오토인코더를 이용한 CNN 이미지 분류 네트워크의 feature 압축 방안

고성영, 권승욱, *김규헌
경희대학교
kosy0413@khu.ac.kr, ponm9704@naver.com, *kyuheonkim@khu.ac.kr

Compression method of feature based on CNN image classification network using Autoencoder

Sungyoung Go, Seunguk Kwon, *Kyuheon Kim
Kyunghee University

요약

최근 사물인터넷(IoT), 자율주행과 같이 기계 간의 통신이 요구되는 서비스가 늘어감에 따라, 기계 임무 수행에 최적화된 데이터의 생성 및 압축에 대한 필요성이 증가하고 있다. 또한, 사물인터넷과 인공지능(AI)이 접목된 기술이 주목을 받으면서 딥러닝 모델에서 추출되는 특징(feature)을 디바이스에서 클라우드로 전송하는 방안에 관한 연구가 진행되고 있으며, 국제 표준화 기구인 MPEG에서는 '기계를 위한 부호화(Video Coding for Machine: VCM)'에 대한 표준 기술 개발을 진행 중이다. 딥러닝으로 특징을 추출하는 가장 대표적인 방법으로는 합성곱 신경망(Convolutional Neural Network: CNN)이 있으며, 오토인코더는 입력층과 출력층의 구조를 동일하게 하여 출력을 가능한 한 입력에 근사시키고 은닉층을 입력층보다 작게 구성하여 차원을 축소함으로써 데이터를 압축하는 딥러닝 기반 이미지 압축 방식이다. 이에 본 논문에서는 이러한 오토인코더의 성질을 이용하여 CNN 기반의 이미지 분류 네트워크의 합성곱 신경망으로부터 추출된 feature에 오토인코더를 적용하여 압축하는 방안을 제안한다.

1. 서론

최근 사물인터넷(IoT) 및 자율주행과 같이 기계에 각종 센서와 무선 통신 모듈들을 내장하여 기계 간의 통신을 이용한 서비스가 대폭 증가하였다. 최근 보도된 자료에 따르면, 전 세계 IoT 솔루션과 서비스 시장은 2024년까지 연 14.9%의 상승세를 유지할 것이며 사물인터넷 연결 수가 약 250억 개에 달할 것으로 추정했다.[1] IoT 장치들은 서로 인터넷으로 연결되어 데이터를 주고받으며 스스로 분석하여 결론을 내려야 한다. 이처럼 기계가 스스로 판단하고 분석하는 데이터가 증가함에 따라, 기계 임무(task) 수행에 최적화된 데이터를 생성해야 할 필요성이 증가하고 있다.

또한, 사물인터넷에 인공지능(AI)이 접목된 기술이 주목을 받으며,[2] 그림 1과 같이 딥러닝 모델에서 추출되는 특징(feature)을 디바이스에서 클라우드로 전송하는 방안에 관한 연구가 진행되고 있다.[3][4] 이와 같은 추세에 맞추어, 국제 표준화 기구인 MPEG에서는 딥러닝 모델에서 추출된 feature를 압축하는 방안인 '기계를 위한 부호화(VCM)'에 대한 표준 기술 개발을 진행하고 있다.[5]

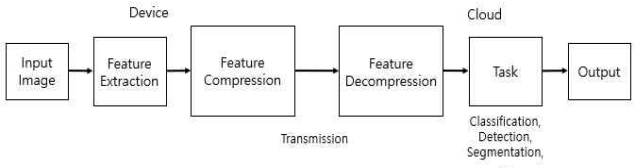

그림 1. 디바이스에서 클라우드로의 feature 전송 구조도

인공지능을 활용하여 feature를 추출하는 방법으로는 합성곱 신경망(Convolutional Neural Network: CNN)이 대표적이다.[6] CNN 이미지 분류 네트워크는 Kernel이라는 2차원 필터로 이미지 전체에 대하여 합성곱 연산을 함으로써 특징을 추출하는 convolutional layer(Conv layer)와 feature 상의 특정 크기의 영역 내에서 하나의 값만을 선택하여 feature의 크기를 줄이는 pooling layer, 추출된 feature를 통하여 이미지를 분류하는 fully connected layer(FC layer)로 구성된다.

