효율적인 게이트웨이 오프로딩을 위한 동적 채널 프루닝 기법

최성헌○, 이하윤, 신동군
성균관대학교 전자전기컴퓨터공학과
csh9580@skku.edu, lhy920806@gmail.com, dongkun@skku.edu

Dynamic channel pruning of efficient gateway offloading

Sunghern Choi○, Hayun Lee, Dongkun Shin
Department of Electrical and Computer Engineering, Sungkyunkwan University

요 약

최근 모바일 장치에서 딥 러닝 모델을 사용하는 어플리케이션들이 많이 개발되고 있다. 하지만 모바일 장치의 하드웨어 제약조건 때문에 모바일 장치 내에서 딥 러닝 모델을 사용하는 것은 성능이 떨어진다는 문제점이 있다. 이러한 문제를 해결하기 위해 계산의 일부를 모바일 장치에서 처리하고 남은 계산을 클라우드에서 처리하는 분할 연산을 수행하는 연구들이 진행되고 있다. 이 연구들은 주로 모바일 장치와 클라우드의 분할지점 선정, 모바일 장치에서 클라우드로 전송하는 데이터 압축 기법 등을 제안하지만 많은 연구가 실제 네트워크 상황에서 실행될 때 받는 영향을 고려하지 않고 있다. 본 논문에서는 실제 네트워크 상황에 따라 동적으로 압축 비율이 달라질 수 있는 프루닝 모델을 제안한다. 또한 이런 프루닝 모델을 사용해서 압축을 한 데이터의 정확도가 압축을 하지 않은 데이터의 정확도와 비교했을 때 정확도 손실이 7% 이하인 것을 보여준다.

1. 서 론

최근 애플의 Siri, 구글의 Now 등 모바일 장치에서 딥 러닝 모델을 사용하는 애플리케이션들이 많이 개발되고 있다. 애플리케이션에서 사용되는 딥 러닝 모델들은 상당한 계산량이 요구된다. 따라서 이러한 애플리케이션은 주로 클라우드 서버에서 계산을 처리하는 방식으로 사용되고 있다. 클라우드 서버는 고성능 GPU를 사용하기 때문에 모바일 장치에서 계산을 처리하는 것보다 빠르게 계산을 처리할 수 있다. 하지만 처리할 데이터를 모바일 장치에서 클라우드로 전송할 때 발생하는 에너지, 전송 지연시간이 발생한다. 이때 전송하는 데이터는 모바일 장치에서 계산돼서 나온 결과인 피쳐 맵(feature map)이다.

이러한 비용을 없애기 위해 모바일 장치 자체에서 계산을 처리할 수도 있지만 모바일 장치가 갖고 있는 하드웨어 제약조건 때문에 계산을 처리하는데 상당한 시간이 소요된다.

따라서 최적의 성능을 내기 위해 데이터를 모바일 장치와 클라우드 서버에서 나눠서 처리하는 연구들이 진행되었다[1, 2, 3, 4]. 모바일 장치와 클라우드 서버로 분할해서 계산을 처리하면 모바일 장치에서 클라우드로 전송하는 데이터의 크기를 줄일 수 있다. 적은 연산으로 보내는 데이터의 크기를 크게 줄일 수 있기 때문에 활발히 연구가 진행되었다. 이러한 연구들은 최적의 성능을 낼 수 있는 분할지점 선택, 분할지점에서 전송하는 데이터의 압축에 초점을 맞추고 있다.

기존연구[2, 4]에서 분할지점을 선택할 때는 데이터가 모바일 장치와 클라우드에서 처리될 때 각각에서 발생하는 지연시간과 에너지, 네트워크 상태, 클라우드 서버와 모바일 장치의 로드를 모두 고려한다. 딥 러닝 모델을 분할 하는 Neurosurgeon[1]에서는 지연시간, 에너지, 네트워크 상태까지는 고려하고 있고 클라우드 서버와 모바일 장치의 로드는 고려하지 않았다. 하지만 SPINN[2]과 Clio[4]에서는 클라우드 서버와 모바일 장치의 로드까지 모두 고려하고 있다.

하지만 분할지점에서 데이터를 전송할 때 보내는 데이터를 압축하는 연구[3]에서는 네트워크 상태, 클라우드 서버와 모바일 장치의 로드를 고려하지 않고 있다. 최적의 압축 작업을 위해서는 동적으로 바뀌는 네트워크 상황을 반영해야 한다. 동적인 네트워크 상황을 압축기법에서도 반영하기 위해서는 다양한 압축률로 압축을 할 수 있는 모델이 필요하다.

