Journal of the KIMST, Vol. 24, No. 4, pp. 382-391, 2021
ISSN 1598-9127(Print) · 2636-0640(Online)
DOI https://doi.org/10.9766/KIMST.2021.24.4.382

Research Paper 센서·신호처리 부문

자기 정규화를 통한 도메인 불변 특징 학습

현재국*,1) · 이찬용1) · 김호성1) · 유현정1) · 고은진1)

1) 국방과학연구소 미사일연구원

Learning Domain Invariant Representation via Self-Rugularization

Jaeguk Hyun*,1) · ChanYong Lee1) · Hoseong Kim1) · Hyunjung Yoo1) · Eunjin Koh1)

1) *The 1st R&D Institute, Agency for Defense Development, Korea*

(Received 4 March 2021 / Revised 6 May 2021 / Accepted 4 June 2021)

Abstract

Unsupervised domain adaptation often gives impressive solutions to handle domain shift of data. Most of current approaches assume that unlabeled target data to train is abundant. This assumption is not always true in practices. To tackle this issue, we propose a general solution to solve the domain gap minimization problem without any target data. Our method consists of two regularization steps. The first step is a pixel regularization by arbitrary style transfer. Recently, some methods bring style transfer algorithms to domain adaptation and domain generalization process. They use style transfer algorithms to remove texture bias in source domain data. We also use style transfer algorithms for removing texture bias, but our method depends on neither domain adaptation nor domain generalization paradigm. The second regularization step is a feature regularization by feature alignment. Adding a feature alignment loss term to the model loss, the model learns domain invariant representation more efficiently. We evaluate our regularization methods from several experiments both on small dataset and large dataset. From the experiments, we show that our model can learn domain invariant representation as much as unsupervised domain adaptation methods.

Key Words : Domain Adaptation(도메인 적응), Domain Generalization(도메인 일반화), Style Transfer(스타일 이전), Convolutional Neural Nework(컨볼루션 뉴럴 네트워크)

1. 서 론

풍부한 데이터셋을 동일한 학습 데이터와 시험 데이터로 구분한 후 딥 뉴럴 네트워크(Deep Neural Network, DNN)를 이용하여 데이터의 분포를 학습 할 경우 대부분 매우 높은 시험 정확도(test accuracy)를 보인다. 사전 학습된(pre-trained) 네트워크를 전이 학습(transfer learning)에 이용할 경우 적은 데이터로도 충분한 시험 정확도 성능을 발휘한다. 하지만 학습 데

* Corresponding author, E-mail: hyunjk@add.re.kr

이터와 시험 데이터의 분포가 상이한 경우 도메인 변화(domain shift)에 대해서는 딥 뉴럴 네트워크의 성능이 현저히 감소한다. 도메인 변화에 따른 딥 뉴럴 네트워크 성능 저하 문제를 해결하기 위해 현재 연구되는 분야는 크게 두 가지이다. 첫 번째는 도메인 적응(domain adaptation) 분야로 시험 데이터와 동일한 도메인의 데이터를 학습 데이터와 함께 학습하여 학습 도메인에 편향되지 않는 특징을 검출하는 방법이다. 이 때 시험 데이터와 동일한 도메인의 데이터는 일반적으로 비지도(unsupervised) 학습 방법이 적용된다. 두 번째는 도메인 일반화(domain generalization) 방법이다. 도메인 일반화 방법은 시험 데이터와 동일한 도메인의 데이터에는 접근할 수 없지만, 제 3의 도메인의 데이터를 이용하여 특정 도메인에 편향되지 않는 특징을 검출하도록 학습하는 방법이다.