본 논문에서는 CNN 이미지 분류 네트워크에서 추출된 feature를 딥러닝을 활용한 이미지 압축 방법인 오토인코더를 이용하여 압축하는 방안에 대하여 제안하고자 한다. 본 논문의 2장에서는 CNN을 활용한 모델 중의 하나인 VGG16과 오토인코더를 이용하여 이미지를 압축하는 방법을 간단히 살펴보고, 3장에서는 오토인코더를 이용한 feature 압축 방안에 대하여 설명한다. 4장에서는 앞선 방법으로 압축된 feature를 통한 classification을 통해 제안한 방법을 검증하고, 5장에서는 결론을 맺는다.

2. 배경 기술 분석

2.1 VGG16

Conv layer는 필터의 크기와 padding, stride라는 값을 조절하여

출력의 크기를 조절할 수 있다. Padding은 입력 데이터의 외곽에 채워 넣을 픽셀의 수를 의미하며, Stride는 필터가 한 번 이동할 때 이동하는 간격을 의미한다. 수식 1은 Conv layer를 거쳐 나온 출력 데이터의 크기를 나타내는 수식이다. O는 출력 이미지의 크기, I는 입력 이미지의 크기, P는 padding, F는 필터의 크기, S는 stride이다.

$$O = \frac{I + 2P - F}{S} + 1$$

VGG16은 224×224 크기의 RGB 이미지를 13개의 Conv layer와 5개의 최대 풀링(Max pooling) layer 사용하여 이미지의 feature를 추출한다.[7] Max pooling은 특정 크기의 영역 내에서 최댓값을 선택하여 데이터를 줄이는 역할을 한다. 3×3 크기의 필터를 사용하고 padding과 stride를 1로 설정하여 Conv layer에서는 크기 변화 없이 채널 수에만 변화를 주고, 2×2 Max pooling을 5번 거치며 3×224×224 크기의 이미지를 512×7×7 크기의 feature로 추출한다. 추출된 feature는 3개의 FC layer를 통과하여 classification을 수행한다.

2.1 오토인코더

오토인코더는 입력층과 출력층의 구조를 같게 하여 가능한 한 적은 왜곡으로 출력을 입력과 유사하게 하고, 은닉층의 크기를 그보다 작게 하여 차원을 축소함으로써 데이터를 압축하는 비지도 학습 네트워크이다.[8][9] 비지도 학습이란 데이터에 대한 기댓값을 부여하는 라벨링 등의 전처리 없이 학습하는 과정을 의미한다. 오토인코더는 손실함수를 입력과 출력의 차이를 통하여 계산하므로, 데이터에 대한 별도의 전처리가 없어도 학습이 가능하다. 출력이 입력과 비슷해지도록 학습하되, 은닉층의 차원이 입력층보다 작게 구성되어 입력과 출력이 같아지는 것은 불가능하므로 데이터를 압축하면서 핵심적인 특징을 학습하게 된다. 오토인코더를 이미지에 적용하여 압축하면 이미지 품질은 감소하지만, 주요한 특징에 대한 부분은 손실되지 않는다.[10]

오토인코더의 구조는 그림 2와 같다. 입력층으로부터 은닉층까지의 과정을 인코더(encoder), 은닉층으로부터 출력층까지를 디코더(decoder)라고 한다.

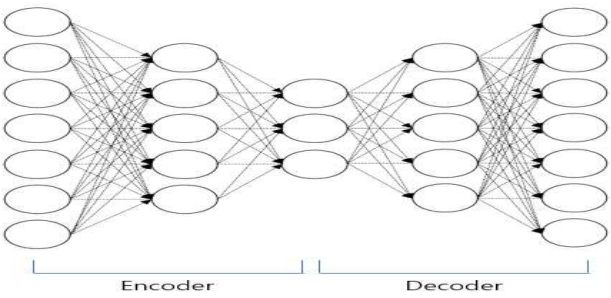

그림 2. 오토인코더의 구조

오토인코더의 종류는 크게 두 가지가 있다. 하나는 인코더를 convolutional layer로 구성하고 디코더는 그 역과정인 Transposed convolutional layer(Transposed Conv layer)로 구성하는 Convolutional Autoencoder(CAE)이고, 다른 하나는 입력층과 출력층을 이미지 크기와 같게 일렬로 재배치(reshape)하고 입력층, 은닉층, 출력층의 3개의 layer를 FC layer로 구성하는 Simple Autoencoder(SAE)이다. 연구 결과에 따르면, CAE를 사용하여 이미지를 압축할 때 SAE를 사용할 때보다 더 적은 파라미터를 사용하고도 더 적은 정보 손실을 보인다.[11]

3. 오토인코더를 이용한 feature 압축

앞서 설명한 바와 같이, CNN으로부터 추출된 feature는 합성곱 연산과 pooling 과정의 단순 연산만을 거쳐 추출되므로 데이터에 대한 기댓값이 존재하지 않는다. 따라서 비지도 학습의 데이터로 사용하기에 적합하며, 오토인코더는 대표적인 비지도 학습 방법으로써 데이터 압축이 가능하다. 이에 본 논문에서는 오토인코더를 이용한 feature 압축 방안에 대하여 제안하고자 한다.

