Note

- About
-
-
-

Knowledge Distillation 리뷰

January 2, 2020 - Ezobear

Introduction

과거부터 지금까지 기계학습 방법을 이용한 알고리즘들은 앙상블 기법을 통해 SOTA를 보여주었습니다. 그러나 이러한 앙상블 방법의 장점때문에 매우 느리고, 무겁다는 큰 단점들이 간과되고있었습니다. 따라서 이러한 문제가 Cristian al.3 에 의해 제시되었고, 이러한 문제를 해결하기 위해 Mimic Learning이라는 학습기법을 제안하였습니다. 이는 기존의 앙상블의 거대하고, 큰 모델에서 예측된 확률을 Pseudo Label로 사용하는 방법으로 KD분야의 기원이 되었습니다[1].

이 Mimic Learning의 배경과 마찬가지로, Deep Learning에서도 모델이 너무 무겁고 너무 느리다는 문제가 제기되었습니다.[2]. 또한 2014년에는 Gradient Vanishing 문제로 네트워크를 깊게 쌓는것 마저 어려운 일이었습니다. 따라서 저자들은 심층 학습 네트워크가 더 깊어야 하는지에 대해 의문을 제기하였고, 또한 심층 네트워크에서 얕은 네트워크로 미믹 학습을 수행함으로써 얕은 네트워크로 깊은 네트워크의 성능을 내기 위해 연구를 진행하였습니다. 그러나 본 논문의 한계는 모델을 압축하는 것이 아니라, 얕은 모델로 깊은 모의 예측 분포를 모방하여 정확도를 높일 수 있다는 점에서만 기술하고 있다는 점 입니다.

이러한 문제는 이후 2015년에 이 Mimic Learning 연구에서 영감을 받은 구글의 Knowledge Distillation을 이용한 딥러닝 모델 압축에 관한 연구가 시작되었고, Fitnet, 연구를 지나면서 점차 틀이 잡히고, 구체화됩니다. 그러면서 Attention Transfer, Jacobian Trnasfer 등 다양한 연구들이 등장하게 되는데, 이러한 흐름과 각 모델에 대해서는 아래서 더 자세하게 다루겠습니다.

본 리뷰에서는 세가지 파트로 나눠 KD 분야에 대해 설명하고있습니다. 첫째 파트에서는 위의 큰 흐름에 대해 조금 더 자세하게 다룹니다. 두번째 파트에서는 이러한 큰 흐름에서 파생되어나온 다양한 KD 방법론에 대해 다룹니다. 세번째에서는 이러한 KD가 적용된 Object Detection 논문에 대해 다루도록합니다.

A large Stream of Knowledge Distillation’s Development

제1장은 KD의 분야의 큰 흐름에 대해 다룹니다. 이 큰 흐름은 4개의 주요 연구로 이어지는데, Mimic Learning과 Deep Learning의 Model Compression을 제안한 연구, 그리고 그것을 바탕으로 한 KD라는 용어를 제안한 연구입니다. 이후 FitNet 연구를 거치면서 점차 Model Compression의 틀을 발전시킬 뿐만 아니라 정량적 평가를 위한 큰 틀또한 구축하게 됩니다[1][2][3][4].

1. Model Compression & Do Deep Nets Really Need to Deep?

과거의 기계학습 분야에서는 앙상블과 거대 모델이 좋은 성과를 거두었습니다. 그러나, 이러한 모델들은 많은 컴퓨팅 자원을 필요로 했고, 이러한 문제들을 해결하기 위해 Mimic Learning이 제안되었습니다[1]. Mimic Learning의 핵심 아이디어는 대형 혹은 앙상블 모델의 Prediction을 Soft Label로 사용하여 소규모 모델을 Supervised Learning하는 기법입니다.

표 1은 본 논문에서 제시하는 데이터를 제가 재구성 한 표입니다. 여기서 정확도의 지표로 볼 수 있는 RMSE는 비슷하지만, 추론 시간과 모델 크기는 매우 큰 차이로 성공적인 Model Compression을 하였다고 볼 수 있습니다.

이후 딥러닝이 Gradient Vanishing 문제로 더 이상 깊게 네트워크를 쌓는것이 어려웠던 시기에 이와 같은 방법을 딥러닝에 적용시킨 Do Deep Nets Really Need to Deep? 논문도 발표되었습니다[2]. 이러한 논문들은 이후 KD연구분야의 기원이됩니다.