도메인 적응 분야와 도메인 일반화 분야에서 이뤄진 연구들은 도메인 변화 문제를 비교적 잘 해결하지만 각 분야에서 사전에 정의된 형식들을 현실에 바로 적용하는 데에는 문제가 발생할 수 있다. 예를 들어 데이터의 수집이 매우 어려워 가상의 데이터를 생성하여 활용하는 경우, 시험에 사용될 소수의 실 데이터 외에는 또 다른 데이터가 없을 수 있다. 이런 경우 도메인 적응을 위한 시험 데이터와 동일 도메인의 비지도 데이터 혹은 도메인 일반화를 위한 제 3의 도메인의 데이터가 없기 때문에 위에 언급한 연구 방법론들을 적용하기가 곤란하다.

따라서 본 연구에서는 오직 학습 도메인의 데이터만 존재할 때, 학습 도메인의 데이터에 과적합(over fitting)되지 않도록 모델을 학습하는 방법을 제시한다. 본 연구에서 제시한 문제는 기존의 연구방법론과는 달리 학습 데이터와 시험 데이터간의 매개 역할을 하는 데이터가 없기 때문에 학습 단계에서 추가적인 자기-정규화(self-regularization) 단계가 필요하다. 이에 대해 본 연구에서는 두 가지의 자기-정규화를 제시한다.

첫 번째는 임의 스타일 변환(arbitrary style transfer)을 통한 픽셀 정규화이다. 이미지넷(ImageNet)[2]을 통해 사전 학습된 컨볼루션 뉴럴 네트워크(Convolutional Neural Network, CNN)의 경우 데이터의 모양 보다는 데이터의 질감(texture)에 편향되어 있음이 Geirhos[3]로부터 지적되었다. 데이터의 질감의 경우 도메인 변화 시 공유되기 어려운 특성이므로 모델 학습 시 모델이 학습 데이터의 질감에 편향되지 않도록 학습해야한다. 픽셀 정규화를 통해 매 학습 때마다 데이터의 질감을 임의의 스타일로 변환함으로써 모델은 특정 질감에 편향되지 않은 데이터의 특징들을 학습할 수 있게 된다. 학습 데이터의 질감을 매 학습 때마다 변환하기 위해서는 임의 스타일 변환 알고리즘이 두 가지 조건을 만족해야한다. 학습 시 매 스텝마다 스타일을 변환해야하기 때문에 처리 속도가 매우 빨라야 하며, 다음으로는 데이터의 질감 외의 공간 정보는 유지해야한다. 본 연구에서는 위 두 조건을 만족하는 임의 스타일 변환 알고리즘을 제시한다.

두 번째 자기 정규화 과정은 특징 정규화(feature regularization)이다. 특징 정규화는 픽셀 정규화된 데이터를 학습하는 모델에게 올바른 학습 방향을 제시하는 역할을 한다. 특징 정규화항이 추가된 손실 함수를 역전파(back propagation)하는 과정을 통해 모델은 특정 도메인에 편향되지 않는 특징을 추출하는 법을 학습한다.

본 연구에서는 규모가 작은 데이터셋과 규모가 큰 데이터셋에서의 실험을 통해 자기 정규화를 적용한 학습모델이 비지도 도메인 적응 기법을 적용한 학습 모델과 유사한 성능을 보이거나 보다 우수한 성능을 내는 점을 확인하였다. 규모가 작은 데이터로는 숫자 분류 데이터들(MNIST-M[1], SVHN[24], MNIST[28])을 사용하였으며, 규모가 큰 데이터로는 Visda2017-C[4] 데이터를 사용하였다.