제안 기술은 CNN의 한 모델인 VGG16을 backbone 네트워크로 사용한다. VGG16으로 추출한 feature를 입력으로 하는 오토인코더를 학습하는 과정에서 기존 Conv layer와 FC layer에 영향을 미치는 것을 방지하기 위해 Conv layer와 FC layer를 분리하여 학습시킨 후에 Conv layer를 거쳐 추출된 feature를 입력으로 하는 오토인코더를 작성하였으며, 오토인코더를 학습시킬 때에는 Conv layer와 FC layer를 제외한 오토인코더의 파라미터만을 학습시켰다. 2장에서 언급한 연구결과에 따라, 오토인코더는 CAE의 구조를 사용하였다.

VGG16으로부터 추출된 512×7×7 크기의 feature를 압축하기 위해 순서대로 1×1 Conv layer를 2번, 3×3 Conv layer를 1번을 사용하여 인코더를 구성하였다. 1×1 Conv layer의 경우 stride를 1, padding을 0으로 설정하여 feature의 너비와 높이는 유지하되 채널 수를 반감하도록 작성하였고, 3×3 Conv layer의 경우 stride를 1, padding을 0으로 채널 수는 유지하되 너비와 높이를 2만큼 감소시키도록 작성하였다. 그 결과, 오토인코더의 인코더를 거치면 512×7×7 크기의 feature의 크기가 128×5×5로 압축된다. 압축된 feature는 인코더의 역순으로, 3×3 Transposed Conv layer를 1번, 1×1 Transposed Conv layer를 2번 사용하여 구성된 디코더를 거쳐 원래의 크기인 512×7×7로 복원된다.

이후 복원된 feature를 통하여 classification을 수행하여 정확도를 측정하고, 압축 대비 성능을 검증하기 위해 기존의 feature로 수행한 classification 정확도와 비교한다. 압축률은 VGG16으로부터 추출된 feature와 오토인코더의 인코더로 압축한 feature의 크기를 비교함으로써 계산한다.

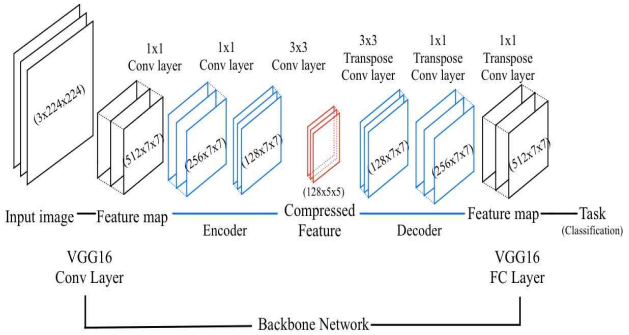

그림 3. 오토인코더를 이용한 feature 압축 방안 구조도

4. 실험 결과

본 논문에서는 CNN으로부터 추출된 feature를 오토인코더를 사용하여 압축하는 방법을 제안하고 있다. CNN 모델로는 VGG16을 사용하였으며, 본 논문의 4장에서는 압축 대비 성능을 평가하기 위해 CNN으로부터 추출된 feature와 오토인코더로 압축 후 복원된 feature로 각각 수행한 classification의 정확도를 비교한다. 성능 평가를 위한 실험은 Linux 18.04, RTX 2060의 GPU와 Jupyter notebook 환경에서 제작되었다.

데이터셋은 10개 class에 대하여 32×32 크기의 이미지를 각각 6000장씩 포함한 CIFAR10 데이터셋을 사용하였고 training에는 1epoch당 12500장, test에는 2500장을 사용하였으며, batch size는 4로 설정하였다. Optimizer와 learning rate는 VGG16에 대하여는 Adam과 0.00001, 오토인코더에 대하여는 MSE와 0.0001을 사용하였다. 학습은 VGG16과 오토인코더에 대하여 각각 100 epoch, 10 epoch씩 진행하였다.