본 논문에서는 동적으로 바뀌는 네트워크 상태, 클라우드 서버 및 모바일 장치의 로드를 고려하기 위해 조건에 따라 압축 정도가 동적으로 달라지는 프루닝 기법을 소개한다.

2. 배경지식 및 관련 연구

2.1 슬라이싱(slicing) 및 프루닝(pruning)

피쳐 맵의 압축을 위해 사용하는 기법으로는 슬라이싱과 프루닝이 있다.

슬라이싱 기법은 그림 1(a)와 같이 피쳐 맵을 일정 비율로 잘라내는 기법이다. 슬라이싱 기법은 일정 비율에 해당하는 부분의 채널을 연속적으로 자른다. 연속적으로 채널을 자르기 때문에 하드웨어에서 처리하기가 좋다는 장점이 있다. 하지만 채널의 중요도를 고려하지 않고 연속적으로 자르기 때문에 중요도가 높은 채널들이 잘리는 경우가 발생할 수 있다. 이는 accuracy에 저하를 발생시키는 원인이 될 수 있다. 슬라이싱과 관련된 기존연구 중 네트워크 상태에 따라서 채널을 슬라이싱하는 연구[4]가 있다. 연구에서는 네트워크 상태에 따라서 슬라이싱하는 비율을 다르게 설정한다.

프루닝 기법은 그림 1(b)와 같이 피쳐 맵에서 중요하지 않은 채널을 선택적으로 잘라내는 기법이다. 프루닝 기법은 중요한 채널과 중요하지 않은 채널을 구분해서 잘라내기 때문에 슬라

이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.IITP-2017-0-00914, 지능형 IoT 장치용 소프트웨어 프레임워크)

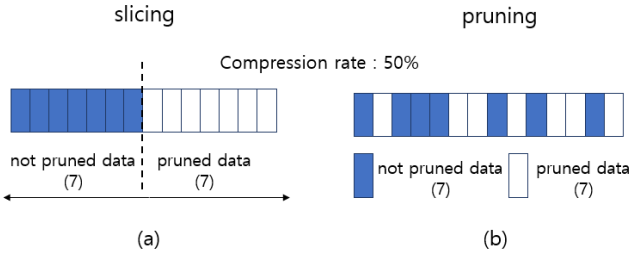

그림 1 slicing 기법과 pruning 기법. (a) slicing 기법을 나타낸다. 피처 맵의 일정 비율을 연속적으로 잘라낸다. (b) pruning 기법을 나타낸다. 피처 맵의 중요도가 낮은 채널을 잘라낸다.

이싱 기법보다 accuracy에 저하를 적게 발생시킨다. 프루닝을 하는 방법은 채널들의 중요도 순위에 따라 프루닝하는 방법[5] 등 여러 가지가 있다.

2.2 joint training 및 knowledge distillation

joint training은 한 모델에서 output이 여러 가지 일 때 학습을 하기 위한 방법이다. 여러 output으로부터 나오는 loss들을 같이 학습시킨다. joint training을 사용하면 한 모델에서 여러 task를 추론할 수 있다.

knowledge distillation은 미리 학습된 큰 모델(teacher)의 지식을 실제 사용하려고 하는 작은 모델(student)에게 전달하는 것이다. 이는 적은 비용으로 작은 모델도 큰 모델만큼의 성능을 내도록 하기 위한 학습전략이다.

3. 동적 채널 프루닝(pruning) 기법

이 연구에서는 채널의 중요도에 따라 프루닝을 하기 위해 Top-k를 선정한다. Top-k를 선정해서 프루닝을 하는 방식은 FBS[5]에서 사용된 방식을 사용했다. Top-k 선정 이란 채널들의 중요도를 숫자로 나타낸 후 그중 가장 높은 k 개의 채널을 선정하는 것이다. Top-k를 선정한 후 Top-k에 선정되지 못한 채널의 피쳐(feature) 들을 0으로 만든다. 0이 된 피쳐들은 계산하지 않는 피쳐들이기 때문에 클라우드로 전송되지 않는다. Top-k를 선정해 프루닝을 하는 과정은 그림 2에 나온 과정으로 진행된다. 서브 네트워크에서 채널의 중요도를 판단한 후 프루닝한다. Top-k를 선정하는 과정에서 PR(Pruning Rate)을 입력받아 Top-k를 선정한다.