2.Distilling the Knowledge in a Neural Network

KD라는 용어가 처음 사용된 구글의 논문입니다[3]. 여기서 Distilling은 증류로 원하는 특성 성분을 불순물과 혼합한 혼합물에서 분리하는 방법인데, 구글의 KD 논문에서는 신경망에서 학습된 파라미터의 결과를 지식이라고 하며, 이러한 지식은 많은 불순물이 혼합되어있는 파라미터로부터 순수한 지식을 증류하는 방법을 제안하였고, 이를 통해 Model Compression을 수행하고자 하였습니다.

위의 수식은 논문에서 제안하는 KD방법이며, 기본적으로는 Mimic Learning과 같습니다. 이때 q는 각 클래스의 확률입니다. 다만 지식을 Distilling하는 온도 하이퍼파라미터 T가 추가되었는데, 이 T를 통해 Distilling된 정보(Softmax로 구해진 확률맵)을 학습하게 됩니다. T는 1로 설정하면 일반적인 SoftMax함수가됩니다. 이때 T 값이 커지면 Expoential로 들어가는 입력값이 작아지므로 결과값이 천천히 증가하게 됩니다. 즉, SoftMax함수를 거칠때 한 값이 너무 큰 값을 가지는 것을 방지합니다. 이는 전체 확률 분포를 부드럽게하며, 이를 통해 학습할 경우 좀 더 일반적인 지식을 배움으로써 모델이 일반화된다 하였습니다. Inference Phase에서는 T값을 1로 사용합니다. 이때 모델이 일반적이게 학습된 것을 보여주기 위해 Mnist 데이터셋에서의 KD를 보여주는데, 숫자 3을 제외하고 학습하였는데도 테스트에서 98.6%의 정확도를 보였다고하였습니다. 이 실험을 통해 KD논문에서는 학습하지 않은 데이터까지 어느정도 맞출수 있고, 모델 자체가 일반화된 것을 보여주었습니다. 다만 본 논문의 한계로는 모델의 압축정도에 대한 내용은 자세히 나와있지 않다는 점입니다. 이러한 모델 압축의 정량적인 평가 틀은 이후 Fitnet에서 잘 다져놓게 됩니다.

3.FITNETS: HINTS FOR THIN DEEP NETS

Fitnet은 KD분야에서 가장 중요한 논문이며 많은 논문의 Baseline으로 으로 사용되고있습니다[4]. 기존의 Model Compression 혹은 KD논문들은 대부분 정확도 상승 측면에 초점을 맞추어 실험결과를 보여주었습니다. 그러나 Fitnet논문에서는 정확도 측면뿐만 아니라 모델의 파라미터 수, 계산 속도, 정확도 등을 고려하여 압축과 속도 측면에서의 정량적인 측정을 수행했습니다. 따라서 이러한 정량적 평가의 틀을 구축함으로써 이후 많은 논문들에 영향을 미쳤습니다.

그림 1의 (a)는 Fitnet의 기본 아이디어인 Teacher Student Framework를 볼 수 있습니다. 이는 Fitnet의 기본 아이디어로, Teacher(넓고 얇은 모델,그러나 여전히 깊은)의 행동을 Student(좁고 깊은 모델)이 배우는다는것인데 기존의 Mimic Learning 방법들과의 차이점은, 단순히 이러한 모델의 형태가 정해져있을뿐만 아니라 Hints(Intermediate Hidden Layer)를 기반으로 학습한다는 점입니다. 이를 통해 오른쪽 빨간색 상자(Student)가 오른쪽 초록색 상자(Teacher)를 흉내내게 하는것입니다. 다만 좁은 Student는 넓은 Teacher의 Hints를 흉내낼 수 없습니다. 따라서 이를 해결하기 위해 그림 1의 (b)를 도입하게되는데, 그림 b의 W_r은 흉내내는 것을 도와주는 모듈입니다. 좀 더 기술적으로 얘기하면 좁은 채널을 갖는 Student가 넓은 채널을 갖는 Teacher의 Hints의 Feature Map을 학습하기 위해 사이즈를 매핑시켜주는 Regressor입니다. 즉,(a)와 같은 방법으로 전체적으로 파라미터수를 줄이면서 Multipication 횟수를 줄이므로써 가볍고, 빠른 모델을 만들고, (b)의 방법을 통해 이러한 모델인 Teacher를 따라할 수 있는 Student와 학습 도우미(Regressor)를 만들었습니다. 이 후 (c)와 같이 KD를 수행함으로써 가볍고, 빠르고, 정확한 모델을 만들 수 있다고하였습니다.

