ISSN 1975-8359(Print) / ISSN 2287-4364(Online)
The Transactions of the Korean Institute of Electrical Engineers Vol. 67, No. 6, pp. 767~772, 2018
http://doi.org/10.5370/KIEE.2018.67.6.767

CNN 구조의 진화 최적화 방식 분석

Analysis of Evolutionary Optimization Methods for CNN Structures

서 기 성*
(Kisung Seo)

Abstract - Recently, some meta-heuristic algorithms, such as GA(Genetic Algorithm) and GP(Genetic Programming), have been used to optimize CNN(Convolutional Neural Network). The CNN, which is one of the deep learning models, has seen much success in a variety of computer vision tasks. However, designing CNN architectures still requires expert knowledge and a lot of trial and error. In this paper, the recent attempts to automatically construct CNN architectures are investigated and analyzed. First, two GA based methods are summarized. One is the optimization of CNN structures with the number and size of filters, connection between consecutive layers, and activation functions of each layer. The other is an new encoding method to represent complex convolutional layers in a fixed-length binary string, Second, CGP(Cartesian Genetic Programming) based method is surveyed for CNN structure optimization with highly functional modules, such as convolutional blocks and tensor concatenation, as the node functions in CGP. The comparison for three approaches is analysed and the outlook for the potential next steps is suggested.

Key Words : Convolutional neural network, Optimization, Genetic algorithm, Cartesian genetic programming

1. 서 론

딥러닝은 기존의 신경망 및 다른 기계학습 방법에 비해서 뛰어난 학습 능력을 보이며 거의 모든 분야에 널리 응용되고 있다[1,2]. 딥러닝 기법의 종류로는 DNN (Deep Neural Network), CNN (Convolutional Neural Networks), Auto-Encoders, Deep Belief Networks (DBNs), Recurrent Neural Networks (RNN), 그리고 LSTM(Long Short-Term Memory) 등이 있다[1,2]. 딥러닝 네트워크의 규모가 깊고 커짐에 따라, 딥러닝 구조에 대한 최적화 문제가 점차 관심을 끌고 있다. 딥러닝의 각 기법들은 구조적인 측면에서 유사성과 상이성이 모두 존재하며, 이에 대해서 개별적인 접근법이 필요하다. 본 연구에서는 영상 인식분야에서 탁월한 성능을 보고이고 있는 CNN 구조의 최적화에 대해서 논의한다.

진화연산은 다윈의 진화원리를 컴퓨터 알고리즘화한 방향성 있는 확률적 최적화 방법으로, NP Hard 또는 NP Complete 문제에 매우 효과적인 것으로 알려져 있다[3]. 다른 접근법에 비해서 최적화 성능이 우수하기 때문에, 일반적인 구조 최적화 문제뿐만 아니라 딥러닝 구조 최적화 연구에도 적용이 되고 있다. 일반적인 DNN에 대해서는 최적화 연구가 일부 시도되어 왔으며, 그 중 가장 주목할 만한 연구는 NEAT[4] 관련 기법들이다. 네트워크내의 층 수 및 층 내의 노드수, 그리고 이들의 연결 방법을 진화연산을 사용하여 최적화를 수행하였다. 이후 HyperNEAT [5], CPPN[6], DPPN[7]등으로 발전하여 왔고, 딥러닝이 대세를 차지하고 있는 와중에 경쟁기법으로 꾸준히 두각을 나타내고 있다. 특히, 학습 네트워크 구조의 진화최적화 측면에서는 탁월한 기여를 하고 있다.

한편 CNN은 DNN에 비하여 종류가 다른 층, 즉 컨볼루션과 풀링층의 조합으로 이루어져 더욱 구조가 복잡하고, 필터 크기 및 수, 다양한 컨볼루션 연산방식이 존재하여 경우의 수가 대폭 늘어난다. CNN 구조의 최적화는 방대한 연산으로 인해 거의 시도되지 않고 있었으나 최근에 GPU등의 발달로 일부 접근이 이루어지고 있다. 이러한 문제에 대한 최적화 접근은 문제의 탐색공간이 방대하므로 우수한 성능을 가진 기존 CNN 네트워크 구조를 일부 변형하거나 부분 탐색을 하는 경우가 대부분이다.

