델로부터 출력된 예측값을 결합하면 가장 우수한 모델 하나보다 더 좋은 예측값을 얻을 수 있다. 단일 CNN 모델이 영상 내 모든 패턴을 포착하지 못하는 경우, 이와 같이 문제를 분리하여 다수의 모델을 학습하는 앙상블 방법은 분류 성능을 향상시키기 위한 방법 중 하나다. 딥러닝에서는 앙상블 방법도 규제(Regularization) 기법의 하나로 사용된다.

2.1 앙상블 생성

앙상블 생성단계에서는 서로 다른 구조를 가진 모델을 여러 개 생성할 수도 있고, 유사한 구조의 모델을 사용하되 데이터 증대(Data Augmentation), 가중치 초기화(Weight Initialization), 최적화기(Optimizer), 학습률(Learning Rate)등을 다르게 설정하고 훈련시킬 수도 있다. ILSVRC(Imagenet Large Scale Visual Recognition Challenge)에 참가한 AlexNet [12], VGGNet[7], GoogLeNet[8]은 모두 유사한 구조의 모델 5개, 2개, 7개의 예측값을 결합하여 오류율(Error Rate)을 줄였다. 일반적인 앙상블 방법은 Fig. 1과 같다. Fig. 1의 앙상블 방법은 생성 단계와 결합 단계로 구분된다. 생성 단계에서 n개 모델은 훈련 및 검증(Training and Validation) 데이터세트를 이용해 학습되고, 학습이 완료되면 각 모델을 모두 저장한다. 이후 학습된 모델을 결합하기 위해 저장된 모델을 모두 불러온다. 결합 단계에서는 테스트(Test) 데이터세트를 예측에 이용한다. 결합된 n개 모델의 예측이 완료되면 선정한 결합 방법으로 각 예측을 결합하고 최종 예측값을 출력한다.

2.2 앙상블 결합

앙상블 결합 방법은 모델의 출력 형태에 따라 다르다. 출력 형태는 크게 원핫(One-hot) 방식을 취하는 라벨(Label), 순위, 확률 3가지로 구분한다. 확률값은 원핫 코드 및 순위로 변환할 수 있으나 역은 성립하지 않는다. 다중 클래스 분류문제에서 CNN은 보통 소프트맥스(Softmax) 활성 함수에서 출력된 확률 값을 평가에 사용한다. 확률 값은 원핫 코드 및

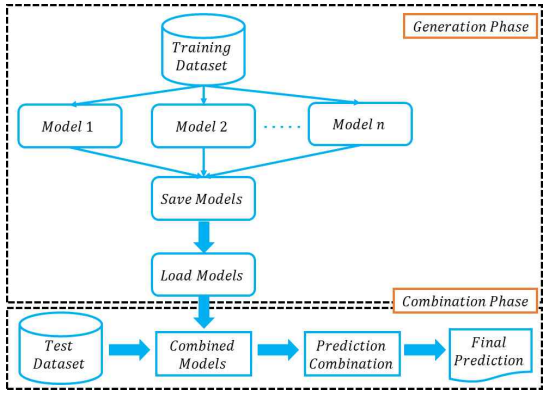

Fig. 1. Ensemble Methods.

순위 보다 많은 정보를 내포하고 있으므로 변환하지 않아도 무관하다.

식(1)~식(6)은 예측값 결합에 사용되는 규칙들이다. Kittler et al[13]와 Fumera et al[14]은 본 논문에서처럼 깊은 컨벌루션 신경망(Deep Convolutional Neural Network)[15]이 아닌 머신 러닝(Machine Learning) 알고리즘에 식(1)~식(6)을 적용하여 성능 비교 실험을 했다.

$$Average\ Rule: P_i = \frac{1}{N}\sum_{k=1}^{N} o_i^k \tag{1}$$

$$Weighted\ Average\ Rule: P_i = \frac{1}{N}\sum_{k=1}^{N} \alpha_k o_i^k \tag{2}$$

$$Maximum\ Rule: P_i = \underbrace{\max_k}\ o_i^{(k)} \tag{3}$$

$$Minimum\ Rule: P_i = \underbrace{\min_k}\ o_i^{(k)} \tag{4}$$

$$Median\ Rule: P_i = \underbrace{median}_k\ o_i^{(k)} \tag{5}$$

$$Product\ Rule: P_i = \prod_{k=1}^{N} o_i^{(k)} \tag{6}$$

6가지 결합 규칙 중 하나를 사용하여 $P=(P_1,P_2,...,P_c)^N$를 구한 후 최곳값 값을 가진 클래스를 선정해 최종 예측 결과에 반영한다. 식(1) ~ 식(6)에서 P는 확률, N은 모델 개수, k번째 모델의 출력을 $o_i^{(k)}$, i는 색인, c는 클래스 개수, α_k는 k번째 모델의 신뢰도(Confidence)다.

2.3 신경망 구조

VGGNet의 모든 층은 Fig. 2처럼 스트라이드(Stride)와 패딩(Padding) 크기가 1인 3*3 크기의 컨벌루션 연산과 스트라이드가 2인 2*2 크기의 최대 풀링(Max Pooling)을 사용한다. 그리고 마지막에는 완전연결(Fully-Connected, FC)층을 거치면서 결과를 출력하는 단순하게 층을 적층시킨 모델이다.