수식은 그림(c)의 학습방법에 대한 내용입니다. W_{Guided}는 연구의 특정 Layer의 Weight 파라미터인 Feature Map을 나타내며, W_{hint}는 힌트가 되는 Teacher의 특정 Layer의 Weight 파라미터인 Feautre Map을 가리킵니다. 모델은 Regressor를 사용하여 High Dimension인 Teacher의 Hints과 Low Dimension인 Student의 Hints간의 차이를 학습합니다.

Figure 3의 좌측 표는 Fitnet의 실험 결과로, 기존의 방법과 비교하였을때 파라미터 대비 정확도가 향상된 것을 볼 수 있습니다. 우측 그래프는 추가적으로 기존의 일반적인 Back Propagation 방식과 Knowledge Distillation 방식, Hints Training 방식을 비교한 실험인데, 오퍼레이션 수를 고정 후 Layer 수를 변화하면서 정확도를 측정한 실험입니다. 오퍼레이션 수가 많을때(107M)일때는 KD방식과 Hints방식이 모두 잘되었는데, 오퍼레이션 수가 적을때 KD로는 학습이 안되는 문제 또한 극복하였습니다. 따라서 보다 적은 파라미터로 학습이 가능하게 된것입니다. 아래 Figure4는 Fitnet의 학습 결과로, 모델의 레이어별, 파라미터의 개수, 연산 횟수, 정확도, 속도 증가율, 압축율 등 다양한 관점에서 Fitnet의 성능을 정량적으로 보여주고 있습니다. 이러한 기반은 향후 많은 논문에 영향을 미치고, Baseline으로 사용됩니다.

Various version of Knowledge Distillation

Fitnet 이후로 KD에 대한 다양한 버전의 테크닉들이 발표되었는데, 이러한 연구 흐름은 크게 2가지 방향으로 진행됩니다.

첫번째 연구 흐름은 KD 자체를 강화하는 방향입니다. 이는 주로 어떤 파라미터를 학습하는데 사용할건지에 대해 다룹니다. 두번째 흐름은 KD 후 Transfer를 강화하는 방향압니다. 주로 네트워크가 서로 많이 다를 경우(크기, 깊이 등) Transfer가 잘 되지 않는 문제를 해결하기 위한 방법을 다룹니다.

이러한 흐름에서 첫번째 케이스의 대표적인 연구로는 Attention을 학습하거나, Gan을 사용하거나, Jacobian을 사용하는 등의 연구가 있습니다[5][6][7]. 두번째 케이스의 대표적인 연구로는 Mutual Learning, Knowledge Projection, Teacher Assistant 등을 사용한 연구가 있습니다[8][9][10]. 각 논문의 자세한 내용들은 각 장에서 Key 아이디어 위주로 다시 보도록 하겠습니다.

1. PAYING MORE ATTENTION TO ATTENTION: IMPROVING THE PERFORMANCE OF CONVOLUTIONAL NEURAL NETWORKS VIA ATTENTION TRANSFER

이전장에서의 Fitnet의 논문에서의 Knowledge Transfer는 Wide Network에서 Narrow Network로 이루어졌습니다. 그러나 이러한 접근에서는 Regressor가 필요하다는 한계를 갖습니다. 따라서 본 논문에서는 추가적인 모듈이 없어도 Knowledge Transfer 가능한 방법론을 제안합니다. 본 논문의 핵심 아이디어는 Deep Network의 Attention을 Shallow Network Transfer한다는 점입니다. 이때 무엇을 Distillation하고, 어떻게 Transfer하는지는 아래서 다루도록 하겠습니다.

 위의 그림은 본 논문에서의 Knowledge Transfer 방법에 대한 그림입니다. Teacher의 경우 Deep Network로 이루어져있고, Student의 경우 Shallow Network입니다. Teacher와 Student의 다른점은 단순히 깊이에 있습니다. 따라서 학습 포인트에서의 Channel이 같기때문에 별도의 Regressor 없이 Transfer가 가능합니다. 이때 학습포인트에서 Transfer하는 Knowledge는 Attention이라하며, 학습포인트 구간에서의 Attention간의 차이의 합을 학습하는 것으로 Knowledge Transfer를 수행합니다.

