Journal of The Institute of Electronics and Information Engineers Vol.59, NO.11, November 2022 https://doi.org/10.5573/ieie.2022.59.11.39
ISSN 2287-5026(Print) / ISSN 2288-159X(Online)

논문 2022-59-11-5

다중목적 보상함수가 개선된 강화학습 기반 NAS

(NAS based on Reinforcement Learning with Improved Multi-objective Reward Function)

임 철 순*, 김 명 선**

(Cheolsun Lim and Myungsun Kim©)

요 약

DNN의 효용성이 넓은 범위의 응용에서 검증되면서 모바일과 같은 임베디드 환경에서 사용할 수 있도록 DNN을 경량화 하는 연구도 많이 이루어지고 있다. 하지만 사람이 직접 DNN구조를 설계하는 데에는 많은 노력과 시간이 소요되므로 이를 해결하기 위해 NAS(Neural Architecture Search)가 등장했다. NAS는 DNN 구조를 자동으로 탐색하며 여기서 탐색되는 DNN들은 현존하는 SOTA(State-of-the-Art) DNN들과 비교해도 뒤쳐지지 않는 성능을 보여주고 있다. 대표적인 모바일용 NAS인 MnasNet은 높은 성능과 낮은 실행시간을 만족하는 DNN을 탐색할 수 있다. MnasNet은 추론 실행시간 제약조건을 만족하면서 정확도는 가장 높은 DNN을 탐색하는 것이 목표이다. 본 논문에서는 MnasNet을 기반으로 정확도와 실행시간에 대한 최적화를 하는 다중목적 보상함수에 집중한다. 먼저 MnasNet에서 사용한 보상함수를 분석하고 주어진 제한시간보다 짧은 추론 실행시간을 가지면서 정확도가 최대가 되는 DNN을 탐색할 수 있도록 새로운 보상함수를 제안한다. 본 논문에서 제안된 보상함수를 사용하였을 때 NAS가 생성하는 DNN 모델들이 추론 실행시간 제약조건을 만족하는 시점이 MnasNet의 보상함수를 사용했을 때에 비해 15%빨라졌다.

Abstract

As the utility of DNNs is verified in a wide range of applications, many research works have been conducted to lighten DNN models for the use in embedded environments such as mobiles. However, since it takes lots of effort and time to design a light-weight DNN structure directly, NAS (Neural Architecture Search) has been introduced to figure out this problem. NAS automatically explores the DNN architectures, where the explored DNNs have the execution performances that are not inferior to the existing state-of-the-art (SOTA) DNNs. MnasNet, a typical mobile NAS, can explore DNNs that satisfy high performance and low execution time. The goal of MnasNet is to explore DNNs with the highest accuracy while satisfying the inference execution time constraints. In this paper, we focus on a multi-objective reward function that optimizes for the accuracy and the inference time based on MnasNet. First, we analyze the reward function used by MnasNet and propose a new reward function to explore DNNs with the maximum accuracy while having a reduced inference execution time than a given time limit. When we use the reward function proposed in this paper, the point at which the DNN models generated by NAS satisfy the inference execution time constraint is 15% faster than when using the reward function of MnasNet.

Keywords : Neural architecture search, DNN architecture, Mobile, Reinforcement learning, Embedded system

*학생회원, 한성대학교 IT융합공학과(Department of IT Convergence Engineering, Hansung University)
**정회원, 한성대학교 AI응용학과(Department of Applied Artificial Intelligence, Hansung University)
© Corresponding Author(E-mail : kmsjames@hansung.ac.kr)
※ 본 연구는 한성대학교 교내학술연구비 지원과제임.
Received ; July 9, 2022 Revised ; July 20, 2022 Accepted ; August 15, 2022

Ⅰ. 서 론

DNN(Deep Neural Network)의 발전으로 몇 해 동안 높은 성능을 내는 DNN들이 많이 등장하였다. Resnet[1]은 잔차학습 구조를 이용하여, Densenet[2]은 Dense 블록을 이용하여 망이 깊어지더라도 이전의 정보를 유지할 수 있도록 디자인되었다. 높은 성능만을 추구하는 DNN들과 달리 서버 대비 적은 연산 처리량을 가진 모바일과 같은 임베디드 환경을 위한 DNN들도 등장하기 시작하였다. 이러한 DNN들의 특징은 모바일 환경에서도 원활하게 실행할 수 있도록 연산량을 줄이는 구조로 디자인된다. MobileNet[3]은 연산량이 큰 일반 합성곱 대신 depthwise-seperable 합성곱을 통하여 연산량을 줄였고, Shufflenet[4]은 1x1 그룹 합성곱을 사용하여 MobileNet에서 사용한 1x1 합성곱 숫자를 더욱 줄여 전체적인 연산량을 줄였다.

