자원제약 내장형 시스템을 위한 동적 뉴럴 네트워크 레이어 생략 기법

허성진, 이다솜, 정종훈, 양회석
아주대학교
{heosungjin23, ekthadl40, kkjjh223, hyang}@ajou.ac.kr

Runtime Neural Network Layer Skipping Technique for Resource-Constrained Embedded Systems

Sungjin Heo, Dasom Lee, Jonghun Jeong, Hoeseok Yang
Dept. of ECE, Ajou University.

요 약

기존 자원제약 내장형 시스템에서 뉴럴 네트워크 응용프로그램의 메모리 사용량, 연산량이 고정적이기 때문에 필요에 따라 메모리 사용량, 연산량을 변경하기 힘들다. 이와 같은 문제를 해결하기 위해 본 논문에서는 다양한 상황에 따라 가변적으로 특정 뉴럴 네트워크의 layer를 뛰어넘거나 모든 layer를 사용하여 뉴럴 네트워크를 동작시키는 방식을 제안한다. 본 논문에서는 nnom 라이브러리를 사용하여 제안하는 네트워크 모델을 ARM Cortex-M 시리즈 보드에서 동작시킬 수 있도록 변환한다. 그 결과, layer를 뛰어넘은 경우에 모든 layer를 사용한 경우보다 정확성이 0.25% 떨어지지만 38.8%의 연산량을 감소시켰다. 이를 통해 제안하는 최적화 기법이 정확성과 연산량의 trade-off를 조절하여 자원제약 내장형 시스템에서 동적으로 적용될 수 있음을 검증하였다.

Ⅰ. 서 론

최근 다양한 사물인터넷(Internet of Things) 장치의 확산에 따라 뉴럴 네트워크 응용 프로그램을 저사양 디바이스에서 구동하려는 시도가 증가하고 있다. 하지만 내장형 시스템의 연산 속도와 메모리의 제한은 이러한 시도들에 큰 걸림돌이 되고 있다. 이를 해결하기 위해 뉴럴 네트워크 경량화 기법, 연산 최적화 기법이 다양하게 연구되고 있다. 이러한 최적화 기법들을 활용하여 저사양 내장형 시스템에서 뉴럴 네트워크 응용 프로그램을 동작시킬 수 있다. 하지만 기존 기법들은 항상 자원 사용량이 고정되어있기 때문에 디바이스에 기존보다 태스크가 동적으로 추가 할당되는 경우, 실시간성을 만족시키지 못할 수 있다.

따라서 본 논문에서는 그림1과 같이 뉴럴 네트워크 layer의 깊이를 유동적으로 조절할 수 있는 뉴럴 네트워크 응용에서, 동적으로 연산 요구량에 변화가 생기더라도 연성 실시간성을 확보하는 추론 엔진을 개발하는 것을 목표로 한다.

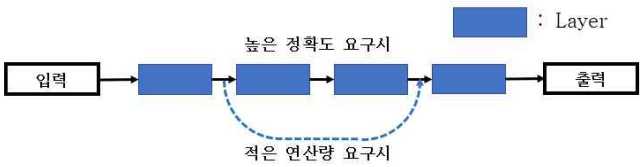


그림 1. 제안하는 유동적 뉴럴 네트워크 layer 선택 기법 동작 예시

Ⅱ. 배경지식

본 장에서는 서론에서 언급한 네트워크 경량화 기법과 연산 최적화 기법, 즉, 프루닝(Pruning), CMSIS-NN[1], NestedNet[2] 그리고 nnom[3]을 간략하게 소개한다. 프루닝 기법은 CNN(Convolution Neural Network)에서 결과에 미치는 영향이 적은 가중치(Weight)를 제거함으로써 정확도 감소를 최소화하며 메모리 사용량과 연산량을 줄인다. CMSIS-NN은 ARM 사에서 개발한 소프트웨어 라이브러리로서, convolution, activation 등과 같이 뉴럴 네트워크 추론에 필요한 함수들을 ARM 프로세서에 최적화하여 배포하고 있다. nnom 프로젝트는 Keras에서 생성한 모델을 CMSIS-NN 기반 C 파일로 변환하는 오픈소스 프로젝트이다. 본 논문에서는 ARM Cortex-M 시리즈에서 뉴럴네트워크 응용프로그램을 동작시키기 위해 nnom을 사용하여 실험을 진행하였다.