동적으로 달라질 수 있는 k 값을 학습시키기 위해 모델을 joint training 하고 학습에 도움을 주기 위하여 knowledge distillation 기법을 사용한다.

모델을 joint training 하기 위해 여러 k 값에 따라 나오는 output에 대한 loss를 구하고 학습을 시킨다. 이렇게 학습을 시키게 되면 모델은 여러 k 값에 대해서 학습을 할 수 있다. 실제 실험에서는 k 값을 변경하기 위해 PR을 변경하면서 실험한다. PR은 [0,1]의 범위를 가지고 있다. PR이 0일 경우에는 k 값이 0인 경우로 프루닝을 하지 않는 경우이다. PR이 0이 아닌 경우에는 PR에 따라 k 값이 정해진다.

위의 내용을 바탕으로 본 논문에서 학습에 사용할 loss를 식으로 나타내면 아래와 같다.

$$ss = CE(M(x), y) + \sum_{i=1}^{N} CE(M_i(x), M(x)) \quad (1)$$

CE는 Cross Entropy loss, M(x)는 teacher 모델의 결과(output), $M_i(x)$는 student 모델의 결과이다. N은 student 모델들의 수이다.

그림 3에 나온 예시를 통해 학습이 진행된다. knowledge distillation 기법을 적용하기 위해 프루닝을 하지 않은 모델, 즉 PR을 0으로 설정한 모델을 teacher 모델로 사용한다.

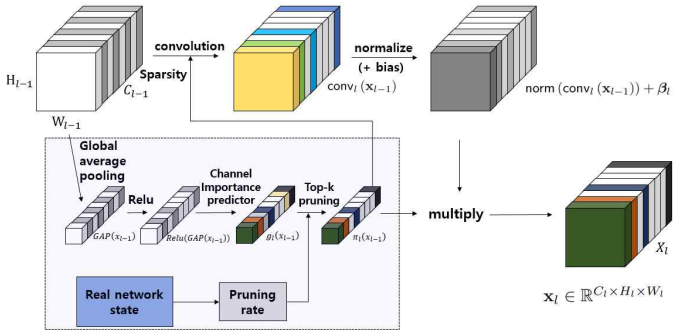

그림 2 프루닝 과정. 채널의 중요도를 판단하기 위한 서브 네트워크를 생성한다.

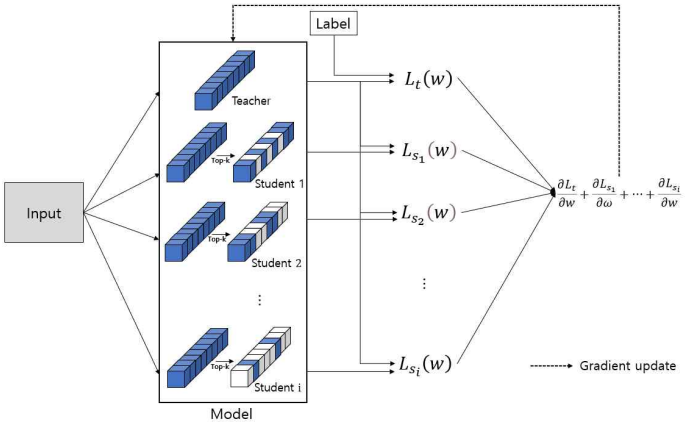

그림 3 모델 학습 예시. Teacher의 loss를 계산할 때만 Label을 사용하고, student의 loss를 계산할 때는 teacher의 output을 label로 사용한다.

knowledge distillation 기법은 US-Net[6]에서 사용한 방식을 사용했다. 여러 k에 대해 프루닝 했을 때, 즉 PR을 0으로 설정했을 때를 제외한 모델을 student 모델로 사용한다. 주어진 label과 teacher의 output의 차로 teacher의 loss를 계산한다. 이후 student들의 loss는 teacher의 output과 student의 output의 차로 계산된다. 계산은 teacher부터 student i까지 순차적으로 진행된다. student i까지 loss가 다 계산되면 계산된 loss를 통해 구해진 teacher부터 student i까지 gradient를 전부 더한다. 그리고 새롭게 구해진 gradient를 사용해서 모델의 가중치 업데이트가 진행된다.