Class	Accuracy (%) (VGG16)	Accuracy (%) (VGG16 + Autoencoder)	Difference of accuracy
Plane	87	87	-
Car	95	96	+1
Bird	75	69	-6
Cat	60	55	-5
Deer	78	75	-3
Dog	75	79	+4
Frog	91	91	-
Horse	87	82	-5
Ship	89	82	-7
Truck	87	82	-5
Average	**82.4**	**79.8**	**-3.16%**

표 1. VGG16과 제안 방법으로 수행한 classification 정확도

3장에서 설계한 오토인코더로 VGG16으로부터 추출한 feature를 오토인코더로 압축한 결과는 표 1과 같다. 512×7×7의 feature를 128×5×5로 압축하여 87.2%의 압축률을 보였고, 압축된 feature를 복원하여 classification을 수행한 결과 10개 class의 객체에 대하여 평균 -3.16%의 성능 저하를 보였다.

5. 결론

본 논문에서는 VGG16 모델을 사용한 CNN 이미지 분류 네트워크에 대하여 Conv layer로부터 추출된 feature를 압축하는 방안에 대하여 제시하고 classification을 통해 성능을 검증하였다. 그 결과 feature가 87.2%의 압축률을 보인 것에 비해, 성능은 3.16% 감소하여 압축률 대비 성능 저하가 크지 않은 것으로 나타났다.

2장에서 언급한 바와 같이 오토인코더로 이미지를 압축할 경우, 이미지 품질은 감소하지만 특징적인 부분은 손실되지 않는다. 이를 근거로 오토인코더를 이용하여 feature를 압축할 경우 압축된 feature를 디코더로 복원하여 classification으로 검증할 시, 성능 저하가 발생할 수 있으나 그 폭이 크지 않을 것으로 예상되었다. 실험 결과를 통하여 이를 확인하였지만, 이미지 class 별로 정확도가 오히려 증가하거나 변하지 않는 경우가 존재했다. 이는 이미지 class 수가 10개로 적은 것이 원인일 것으로 예상하며, class의 개수를 증가시킴으로써 극복할 수 있을 것으로 기대한다.

이 논문은 2020년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No. 2020-0-00011, (전문연구실)기계를 위한 영상부호화 기술)

참고문헌

[1] “IoT Solutions and Services Market by Component (Platform, Solution And Services), Service (Consulting, and Integration and Deployment), Vertical (Smart Manufacturing, Smart Energy and Smart Transportation), and Region”, Global Forecast to 2024, 2020.
[2] “[CES2020]Bosch, ‘AI in all products by 2025’”, etnews, accessed Jan 7, 2020, https://m.etnews.com/20200107000231
[3] H. Unnibhavi, et al. “DFTS: Deep Feature Transmission Simulator”, IEEE Multimedia Signal Processing Workshop (MMSP), Aug, 2018.
[4] Zhuo Chen, et al. “Toward Intelligent Sensing: Intermediate Deep Feature Compression”, IEEE Transactions on Image Processing, Vol.29, 2020.
[5] Ling-Yu Duan, “Video Coding for Machines: A Paradigm of Collaborative Compression and Intelligent Analytics”, Computer Vision and Pattern Recognition (cs.CV), Jan, 2020.
[6] A Krizhevsky, I Sutskever, GE Hinton, “ImageNet Classification with Deep Convolutional Neural Networks”, Communications of the ACM, 2017.
[7] K. Simonyan, A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition”, arXiv 1409.1556, 2014.
[8] Y. Yaginuma, T. Kimoto and H. Yamakawa, "Multi-sensor fusion model for constructing internal representation using autoencoder neural networks," Proceedings of International Conference on Neural Networks (ICNN'96), pp. 1646-1651 vol.3, 1996.
[9] Pierre Baldi, “Autoencoders, Unsupervised Learning, and Deep Architectures”, Journal of Machine Learning Research(Proc. 2011 ICML Workshop on Unsupervised and Transfer Learning) 27, pp.37-50, 2012.
[10] L. Theis, et al. “Lossy image compression with compressive autoencoders”, Int. Conf. on Learning Representations (ICLR), 2017.
[11] Yifei Zhang, “A Better Autoencoder for Image: Convolutional Autoencoder”, International Conference on Neural Information Processing 2017, 2018.