본 연구에서는 진화연산 기반의 CNN 구조 최적화에 대한 주요 연구를[8-10] 분석하고, 문제점 및 개선 방안을 고찰한다. 특히, 네트워크 구조의 유전자 표현 방법과, 생성 및 탐색 측면에서 주요 기법들인 GA, GP를 중심으로 비교 분석한다.

2. CNN(Convolutional Neural Network) 구조 최적화

2.1 CNN과 구조 문제

딥러닝의 한 종류로 영상인식에 주로 사용되는 CNN은 1989년 LeCun이 발표하였으며[11], 특징점을 사전에 추출하지 않고, 입

* Corresponding Author : Dept. of Electronics Engineering, Seokyeong University, Korea.
E-mail : ksseo@skuniv.ac.kr
Received : May 2, 2018; Accepted : May 29, 2018

Copyright ⓒ The Korean Institute of Electrical Engineers

This is an Open-Access article distributed under the terms of the Creative Commons Attribution Non-Commercial License (http://creativecommons.org/licenses/by-nc/3.0/)which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.

력 층에 영상 데이터를 직접 입력하여 컨볼루션 층을 통해 특징이 자동 추출되는 특징을 가지고 있다. CNN의 구조는 다음과 같이 구성된다. 그림 1과 같이 층 사이의 노드 쌍들 중 일부만 연결하는 컨볼루션(convolution) 층과 풀링(pooling) 층이 교대로 반복되며 특징 추출과 강인화에 관여하고, 후반부에 분류 목적에 사용되는 층 사이의 노드를 모두 연결하는 완전연결(fully connected)이 위치한다.

CNN 구조에 관한 변수들은 다음과 같다. 컨볼루션 층수, 풀링 층수, 층간 연결, 컨볼루션 층내의 필터 수 및 크기등이 있으며, 컨볼루션 연산에 필요한 패딩, 보폭, 풀링 연산의 방식등이 함께 존재한다. 또한 여러 개의 컨볼루션 층이 연이어질 수 있으며, 층간의 분기도 가능하여 연결 가능 복잡도는 더욱 증가한다. 네트워크 구조외에 학습률, 가중치 갱신법, 가중치 초기화, 배치 정규화, 활성화 함수등의 하이퍼 파라미터가 부가적으로 존재하나, 구조와 파라미터를 결합해서 최적화하기에는 거의 불가능한 문제이므로 논의에서 제외한다.

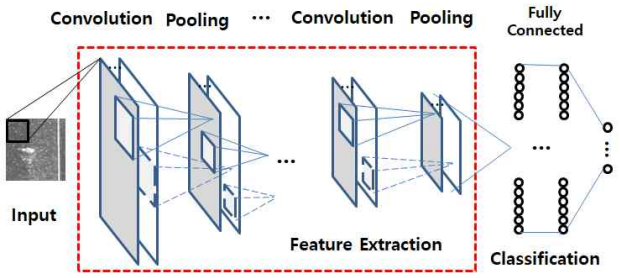

그림 1 CNN 모델 구조
Fig. 1 CNN model structure

2.2 진화 연산

진화연산은 유전자 표현, 유전자 생성, 적합도 평가, 선택, 유전연산 부분으로 구성되어 있다. 이중에서 구조/파라미터 최적화 문제에 대해서 밀접하게 관련된 부분은 해의 유전자 표현 방법과 생성 및 탐색이다.

먼저, 어떤 방식의 유전자 표현을 통해 구조 최적화를 접근할 것인가가 중요하다. 일반적 GA는 고정된 스트링의 유전자를 사용하기 때문에, 구조가 고정이거나 제한된 범위내에서 일부분의 가변이 가능하다. GP(Genetic Programming)[12]는 유전자 표현으로 트리구조를 사용함으로써 열린 가변 구조의 표현이 가능하다. CGP(Cartesian Genetic Programming)[13]는 GP의 변형으로 유전자 표현에 트리 대신 네트워크를 사용한다.

유전자 표현과 더불어 유전자의 생성 및 탐색이 역시 큰 관련이 있다. GA는 고정된 유전자 구조를 사용하기 때문에, 개체 해의 생성 역시 고정된 크기로 통일되며, 이들의 유전 연산 역시 닫혀져 있어 탐색시 특별한 문제가 발생되지 않는다. GP는 서브트리를 서로 교체하는 교배 연산이 잘 정의가 되어 있어, 제한없는 해의 생성 및 탐색이 가능하나, 현실적으로 트리의 크기가 너무 커질 경우 의미 없는 서브트리가(bloat) 나타나는 현상이 발생하여 탐색을 어렵게 하는 단점이 존재한다.