GoogLeNet은 크기가 다른 필터와 풀링을 여러 개 적용하여 그 결과를 연결(Concatenation)한다. 이 구조를 인셉션 모듈(Inception Module)이라고 한다. 가장 기본적인 인셉션 모듈 형태는 Fig. 3과 같이 3~4개의 가지를 가진다. 일찍이 Lin et al[16]의 NIN (Network In Network) 구조의 영감을 받아, 1*1 크기 필터를 많은 컨벌루션층에서 사용한다.

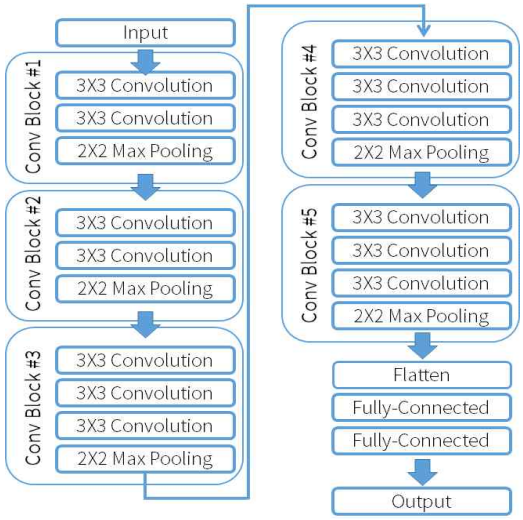

Fig. 2. Structure of VGG16 model.

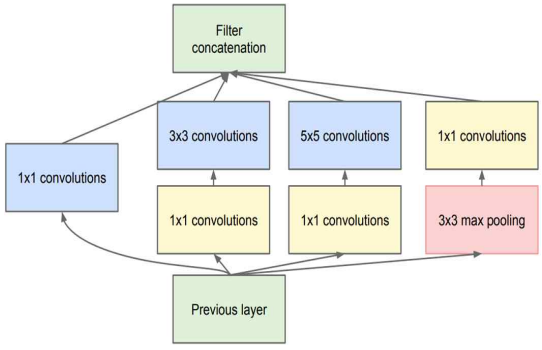

Fig. 3. Inception module with dimension reductions[8].

ResNet은 Fig. 4와 같이 하위층의 출력 x를 상위층의 $F(x)$와 합산(Addition)한다. 이때, 두 출력의 크기는 동일해야 하는데 크기가 일치하지 않을 경우 선형 변환(Linear Transformation)을 사용해 문제를 해결한다.

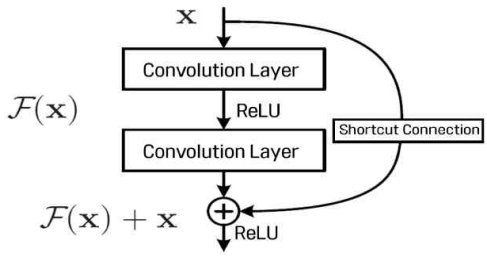

Fig. 4. Principle of residual learning[3].

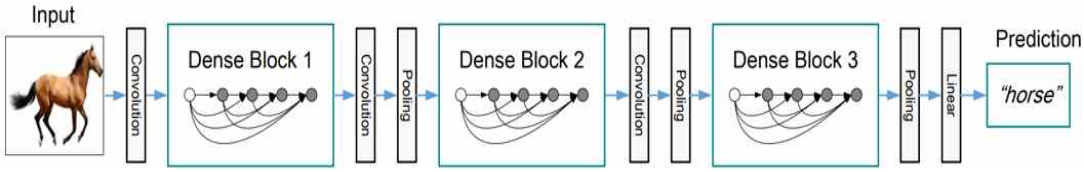

Fig. 5. DenseNet with three dense blocks[9].

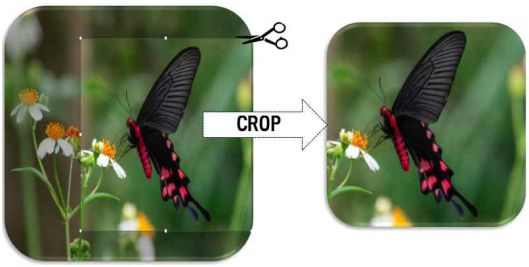

Fig. 6. Example of a cropped insect image.

DenseNet은 ResNet과 달리 합산이 아닌 연결을 통해 이전층의 출력과 현재층의 출력이 합쳐진다. 특징맵(Feature Map)의 크기를 줄이는 다운 샘플링(Down Sampling)을 용이하게 하기 위해 Fig. 5처럼 모델을 여러 개의 밀집된 블록(Dense Block)으로 나누고 블록 내 하위층 가중치(Weight)를 상위층에 모두 분산시킨다.

3. 실험 데이터세트 구성 및 환경설정

3.1 산림곤충 데이터세트 구성

실험 영상은 주변에서 쉽게 관찰할 수 있는 30종의 산림곤충으로 선정했다. 이미지넷(ImageNet)에서 5종, 웹 크롤링(Web Crawling)으로 25종 클래스의 영상을 수집했고 별도의 전처리(Preprocessing) 작업을 했다. 데이터는 현실 상황에 최대한 가까운 모습을 가지는 것이 이상적이라 사료되어 잡음(Noise)이 심한 데이터를 최대한 제거했고 Fig. 6와 같이 클래스 정보를 훼손하지 않는 범위에서 영상을 크롭(Crop)하여 실험 데이터세트로 구성했다.