앞서 언급한 DNN들의 특징은 사람이 직접 특정 목적을 가지고 디자인한 구조라는 것이다. 새로운 DNN을 만들기 위해서는 많은 시간과 노력이 필요하다. 그래서 DNN을 이용하여 DNN을 찾기 위한 NAS(Neural Architecture Search)라는 개념이 등장하였다. 이 개념이 처음 등장한 NAS[5]에서는 컨트롤러라는 RNN 모델을 강화학습을 이용하여 학습한다. 컨트롤러는 하나의 DNN을 선택하고 이 DNN을 CIFAR10[6], ImageNet[7]과 같은 데이터셋으로 학습시켜 정확도를 계산하고 이 정확도를 강화학습의 보상으로 사용하여 컨트롤러가 학습된다. 학습이 진행되면서 컨트롤러는 정확도가 높은 DNN을 생성하게 된다. 그림 1의 (a)는 [5]의 컨트롤러가 레이어 단위로 DNN을 선택 및 출력하는 NAS를 나타낸다. NAS의 컨트롤러는 학습이 진행되는 동안 (a)의 각 OP(Operation)에 들어갈 연산자를 선택하게 된다. 이 OP에 들어갈 수 있는 연산자들을 미리 정의한 목록을 탐색공간이라 한다. 탐색공간은 사용자에 의해 자유롭게 정의될 수 있다.

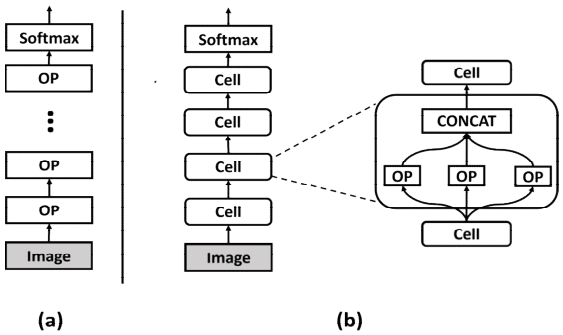

그림 1. 레이어 기반 NAS의 출력 DNN(a)과 셀 기반 NAS의 출력 DNN(b)

Fig. 1. Output DNN of a layer-based NAS (a) and that of cell-based NAS (b).

그림 1의 (a)와 같은 구조는 다양한 구조를 탐색할 수 있지만 탐색공간의 크기가 지나치게 커질 수 있다는 단점이 있다. 탐색공간의 크기를 줄이고 보다 일반화된 DNN을 얻기 위해 NASNet[8]에서는 그림 1의 (b)와 같은 셀 기반 구조를 이용하여 탐색한다. NASNet의 셀 기반 구조에서 컨트롤러는 하나의 셀 구조를 선택한다. 셀에 있는 각 OP는 컨트롤러에 의해 선택되며, 이 셀을 여러 번 반복하여 쌓는 형태로 DNN을 완성한다. 각 셀들은 같은 구조를 가지며 하나의 셀을 몇 개 배치하느냐에 따라서 여러 DNN을 얻을 수 있다. 이 구조는 탐색공간을 줄이는 것뿐만 아니라 확장성에서도 장점이 있다. NASNet에서는 실제 CIFAR10을 통해 찾은 최적의 셀을 몇 개 더 쌓는 형태로, 더 큰 이미지인 ImageNet에서도 높은 정확도를 갖는 DNN을 탐색할 수 있었다.

MobileNet, ShuffleNet과 같은 모바일 환경에 적합한 DNN을 찾기 위해 MnasNet[9]에서는 NAS를 적용하고 정확도와 추론 실행시간을 이용하여 컨트롤러를 학습한다. MobileNet과 ShuffleNet은 모바일 환경의 상대적으로 낮은 성능의 연산장치를 고려하여 MAC (multiply-accumulate) 연산량을 줄이기 위해 노력했다. 이와 달리 MnasNet에서는 DNN이 실행될 환경에서 직접 추론 실행시간을 측정하여 이를 일정 기준 이하로 줄이는 것에 초점을 둔다. 그리고 측정된 추론 실행시간과 정확도를 이용하여 보상을 계산하고 계산된 보상으로 컨트롤러를 학습시킨다. MnasNet에서는 정확도와 추론 실행시간에 대한 일정한 관계를 정한다.