GA의 수행과정은 다음과 같다. 초기에 스트링으로 구성된 각 개체들을 임의로 생성한다. 그리고 각 개체를 해석하여 구한 후보해를 적합도 함수로 평가한다. 이후, 주어진 선택 방법에 의해 유전 연산에 참여할 개체들을 선택한다. 선택된 개체들을 대상으로 유전 연산을(교배, 돌연변이) 수행한다. 그리고 이 전체과정을 종료조건이 만족될 때까지 반복한다. GA에서의 교배 연산자는 임의의 점에서 부모의 스트링의 일부를 교체하는 방식으로 수행된다.

GP는 GA의 수행과정과 유사하며 유전자 표현에 스트링 대신 트리를 사용한다. 그리고 사전에 트리로 표현되는 유전자를 구성하는 함수와 터미널을 정의한다. 함수와 터미널은 해를 조합적으로 구성할 수 있는 요소들을 선택한다. GP에서의 교배와 돌연변이 연산자는 서브트리를 교체하는 방식으로 수행된다.

2.3 진화적 CNN 구조 최적화

일반적 진화연산에 의한 최적화와는 달리 CNN 구조의 최적화는 개체의 평가(evaluation) 과정이 CNN의 학습을 포함하기 때문에 전체 세대의(generations) 반복에 장시간의 연산이 필요하다. 즉 한 개체를 평가하기 위해서 수백 회(epochs)의 학습과정이 수행되어야 한다. 전체 수행 알고리즘은 그림 2와 같이 정리될 수 있다.

```
Algorithms - Evolution of CNN Structures {
  t <- 0
  initialize P(t)
  evaluate P(t) {
    CNN training for train dataset
    CNN test for test dataset
    calculate accuracy
  }
  while (not termination-condition) {
        t <- t+1
        select P(t) from P(t-1)
        recombination P(t)
        evaluate P(t) {
          CNN training for train dataset
          CNN test for test dataset
          calculate accuracy
        }
    }
  }
```

그림 2 CNN 구조의 진화 최적화 알고리즘
Fig. 2 Evolutionary optimization algorithm for CNN structures

3. GA 기반 CNN 최적화

3.1 GA 기반 CNN 구조 최적화 접근 1

GA로 CNN 구조를 표현할 경우, 조합최적화와 유사한 방식이 적합하다. 즉, 실수 대신 정수를 사용하여 층의 개수 및 연결 관계등을 나타내야 한다. 다음 연구는[8] 6개 층(conv 3, 풀링 2)으로 구성된 네트워크를 GA로 진화시킨 예이다. 여기서 최적화

변수는 각 층의 필터 크기와 수, 그리고 층간의 연결, 각층의 활성화 함수를 포함한다. 구체적인 유전자 표현은 다음과 같다. 총 4가지 그룹으로 구분되며, 첫 번째 3개의 필드는 컨볼루션 C1, C3, C5에 대한 필터 수를 나타낸다. 두 번째 3개의 필드는 컨볼루션 C1, C3, C5에 대한 필터 크기를 의미한다. 여기서, 각 필드마다 필터의 가로와 세로 변수가 존재한다. 세 번째 2개의 필드는 컨볼루션 층으로부터 풀링 층으로 노드별 연결관계를 나타낸다. 네 번째 6개의 필드는 3개의 컨볼루션 층과 2개의 풀링 층, 하나의 완전연결 층을 포함한 총 6개 층에 대한 활성화 함수 선택을 표시한다. 유전자 길이는 아래 그림과 같이 총 3+6+2+6=17개로 구성된다.