NestedNet은 다양한 단계로 프루닝된 여러 뉴럴 네트워크를 하나의 큰 뉴럴 네트워크가 포함할 수 있도록 하는 기법이다. NestedNet에서는 총 세 가지 단위로 프루닝을 적용할 수 있는데 가장 자유도가 높은 가중치 수준, 중간 단계의 필터 수준, 마지막으로 그림 1에 도시된 바와 같이 하나의 layer 전체를 생략할 수 있는 layer 수준을 지원한다. 본 논문에서는 layer 수준 프루닝을 주로 다루며 이는 Wide ResNet[4]과 같이 residual block을 지원하는 특수한 뉴럴 네트워크 모델에만 적용가능하다.

Ⅲ. 제안하는 기법

본 장에서는 유동적인 뉴럴 네트워크를 생성하기 위해 NestedNet에 제안된 layer 프루닝 방식을 자원 제약 내장형 프로세서에 적용하는 것을 제안한다. 이를 활용하면 시스템의 연산 요구량이 동적으로 변화하는 응용의 경우 적은 연산량을 요구할 때, 특정 layer를 건너뜀(layer skip)으로써 연산량을 감소시킬 수 있다.

수식 (1)은 layer skip이 적용된 경우와 적용되지 않은 경우에 대한 두 가지 경우의 평균으로 loss를 최소화하도록 뉴럴 네트워크의 가중치 W를 훈련하도록 변경된 loss 함수이다. 함수 L은 cross entropy 함수이며, Y_{pred}는 모든 layer를 거쳤을 때의 예측값, Y'_{pred}는 layerskip을 적용한 예측값, Y_{true}는 dataset의 true label을 의미한다.

$$\min_{W} \frac{L(Y_{true}, Y_{pred}) + L(Y_{true}, Y'_{pred})}{2} \quad (1)$$

본 논문에서는 그림 2와 같이 기존 Wide ResNet[4]에서 사용하는 Residual block으로 내장형 시스템에서 구동할 뉴럴 네트워크 모델을 설계하였다. layer skip을 적용하지 않는 경우에는 Residual block 1이 수행된 후, Residual block 2이 수행된 결과가 fully connected layer에 입력으로 들어간다. 반면 layer skip을 적용한 경우, Residual block 1만이 수행된 결과가 fully connected layer의 입력이 된다.

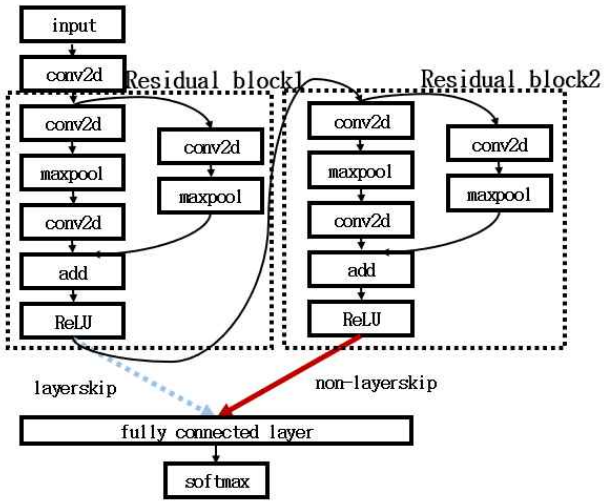
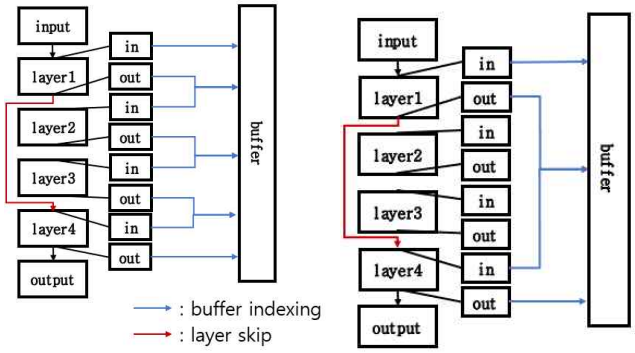


그림 2. 실험에서 사용한 유동적 뉴럴 네트워크 모델



a. 수정 전 nnom b. 수정 후 nnom

그림 3. layer skip을 고려한 기존 nnom과 수정한 nnom의 버퍼인덱싱 과정

기존 nnom의 경우 layer skip을 고려하지 않았으므로 그림3 (a)와 같이 layer skip을 적용할 때 일부 layer의 입출력이 전체 buffer 상에서 제대로 indexing 되지 않는 문제가 존재하였다. 따라서 본 논문에서는 각 layer의 입출력이 전체 buffer상에서 indexing 되는 과정을 그림 3(b)와 같이 수정하였다. 즉, 기존 nnom에서 buffer indexing시 i번째 layer의 입력 buffer의 주소(in)와 i-1번째 layer의 출력 buffer의 주소(out)는 같은 주소를 갖게 된다. 그러나 기존 nnom buffer indexing 방식에서 layer2와 3을 건너뛴다고 가정했을 때, layer4의 입력이 layer1번의 출력 결과가 아닌 layer3의 출력 buffer가 되어야 하므로 그림3 (b)와 같이 buffer를 indexing하는 함수를 수정하여 layer skip 적용 시 layer4번의 입력으로 layer1번의 출력이 오도록 수정하였다.