본 논문에서는 MnasNet을 기반으로 서버보다 상대적으로 연산 속도가 느린 모바일, 임베디드 환경을 위한 NAS에서 정확도와 실행시간으로 보상을 계산하는 새로운 보상함수를 제안한다. MnasNet의 보상함수는 정확도와 추론 실행시간에 대한 일정한 관계를 갖도록 정의되어있어 추론 실행시간이 제한시간을 넘어가는 DNN에 보상을 크게 감소시키지 않는다. 또 추론 실행시간이 짧아질 수록 보상은 크게 증가하여 정확도가 더 작더라도 추론 실행시간이 짧은 DNN에 더 큰 보상을 주는 경우가 생긴다. 따라서 본 논문에서는 추론실행시간에 대한 제한시간이 있을 때 정확도가 최대가 되는 DNN을 찾기 위한 새로운 보상함수를 제안한다. 그리고 추론 실행시간이 제한시간보다 큰 DNN에 대한 탐색을

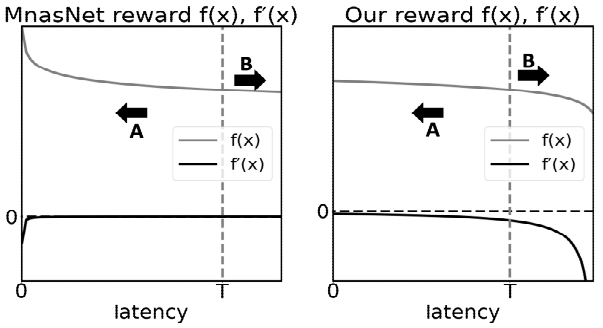

그림 2. 추론 실행시간 제약조건 T가 주어질 때 보상함수 f(x)와 이의 미분된 함수 f'(x)

Fig. 2. Given the inference execution time constraint T, the reward function f(x) and its differentiated function f'(x).

줄여 컨트롤러가 더 빠르게 제한시간보다 작은 추론 실행시간을 가지는 DNN을 탐색할 수 있게 한다.

Ⅱ. MnasNet 및 다중목적 보상함수 제안

MnasNet은 NAS를 이용하여 모바일 환경에 맞는 DNN구조를 탐색하였다. 보통 DNN을 얼마나 경량화

$$\begin{aligned} &\underset{m}{\text{maximize}} && ACC(m) \\ &\text{subject to} && LAT(m) \le T \end{aligned} \tag{1}$$

$$\underset{m}{\text{maximize}} \quad ACC(m) \times \left[\frac{LAT(m)}{T}\right]^{w} \tag{2}$$

$$w = \begin{cases} \alpha, & \text{if } LAT(m) \le T \\ \beta, & \text{otherwise} \end{cases} \tag{3}$$

했는지에 대한 지표로 MAC 또는 FLOPS를 사용하는데 MnasNet의 컨트롤러는 생성하는 DNN을 모바일환경에서 직접 추론 실행시간을 측정하여 일정 기준보다 작은 추론 실행시간을 가지는 DNN 구조를 생성하도록 학습된다.

식 (1)에서 m은 하나의 DNN이며 ACC는 정확도, LAT는 1번 추론에 걸리는 실행시간, T는 실행시간에 대한 제한 조건이다. 식 (1)은 정확도는 최대이며 지연시간은 T보다 작은 DNN을 찾는 것을 목표로 한다. 이를 위해 MnasNet에서는 DNN의 정확도와 실행시간의 파레토 최적을[10] 위한 보상함수를 식 (2)와 같이 정의하였다. 식 (2)의 w는 실행시간에 대한 일종의 가중치이며 이 값을 조절하여 실행시간을 어느 정도의 비율로 고려할 것인지 정한다. w의 값은 식 (3)에 의해 결정되며 DNN의 추론 실행시간이 T보다 작을 때와 클 때 다른 값으로 설정함으로써 T를 기준으로 보상의 정도를

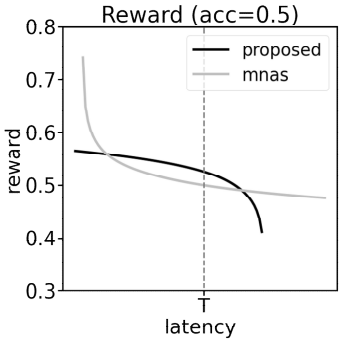

그림 3. 보상 비교

Fig. 3. Comparison of the reward function.