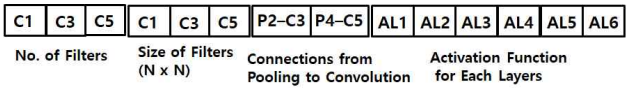

그림 3 CNN 구조에 대한 GA 유전자 표현
Fig. 3 GA representation for CNN structures

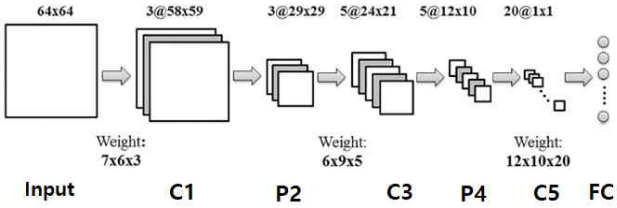

그림 4 CNN 구조에 대한 GA 표현
Fig. 4 Encoding of GA

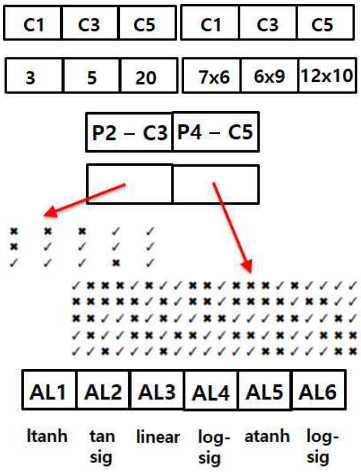

그림 5 CNN 구조에 대한 GA 유전자 예
Fig. 5 Example of GA Encoding for CNN Structures

그림 4는 총 6개 층에 대해서 GA 연산을 통해 얻어진 CNN 구성도가 나와 있으며, GA 유전자 해의 상세 예가 그림 5에 나와 있다. CNN 구조에 필요한 상당수 변수를 GA 유전자로 표현하고, 진화 최적화를 수행했다. 특히 풀링 층에서 컨볼루션 층으로 연결 관계를 일반적인 모든 연결 대신 연결을 특정하게 몇 종류로 구성하여 이를 선택하게 한 것으로 보인다. ORL과 Yale 얼굴 데이터베이스에 대해서 기존 CNN과 비교하여 약 1-2% 정도의 적은 성능 향상만을 보였다[8].

3.2 GA 기반 CNN 구조 최적화 접근 2

성능이 우수한 CNN 구조를 설계하는 것은 매우 어려운 일이나, 깊은 네트워크가 더 나은 인식결과를 보인 연구가 일반적으로 받아들여 진다[14][15]. 분기 구조의 추가가 유용한 결과를 보였고[16][17], 다양한 변형이 시도되었다[18].

다음 연구는[9] GA 유전자의 표현에 중점을 둔 연구로서, 컨볼루션 층의 개수와 컨볼루션 층간의 연결관계에 초점을 맞추었다. 이진 스트링으로 구성된 GA를 사용하여 CNN 네트워크 구조를 표현하였는데 복수개로 구성된 컨볼루션 층의 연결구조를 진화시켰다. 그림 6의 위의 그래프와 코드는 유전자의 예와 이에 대한 네트워크의 연결관계를 나타낸다. 예로, 1-00-111에서 처음 1은 A1-A2의 연결을 나타내고, 다음 두 00은 A1-A3와 A2-A3가 연결이 안되었음을 의미한다. 세 번째 111은 A1-A4, A2-A4, A3-A4들이 연결되었다는 것을 표시한다. 인코딩 규칙은 그룹당 첫 노드부터 차례로 노드 번호를 증가시켜가며 노드 쌍의 연결을 표현한다. A0는 디폴트 입력이고 A5는 디폴트 출력이다. 그림 6의 하단은 입력과 컨볼루션 층, 그리고 풀링층으로 이어지는 전체 CNN 네트워크에서 상단의 노드 연결이 컨볼루션 단계에 해당하는 것을 보여준다.

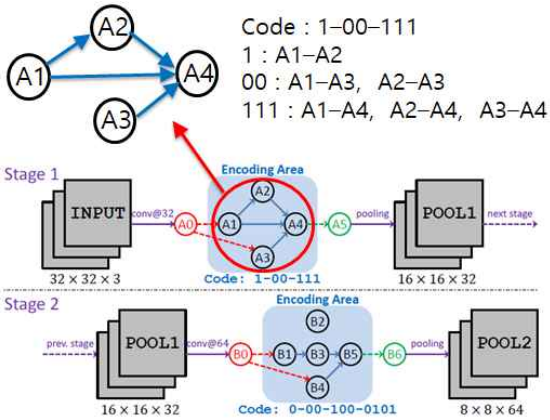

그림 6 CNN 구조에 대한 GA 해 표현 2
Fig. 6 Encoding of GA 2, and their optimum values as found by GA