 위의 그림은 본 논문에서 Distillation하는 Attention에 대한 그림입니다. (a)의 Input Image에서 뽑아낸 딥러닝 네트워크가 주목하고있는 정도에 대한 Feature가 Attention입니다. 이러한 Attention은 (b)와 같은 방법으로 구해지는데, N-Channel의 Feature를 Point-wise하게 평균내는 방법으로 구할 수 있습니다.(Mean, Average, Max등 다른 방법을 이용해도 됩니다.) 이렇게 Teacher Network에서 뽑아낸 Attention을 Student Network로 Transfer하면 됩니다. 따라서 본 논문에서는 이러한 방법을 통해 별도의 Regressor 없이 Deep Network에서 Shallow Network로 Transfer를 수행하였습니다.

표는 AT의 결과입니다. 데이터 셋은 CIFAR10이고, 각 표의 수치는 Error Rate입니다. 이대 표의 좌측 Student 부터 우측 Teacher까지의 각 기법을 적용하였을때의 성능을 보여주고있습니다. 표에서 보면 Wide에서 Thin으로의 Transfer는 여전히 KD가 성능이 좋은 것을 알 수 있습니다. 그러나 Deep에서 Shallow로의 Transfer는 AT기법이 더 우수한 성능을 보여줍니다. 추가적으로 이 둘을 같이 적용하였을때의 성능이 가장 좋은 것을 확인할 수 있습니다.

2. Training Shallow and Thin Networks for Acceleration via Knowledge Distillation with Conditional Adversarial Networks

본 논문의 핵심 아이디어는 KD를 수행할 때 GAN의 개념을 적용한 학습방법론입니다. 그림에서 보듯 먼저 입력 x_i 가 들어갔을때, Student와 Teacher Network가 각각 Inference를 수행하게 됩니다. 이때 Teacher는 Offline에서 pre-trained model입니다. 이후 student는 두번 학습을 진행하게되는데, 첫번째 학습은 Ground Truth와의 일반적인 학습입니다. 두번째 학습은 본 논문에서 제안하는 키 아이디어로, 이전에 Teacher에서 나온 Soft Label과 Student에서 나온 Soft Label을 보고 Disciminator가 Student가 Real인지 Fake인지 맞추게 됩니다. 이때 Disciminator는 일반적으로 사용되는 binary-cross entropy를 이용하여 업데이트 되고, 마찬가지고, Student 또한 일반적인 Gan에서의 loss인 KL Divergence를 이용하여 Teacher와의 차이만큼 학습을 진행하게 됩니다.

이렇게 학습을 진행하게 될 경우 위의 그림과 같이 Student가 업데이트 된 것을 확인할 수 있습니다. 보면 Our를 보면 Gan 기반의 Knowledge Transfer 학습을 하였을때 Student의 분포와 Teacher간의 분포차이가 줄어든 것을 확인할 수 있습니다.

이러한 결과를 정량적으로 위의 표에대해 확인해볼 수 있습니다. 이는 각 벤치마킹에 대한 Error Rate이며, 기존 KD방식에 비해 CIFAR10 Error Rate가 1%정도 낮아진 것을 확인할 수 있습니다. 그 외의 데이터셋에서도 마찬가지로 성능이 향상된 것을 확인할 수 있습니다.

3. Knowledge Transfer with Jacobian Matching

본 논문의 사실 심플한 솔루션이나 방법을 제시하지는 않습니다. 사실상 기존의 원자적 연구들을 조합하여 가장 성능 좋은 모델을 만든 연구이기 때문에 성능은 좋지만, 복잡하고 적용하기 어려운 점이 많습니다. 그럼에도 불구하고 리뷰를 하는 이유는 여태까지 리뷰했었던 논문들의 총 집합체 같은 느낌이기 때문입니다. 본 논문에서 사용한 테크닉은 4가지로 나뉩니다. 첫째, 제목 차원이 다른 경우의 Transfer 문제를 다룬 Fitnet에서의 연구, 둘째, 위에서 리뷰했었던 Attention Transfer를 이용한 연구, 셋째, 그대로 Jacobian을 통해 Knowledge Transfer를 수행한 연구, 넷째, Noise Input을 통한 Model Generalization 연구입니다. 각각의 활용방안은 아래에서 자세히 다루도록 하겠습니다.