훈련, 검증, 테스트 시 영상 크기는 VGGNet, GoogLeNet, ResNet, DenseNet 모두 299*299 화소(Pixel)로 조절했다. 불균형 클래스의 데이터세트는 결과의 신뢰도를 저해하기 때문에 각 곤충 클래스 영상에 플립핑(Flipping), 시어링(Shearing), 확대/축소, 회전 등의 데이터 증대를 적용하여 훈련세트 1,458장, 검증세트 162장, 테스트세트 180장으로 조정했다. 결과적으로 훈련세트는 43,740장, 검증세트는 4,860장, 테스트세트는 5,400장이 되고 실험은 총합 54,000장의 영상을 이용했다. 데이터세트 세부사항은 Table 1, 대표 곤충 영상은 Fig. 7과 같다.

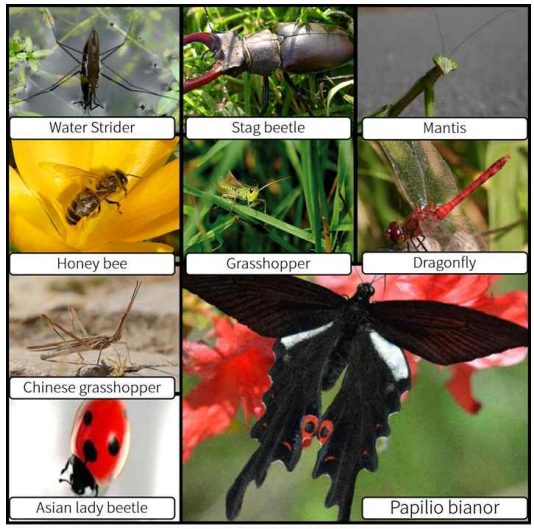

Fig. 7. Representative insect images used in the experiment.

3.2 실험환경

하드웨어 사양의 경우는 CPU(Central Processing

Table 1. Data set information

Image Input Size	Training Data set	Validation Data set	Test Data set
299 * 299	43,740 (1,458 * 30)	4,860 (162 * 30)	5,400 (180 * 30)

Table 2. Operating System and software version for reproduction

	Version
Operating System	Ubuntu 16.04.4 LTS
CUDA	9.0.176
cuDNN	7.1
Tensorflow	1.12.0
Keras	2.2.4
Python	3.5.2

Unit)는 Intel Core i7 7세대 Kaby Lake 7700K, Graphics Card는 NVIDIA TITAN Xp 12GB, RAM은 삼성 DDR4 32GB, SSD(Solid State Drive)는 삼성전자 850 Pro 512GB를 사용했다. 실험에 사용된 운영체제(Operating System) 및 소프트웨어 버전은 Table 2와 같다.

앙상블 생성 조합에 기준이 되는 모델을 선정하기 위해 VGGNet(VGG16, VGG19), ResNet(ResNet18, ResNet34, ResNet50, ResNet101, ResNet152), DenseNet(DenseNet121, DenseNet169, DenseNet201), GoogLeNet을 훈련시켰다. VGGNet, ResNet, DenseNet, GoogLeNet은 2.3절에서 언급했던 것처럼 학습 알고리즘이 모두 다르고, 공개 데이터세트인 이미지넷 데이터세트를 이용한 벤치마킹(Benchmarking) 점수가 높다. 위 신경망들을 실험 모델로 선정한 이유는 층의 깊이만 다른 모델을 앙상블 하는 것보다 서로 다른 구조의 모델을 앙상블 하는 것. 즉, 앙상블의 다양성이 더 좋은 성능을 발휘한다는 것을 보이기 위함이다. 총 11개 모델로, 1 에폭(Epoch)마다 검증 데이터로 훈련 결과를 모니터링 했다.

Fig. 8는 VGGNet, Fig. 9는 ResNet, Fig. 10은 DenseNet이며 100 에폭까지의 검증 정확도 및 손실

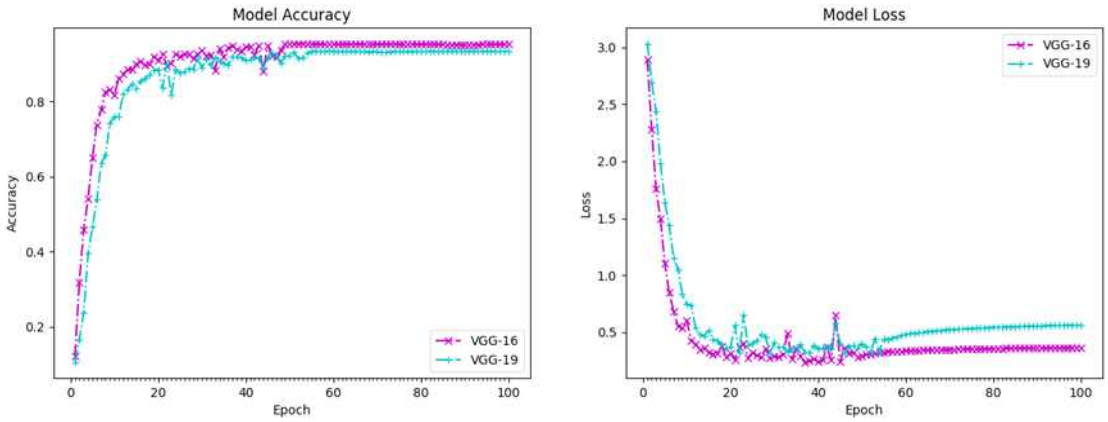

Fig. 8. Accuracy and loss rate curve of VGGNet.