조절할 수 있다. MnasNet의 보상함수는 정확도와 실행시간의 관계를 일정하게 유지하는 것에 초점을 둔다. DNN의 실행시간이 두 배 상승하면 정확도도 현재보다 5% 상승해야하는 관계를 유지한다. 이 관계를 유지하는 w의 값은 α, β 모두 -0.07이며 MnasNet에서는 w를 이 값으로 설정하여 DNN을 탐색한다. 그림 2의 왼쪽은 MnasNet의 보상함수와 이를 미분한 함수를 표현하며, x축은 실행시간 y축은 ACC가 50%일 때의 보상값이다. T의 값은 17ms이며 이후 특별한 언급이 없는 한 T는 17ms로 고정한다. MnasNet의 보상함수는 DNN의 추론 실행시간이 줄어드는 A방향으로 갈수록 보상이 커지는 정도가 증가한다. 반대로 DNN의 추론 실행시간이 커지는 B방향으로 갈수록 보상이 줄어드는 정도는 감소한다. 이는 같은 정확도를 가지는 두 DNN의 추론 실행시간이 20ms, 10ms일 때의 보상값의 차이와 15ms, 5ms 일 때의 보상값의 차이가 더 크다. 실행시간이 T를 넘는 DNN에 대한 보상의 감소량이 크지 않기 때문에 MnasNet은 추론 실행시간이 T를 넘는 DNN도 많이 생성한다.

$$latency(m) = \begin{cases} \log_T(-LAT(m) + 1.5T), \\ \quad \text{if } LAT(m) \le 1.5T - 1 \\ 0, \ otherwise \end{cases} \tag{4}$$

$$\underset{m}{\text{maximize}} \quad x \times ACC(m) + y \times latency(m) \tag{5}$$

우리는 MnasNet과 같이 MAC, FLOPS 대신 추론 실행시간을 이용하여 정해진 T보다 낮은 추론 실행시간을 가지면서 정확도가 가장 높은 DNN 구조를 빠르게 탐색할 수 있도록 새로운 다중목적(Multi-objective) 보상함수를 제안한다. 제안하는 보상함수는 식 (4)와 식 (5)와 같이 정의한다. 식 (4)는 추론 실행시간에 대한 보상을 정의한 식이며 이때 추론 실행시간이 조건보다 크면 0으로 고정한다. 식 (5)는 정확도와 추론 실행시간의 파

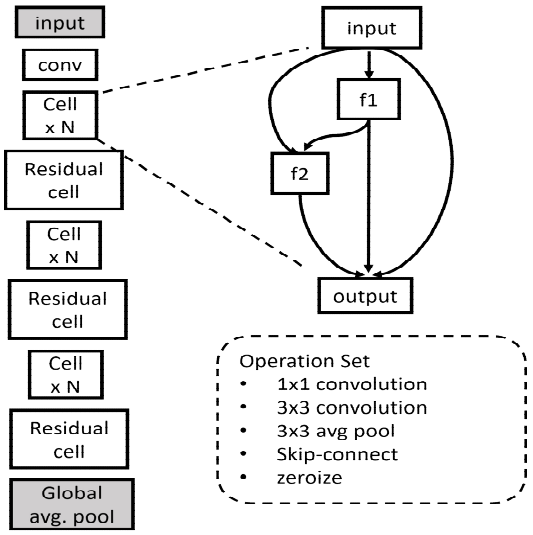

그림 4. NAS-Bench-201 탐색공간과 DNN 구조[11]
Fig. 4. DNN architecture of NAS-Bench-201[11].