위 그림은 본 논문에서 제안하는 Jacobian Matching을 이용한 Knowledge Transfer 방법입니다. 그림에서 Input의 경우 Teacher Network와 Student Network로 들어 갑니다. 이때 Teacher Network의 동작먼저 살펴보면 Teacher Network는 기존의 pre-trained model이며, 기존 방법과 다른점은 새로운 Input(Target Dataset)에 대해 학습을 Online으로 진행한다는 점입니다. 이후 일반적인 Deep Learning 프로세스와 같이 Class Probability Map을 Output합니다. 여기서는 이러한 Map을 Activation Map이라 하였습니다. Student의 경우도 Teacher와 마찬가지로 Input을 받아 Output을 내놓는데, 이때 주의하여 살펴볼 점은 Output이 2-way 인 점 입니다. 첫번 째 방법은 일반적인 학습 방법과 같이 Target Dataset에 대해 Hard Label을 통해 Cross Entropy Loss를 이용하여 학습하는 방법이고, 두번 째 방법은 Teacher의 Output으로 부터 학습하는 방법입니다. 이는 Fitnet에서 사용한 방법과 같습니다. 이때 Fitnet의 경우에는 Student와 Teacher의 Channel이 안맞는 문제를 Regressor를 통해 풀었었는데, 본 논문에서는 Regressor 없이 Teacher와 같은 Channel의 Output을 내놓는다고 하였습니다만 사실상 Fitnet의 Regressor와 다를 바 없는것같습니다. 이 후 Teacher와 Student간의 Activation의 Loss를 구하는데, 이는 Attention Transfer에서 사용하였던 Loss를 그래도 사용하게 됩니다. 이 후 같은 방법으로 Attention에서 Jacobian을 구하는데 이때 Full Jacobian을 구하는 것은 어려운 문제이므로 Approximate하여 구하게 됩니다. 자세한 Approximate 방법은 논문에서 확인 가능합니다.

다음은 Noise Input을 사용하면 Model이 Generalization된다는 연구에 기반해 Noise를 주게 되는데, 이때 Noise는 Jacobian에 섞이게 됩니다. 이러한 방법의 수학적 표기는 위와 같습니다.

이를 좀 더 보기 좋게 시각화 한 그림입니다. 첫번째 그림을 보면 우리에게 어떤 분포에서 샘플링 된 Limited Data Point가 주어졌을 때, 딥러닝의 목적은 Global Approximator를 찾는 것 입니다. 그러나 사실상 이러한 것은 불가능하고, Empirical Approximator가 Global 과 같다고 기대하는 문제로 변형하여 풀게됩니다. 그러나 그림 2와 같은 수많은 Empirical Approximator가 존재할때, 우리는 보다 더 좋은 Approximator를 선택하기 위해 과거에는 3번째 그림과 같이 Noise를 섞는 방법을 통해 접근하였습니다. 그러나 이러한 방법은 범위 내의 여러개의 데이터 포인트 중 하나를 랜덤하게 뽑는것과 같은 효과이기 때문에 저자들은 보다 Slopes를 직접적으로 Matching할 수 있도록 Noise Jacobian Matching을 수행하였습니다. 이를 통해 4번째 그림과 같이 특정 데이터 포인트에서의 범위를 선택할 수 있게되는 효과를 얻게됩니다.

표는 VGG-9에서 VGG-4로의 KD를 수행하는 Ablation Study 결과입니다. 데이터셋은 CIFAR100을 사용하였으며, of Data points per class는 사용한 class별 데이터의 개수를 말합니다. 이때 각 수치는 정확도를 의미합니다. 이때 CE Training은 일반적으로 Cross Entropy를 사용하였을때의 Training 결과고, CE+ match activations는 Student와 Teacher갠의 match만을 사용하였을때의 결과를 얘기합니다. 마찬가지로 match Jacobians는 jacobian matching을 사용하였을때의 결과를 얘기합니다. 이때 예상대로 CE+match{activation + Jacobians}를 모두 했을때 성능이 가장 높게 나왔으며, Hard Label로 학습을했을때와 안했을때는 data points 1개 기준으 로 약 0.7, data points 100개를 기준으로 약 2.4%정도 차이나게 됩니다. 따라서 본 연구에서는 기존의 연구들을 종합하여 가장 좋은 성능을 보여줬습니다. 다만 한계로는 여러 방법을 종합하였기 때문에 원자성이 부족하고 확장 가능성이 약간 떨어진다는 한계를 갖는다고 생각합니다.

4. Deep Mutual Learning

기존 1,2,3번의 KD방법은 어떻게하면 좀 더 KD가 잘될까의 고민이었다면, 4,5,6 번은 Transfer를 잘하는 방법에 대한 고민입니다. 특히 Deep Mutual Learning 논문 저자는 기존의 KD방식은 Deep Network에서 Shallow Network로의 학습이 잘 안되는 점과 KD를 위해 너무 큰 모델이 필요하다는 2가지 문제를 지적하였습니다. 따라서 이러한 문제를 해결하기 위해 Deep Mutual Learning 방식을 제안하였는데, 자세한 내용은 아래서 다루도록 하겠습니다.