2. 이론적 배경

2.1 도메인 변화

도메인 변화는 머신 러닝 모델이 학습할 때 적용된 학습 도메인의 데이터($(x_s, y_s) \in D_s$)와 학습된 모델이 시험할 때 쓰인 시험 도메인 데이터($(x_t, y_t) \in D_t$)간의 차이가 발생한 현상을 의미한다. 학계에서 통용되는 학습/시험 데이터 분할 방식은 전체 데이터 중 특정 부분을 시험 데이터로 분류하고 나머지를 학습 데이터로 지정하기 때문에 학습 도메인의 데이터와 시험 도메인의 데이터 간의 차이가 없지만 현실에서는 학습 시 사용하는 데이터와 학습된 모델이 적용되는 데이터간의 도메인 격차가 큰 경우가 종종 발생한다. 예를 들면, 유럽의 자율주행 데이터셋으로 학습을 한 모델을 바로 한국의 도로에서 획득된 데이터에 적용을 했을 때 예상했던 성능보다 현저히 떨어진 성능을 마주하게 된다. 이러한 도메인 변화에 따른 모델 성능의

하락 현상을 해결하기 위해 사전 연구들은 학습 도메인의 데이터를 학습 시 다양한 매개 데이터를 같이 학습함으로써 모델의 학습 도메인에서의 성능과 시험 도메인에서의 성능 차를 줄이고자 했다[1,5,11].

매개 데이터로 가장 많이 사용되는 데이터는 시험 도메인과 동일한 도메인의 비지도 데이터($(x_t, \not{y_t}) \in D_t$)이다. 비지도 데이터란 데이터의 정답이 주어지지 않은 데이터를 말한다. 시험 도메인과 동일한 도메인의 비지도 데이터를 이용하여 학습 도메인과 시험 도메인간의 격차를 줄이려는 방법론을 비지도 도메인 적응(Unsupervised Domain Adaptation, UDA)이라고 한다. 비지도 도메인 적응 방법은 매개 데이터의 활용 방법에 따라 크게 두 가지로 분류된다.

첫 번째로 매개 데이터에서 추출된 특징 벡터와 학습 데이터에서 추출된 특징 벡터간의 차이를 줄이는 방법이다. Ganin[1]은 입력된 특징 벡터들이 어느 도메인에 속해있는지를 판별하는 도메인 판별모델을 도입하여 처음으로 적대적 학습 방법을 이용한 비지도 도메인 적응 문제의 해결 방법을 제시하였다. Ganin[1]은 실제 학습 모델의 경우 도메인 판별 모델의 미분방향의 역으로 학습을 시키는 방법을 통해 특정 도메인에 편향되지 않으며, 학습 도메인에서 성능이 좋은 특징을 추출하는 학습하는 방법을 소개하였다. Tzeng[5]는 Ganin[1]의 모델구조를 일반화 하여 보다 효과적인 적대적 학습 방법을 제안하였다.

비지도 시험 도메인 매개 데이터를 이용하는 두 번째 방법은 적대적 생성 모델(Generative Adversarial Network, GAN)[6]을 이용하여 학습 도메인의 데이터를 시험 도메인의 데이터로 변환하는 것이다. Bousmalis[7]는 적대적 생성 모델을 이용하여 학습 도메인의 데이터를 시험 도메인의 데이터로 변환시킨 데이터와 본래 학습데이터를 모두 학습하여 시험 도메인에 좋은 성능을 내는 모델을 학습하는 방법을 제안했다. Hoffman[8]은 사이클 손실 함수(cycle consistent loss) 및 데이터 의미 손실 함수(semantic consistent loss)를 추가하여 시험 도메인에서의 모델 성능을 향상시켰다.

비지도 도메인 적응 기법은 다수의 시험 도메인 매개 데이터를 이용하여 학습 데이터와 시험 데이터에서의 모델 성능 차를 줄이는 데 큰 공헌을 하였다. 하지만 현실에서는 시험 도메인의 비지도 데이터의 수집이 어려운 경우가 종종 발생한다. 이를 해결하고자 시험 도메인의 소수의 지도 데이터($(x_t, y_t) \in D_t$)를 매개 데이터로 활용한 방법론들이 제시되었다. Motiian[9]에서 제시한 방법은 학습 도메인에 대해 모델을 한 번 학습한 후 해당 모델을 이용하여 도메인 판별자와 적대적으로 학습한다. 이후 모델을 학습 도메인과 시험 도메인 그리고 학습된 도메인 판별자 모두에 대해 학습하여 학습 도메인 데이터와 시험 도메인 데이터에 대해 성능 차가 없도록 한다. Xu[10]은 확률적 이웃 임베딩(Stochastic Neighbor Embedding, SNE)을 이용하여 효과적인 소수의 시험 도메인 데이터를 활용하는 방법을 제안하였다.

