PBL Paper Review

2016026080 한다진

**- DeepXplore**

모든 Deep Neural Network에는 취약점이 있습니다. 학습데이터로 사용한 이미지가 아닌 새로운 이미지에 대해서는 어쩔 수 없이 정확도가 떨어지기도 하고, 이미지에 약간의 변화만 주어도 전혀 다른 결과를 내놓기도 합니다. 그래서 새로운 데이터셋을 생성하는 것이 중요하게 여겨졌고, 기존 데이터셋에서 새로운 데이터셋을 만드는 여러 고전적인 방법론이 제시되었습니다. 하지만 Edge-case 이미지를 생성하는 방식들은 대부분 레이블링을 매번 하는 수고가 동반되고, 특정 픽셀에만 큰 변화를 주는 방식이었습니다. 인간이 직접 하기 때문에 새로운 이미지를 생성하는 방식에 패턴이 존재했고, 이렇게 생성된 이미지는 실제 상황에서도 매우 edge-case였습니다. 그래서 효율적인 방식으로 새로운 데이터셋을 생성하고자 하는 방법이 많이 연구되었고 그중 하나가 DeepXplore입니다.

DeepXplore는 앞의 문제를 해결하기 위해, 학습된 딥러닝 모델을 활용하여 기존 학습데이터로부터 새로운 학습데이터를 생성하는 방식을 취했습니다. DeepXplore는 pre-trained된 모델을 사용하여 기존 모델에 취약한 이미지를 생성하고, 기존 모델을 fine tuning 합니다.

새로운 이미지를 생성하는 방식은 다음과 같습니다. 모델에 현재 이미지를 입력으로 줍니다. 모델은 현재 이미지를 받아서 결과값을 내고, 정답값과 비교하여 gradient값을 계산합니다. Pre-training 시에는 이 gradient 값을 모델의 layer의 weight값에 반영해줬지만, 지금은 이 gradient값을 첫번째 tensor에 영향을 주도록 합니다. 즉 입력으로 주어진 이미지에 반영해줍니다. 그렇게 새로운 이미지를 생성하는 방식입니다.

Fine-tuning의 핵심은 neuron coverage입니다. 모델 내부의 가능한 모든 뉴런이 활성화 되도록 추가 학습을 시킵니다. 여기서 활성화 되었다는 것은 해당 뉴런의 가중치값이 일정 threshold 값 이상이라는 것입니다. 모든 뉴런이 결과에 유효한 영향을 끼치도록 하는 방식입니다.

앞의 두 과정을 효율적으로 진행하기 위해서 DeepXplore는 input tensor와 output tensor 그리고 모델의 용도가 동일한 모델을 3개 사용하여 loss function을 normalize 했습니다. 특정 모델이 이미지 결과값에 방해를 주도록 영향을 끼치도록 하고 다른 모델은 이를 막도록 하여 특정 모델의 취약점이 부각되도록 새로운 이미지를 생성하도록 했습니다. 그리고 매번 각각의 모델들로부터 활성화 시킬 레이어의 노드를 찾아 그 값을 중심으로 fine tuning이 일어나도록 했습니다.

**- Poseidon**

Poseidon은 gpu cluster 구조의 컴퓨팅 환경에서 딥러닝 학습 시 발생하는 오버헤드를 최소화하는 방법이다.

사용해야 하는 파라미터가 매우 많거나, 입력으로 주어지는 데이터셋의 크기가 너무 클 경우, 클러스터 구조로 여러 컴퓨터를 사용하여, 딥러닝 모델을 학습시킨다. 이때, 딥러닝 모델의 weight 값 등 여러 노드 간 공유되어야 하는 값은 컴퓨터간 네트워크 통신을 통해 주고받아진다. 하지만 기존의 방식에서는, 한 학습이 진행될 때 마다, 필요로 하는 노드를 서로 통신해야 하고, 학습이 완료될 때 마다, 갱신되어야 하는 값이 다른 컴퓨터로 보내져야 하기 때문에 네트워크 망에서의 과부하가 발생하곤 했다. Poseidon은 gpu cluster 구조에서 네트워크 과부하로 인한 gpu사용률 저하를 방지하기 위해, 병렬적으로 데이터를 통신하는 새로운 방법을 제시했다.

딥러닝 클러스터 환경을 가속화시키는 방법은 크게 Data-Parallelism과 model parallelism이 있다. 각각 데이터를 병렬화 하여 처리 할 것인지, 아니면 연산을 맡는 모델을 여럿으로 분리하여 병렬적으로 계산하는 방식이다. Poseidon은 이중 Data-Parallelism 방식으로 클러스터 환경을 가속화한다.

Poseidon이 target으로 선정한 문제점은 다음과 같다. Gpu의 계산 속도는 빠르지만, gpu 전용 메모리에서의 메모리 복사 오버헤드가 발생하고, 매 학습이 끝날 때 마다, 이를 반영시키기 위해서는 많은 통신량이 필요한데, 대역폭이 제한되어있기 때문에, 통신부하와 네트워크 지연이 동시에 발생하는 것이다. 그렇게 gpu를 실제 성능보다 덜 활용하게 되는 문제를 target으로 삼았다. 이 문제를 해결하기 위해 Poseidon은 Parameter 업데이트를 스케쥴링하였고, Parameter 업데이트의 구조 자체에 변화를 주었다.

Parameter 업데이트를 스케쥴링한 방식은 WFBP(Wait-Free BackPropagation) 이다. 기존에는 모델 학습시에는 파라미터값 로드만 하고, 학습이 완료된 시점에 전체 모델의 파라미터값을 갱신해주었다. WFBP 방식은 레이어 별로 나누어서 파라미터 이동 스케쥴링을 진행했다. 레이어 단위로, 연산하고, 연산결과 내보내고, 동기화된 값을 다시 불러오는 세 작은 동작을 하나의 큰 동작으로 삼았다. 그리고 레이어별로 큰 동작을 순차적으로 진행하며 데이터 통신이 특정 시간대에 몰리는 것을 피했다.

하지만 위의 방식은 통신량 자체를 줄이지는 못하였다. 통신이 몰리는 시간대만 피한 것이다. 두번째로 파라미터 업데이트 구조를 바꾸는 방식으로 hybrid communication 방법이 제시되었다. 이는 파라미터의 개수를 가지고 통신량을 계산하여, 현재 시점에서 어떤 방식으로 통신하는 것이 통신량이 적은지 판단하여, 보다 효율적인 방식으로 통신을 하는 것이다. 통신 방식은 star 방식과 full-mesh 방식이 있는데, 파라미터 값을 가지고 전체 통신량을 계산하는 수식이 존재한다. 그렇기 때문에, 매번 효과적인 통신방식을 채택하여 통신량 자체를 줄이는 hybrid communication 방식을 적용하여 gpu cluster 구조의 성능을 올렸다.