레토 최적을 찾기 위한 가중치 합 형태의 보상함수이다. x는 정확도에 대한 비율, y는 실행시간에 대한 비율을 조절할 수 있는 변수이며 설정할 때에는 x+y = 1을 만족하도록 설정한다[10]. 본 논문에서 x는 0.9, y는 0.1로 설정한다. 그림 2의 오른쪽은 제안된 보상함수와 이를 미분한 함수를 표현한다. 그림에서 DNN의 실행시간이 A 방향으로 갈수록 보상이 커지는 정도가 감소하며 반대로 B 방향으로 갈수록 보상이 작아지는 정도가 증가한다. 따라서 MnasNet과는 반대로 같은 정확도를 가지는 두 DNN의 추론 실행시간이 20ms, 10ms일 때의 보상값의 차이와 15ms, 5ms 일 때의 보상값의 차이가 더 작다. 즉, 더 작은 추론 실행시간을 가지는 DNN에 대한 보상을 크게 증가시키지 않는다는 것이다. 제한시간 T는 직접 정하는 값이므로 만약 30fps의 처리량이 필요하다면 33ms 로 설정 할 것이며 추론 실행시간이 T만 넘지 않으면 정확도가 더 높은 DNN을 선택하는 것이 좋다. 만약 DNN 1의 정확도가 60%이고 지연시간이 8ms, DNN 2의 정확도가 63%이고 지연시간이 16ms일 때 MnasNet의 보상함수로 계산한 보상값은 DNN 1과 DNN 2가 비슷하다. 하지만 우리의 보상함수로 계산한 보상은 두 DNN 모두 지연시간은 T보다 낮으므로 그 중 정확도가 더 높은 DNN 2에 더 큰 보상을 줄 수 있도록 보상함수를 설정하였다.

그림 3은 DNN정확도가 50%일 때 추론 실행시간에 따른 MnasNet과 제안된 보상함수를 비교한 그림이다. 추론 실행시간이 T보다 작을 경우에는 MnasNet의 보상함수는 급격히 보상이 증가하며 제안된 보상함수는 완만하게 증가시켜 보상에 대한 추론 실행시간의 영향력을 조절한다. 반대로 T보다 큰 경우에는 추론 실행시간이 커질수록 보상값을 크게 줄여서 T보다 작은 DNN을 탐색하도록 유도한다.

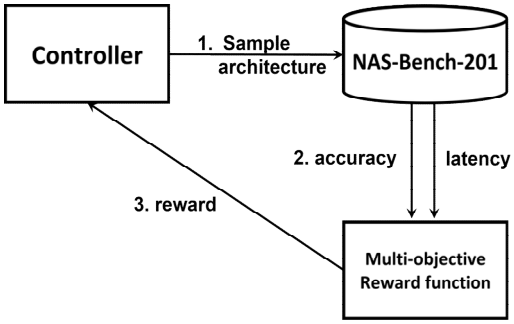

그림 5. 제안된 다중목적 보상함수를 적용한 강화학습 기반 NAS
Fig. 5. NAS based on reinforcement learning with the proposed multi-objective reward function.

Ⅲ. NAS-Bench-201 및 NAS 제안

1. NAS-Bench-201

NAS를 이용하여 DNN 구조를 탐색하는 시간에서 가장 많은 비율을 차지하는 것이 컨트롤러가 생성한 DNN의 정확도를 평가하기 위해 생성된 DNN을 학습하는 시간이다. 심지어 CIFAR-10 데이터셋 대신 더 큰 크기를 가진 ImageNet 데이터셋을 사용하게 되면 그 시간은 더 크게 증가할 것이다. 실제로 처음 등장한 NAS[5]에서는 800개의 GPU로 28일 동안 탐색하였고 NASNet에서는 500개의 GPU로 4일 동안 탐색하였다. MnasNet은 ImageNet에 대해 5 epoch씩만 학습하여 정확도를 평가하였음에도 불구하고 64개의 TPU를 사용하여 4.5일 동안 탐색하였다.

NAS-Bench-201[11]는 NAS를 위한 DNN의 구조 및 정확도와 같은 데이터를 모아놓은 데이터셋이다. 이 데이터셋은 입력으로 DNN이 들어오면 출력으로 이 DNN에 대한 정확도와 추론 실행시간, 용량, FLOPS가 나온다. 결과들은 미리 측정하여 저장해놓은 값이므로 이 데이터셋을 이용하면 컨트롤러가 생성한 DNN의 정확도를 얻기 위해 따로 학습할 필요 없이 바로 정확도를 얻을 수 있다. NAS-Bench-201은 셀 구조 기반의 DNN을 입력으로 사용하며 그림 4의 왼쪽은 NAS-Bench-201의 DNN 구조를 나타낸다. 여러 개의 셀을 반복적으로 배치한 구조이며 각 셀은 그림 4의 오른쪽과 같이 구성된다. 각 화살표는 하나의 연산이며 선택

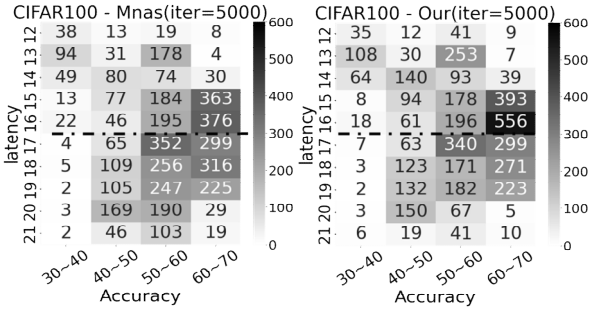

그림 6. 컨트롤러가 생성하는 DNN의 정확도와 실행시간의 히트맵 분석

Fig. 6. Heat map analysis of the accuracy and the execution time of the DNNs generated by the controller.