4. 실험 환경 및 결과

4.1 실험 환경

실험에서 사용된 DNN 모델은 ResNet20 이고, 데이터 셋으로 CIFAR-10을 사용했다. 실험결과에 사용되는 용어로 LPR(Learning Pruning Rate)은 training 시 사용된 PR이고, TPR(Test Pruning Rate)은 테스트 시 사용된 PR이다. STP(Single Target PR Model)는 단일 PR에 대해서 training을 시킨 모델이고, MTP(Multi Target PR Model)는 여러 PR에 대해서 joint training을 시킨 모델이다.

실험에서 0, 0.2, 0.4, 0.6 네 가지 PR을 MTP를 학습시킬 때 사용하고 TPR이 0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6인 경우에 대해 테스트했다.

MTP의 성능을 분석하기 위해 비교군으로 LPR이 0, 0.2, 0.4, 0.6인 STP를 MTP와 동일하게 TPR이 0, 0.1, 0.2, 0.3, 0.4, 0.5,

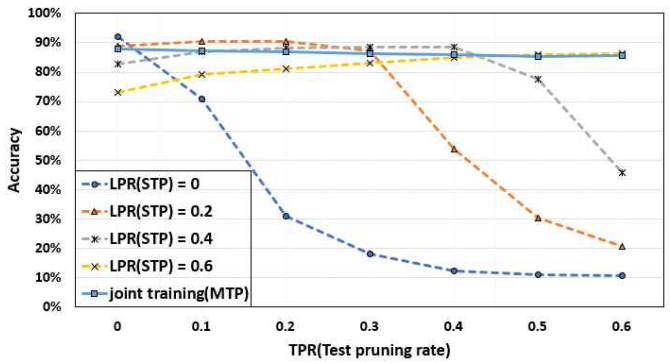

그림 4 MTP와 STP 결과 비교 그래프. 각 선들은 LPR들을 나타내고 x축의 값들은 TPR이다.

0.6인 경우에 대해 테스트했다.

4.2 실험 결과

그림 4는 TPR에 따른 모델의 정확도를 보여준다. STP 모델들은 LPR과 동일한 TPR에서 가장 높은 정확도를 가진다. LPR과 다른 TPR에 대해서는 LPR과 동일한 TPR의 경우보다 낮은 정확도를 가진다. 특히 LPR이 0인 STP의 경우, TPR이 0인 경우를 제외한 다른 TPR에서의 정확도가 상당히 낮다.

STP와 비교해 봤을 때 MTP는 모든 TPR에 대해서 85% 이상의 정확도를 얻어냈다. STP는 LPR < TPR인 경우 상당히 낮은 정확도가 나왔고, LPR > TPR인 경우 감소는 하지만 소폭 감소했다. 이 이유는 4.3장에서 분석한다.

4.3 실험 결과 분석

STP는 LPR > TPR인 경우 LPR과 비슷한 수의 채널을 선택한다. 그림 6은 layer 별로 선택되는 채널 수의 비율을 보여준다.

그림 5 (a)~(c)를 보면 LPR > TPR인 TPR들이 LPR과 비슷한 수의 채널을 선택한다. 따라서 TPR가 LPR과 다르지만, 채널을 선택할 때 LPR과 비슷한 수의 채널이 선택되기 때문에 정확도의 감소가 크지 않다. LPR과 비슷한 수의 채널이 선택되는 이유는 서브 네트워크에서 Global Average Pooling이 된 feature가 top-k를 선정하기 전 Relu를 통과하면서 LPR과 비슷한 PR로 프루닝이 되기 때문이다.

반면 LPR < TPR인 경우에 대해서는 그림 5 (a)~(c)를 보면 각 TPR에 맞는 채널의 수가 선택되는 것을 볼 수 있다. LPR보다 더 많이 채널을 프루닝하기 때문에 정확도의 감소가 크다.

그림 5 (d)에서는 MTP의 선택되는 채널 수의 비율이 다양한 것을 볼 수 있다. 앞쪽 layer에서는 PR이 0.2일 때만큼의 채널의 수가 선택되고 뒤쪽 layer로 갈수록 PR이 높을 때의 채널 수가 선택된다. 4번, 16번 layer에서는 PR이 0.4일 경우보다 더 적은 채널이 선택되고 중간 layer에서는 다양하게 채널 수가 선택된다. 이처럼 다양하게 채널의 수를 선택하기 때문에 모든 TPR에 대해서 높은 정확도를 얻어낼 수 있다.