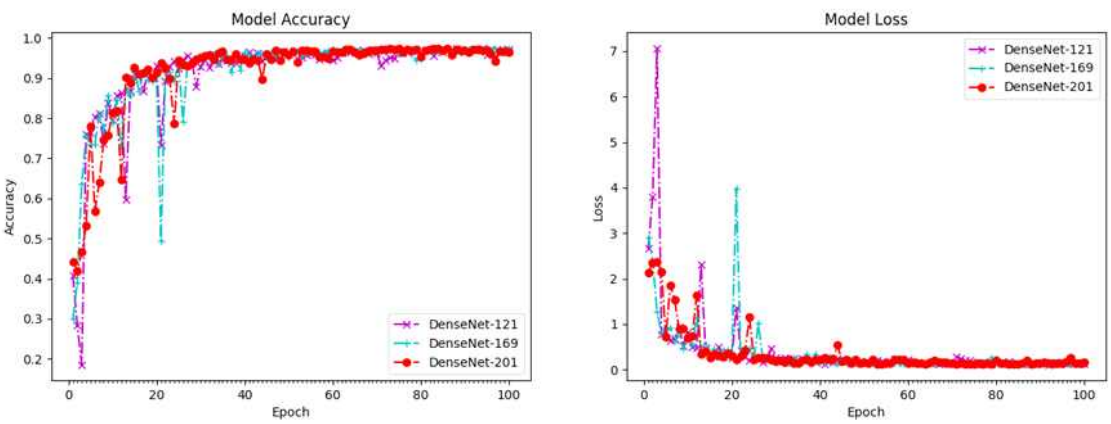

Fig. 9. Accuracy and loss rate curve of DenseNet.

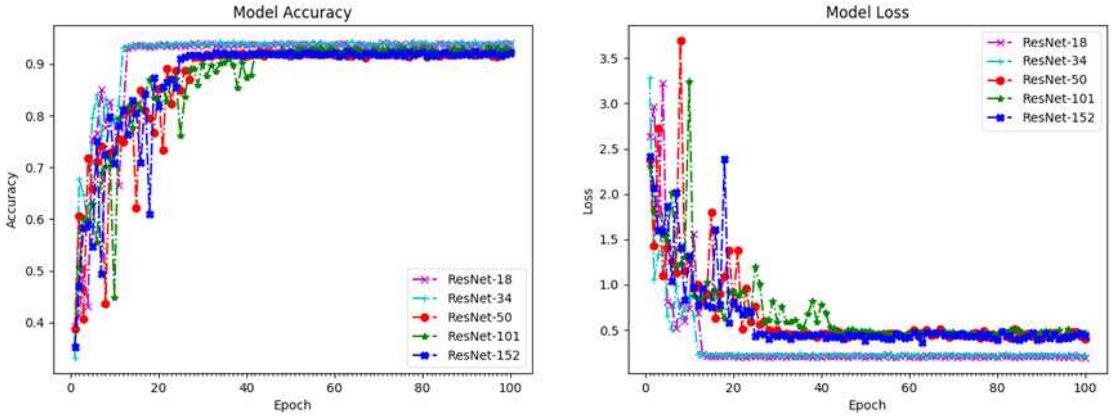

Fig. 10. Accuracy and loss rate curve of ResNet.

률 변화를 나타낸다. 검증 데이터의 정확도가 최고점이면 모델을 저장하는 모델 체크포인트(Model Checkpoint) 알고리즘을 적용했고, 최적 학습 에폭을 자동으로 탐색하게 설정했다. 훈련 결과는 VGGNet은 VGG16 모델이 52 에폭에서 95.40%, ResNet은 ResNet34 모델이 68 에폭에서 93.68%, DenseNet은 DenseNet169 모델이 95 에폭에서 97.46%의 가장 높은 정확도를 기록했다. GoogLeNet은 75 에폭에서 94.51%의 가장 높은 정확도를 기록했다. GoogLeNet은 단일 모델로 훈련했기 때문에 따로 그래프를 삽입하지 않았다. 최적 학습 에폭에서 VGG16의 손실률(Loss Rate)은 0.3098%, ResNet34는 0.3134%, DenseNet169는 0.1185%, GoogLeNet은 0.3134%를 기록했다. 모든 모델의 훈련 결과 세부사항은 Table 3과 같다.

훈련에 참여한 CNN 모델 모두 손실 함수(Loss Function)는 크로스 엔트로피(Cross Entropy)[17]를 사용했고, 활성 함수는 정규화 선형 유닛(Rectified Linear Unit, ReLU)[18]을 사용했다. 크로스 엔트로피는 오류가 클수록 높은 패널티(Penalty)를 부과하기 때문에 딥러닝에서 느린 학습 문제를 보완한다. ReLU는 양의 영역이 선형이므로 포화 현상이 발생하지 않고 음의 영역이 0이므로 신경망을 희소하게 만드는 효과가 있다.