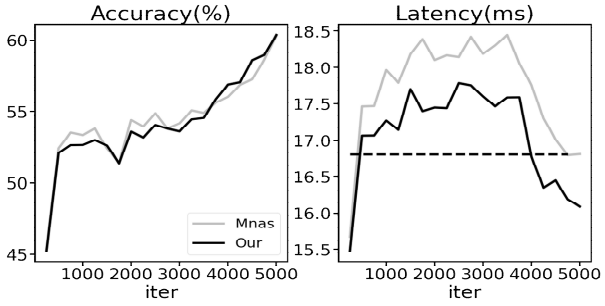

그림 7. 컨트롤러 학습 과정에서 생성되는 DNN들의 정확도 및 지연시간

Fig. 7. Accuracy and execution time profiles of DNNs produced during the learning phase of the controller.

될 수 있는 연산은 아래의 Operation Set에 정의되어 있고 이 중 하나의 연산이 화살표에 배치된다. f1, f2는 중간 피쳐맵이며 여러 개의 화살표가 향하는 곳에서는 concat으로 피쳐맵을 합친다. Residual Cell은 같은 셀 구조에 stride만 2로 설정하여 피쳐맵의 크기를 반으로 줄이는 역할을 한다. Cell을 몇 번 반복하는지에 대한 변수 N은 5로 설정되어 있다. NAS-Bench-201의 탐색 공간에서 나올 수 있는 DNN의 경우의 수는 15625가지이며 NAS-Bench-201은 각 DNN에 대한 정확도, 실행시간, 용량, FLOPS를 3가지 이미지 데이터셋(CIFAR-10, CIFAR-100[6], ImageNet-16-120[12])으로 학습시킨 결과를 포함한다.

2. 제안된 다중목적 보상함수를 적용한 강화학습 기반 NAS

그림 5는 본 연구에서 제안된 다중목적 보상함수를 기반으로 하는 NAS를 나타낸다. 이는 컨트롤러, NAS-Bench-201, 보상함수계산으로 이루어져 있다. 컨트롤러는 RNN으로 구성되어있으며 컨트롤러는 DNN 을 NAS-Bench-201의 구조에 맞게 생성한다. 그리고 생성된 DNN을 NAS-Bench-201의 입력으로 사용하여 출력으로 정확도와 추론 실행시간을 얻는다. 그리고 받아온 정확도와 추론 실행시간을 통해 보상함수로 보상값을 계산한다. 계산된 보상값은 컨트롤러를 학습하는데 쓰이며 이때 컨트롤러는 Policy Gradient(PG)[13]를 통해 학습한다. 컨트롤러는 정확도가 최대가 되면서 추론 실행시간이 T보다 작은 DNN을 생성하도록 학습된다.

Ⅳ. 실 험

1. 실험환경

본 실험에서는 MnasNet에서 사용한 보상함수와 본 논문에서 제안된 보상함수를 이용하여 신경망 탐색을 하였을 때 컨트롤러가 생성하는 DNN의 정확도 및 추론 실행시간의 차이를 비교한다. Mnasnet 보상함수의 w값은 -0.07로 사용한다. 컨트롤러는 DNN을 총 5000번 생성할 때까지 학습되며 T=17ms로 고정한다. 정확도와 추론 실행시간 및 DNN의 탐색공간은 NAS-Bench-201의 CIFAR-100로 평가한 데이터를 사용한다.