그림 7은 VGGNet[19], ResNet[20], 그리고 DenseNet[21]에 대한 컨볼루션 층의 연결이 맨 밑의 유전자 해로 표현될 수 있음을 보여주고 있다. 전체 유전자의 일부가 빌딩 블록의 형태로 반복해서 제공될 수 있음을 보여준다. 컨볼루션 층내의 필터 수 및 크기는 고정되어 있어, 인셉션 모듈과 같이 다중 스케일 정보를 처리하는 데는 제한이 있다. 이는 대부분의 인코딩 기반 접근이 가지고 있는 문제점이다. 실험은 MNIST와 CIFAR10을 대상

으로 Titan-X GPU 상에서 수행되었다. 이진 스트링 유전자의 길이는 19이며, 군집수는 20개의 개체를 사용하고, 총 50세대를 수행하였다. 교배확률은 0.2로 매우 낮게, 돌연변이 확률은 0.8로 매우 높게 설정했다. 이는 유전자 길이가 길기 때문에 새로운 구조의 생성을 용이하기 위해서라고 설명이 되어있다. MNIST에 대한 개체당 학습 시간은 2.5분간이 걸리고, 전체 GA 진화과정은 2 GPU-days가 소요되었다. CIFAR10에 대해서는 개체당 학습 시간은 0.4 시간이 걸리고, 전체 GA 진화과정은 17 GPU- days가 소요되었다.

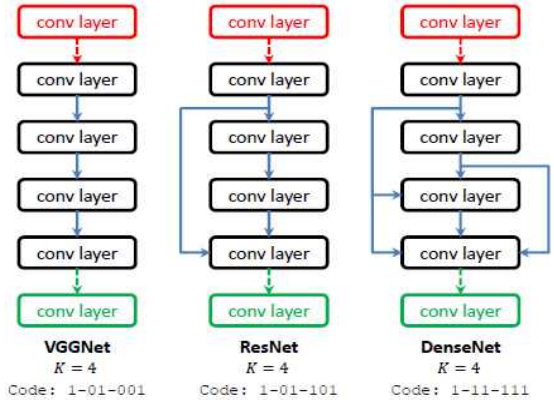

그림 7 VGGNet, ResNet and DenseNet의 기본 빌딩 블럭
Fig. 7 The basic building blocks of VGGNet, ResNet and DenseNet

CIFAR10 실험 결과를 보면 오류율이 7.1%로 기존 주요 연구들의 평균 성능 수준이거나 약간 더 높게 나왔다. 특히 DenseNet[13,14]의 성능 3.74% 보다는 상당히 떨어졌다. 이는 GA에 의한 최적화 대상이 컨볼루션 층내의 연결구조에 대한 것이므로, 성능에 영향을 주는 다른 구조적 부분이나 수 많은 관련 파라미터가 존재하기 때문이라고 생각된다.

4. GP 기반 CNN 최적화

CGP[13]는 Genetic Programming의 변형으로, 인덱스된 그래프 형식을 사용한다. 입력 노드, 출력노드, 노드 함수로 구성 된 셀 네트워크로 구성되며, 노드의 입력과 함수를 정수형 스트링으로 표현한다. 부분 모듈의 재사용성이 특징이다. CGP를 이용하여 CNN 구조를 구성하는 연구가 시도되었고, 기존의 GA 기반 방법에 비해 구조적 제한과 최적화 대상의 확대가 이루어졌다. 유전자 인코딩을 통한 노드들의 연결과 함께 CNN 모듈을 함수화하여 블록으로 사용함으로써, 높은 집적도와 함께 컨볼루션, 풀링, 그리고 연산함수를 포함하였다. 그림 8의 유전자 표현에서 첫 번째 노드의 C0 함수는 conv(32,3) 모듈의 입력이 0(input)으로부터 왔다는 표시이고, 그 다음 노드의 P1 함수는 pool(max)의 입력이 역시 0(input)에서 연결되었다는 의미이다. 세 번째 노드의 C3 함수는 conv(64,5) 모듈이고, 입력은 1번 블록인 conv(32,3) 모듈로부터 나왔다는 것을 나타낸다. 회색으로 음영진 부분은 실제 유전형(genotype)에서는 존재하나 표현형(phenotype)에서는 발현이 안된다. 그림 8의 상단에 있는 CGP 유전자 표현을 해석하여 CNN 구조로 나타나면 하단과 같다.