소수의 시험 도메인의 지도 데이터를 사용하여 도메인 변화 문제를 해결하는 방법은 접근가능한 시험 도메인의 데이터가 적을 때에도 도메인 적응 기법을 사용할 수 있게 했으나 여전히 현실적인 문제점을 갖고 있다. 즉 학습에 사용될 수 있는 시험 도메인의 데이터가 존재하지 않을 때에는 해당기법들을 사용할 수 없다. 학습 시 시험 도메인의 데이터를 사용하지 못하는 경우 여러 개의 제 3의 도메인의 데이터를 활용하여 특정 도메인에 편향되지 않은 특징을 학습하는 방법론을 도메인 일반화라고 한다. Li[11]는 컨볼루션 뉴럴 네트워크를 이용한 도메인 일반화를 제시한 논문으로써 학습 도메인과 매개 도메인 어디에도 편향되지 않은 모델 학습법을 소개했다. Balaji[12]은 메타 러닝(meta learning)기반의 최적화 방법을 이용하여 스스로 학습 도메인과 매개 도메인에 편향되지 않도록 하는 정규화 방법을 학습하도록 설계하였다.

비지도 도메인 적응 방법론 혹은 도메인 일반화 방법론에서 요구하는 매개 데이터는 모두 학습 도메인과 시험 도메인의 라벨 분포(label distribution)를 공유해야한다. 하지만 데이터 수집이 어려운 특수한 라벨 분포를 갖고 있는 학습 도메인 데이터와 시험 도메인 데이터의 경우, 두 방법론을 적용하기 위한 특정 라벨 분포를 공유하는 매개 데이터의 수집이 불가능하게 된다. 본 연구에서는 원하는 라벨 분포를 공유하는 매개 데이터가 없는 상황에서 인터넷에서 쉽게 구할 수 있는 데이터셋을 이용하여 특정 도메인에 편향되지 않은 특징을 학습하는 방법을 소개하고자 한다.

2.2 스타일 변환

Gatys[13]는 컨텐츠 이미지와 스타일 이미지의 그램 행렬(Gram Matrix)을 맞춤으로써 컨텐츠 이미지의 스타일을 스타일 이미지의 스타일로 변환하는 방법을 제시했다.

Johnson[14]은 스타일 변환된 결과 자체를 학습함으

로써 Gatys[13]에 비해 훨씬 빠른 추론 속도를 낼 수 있음을 보였다. Johnson[14]은 스타일 변환 속도에서는 비약적인 발전을 보였지만 변환하고자 하는 스타일 이미지가 바뀔 때마다 스타일 변환 네트워크를 매번 학습해야한다는 단점이 존재했다. 이를 해결하기 위해 Huang[15]는 컨텐츠 이미지의 평균과 분산을 스타일이미지의 평균과 분산으로 매칭하는 AdaIN(Adaptive Instance Normalization) 방법을 통해 그램 행렬의 일치를 통한 스타일 변환과 유사한 효과를 낼 수 있다는 점을 제안하였다. Li[16]는 컨텐츠 이미지와 스타일 이미지의 평균과 공분산을 매칭시키는 방법 역시 효과적인 스타일 변환이 가능함과 이를 통해 학습하지 않은 스타일도 변환이 가능함을 보였다.