2. 탐색한 DNN들의 결과

그림 6은 컨트롤러를 5000번 학습하였을 때 생성되는 DNN들의 정확도와 실행시간의 히트맵을 나타낸다. 히트맵에서 세로축의 숫자는 소수점을 버린 값들을 나타낸다. 즉, 16은 16.00 ~ 16.99 사이의 값을 의미한다. 그림 6의 검은 점선은 17ms를 기준으로 나눈 것이다. 그림을 보았을 때 MnasNet은 17ms를 기준으로 넓은 범위를 탐색한다. 하지만 우리의 보상함수를 사용하였을 때에는 17ms보다 작은 실행시간을 가지는 DNN을 더 많이 생성한다. 실행시간이 작은 것만이 아닌 정확도도 높은 DNN을 탐색하는데 17ms 이하이면서 정확도가 50~60%인 DNN은 MnasNet의 보상함수는 659개, 제안된 보상함수는 780개이며 정확도가 60~70%인 DNN은 MnasNet의 보상함수에서 781개, 제안된 보상함수에서는 1004개가 생성되었다. 이를 통해 제안된 보상함수가 17ms보다 작은 모델에만 집중하는 것이 아니라 정확도도 같이 최대가 되는 DNN을 찾

표 1. 각 DNN의 정확도, 실행시간에 따른 각 보상함수의 보상값 비교

Table 1. Comparison of reward values of each reward function according to the accuracy and the execution time of each DNN.

(a)	DNN 1 (80%, 8ms)	DNN 2 (84%, 16ms)
MnasNet	0.84334	0.84352
Proposed	0.82102	0.83546
(b)	DNN 1 (80%, 12ms)	DNN 2 (84%, 24ms)
MnasNet	0.81974	0.81996
Proposed	0.81186	0.77031
(c)	DNN 1 (80%, 8ms)	DNN 2 (83%, 16ms)
MnasNet	0.843344	0.833527
Proposed	0.82102	0.826460

도록 컨트롤러를 학습시킨다는 것을 알 수 있다.

그림 7은 총 5000번의 반복 동안 컨트롤러를 학습하였을 때 컨트롤러에서 생성되는 DNN의 정확도 및 실행시간을 나타내는 그래프이다. 250번씩 20구간으로 나누어 각각 정확도와 실행시간으로 평균을 낸 그래프이다. 그림에서 볼 수 있듯이 두 개의 보상함수 모두 정확도는 상승한다. 하지만 생성되는 DNN의 평균 실행시간은 우리의 보상함수를 사용하였을 때 더 빠르게 도달하는데, 우리의 보상함수를 사용하였을 때 4000번째부터 실행시간의 평균이 17ms이하로 떨어졌으며 MnasNet의 보상함수는 4750번째 에 17ms이하로 떨어졌다. 즉, 본 연구에서 제안한 다중목적 보상함수 기반의 NAS는 추론 실행시간을 T이하로 제한을 두는 조건에서 기존보다 750번 덜 동작해도 T보다 작은 추론 실행시간을 가지는 DNN위주로 탐색한다. 따라서 더 빠르게 원하는 DNN 모델을 찾게 해줘서 전체적인 NAS의 동작 속도를 개선한다. 이는 제안된 보상함수가 T보다 큰 추론 실행시간을 가지는 DNN에 대한 보상을 MnasNet의 보상함수보다 크게 낮추어 컨트롤러가 T보다 작은 추론 실행시간을 가지는 DNN을 탐색하도록 더 빠르게 학습되기 때문이다.

표 1은 두 개의 DNN의 조건에 따른 각 보상함수의 결과값을 나타낸 것이다. (a), (b), (c)는 각 비교군이며 괄호 안의 값은 (정확도, 실행시간)의 순서이다. (a)와 (b)의 DNN들은 MnasNet의 5% 정확도 상승 시 실행시간 두 배가 되는 관계를 가지는 DNN들이며 (a)의 DNN 2는 추론 실행시간이 T보다 작은 경우, (b)의 DNN 2는 추론 실행시간이 T보다 큰 경우이다. (a)와 (b)의 두 DNN은 MnasNet의 5% 정확도 상승 시 추론 실행시간이 두 배가 되는 관계를 가지므로 DNN 1과 DNN 2의 정확도와 추론 실행시간을 이용하여 계산한 MnasNet 보상함수의 결과값은 (a), (b)모두 비슷한 값이 나온다. 하지만 제안된 보상함수를 이용하여 계산하면 (a)에서는 DNN 1보다 DNN 2가 큰 값이고, (b)에서는 DNN 1보다 DNN 2가 작은 값을 가진다. (a)에서는 DNN 1과 DNN 2의 추론 실행시간이 모두 T 이하이므로 정확도가 더 높은 DNN 2에 큰 보상을 주고, (b)에서는 DNN 2의 추론 실행시간이 T보다 크기 때문에 더 작은 보상을 준다. (c)는 DNN 1과 DNN 2의 실행시간은 두 배 차이나지만 정확도는 5% 이하로 증가한 경우이다. MnasNet의 보상함수로 계산한 결과는 DNN 1이 크고 우리의 보상함수로 계산한 결과는 DNN 2가 크다. DNN 2가 T보다 추론 실행시간이 짧고 정확도는 DNN 1 보다 크기 때문에 우리는 DNN 2를 선택하는 것이 더 바람직하며 DNN 2에 더 큰 보상을 주는 것을 확인할 수 있다.