모든 모델의 최상위 2개 층은 제거하고 전역 평균 풀링(Global Average Pooling, GAP)[16]층과 FC층을 추가시켰다. GAP은 별도의 파라미터(Parameter) 최적화 작업이 필요 없고 플래튼(Flatten)층과 달리 공간 정보를 반영하기 때문에 과적합(Overfitting)을 피할 수 있다. 최종 출력층의 활성 함수는 소프트맥

Table 3. Validation performance for each model

	Optimum Epoch	Val Acc[%]	Val Loss[%]
VGG-16	52	95.40	0.3098
VGG-19	56	93.38	0.4331
GoogLeNet	75	94.51	0.3134
ResNet-18	87	92.86	0.2947
ResNet-34	68	93.68	1.1086
ResNet-50	87	92.18	0.3487
ResNet-101	89	93.02	0.3398
ResNet-152	94	92.57	0.3385
DenseNet-121	94	97.32	0.1176
DenseNet-169	95	97.46	0.1185
DenseNet-201	73	97.38	0.1300

스를 사용했다. 소프트맥스는 최종 출력에서 가장 높은 확률값을 정답 클래스로 분류한다. 최적화기의 경우, VGGNet, ResNet, DenseNet은 확률적 경사 하강법(Stochastic Gradient Descent, SGD)[19]을 사용했고, 학습률='0.01', 모멘텀(Momentum)='0', 가중치 감쇠(Weight Decay)='0', 네스테로프(Nesterov)[20]="False"로 설정했다. GoogLeNet은 실효값 전파(Root Mean Square Propagation, RMSProp)[21]를 사용했고, 학습률='0.001', 실효값 감쇠(RMS Decay)='0.9', 엡실론(Epsilon)="None", 가중치 감쇠='0'으로 설정했다. 가중치 초기화의 경우, 참여한 CNN 모델 모두 글로로트 균등 분포(Glorot Uniform Distribution)[22] 방법을 사용했고, 미니 배치(Mini Batch) 크기는 '32', 에폭은 '100'으로 설정했다.

4. 실험 결과 및 고찰

하이퍼파라미터(Hyper-parameter) 설정은 3.2절에서 언급했던 내용과 동일하며 테스트 데이터세트 5,400장을 사용하여 출력된 정확도를 통해 앙상블 생성 조합 및 6가지 결합 규칙들의 성능을 평가했다. Table 4~Table 6의 앙상블 생성 실험에서는 단일 모델과 층의 깊이만 다른 모델 앙상블을 비교하고 층의 깊이만 다른 모델 앙상블과 서로 다른 구조의 모델 앙상블을 비교했다. 앙상블 생성 실험의 목적은 층의 깊이만 다른 모델을 앙상블 하는 것보다 서로 다른 구조의 모델을 앙상블 하는 것. 즉, 앙상블의 다양성이 더 좋은 성능을 발휘한다는 것을 보이기 위함이다. Table 7~Table 8의 앙상블 결합 실험에서는 ILSVRC 2012, 2014, 2015 우승팀(AlexNet, GoogLeNet, ResNet)들이 사용했던 평균 규칙과 가중 평균(Weighted Average), 최대(Maximum), 최소(Minimum), 메디안(Median), 곱 규칙들을 비교했다.

Table 4는 테스트 데이터세트 5,400장을 사용하여 출력된 단일 모델 및 앙상블 생성 조합의 테스트 정확도다. Table 4에서 VGG16 모델의 정확도는 95.67%다. Table 3의 검증 데이터세트 정확도에서 VGG16 모델이 VGG19 모델보다 2.02% 높았기 때문에 VGG16을 Table 4의 기준 모델로 선정했다. Table 5~Table 6의 기준 모델도 이와 같은 원리로 선정했다. Table 4에서 VGG16, VGG19 2개 모델 앙상블은 95.93%로 단일 모델과 0.26%의 성능 차이를 보였다. VGG16-DenseNet169 조합이 97.31%로 가장 높은 정확도를 보여줬고 VGG16-GoogLeNet이 96.98%, VGG16-ResNet이 96.51%로 그 뒤를 이었다. 층의 깊이만 다른 모델을 앙상블해도 단일 모델 보다는 결과가 좋았고 층의 깊이만 다른 모델을 앙상블 하는 것보다 서로 다른 구조의 모델을 앙상블 하는 것이

Table 4. Comparison of accuracy according to ensemble generation methods in VGG model

	Model Ensemble(Average Rule)				
	VGG-16	VGG-16 VGG-19	VGG-16 GoogLeNet	VGG-16 ResNet-34	VGG-16 DenseNet-169
Test Acc [%]	95.67	95.93	96.98	96.51	97.31

Table 5. Comparison of accuracy according to ensemble generation methods in DenseNet model

	Model Ensemble(Average Rule)				
	DenseNet-169	DenseNet-121 DenseNet-169 DenseNet-201	DenseNet-169 VGG-16 GoogLeNet	DenseNet-169 VGG-16 ResNet-34	DenseNet-169 GoogLeNet ResNet-34
Test Acc [%]	97.11	97.44	97.79	97.61	97.70

Table 6. Comparison of accuracy according to ensemble generation methods in ResNet model

	Model Ensemble(Average Rule)		
	ResNet-34	ResNet-18, ResNet-34, ResNet-50, ResNet-101, ResNet-152	ResNet-34, VGG-16, GoogLeNet, DenseNet-169
Test Acc [%]	94.30	96.29	97.25

더 나은 결과를 보였다.