Huang의 방법과 Li의 방법은 빠르게 컨텐츠 이미지의 스타일을 다양한 스타일 이미지의 스타일로 변환할 수 있지만 스타일 변환 과정에서 컨텐츠 이미지 본래의 공간 정보에 손실이 발생하게 된다. 본 연구에서는 임의 스타일 변환된 이미지를 딥러닝 모델의 학습 데이터로 사용하기 때문에 스타일 변환과정에서 공간 정보 손실이 발생해서는 안된다. 이에 이미지 스타일 변환 속도를 유지하면서 공간 정보 손실을 막는 Unpooled AdaIN을 제시한다.

3. 자기 정규화 방법

3.1 픽셀 정규화

픽셀 정규화는 임의 스타일 변환을 통해 학습 데이터의 질감을 무작위로 변환함으로써 모델이 학습할 때 학습 데이터의 질감에 과적합하는 현상을 방지하는 방법이다. 픽셀 정규화에 사용될 임의 스타일 변환 알고리즘은 학습 속도에 지장이 되지 않을만큼 빨라야하며, 학습 성능에 지장이 되지 않도록 본래의 학습 데이터의 공간 정보를 유지해야한다. 본 연구에서는 이 두 가지 조건을 모두 만족하는 임의 스타일 변환 알고리즘인 Unpooled AdaIN을 제시한다. Unpooled AdaIN은 Huang[15]에서 제시한 AdaIN 알고리즘을 개선한 알고리즘으로써 AdaIN 알고리즘의 속도를 유지하면서 데이터의 공간 정보를 유지할 수 있다. Unpooled AdaIN은 기존의 AdaIN의 디코더의 업샘플링 레이어(Upsampling layer)를 언풀링 레이어(Unpooling layer)[17]로 대체하였다. 또한 언풀링 레이어는 업샘플링 레이어와는 달리 인코더의 맥스풀링 레이어(Maxpooling layer)에서 풀링된 부분의 위치정보를 필요로 하므로 AdaIN의 인코더에 맥스풀링 마스크를 추가하여 해당 정보를 전달하였다. Fig. 1은 Unpooled AdaIN의 학습 과정을 나타낸 그림이며 Unpooled AdaIN의 인코더 f와 디코더 g의 손실 함수는 다음과 같이 정의된다.

$$t = \sigma(f(I_s))\left(\frac{f(I_c)-\mu(f(I_c))}{\sigma(f(I_c))}\right)+\mu(f(I_s)) \tag{1}$$

$$\begin{aligned} L_c &= \| f(g(t))-t \|_2 \\ L_s &= \sum_{i=1}^{l} \| \mu(\phi_i(g(t)))-\mu(\phi_i(I_s)) \|_2 + \\ &\quad \sum_{i=1}^{l} \| \sigma(\phi_i(g(t)))-\sigma(\phi_i(I_s)) \|_2 \end{aligned} \tag{2}$$

$$L_{AdaIN} = L_c + \lambda_s L_s \tag{3}$$

식 (1)은 컨텐츠 이미지 I_c의 특징 벡터의 평균과 분산을 스타일 이미지 I_s의 특징벡터의 평균과 분산으로 일치시키는 과정을 나타낸 식이다. 식 (1)의 결과인 t는 컨텐츠 이미지의 공간정보와 스타일 이미지의 스타일 정보를 갖고 있는 특징벡터이다. L_c와 L_s는 각각 컨텐츠 손실함수와 스타일 손실함수를 나타내며, $\mu(\cdot)$, $\sigma(\cdot)$는 각각 채널 방향의 평균과 표준편차를 의미한다. 식 (2)에서 ϕ_i는 인코더 f의 컨볼루션 레이어들을 의미한다. 식 (3)에서 λ_s는 컨텐츠 손실함수와 스타일 손실함수의 가중치를 조절하는 하이퍼파라미터이다. COCO 데이터셋[18] 과 Wikiart[27] 데이터셋으로 사전 학습된 Unpooled AdaIN은 학습 도메인의 데이터들의 질감을 다양하게 만들어줌으로써 딥러닝 모델이 특정 질감에 편향된 특징을 추출하는 것을 방지할 수 있다.