Ⅳ. 실험 결과

뉴럴 네트워크의 훈련, 정확도 측정은 모두 서버 환경에서 진행하였고, 실제 내장형 시스템 상에서의 연산 속도와 메모리 사용량은 cortex-M7을 사용한 NUCLEO - F746Zg 보드에서 측정하였다. 수행 시간 측정을 위해 keil uVision5 통합 개발 환경 프로그램 내부의 디버거를 사용하였다. 연산 속도의 평균을 구하기 위해 9장의 테스트 이미지를 사용해 추론에 소요된 시간의 평균을 구했다. dataset은 MNIST[5]를 사용했으며, 네트워크 모델은 그림 2와 같은 Wide ResNet 형태의 네트워크 모델을 사용하였다.

표1. 기존 모델과 최적화 모델 비교 분석

	정확도	평균 연산속도(tick)
full	99.05%	10.38
layer skip	98.80%	6.88
original	99.15%	10.38

표1에서 full은 수식(1)을 통해 학습한 모델에서 모든 layer를 수행하는 모델이며, layer skip은 특정 layer를 건너뛰었을 때 모델이다. 본 실험에서는 그림 2의 Residual block2 즉, 총 7개의 layer를 생략하여 추론하였다. original 모델은 loss 함수를 본 연구에서 사용하는 수식 (1)이 아닌 모든 Layer를 거치는 경우의 결과만을 비교해 학습하는 수식 (2)를 사용해 학습한 모델을 의미한다.

$$\min_{W} L(Y_{true}, Y_{pred}) \quad (2)$$

표1을 통해 layer skip을 했을 경우, 정확도는 0.25% 감소한 것을 확인할 수 있고, layer skip을 사용했을 때, 연산 속도가 33.73% 감소한 것을 확인할 수 있다.

표2. 기존 모델과 layer skip 모델의 연산량과 메모리 사용량의 차이

	연산량(FLOPS)	메모리 사용량(B)
full	9.80M	96.65K
layerskip	5.99M	76.39K

표2에서 볼 수 있듯 연산량(FLOPS)은 9.80M에서 5.99M로 약 38.8%의 연산량이 감소한 것을 확인할 수 있다. 또한 네트워크의 메모리 사용량 역시 96.65KB에서 76.39KB로 20.96%감소한 것을 확인할 수 있다.

Ⅴ. 결론

본 논문에서는 내장형 시스템에서 한 디바이스 내 다양한 상황에 맞게 유동적으로 동작하는 뉴럴 네트워크 기법에 대해 소개하였다. 실험 결과 특정 layer를 뛰어넘었을 경우, 모든 layer를 사용했을 경우보다 정확성이 0.25% 감소 하지만 연산량을 38.8%, 데이터의 예측에 걸린 시간을 32.73% 줄일 수 있다는 것을 확인하였다. 따라서 본 논문에서 제안하는 최적화 기법이 정확성과 연산량의 trade-off를 유동적으로 조절하여 자원 제약 내장형 시스템에서 동적으로 적용될 수 있음을 검증하였다.

ACKNOWLEDGMENT

본 연구는 과학기술정보통신부 및 정보통신기획평가원의 대학ICT연구센터지원사업의 연구결과로 수행되었음(IITP-2020-2018-0-01424)

참 고 문 헌

[1] LAI, Liangzhen; SUDA, Naveen; CHANDRA, Vikas. Cmsis-nn: Efficient neural network kernels for arm cortex-m cpus. arXiv preprint arXiv:1801.06601, 2018.

[2] KIM, Eunwoo; AHN, Chanho; OH, Songhwai. Nestednet: Learning nested sparse structures in deep neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018. p. 8669-8678.

[3] Neural Network on Microcontroller (NNoM) [Online] Available: https://github.com/majianjia/nnom

[4] MISHRA, Asit, et al. WRPN: wide reduced-precision networks. arXiv preprint arXiv:1709.01134, 2017.

[5] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” Proceedings of the IEEE, vol. 86, no. 11, pp. 2278 - 2323, 1998.