Table 5의 5가지 실험에서는 DenseNet169-VGG16-GoogLeNet 조합이 97.79%로 가장 높은 정확도를 보여줬다. 단일 모델 DenseNet169는 97.11%로 가장 낮은 정확도를 보였고 DenseNet169-VGG16-GoogLeNet과 0.68%의 차이를 보였다. DenseNet169-GoogLeNet-ResNet34가 97.70%로 2위, DenseNet169-VGG16-ResNet34가 97.61%로 그 뒤를 따랐다. DenseNet121-DenseNet169-DenseNet201은 단일 모델 DenseNet169 보다는 정확도가 0.33% 높지만 최종 순위 4위를 기록했다. Table 4와 같이, 층의 깊이만 다른 모델을 앙상블해도 단일 모델 보다는 결과가 좋았고 층의 깊이만 다른 모델을 앙상블 하는 것보다 서로 다른 구조의 모델을 앙상블 하는 것이 더 나은 결과를 보였다.

Table 6의 3가지 실험에서는 단일 모델 ResNet34가 94.30%의 가장 낮은 정확도를 보여줬다. ResNet34-VGG16-GoogLeNet-DenseNet169 조합이 97.25%로 가장 높은 정확도를 보여줬고 ResNet34와 2.95%의 차이를 보였다. 실험에서, 서로 구조가 다른 모델은 4개이기 때문에 4개 모델을 조합해서 실험하고 비교했다. ResNet18-ResNet34-ResNet50-ResNet101-ResNet152의 경우, 96.29%로 2위를 기록했다. 이처럼 앙상블 생성에서 큰 수의 법칙(Law of Large Numbers)이 항상 성립되는 것은 아니다.

Table 7은 앙상블 생성 실험에서 가장 성능이 좋았던 DenseNet169-VGG16-GoogLeNet 조합에 가중 평균 규칙을 적용한 결과다. 모델의 신뢰도를 주어 예측값을 결합하는 경우, 가중 평균 규칙을 사용한다. α는 모델의 신뢰도를 나타내고 α_1은 첫 번째 모델의 신뢰도를 뜻한다. Table 7에서 모든 α값의 합은 1이다. DenseNet169-VGG16-GoogLeNet의 첫 번째 모델은 DenseNet169, 두 번째 모델은 VGG16, 세 번째 모델은 GoogLeNet이다. 실험은 기준이 되는 모델 1개와 나머지 모델 2개의 신뢰도를 변경하며 진행했다. 이때, 나머지 모델들의 신뢰도는 동일하게 설정했다. 총 18가지 실험에서 기준 모델이 DenseNet169일 때 97.98%의 가장 우수한 성능을 보였다. VGG16과 GoogLeNet은 기준 모델의 신뢰도가 0.4, 나머지 모델의 신뢰도가 0.3일 때 가장 높은 정확도를 기록했지만 1위와 0.39%, 0.32%의 성능 차이를 보였다.

Table 8은 앙상블 생성 실험에서 가장 성능이 좋았던 DenseNet169-VGG16-GoogLeNet 조합에 평균, 최대, 최소, 메디안, 곱 규칙을 적용한 결과다. 총 5가지 실험에서 곱 규칙이 98.05%로 가장 높은 정확

Table 7. Comparison of accuracy according to the classifier confidence in a weighted average rule

Weighted Average Rule					
Confidence	Test Acc[%]	Confidence	Test Acc[%]	Confidence	Test Acc[%]
$\alpha_1 = 0.9$ $\alpha_2, \alpha_3 = 0.05$	97.20	$\alpha_2 = 0.9$ $\alpha_1, \alpha_3 = 0.05$	95.79	$\alpha_3 = 0.9$ $\alpha_1, \alpha_2 = 0.05$	95.09
$\alpha_1 = 0.8$ $\alpha_2, \alpha_3 = 0.1$	97.33	$\alpha_2 = 0.8$ $\alpha_1, \alpha_3 = 0.1$	95.85	$\alpha_3 = 0.8$ $\alpha_1, \alpha_2 = 0.1$	95.18
$\alpha_1 = 0.7$ $\alpha_2, \alpha_3 = 0.15$	97.37	$\alpha_2 = 0.7$ $\alpha_1, \alpha_3 = 0.15$	95.92	$\alpha_3 = 0.7$ $\alpha_1, \alpha_2 = 0.15$	95.44
$\alpha_1 = 0.6$ $\alpha_2, \alpha_3 = 0.2$	97.55	$\alpha_2 = 0.6$ $\alpha_1, \alpha_3 = 0.2$	96.22	$\alpha_3 = 0.6$ $\alpha_1, \alpha_2 = 0.2$	95.85
$\alpha_1 = 0.5$ $\alpha_2, \alpha_3 = 0.25$	97.98(A)	$\alpha_2 = 0.5$ $\alpha_1, \alpha_3 = 0.25$	97.03	$\alpha_3 = 0.5$ $\alpha_1, \alpha_2 = 0.25$	97.09
$\alpha_1 = 0.4$ $\alpha_2, \alpha_3 = 0.3$	97.68	$\alpha_2 = 0.4$ $\alpha_1, \alpha_3 = 0.3$	97.59	$\alpha_3 = 0.4$ $\alpha_1, \alpha_2 = 0.3$	97.66