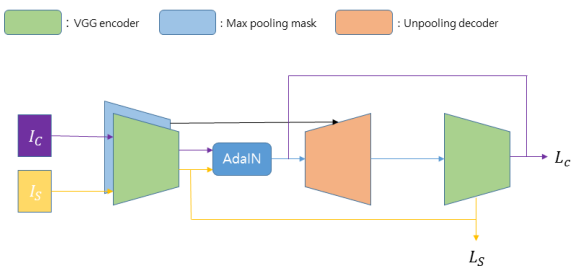

Fig. 1. Training procedure of Unpooled AdaIN

3.2 특징 정규화

픽셀 정규화된 데이터로 학습한 딥러닝 모델의 경우 학습 데이터 도메인의 질감에 편향된 특징을 추출하지 않는다. 픽셀 정규화 이전의 데이터와 픽셀 정규화 이후의 데이터는 질감을 제외한 나머지 특성은 공유하고 있다. 따라서 데이터의 질감에 강인한 모델은 두 데이터에 대해 유사한 특징 벡터를 추출해야한다. 이를 명시한 특징 정규화항을 추가하여 학습한 모델은 보다 더 질감의 편향성에 대응하는 강인한 특징을 검출할 수 있게 된다. 3.1에서 학습한 인코더 f와 디코더 g를 이용하여 학습 데이터 $x_s \in D_s$를 임의의 스타일 이미지 I_s로 픽셀 정규화된 학습 데이터를 x_s'는 다음과 같다.

$$x_s' = g(\sigma(f(I_s))(\frac{f(x_s)-\mu(f(x_s))}{\sigma(f(x_s))})+\mu(f(I_s))) \quad (4)$$

이 때, 특징 검출기(feature extractor) F 대해 특징 정규화는 다음과 같이 정의할 수 있다.

$$L_{reg} = mse(F(x_s),\ F(x_s')) \quad (5)$$

픽셀 정규화된 입력값 x_s'와 픽셀 정규화 이전의 x_s로부터 추출한 특징 벡터간의 차이를 줄이는 방향으로 최적화하는 과정을 통해 특징 검출기 F는 입력값의 질감에 보다 더 불변하는 특징을 검출하는 방법을 학습할 수 있다. 두 특징 벡터간의 차이는 평균 제곱 오차(mean squared error, MSE)로 정의하였다. 추출된 특징을 분류하는 특징 분류기(classifier) G와 함께 정의되는 딥러닝 모델 $M = G \circ F$의 손실 함수는 다음과 같다.

$$L_{task} = -\sum_{i}^{C} y_i \log(G(F(x_s'))_i)$$
$$L_M = L_{task} + \lambda_{reg} L_{reg} \quad (6)$$

식 (6)의 λ_{reg}는 특징 정규화 손실 함수와 테스크 손실 함수간의 영향력을 조절하는 하이퍼 파라미터이다.

Fig. 2는 본 연구에서 제안하는 자기 정규화 과정의 전체적인 과정을 나타낸 그림이다. 이 때 사용되는 Unpooled AdaIN의 경우 사전에 학습된 모델로 특징 정규화 과정에서는 역전파 하지 않는다.

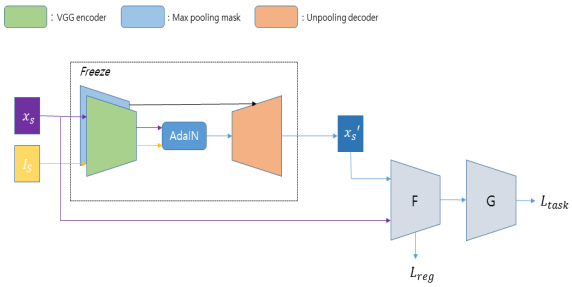

Fig. 2. Process of self-regularization

4. 실험 및 평가

자기 정규화를 통해 학습한 딥러닝 모델의 성능을 평가하기 위해 본 연구에서는 비지도 도메인 적응 분야에서 활용되는 데이터셋들을 활용하였다. 다양한 분석을 위해 규모가 작은 데이터셋과 규모가 큰 데이터셋 모두에서 실험 및 평가를 진행하였다. Fig. 3은 본 연구에서 사용한 데이터셋의 예시이다.