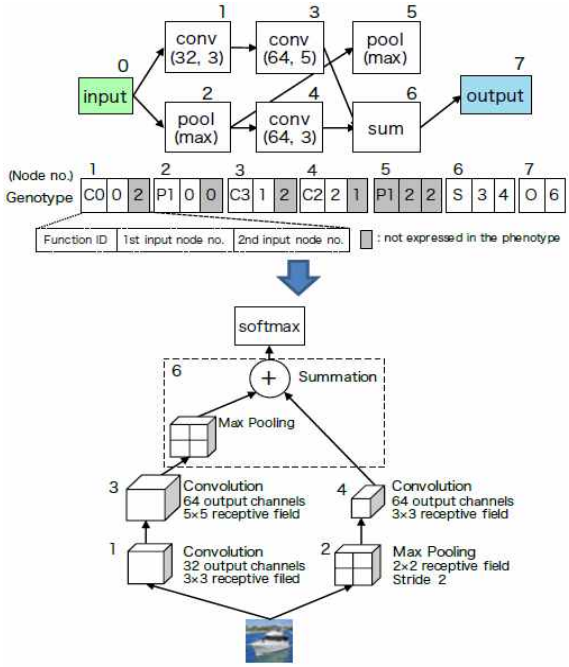

그림 8 CGP 유전자형과 표현형 예
Fig. 8 Example of a genotype and a phenotype

세대별 개체 생성에 (1+λ) steady state 생성 모델을 사용하였다. 여기서 λ=2를 사용했다. CGP 파라미터는 다음과 같다. 돌연변이 확률은 0.05, 행수는 5, 열수는 30, 전단계 연결 범위(levels-back)는 10으로 설정하였다. CPG 노드에 사용되는 함수와 터미널의 정의는 다음과 같이. CB(ConvBlock)는 컨볼루션 층 정의이고, 필터수와 크기를 인수로 가진다. MP(Max Pooling)와 AP(Average Pooling)가 풀링에 대한 모듈로 제공되고, Sum과 Concat(Concatenation) 연산 함수가 포함된다. CIFAR10 데이터에 대한 학습 시간은 12-15 GPU-days가 걸렸고, GPU는 GTX-1080 2대를 사용하였다. 성능은 Res-Net 보다 비슷하거나 약간 우수하여 상당한 경쟁력을 가졌음을 알 수 있다. 실험 대상 데이터와 환경의 엄밀한 비교가 어렵지만, 3.2절에서 설명한 [9]보다는 우수한 성능을 나타내는 것으로 보여진다. 그림 9에 CGP 진화를 통해 얻어진 CNN 구조의 한 예가 나와 있다.

5. 접근법 특성 비교

1) GA-CNN1

CNN 층의 내부구조인 필터의 수 및 크기, 일부 층간의 연결방법, 그리고 층에 대한 활성화 함수 선택을 최적화하는 시도로서 전체 구조보다는 부분 구조 문제 중점을 주었다. 층간의 연결관계

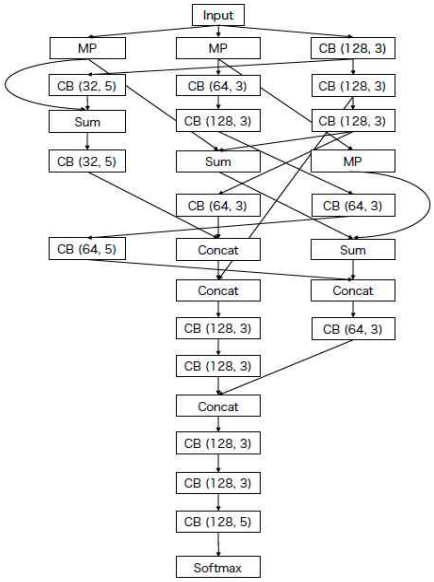

그림 9 CGP 기법으로 설계된 CNN 구조
Fig. 9 The CNN architectures designed by CGP method

를 완전 연결이나 인셉션 구조를 사용하지 않고, 이를 최적화 대상으로 삼은 것이 특이하다. 그러나 몇 개의 연결관계 후보 중에서만 선택을 하게 함으로써 최적화 변수로써의 의미가 반감된다. 게다가 필터의 수 및 크기는 앞 층에서의 수치로부터 규칙적인 추출도 가능하므로, 이의 최적화가 의미가 줄어들 수 있다. 각 층에 대한 활성화 함수의 선택은 CNN 구조보다는 관련 파라미터로 볼 수 있는데, 이 연구에서는 구조 변수외에 활성화 함수 선택을 포함시켰다. 또한 근본적인 구조인 CNN 층수는 6개로 고정시켜놓고 필터의 수 및 크기, 그리고 층간의 연결 관계만을 진화적으로 탐색했기 때문에 한계가 존재한다.