Table 8. Comparison of accuracy according to ensemble combine technique in optimum model

	Combine Technique				
	Average	Maximum	Minimum	Median	Product
Test Acc [%]	97.79	97.51	97.88	97.85	98.05

도를 보여주었고 Table 7의 가중 평균 규칙 결과 A보다도 0.07% 더 높았다. 최대 규칙을 제외한 최소, 메디안 규칙은 정확도가 평균 규칙보다 0.09%, 0.06% 높았다. 최대 규칙은 정확도가 평균 규칙보다 0.28% 낮았지만 단일 모델 및 층의 깊이만 다른 모델을 앙상블 하는 방법보다 성능이 좋았다.

5. 결 론

본 논문에서는 CNN의 분류 성능을 개선하고자 다양한 앙상블 학습 알고리즘들을 실험했다. 여러 CNN 모델(VGG16, VGG19, DenseNet121, DenseNet169, DenseNet201, ResNet18, ResNet34, ResNet50, ResNet101, ResNet152, GoogLeNet)을 이용해 10가지 앙상블 생성 조합을 만들었고 최적 조합에 6가지 결합 규칙(평균, 가중 평균, 최대, 최소, 메디안, 곱)을 적용했다.

10가지 앙상블 생성 실험에서는 DenseNet169-VGG16-GoogLeNet 조합이 97.79%의 정확도로 가장 높은 성능을 보여줬다. Table 4~Table 6 결과에서 층의 깊이만 다른 모델을 앙상블 하는 것보다 서로 다른 구조의 모델을 앙상블 하는 것이 더 좋은 결과를 보였다. Table 6 결과에서 앙상블 생성에서 큰 수의 법칙이 항상 성립되지 않음을 보였다. 위 결과들을 종합해보면, 앙상블 생성에 참여한 모델 개수나 최상위 모델이 얼마나 우수한지보다 후보 모델의 다양성이 성능 향상에 더 크게 기여할 것이라 판단된다.

6가지 결합 규칙을 이용한 23가지 앙상블 결합 실험에서는 곱 규칙이 98.05%의 정확도로 가장 높은 성능을 보여줬다. 곱 규칙은 모델 하나가 아주 낮은 확률을 출력하면 해당 클래스의 최종 예측값이 0에 가까워지는 문제가 생기지만 앙상블이 본 논문처럼 높은 벤치마킹 점수의 모델들로만 구성되었을 때는 우수한 성능을 낼 수 있었다. 가중 평균 규칙 방법은 97.98%의 정확도로 두 번째로 높은 성능을 기록했다. 신뢰도 1을 모든 모델에 균등하게 나누는 방법이 그렇지 않은 방법보다 성능이 좋았고 단일 모델 테스트에서 정확도가 가장 높았던 DenseNet169에 다른 모델보다 높은 신뢰도를 주는 선택이 가장 나은 결과를 보였다.

가중 평균 규칙의 경우, 다른 조합방식으로 추가 실험 하여 곱 규칙과의 성능 차이를 더 좁힐 수 있을 것이라 판단된다. 두 규칙 간 성능 차이는 0.07%로 크지 않다고 볼 수 있지만 전 세계에서 가장 규모가 큰 머신러닝, 딥러닝 경진대회 캐글(Kaggle)[23]에서는 소수점 0.01%의 개선이 순위를 뒤바꾼다[24-27].

결과적으로, 높은 벤치마킹 점수의 상이한 모델을 앙상블 하는 것도 유의미한 결과를 얻을 수 있는 방법이라고 결론 지었다. 테스트 정확도는 임의 탐색(Random Search)[28], 베이지안(Bayesian)[29] 같은 하이퍼라라미터 최적화 알고리즘을 이용하면 보다 향상시킬 수 있을 것으로 사료된다.

향후 과제는 실험에 사용된 학습 알고리즘들의 상하관계를 보다 명확하게 밝혀 앙상블 예측에서 시간과 비용을 절약하는데 도움이 될 수 있는 연구가 필요하다.

REFERENCE

[1] L. Yann, B. Yoshua, and H. Geoffrey, “Deep Leaning,” *Nature*, Vol. 521, No. 7553, pp. 436-444, 2015.

[2] Y. Lecun, L. Bottou, and Y. Bengio, “Gradient-based Learning Applied to Document Recognition,” *Proceeding of The IEEE*, Vol. 86, No. 11, pp. 2278-2324, 1998.

[3] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” *arXiv*, arXiv:1512.03385, 2015.

[4] S.W. Park, J.C. Kim, and D.Y. Kim, “A Study on Classification of Convolutional Neural Network using Ensemble Combining Technique,” *Proceeding of the Fall Conference of the Korea Multimedia Society*, pp. 757, 2018.

[5] D. Ciresan, U. Meier, and J. Schmidhuber, "Multi-column Deep Neural Networks for Image Classification," *Proceedings of the 25th IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3642-3649, 2012.