Figure15는 Deep Mutual Learning의 방법입니다. 본 논문의 Key 아이디어는 서로 비슷한 규모의 Network간의 Mutual Learning을 수행하는 것 입니다. 이떄 기존의 KD가 Teacher Network에서 Student Network로의 Knowledge Transfer였다면, 본 논문에서 제안하는 Mutual Learning은 상호간의 KD를 수행함으로써 Deep Network에서 Small Network로 Transfer가 잘 안되는 문제와 KD를 위해 큰 Network가 필요했었던 문제점을 해결하게 됩니다. 이때 Loss Lc는 Hard Label과의 Cross Entropy Loss + 다른 네트워크의 분포와의 KL Divergence 의 합으로 정의됩니다.

이때 기존의 Cross Entropy와 약간 다른점이있는데 log(probability) 값 앞에 Indicator Function I 가 곱해진다는 점입니다. 이때 Indicator Function의 정의는 우측 수식과 같이 되어있으며, yi는 prediction된 label을 뜻하고, m은 ground truth를 뜻합니다. 즉 예측한것이 실제 gt와 같을때만 학습하는 것을 의미하게 된다는 것 입니다.

표는 CIFAR100에서의 결과로, 같은 네트워크들간의 혹은 서로 다른 네트워크들간의 DML 학습의 효과를 보여주고 있습니다. 이때 Independent는 각기 기존의 학습 방법대로 학습한 방법이고, DML은 DML만을 사용하여 학습시킨 방법입니다. DML-Independent는 DML로 학습 후 Independent하게 다시 학습했을때의 성능 향상폭입니다. 이때 학습 결과는 큰 이변 없이 Independent하게 학습하는 것 보다는 DML 방식이 더 우수한 성능을 보이고, DML 후 Independent하게 다시 학습하면 성능이 오른것을 확인할 수 가 있습니다.

5. Knowledge Projection for Effective Design of Thinner and Faster Deep Neural Networks

DML에서는 큰 네트워크에서 작은 네트워크로의 학습이 잘 안되는 문제를 작은 네트워크들 간의 Mutual Learning을 통해 풀고자 하였습니다. 추가적으로 큰 네트워크가 필요하지 않다는 이점도 있었습니다. 그러나 이제 나올 4,5번의 경우는 큰 네트워크에서 작은 네트워크로의 학습이 안되는 중간 계층을 두어 학습하는 방법을 통해 직접적으로 풀어보고자 하였습니다.

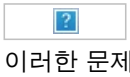
위의 그림의 경우 큰 네트워크에서 작은 네트워크로의 학습이 안되는 문제를 중간 계층을 두어 학습하고자 하였고, 이 중간계층으로 제안한것이 Projection Layer입니다. Projection Layer는 기존의 Domain Adaptation분야에서 적은 데이터만으로도 학습을 진행하기 위한 Transfer Learning의 연구로부터 파생되었습니다. 따라서 본 연구의 Contribution은 Domain과 관련 없는 Domain-invarient Feature를 Teacher에서 Student로 Transfer함으로써 적인 데이터로도 Student를 빠르게 학습시킬 수 있게하였다는 점입니다. 이때 Teacher에서 Student로 Projection Layer를 통해 Transfer된 Feature는 Backpropagation 단계에서 더해져서 학습하게 됩니다.

이때 Projection Layer의 Injection과정 및 Transfer 과정을 자세히 살펴보면 Teacher로부터 Feature Size가 맞는 Convolution Layer의 Output이 Projection Layer로 Injection되게 됩니다. 이 후 Student로 Transfer되는데, 이러한 과정을 Projection Route라 하고 이러한 Route를 선택하기 위해 Iterative Pruning을 사용하였습니다.

표는 Projection Layer의 결과입니다. 본 논문의 목적이 Knowledge Distillation을 통한 Model Compression보다는 Pretrained Teacher로 부터 Student로의 Knowledge Transfer를 목적으로 하였기 때문에 아쉽게도 Compression 측면에서의 지표는 확인할 수 없으나 기존 Fitnet 혹은 Resent Slim과 비교하였을때 메모리도 작을뿐더러, 정확도도 가장 높게 나왔다는것을 확인할 수 있었습니다.