V. 결 론

본 논문에서는 모바일과 같은 환경에서 추론실행시간이 제한될 때 정확도가 가장 큰 DNN을 찾기 위한 새로운 보상함수를 제안하고 이를 NAS에 적용하여 실험하였다. 이를 위해 모바일 환경을 위한 MnasNet의 접근 방법에 영향을 받아 새로운 형태의 보상함수를 만들었고, 빠른 실험을 위해 Nas-Bench-201을 이용하여 MnasNet과 제안된 보상함수의 효과를 실험했다. 우리의 목표는 추론 실행시간에 대한 제한 조건이 주어졌을 때 그 중 가장 높은 정확도를 갖는 DNN을 탐색하는 것이었다. 우리의 보상함수는 제한 조건보다 큰 실행시간일 때에는 보상을 서서히 크게 감소시키고, 제한 조건보다 작은 실행시간일 때에는 보상의 증가량을 줄여서 실행시간이 짧아질 때에 얻는 보상의 양이 크지 않도록 조절하였다. 이를 통해 DNN들의 비교에서 우리가 제시한 조건인 제한시간 T보다 작은 추론 실행시간을 가지면서 정확도가 더 큰 DNN에 대한 보상값이 더 크게 나타나는 것을 확인할 수 있었다. 또 이를 통해 실행시간이 제한 조건보다 큰 DNN에 대한 탐색의 수를 줄여서 컨트롤러가 제한조건보다 낮은 실행시간 위주로 탐색하게 되어 더 빠르게 조건에 맞는 DNN을 탐색하도록 컨트롤러를 학습할 수 있었다.

REFERENCES

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition", in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
[2] G. Huang, Z. Liu, L. v. d. Maaten, and K. Q. Weinberger, "Densely Connected Convolutional networks", in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.
[3] A. G. Howard, M. Zhu, B. chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications", arXiv preprint arXiv:1704.04861, 2017
[4] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices", in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.
[5] B. Zoph, and Q. V. Le, "Neural Architecture Search with Reinforcement Learning", in ICLR, 2017.
[6] A. Krizhevsky, and G. Hinton "Learning multiple layers of features from tiny imagenet", Technical report, 2009.
[7] J. Deng, W. Dong, R. Socher, L. j. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database" In CVPR, 2009.
[8] B. Zhoh, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning Transferable Architectures for Scalable Image Recognition", in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.
[9] M. Tan, B. Chen, R. Pang, V. Vasudevan, M. Sandler, A. Howard, and Q. V. Le, "MnasNet: Platform-Aware Neural Architecture Search for Mobile", in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.
[10] K. Deb, "Multi-objective Optimization", Search methodologies, pages 403-449, 2014.
[11] X. Dong, and Y. Yang, "NAS-Bench-201: Extending the Scope of Reproducible Neural Architecture Search", in ICLR, 2020.
[12] P. Chrabaszcz, I. Loshchilov, and F. Hutter, "A downsampled variant of imagenet as analternative to the cifar datasets", arXiv preprint arXiv:1707.08819, 2017.
[13] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning", In Machine Learning, 1992.

저 자 소 개

임 철 순(학생회원)
2021년 한성대학교 IT응용시스템 공학과 학사 졸업.
2021년~현재 한성대학교 IT융합공학과 석사과정.
<주관심분야: GPU Programing, DNN framework, Deep Learning, DNN 최적화>

김 명 선(정회원)
2000년~2002년 LG전자 액세스네트워크 연구소 주임연구원
2002년~2011년 삼성전자 DMC 연구소 책임연구원
2016년 서울대학교 전기컴퓨터공학부 박사 졸업.
2016년~2019년 삼성전자 SR연구소 수석연구원
2019~현재 한성대학교 IT융합공학부 조교수
<주관심분야: 인공지능 시스템, Linux kernel, HW/SW Co-design, NPU 등>