[6] The BellKor Solution to the Netflix Grand Prize, https://www.netflixprize.com/assets/GrandPrize2009_BPC_BellKor.pdf (accessed Feb., 14, 2019).

[7] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *arXiv*, arXiv:1409.1556, 2015.

[8] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S.

Reed, D. Anguelov, et al., "Going Deeper with Convolutions," *arXiv*, arXiv:1409.4842, 2014.

[9] G. Huang, Z. Liu, L.V.d. Maaten, and K.Q. Weinberger, "Densely Connected Convolutional Networks," *Proceeding of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2261-2269, 2017.

[10] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, et al., "TensorFlow: Large-scale Machine Learning on Heterogeneous Systems," *arXiv*, arXiv:1603.04467, 2015.

[11] Keras, https://github.com/fchollet/keras (accessed Jan., 15, 2019).

[12] K. Alex, S. Ilya, and H. Geoffrey, "ImageNet Classification with Deep Convolutional Neural Networks," *Proceeding of Advances in Neural Information Processing System*, pp. 1097-1105, 2012.

[13] G. Fumera and F. Roli, "A Theoretical and Experimental Analysis of Linear Combiners for Multiple Classifier Systems," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 27, No. 6, pp. 942-956, 2005.

[14] J. Kittler, M. Hatef, R.P.W. Duin, and J. Matas, "On Combining Classifiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 20, No. 3, pp. 226-239, 1998.

[15] Y. Jeong, L. Ansari, J. Shim, and J. Lee, "A Car Plate Area Detection System Using Deep Convolution Neural Network," *Journal of Korea Multimedia Society*, Vol. 20, No. 8, pp. 1166-1174, 2017.

[16] M. Lin, Q. Chen, and S. Yan, "Network in Network," *arXiv*, arXiv:1312.4400v3, 2014.

[17] K. Janocha and W.M. Czarnecki, "On Loss Functions for Deep Neural Networks in Classification," *arXiv*, arXiv:1702.05659, 2017.

[18] V. Nair and G. Hinton, "Rectified Linear Units Improve Restricted Boltzmann Machines," *Proceedings of the 27th International Conference on Machine Learning*, pp. 807-814, 2010.

[19] L. Bottou, "Stochastic Gradient Descent Tricks," *Neural Network, Tricks of the Trade, Reloaded*, Vol. 7700, pp. 430-445, 2012.

[20] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the Importance of Initialization and Momentum in Deep Learning," *Proceeding of the 30th International Conference on Machine Learning*, Vol. 28, pp. 1139-1147, 2013.

[21] T. Tieleman and G. Hinton, *RMSProp: Divide the Gradient by a Running Average of I ts Recent Magnitude*, COURSERA: Neural Networks for Machine Learning Technical Report, 2012.

[22] X. Glorot and Y. Bengio, "Understanding the Difficulty of Training Deep Feedforward Neural Networks," *Proceedings of the International Conference on Artificial Intelligence and Statistics*, pp. 249-256, 2010.

[23] Kaggle: Your Home for Data Science, https://www.kaggle.com/ (accessed Feb., 14, 2019).

[24] Santander Product Recommendation, https://www.kaggle.com/c/santander-product-recommendation (accessed Aug., 14, 2019).

[25] Tensorflow Speech Recognition Challenge, https://www.kaggle.com/c/tensorflow-speech-recognition-challenge (accessed Aug., 14, 2019).

[26] Porto Seguro's Safe Driver Prediction, https://www.kaggle.com/c/porto-seguro-safe-driver-prediction (accessed Aug., 14, 2019).

[27] State Farm Distracted Driver Detection, https://www.kaggle.com/c/state-farm-distracted-driver-detection (accessed Aug., 14, 2019).

[28] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *The Journal of Machine Learning Research*, Vol. 13, No. 1, pp. 281-305, 2012.

[29] J. Snoek, H. Larochelle, and R.P. Adams, "Practical Bayesian Optimization of Machine Learning Algorithms," *Proceedings of the 25th International Conference on Neural Information Processing Systems*, Vol. 2, pp. 2951-2959, 2012.

박 성 욱

2018년 순천대학교 컴퓨터공학과 (공학사)
2018년~현재 순천대학교 컴퓨터공학과 석사과정
관심분야 : 컴퓨터비전, 기계학습

김 종 찬

2000년 순천대학교 컴퓨터과학과 졸업(이학사)
2002년 순천대학교 대학원 컴퓨터과학과 졸업(이학석사)
2007년 순천대학교 대학원 컴퓨터과학과 졸업(이학박사)
2013년 서울대학교 자동화 시스템 연구소 선임연구원
관심분야 : 영상 처리, HCI, 콘텐츠, 컴퓨터그래픽스, 기계학습 데이터 분석 및 예측

김 도 연

1986년 충남대학교 계산통계학과 졸업(이학사)
2000년 충남대학교 대학원 정보통신공학과 졸업(공학석사)
2003년 충남대학교 대학원 컴퓨터공학과 졸업(공학박사)
1986년~1996년 한국원자력연구원 선임연구원
1997년~2008년 한국전력기술(주) 책임연구원
2008년~현재 순천대학교 컴퓨터공학과 교수
관심분야 : 컴퓨터비전, 컴퓨터보안, 기계학습