2) GA-CNN2

단순한 스트링 인코딩으로 일정 규모내이지만 자유로운 연결구조를 표현한 것이 의미가 있다. 전체 구조는 고정 필터를 사용하고 컨볼루션 층내의 구조 변화에만 초점을 맞추었고, 풀링층에 대해서는 고려가 안되었다. 또한 인셉션 모듈과 같이 다중 스케일 정보를 처리하는 데는 제한이 있다. 1)번 접근법에 비해 최적화 대상 변수가 적고 컨볼루션층 내의 연결구조에만 적용을 했다는 제한점이 존재한다.

3) CGP-CNN

1), 2)의 두 가지 GA 기반 CNN 최적화 기법들 보다 진일보한 기법으로, 컨볼루션과 풀, 그리고 Summation 과 Concatenation 모듈을 정의하여 구조의 확장이 유연하고 다양한 네트워크의 조합 생성이 가능하다는 장점이 있다. 또한 컨볼루션 함수의 경우 필터 수 및 크기를 인자화하여 다양화가 가능하다. CGP의 특성상 최대 유전자 길이내에서 가변적인 개체 구성이 가능하고, 기존 정의된 함수외에 다양한 함수가 추가될 수 있어서 확장된 표현이 가능하다.

표 1 3가지 CNN 최적화 접근법 비교
Table 1 Comparison of Three CNN Optimizations

	유전자형	구조	최적화변수
GA-CNN1	정수 스트링	층 구조 고정	필터수 및 크기, 층간 요소의 연결, 활성화 함수
GA-CNN2	이진 스트링	컨볼루션층 가변	컨볼루션층 연결구조
CGP-CNN	함수/터미널노드	모든 층 구조 가변	모든 층 연결구조 및 연산, 필터수 및 크기

6. 분석 및 결론

CNN 구조 최적화에 대한 최신의 대표적 연구 3가지 방법을 비교 분석하였다. 수많은 최적화 변수와 12-15 GPU-days 방대한 연산량 때문에 아직까지도 연구가 초기 단계에 그치고 있다. 상기 연구들도 구조의 일부 또는 상위 구조 위주로 접근한 것들이다. 4절까지의 분석을 토대로 주요 사항을 정리하면 다음과 같다. 이를 통해 목적에 맞는 진화기법의 선택과 유전자 구성등 CNN 구조 최적화를 위한 일종의 참고가 되기를 기대한다.

1) 현재까지 구조에 대한 최적화 변수가 일관화 되지 않았고, 워낙 많은 변수 가운데 일부만을 선택하여 유전자 해로 구성하였기 때문에 변수의 효용성 검토가 부족하다. CNN의 성능에 미치는 영향의 크기를 고려해서 최적화 대상 변수를 결정해야 한다.
2) 개체의 평가 때마다 전체 학습과정을 수행해야하기 때문에 개체 수와 세대수는 수십 이내로 제한하고 있다. 따라서 일반적인 generational 생성 방식보다는 적은 개체부터 점차적으로 늘려나가는 steady state 생성 방식이 타당해 보인다(4절 CGP 기반 접근 방법 참고).
3) GP 접근법은 더 크고 복잡한 공간을 탐색하기 때문에 GA 접근법보다 열려진 구조에 대한 탐색이 가능하기 때문에 구조 최적화에 대한 가능성이 높으나, 탐색의 어려움이 훨씬 크기 때문에 진화 과정에 의해서 좌우된다. 따라서 최적화 대상 변수의 특성을 고려하여 이에 적합한 진화연산 기법의 선택이 필요하다.
4) 트리를 유전자로 사용하는 일반적인 GP보다 네트워크 표현 방식의 CGP가 CNN의 네트워크 구조 진화에 적합한 것으로 생각된다.
5) 연산량을 최소로 하면서 개체를 평가할 수 있는 기법이 필요하다. 예로, 개체 당 전체 학습과정을 수행하지 않고, 초기 학습과정을 통해 성능의 개선 추이를 평가함수로 삼아서 빠르게 해를 진화시킨 후, 후기 세대에서 전체 학습과정을 수행하는 것이 전체 연산량을 줄일 수 있는 대안이다.