5. 결론 및 향후 연구

본 논문에서는 동적인 피쳐 맵 압축을 위한 프루닝 기법을 제안하였다. 다양한 PR에서 동작할 수 있도록 하는 joint training 기법과 knowledge distillation 기법을 사용해 모델을 학습시켰다.

PR이 0, 0.2, 0.4 0.6일 경우로 MTP를 학습시켰고, 테스트한 결과 모든 PR에 대해서 정확도(accuracy)가 85% 이상이었다. 이는 압축을 하지 않았을 때의 정확도와 비교했을 때 7% 이하의 정확도 손실이 있는 것이다. 따라서 7% 이하의 정확도 손실로 여러 압축률의 압축을 할 수 있다.

게이트웨이 오프로딩시 네트워크 상태 등 동적으로 변하는 요소들에 의해 모바일 장치에서 클라우드 서버로 데이터를 전송

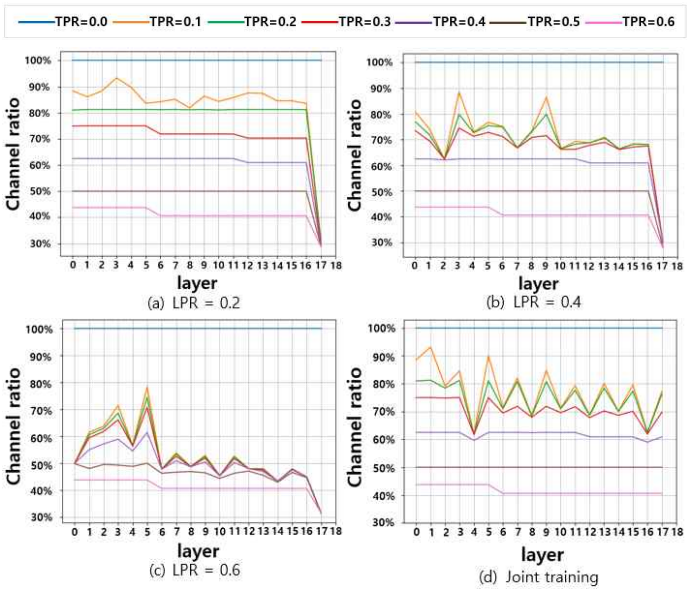

그림 5 layer 별 선택되는 채널의 비율. 각 선들은 TPR이다.

할 때 발생하는 지연시간이 달라진다. 예를 들어 네트워크 상태가 좋지 않으면 데이터를 전송할 때 큰 지연시간이 발생한다. 따라서 동적으로 변하는 요소들이 미치는 영향을 최소화할 필요가 있다. 이를 위해 동적 프루닝 기법을 실제로 게이트웨이 오프로딩에 적용하는 것이 앞으로 연구할 과제이다.

6. 참고문헌

[1] Yiping Kang, Johann Hauswald, Cao Gao, Austin Rovinski, Trevor Mudge, Jason Mars, and Lingjia Tang. Neurosurgeon: Collaborative Intelligence Between the Cloud and Mobile Edge. In International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS), 2017.

[2] Stefanos Laskaridis, Stylianos I. Venieris, Mario Almeida, Ilias Leontiadis, and Nicholas D. Lane. SPINN: Synergistic Progressive Inference of Neural Networks over Device and Cloud. In International Conference on Mobile Computing and Networking (MobiCom), 2020.

[3] A. E. Eshratifar, A. Esmalili, and M. Pedram. BottleNet: A deep learning architecture for intelligent mobile cloud computing services. In International Symposium on Low Power Electronics and Design (ISLPED), 2019.

[4] J. Huang, C. Samplawski, D. Ganesan, B Marlin, and H. Kwon. Clio: Enabling automatic compilation of deep learning pipelines across IoT and Cloud. In International Conference on Mobile Computing and Networking (Mobicom), 2020.

[5] Xitong Gao, Yiren Zhao, Lukasz Dudziak, Robert Mullins, Cheng-zhong Xu. Dynamic Channel Pruning: Feature Boosting And Suppression. In International Conference on Learning Representation (ICLR), 2019.

[6] Jiahui Yu, Thomas Huang. Universally Slimmable Networks and Improved Training Techniques. In International Conference on Computer Vision (ICCV), 2019.