6. Improved Knowledge Distillation via Teacher Assistant: Bridging the Gap Between Student and Teacher

본 논문에서는 Student의 Layer 수를 2개로 구정하고 Teacher의 Layer 수를 4,6,8,10으로 늘려가며 Student의 성능을 확인해보았는데, Teacher가 어떠한 한계를 넘으면 Student에 대해 KD가 잘 수행되지 않는 문제를 발견하였습니다.



이러한 문제를 본 논문에서 제안하는 Teacher Assitant Network의 개념으로 풀어보고자 하였습니다. 이전에 5번에서는 큰 네트워크에서 작은 네트워크로의 학습이 잘 안되는 문제를 Projection Layer라는 Convolution을 도입하여 풀었습니다. 이 논문에서 제안하는 T/A는 이러한 Projectio Layer의 확장 버전이라고 볼 수 있습니다. 단일 Convolution이 아닌 하나의 작은 Netowrk를 중간 계층으로 두어 KD를 수행하는 것으로 Teacher와 Student간의 Capacity Gap을 줄임으로써 Mimicking하기 더 쉽게 만드는 것인데, 이러한 주장을 논문에서는 VC 이론을 통해 수학적으로 왜 잘되는지에 대해 보여주었습니다.

그림은 실제 결과인데, Size의 Column은 Layer로 10,8,6,4,2 로 줄어드는 것을 확인할 수 있습니다. 이때 초록색 원은 10-Layer에서 8,6,4,2 로 각각 직접 KD를 수행하였을때 성능이며, 파란색 원들은 TA를 거쳐 KD를 하였을때의 성능을 나타냅니다. 예를들어 10-Layer인 oval(56.19)에서 6-Layer로으로 KD를 할때 바로 KD를 수행하면 초록색 원인 oval(57.13)이 되는데, 8-Layer를 거쳐갈 경우 oval(57.53)으로 성능이 더 좋은 것을 확인할 수 있습니다. 최종적으로 2-Layer로 KD를 수행할 때 바로 수행할 경우 oval(42.56)의 성능을 보이는데에 비해 모든 TA(10->8->6->4->2)를 거쳐간 케이스의 경우 45.14로 가장 높은 성능을 보여줍니다.

Jacobian Matching 그 이후로...

Knowledge Transfer with Jacobian Matching에서는 이전 논문들에서 KD를 수행할때 잘 되게 만드는 방법론들은 모두 활용하여 좋은 성능을 보여주었습니다. 다만 이러한 방법은 복잡하고, 더 이상의 개선하기가 쉽지 않다는 한계를 가졌는데, 이 이후 나온 클로버팀의 논문에서는 매우 간단한 방법으로 높은 성능향상을 보였습니다[12]. 이는 기존의 Feature Map위주의 KD에서 벗어나 Activation Boundary를 KD하는 방법을 제안하였습니다.

머신러닝 문제에서는 Decision Boundary를 정하는 것은 중요한 이슈였습니다. 최근 연구(Pan and Srikumar et al., 2016)에서는 딥러닝에서의 Decision Boundary가 Activation Boundary의 조합으로 이루어져있다고 주장하였습니다. 따라서 본 논문에서는 이러한 연구에 근거하여 Feature Map으로 사용되는 기존의 Activation Map이 아닌 Activation Boundary를 KD하고자 하였습니다.

기존의 KD에서 사용하는 MSE Loss를 사용할 경우 Strong Response에만 집중하게 되고, 실질적으로 Decision Boundary가 결정되는 Zero Response가 학습이 잘 안될 수 있따는 문제를 지적하였습니다. 따라서 이러한 Zero Response를 학습시키기 위해 activation transfer loss를 제안하였으며, 이를 통해 Zero Response를 학습할 수 있고, 나아가 Deicison Boundary를 학습할 수 있다고 하였습니다.

표는 AB Distillation의 성능표입니다. 저희가 위에서 살펴보았던 다른 방법론들에 비해 매우 심플한 방법으로 월등히 높은 성능을 보여주고있습니다. 특히 Training Data의 퍼센테이지가 낮을때 그 효과를 크게 볼 수 있는데, 실제 결과만 봐도 약 20%가까이 개선된 것을 볼 수 있습니다.

Benchmarking

표는 github에서 KD관련 레파지토리에 나와있는 결과를 비교하기 위해 재구성한 표입니다[11]. 여기에서는 잘 Transfer하는 방법들이 아닌 정확하게 Transfer하는 방법들 위주로 정리하였으며 Teacher와 Student는 같은 모델의 Gap이 작은 모델, 같은 모델의 Gap이 큰 모델 각기 다른 Model간의 케이스에 대해 비교할 수 있도록 재구성하였습니다. 성능지표는 정확도입니다. 저희가 위에서 논문에서 제공하는 밴치마킹으로 비교하였을때처럼 큰 성능차이는 나지 않기때문에 아직 어떤 방법론이 무조건 적으로 좋다고 얘기하기는 어려울 것 같습니다. 따라서 각 도메인에 맞게 적용해보는 연구 또한 필요할 것 같습니다.