감사의 글

본 연구는 2016학년도 서경대학교 교내연구비 지원에 의하여 이루어졌음.

References

[1] J. Schmidhuber, "Deep Learning in Neural Networks: An Overview," Neural Networks, Vol. 61, pp. 85-117, 2015.
[2] Y. LeCun, Y. Bengio, G. Hinton, "Deep learning," Nature Vol. 521, pp. 436-444, 2015.
[3] J. D. Goldberg, *Genetic Algorithms in Search, Optimition and Machine Learning*, Addison-Wesley, Reading, MA, 1989.
[4] K. O. Stanley and R. Miikkulainen, "Competitive coevolution through evolutionary complexification," Journal of Artificial Intelligence Research, vol. 21, pp. 63-100, 2004.
[5] K. O. Stanley, D. B. D'Ambrosio, and J. Gauci, "A hypercube-based indirect encoding for evolving large scale neural networks," Artificial Life, vol. 15, 2009
[6] K. O. Stanley, "Compositional pattern producing networks: A novel abstraction of development," Genetic Programming and Evolvable Machines Special Issue on Dev. Sys., vol. 8, no. 2, pp. 131-162, 2007.
[7] C. Fernando et al. "Convolution by Evolution: Differentiable Pattern Producing Networks," In Proceedings of the 2016 Genetic and Evolutionary Computation Conference, Denver, CO, USA, pp. 109-116. 2016.
[8] A. Rikhtegar, M. Pooyan, M. Manzuri-Shalmani, "Genetic algorithm-optimised structure of convolutional neural network for face recognition applications," IET Computer Vision, Vol. 10, Iss. 6, pp. 559-566, 2016
[9] L. Xie, A. Yuille, "Genetic CNN," CVPR 2017
[10] M. Suganuma, s, Shirakawa, T. Nagao, "A Genetic Programming Approach to Designing Convolutional Neural Network Architectures," Proceedings of GECCO 2017, pp. 497-504, 2017
[11] LeCun, Yann, et al. "Gradient based learning applied to document recognition," Proceedings of the IEEE, pp. 2278-2324, 1998
[12] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, The MIT Press, 1992.
[13] J. Miller, P. Thomson, "Cartesian Genetic Programming," EuroGP 2000. LNCS, vol. 1802, pp. 121-132. Springer, 2000
[14] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," International Conference on Learning Representations, 2014.
[15] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going Deeper with Convolutions," Computer Vision and Pattern Recognition, 2015
[16] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," Computer Vision and Pattern Recognition, 2016.
[17] S. Zagoruyko and N. Komodakis, "Wide Residual Networks," arXiv: 1605.07146, 2016.
[18] L. Xie, J. Wang, W. Lin, B. Zhang, and Q. Tian, "Towards Reversal-Invariant Image Representation," International Journal on Computer Vision, 2016.
[18] L. Xie, J. Wang, W. Lin, B. Zhang, and Q. Tian, "Towards Reversal-Invariant Image Representation", International Journal on Computer Vision, 2016.
[19] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," International Conference on Learning Representations, 2014.
[20] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition,"Computer Vision and Pattern Recognition, 2016.
[21] G. Huang, Z. Liu, and K. Weinberger, "Densely Connected Convolutional Networks," arXiv: 1608.06993, 2016.

저 자 소 개

서 기 성 (Kisung Seo)

1993년 연세대학교 대학원 전기공학과 졸업(박사). 1999~2003년 Michigan State University, Genetic Algorithms Research and Applications Group, Research Associate, 2002~2003년 Michigan State University, Electrical & Computer Engineering, Visiting Assistant Professor, 2011~2012년 Michigan State University, BEACON (Bio/ computational Evolution in Action CONsortium) Center, Visiting Scholar, 1993년~현재 서경대학교 전자공학과 교수. 관심분야는 진화연산, 딥러닝, 머신비전, 기상예측, 지능로봇.