KD For Object Detection

그림은 Object Detector인 Yolo v2에 KD를 적용한 논문입니다. 아이디어는 간단합니다. Backbone에서 나온 Faeture Map간의 L2 Loss를 Conv 마다 구하여 학습시키는 방법입니다. 이때 Model을 압축하기 위해 첫번째, 두번째, 세번째 Layer에 대해 BWN(Binary Weight Network)기법을 적용하는데, 이는 Sign Function을 통해 -1과 1로 인코딩하는 방법을 적용합니다. 이는 위에서 살펴본 AB와 유사하다고 볼 수 있습니다. AB의 경우 Teacher의 Decision Boudnary를 Zero인지 아닌지로 정의했다면 본 논문에서 제안하는 BWN은 음수인지 양수인지로 Decision Boundary를 정했다고 볼 수도 있을것같습니다. 본 논문에서는 이러한 KD 방법을 이용하여 약 mAP가 2%낮으나, 28배 가벼운 Detector를 구현하였습니다. 자세한 성능지표는 아래에서 확인할 수 있습니다.

이때 FP는 Full Precision Model이며, M0,M1,M2는 각각 첫번째, 두번째, 세번째 Layer에 대해 BWN이 적용된 Network입니다. 그리고 KT는 M0로부터 초기화되고 Train된 Network입니다. 이때 성능이 약 2% 낮아졌지만 성능 감소 대비 모델의 압축률은 매우 높은 것으로 확인할 수 있습니다.

마치며...

KD분야를 공부하기 위해 조사하면서 자주 언급되는 논문, 혹은 제가 중요하게 본 논문 위주로 KD분야에 대해 리뷰해보았습니다. 저도 공부하면서 작성한 글이기 때문에 부족한 점이나 몇가지 중요한 논문이 빠져있을수도있습니다. 댓글 남겨주시면 공부해서 업데이트하도록 하겠습니다. 또한 누군가에게 튜토리얼 자료로 잘 사용되었으면 좋겠습니다. 감사합니다

Reference

[1]Buciluǎ, Cristian, Rich Caruana, and AlexandruNiculescu-Mizil. “Model compression.” Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2006.
[2]Ba, Jimmy, and Rich Caruana. “Do deep nets really need to be deep?.” Advances in neural information processing systems. 2014.
[3]Hinton, Geoffrey, Oriol Vinyals, and Jeff Dean. “Distilling the knowledge in a neural network.” arXiv preprint arXiv:1503.02531 (2015).
[4]Romero, Adriana, et al. “Fitnets: Hints for thin deep nets.” arXiv preprint arXiv:1412.6550 (2014).
[5]Zagoruyko, Sergey, and Nikos Komodakis. “Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer.” arXiv preprint arXiv:1612.03928 (2016).
[6]Xu, Zheng, Yen-Chang Hsu, and Jiawei Huang. “Training shallow and thin networks for acceleration via knowledge distillation with conditional adversarial networks.” arXiv preprint arXiv:1709.00513 (2017).
[7]Srinivas, Suraj, and François Fleuret. “Knowledge transfer with jacobian matching.” arXiv preprint arXiv:1803.00443 (2018).
[8]Zhang, Ying, et al. “Deep mutual learning.” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018.
[9]Zhang, Zhi, Guanghan Ning, and Zhihai He. “Knowledge projection for deep neural networks.” arXiv preprint arXiv:1710.09505 (2017).

[10]Mirzadeh, Seyed-Iman, et al. “Improved knowledge distillation via teacher assistant: Bridging the gap between student and teacher.” arXiv preprint arXiv:1902.03393 (2019).
[11]https://github.com/HobbitLong/RepDistiller
[12]Heo, Byeongho, et al. “Knowledge transfer via distillation of activation boundaries formed by hidden neurons.” Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 33. 2019.
[13]Xu, Jiaolong, et al. “Training a Binary Weight Object Detector by Knowledge Transfer for Autonomous Driving.” 2019 International Conference on Robotics and Automation (ICRA). IEEE, 2019.

#Knowledge Distillation #Knowledge Transfer #Model Compression