Fig. 3. Examples of datasets

4.1 Unpooled AdaIN 학습 방법

픽셀 정규화를 진행하기 위해서는 사전에 학습된 Unpooled AdaIN 모델이 있어야한다. 본 연구에서는 컨텐츠 이미지로는 COCO 데이터셋을 사용하였으며, 스타일 이미지로는 Wikiart 데이터셋을 사용했다. 모델 최적화를 위해 Adam[28] 옵티마이저(optimizer)를 이용했으며, 학습률(learning rate)은 0.0001로 설정하였다. 인코더는 이미지넷으로 사전 학습한 VGG 네트워크[19]를 사용했으며, Unpooled AdaIN 학습 시에는 인코더의 변수들은 역전파 하지 않았다. 디코더는 VGG 네트워크 구조를 역으로 하여 설계했다. 이 때, 맥스 풀

링 레이어의 역으로는 업 샘플링 레이어가 아닌 언 풀링 레이어를 사용하였다. 모델 학습은 배치 사이즈(batch size)를 16으로 하여 100,000 스텝을 진행하였다. λ_s의 경우 10으로 설정하였지만, 다수의 실험을 통해 λ_s의 영향력은 크지 않음을 알 수 있었다.

Fig. 4는 학습된 Unpooled AdaIN 모델과 AdaIN 모델을 MNIST 데이터셋과 Visda2017-C train 데이터셋에 적용결과를 나타낸 그림이다. Fig. 4에서 1, 4행은 각각 MNIST 데이터셋과 Visda2017-C 데이터셋에서 추출한 이미지이고 2, 5행은 AdaIN을 이용하여 1, 4행의 이미지를 Wikiart 데이터셋 이미지의 스타일로 변환시킨 것이다. 마지막으로 3, 6행은 본 연구에서 제안하는 Unpooled AdaIN을 이용하여 1, 4행의 이미지를 변환시킨 결과물이다. Fig. 4를 통해 Unpooled AdaIN 알고리즘이 기존의 AdaIN 알고리즘에 비해 컨텐츠 이미지의 공간 정보의 손실이 더 적음을 알 수 있다.

Fig. 4. Results of AdaIN and Unpooled AdaIN

4.2 규모가 작은 데이터셋

규모가 작은 데이터셋으로는 비지도 도메인 적응 분야에서 성능 평가 척도로 쓰이는 숫자 분류 데이터(MNIST, MNIST-M, SVHN)들을 사용하였다. 학습 도메인으로는 MNIST 데이터셋으로 선정했다. 학습 도메인에서 학습된 모델이 학습 도메인에 편향되어있는지를 검증하기 위한 시험 데이터셋으로는 MNIST-M 데이터셋과 SVHN 데이터셋을 선정하였다. MNIST-M 데이터셋은 MNIST 데이터셋의 데이터에 BSDS500[20] 데이터로부터 추출된 색을 합성한 데이터셋으로 기존의 MNIST 데이터셋보다 훨씬 다양한 질감을 가지고 있다. SVHN 데이터셋은 Google Street View를 이용하여 수집한 건물 번호 및 표지판 데이터이다. SVHN 데이터셋 역시 MNIST 데이터셋에 비해 훨씬 다양한 질감을 가지고 있다. 따라서 일반적인 방법으로 MNIST 데이터셋을 학습한 딥러닝 모델의 경우 MNIST-M 데이터셋과 SVHN 데이터셋에 대해 성능 하락이 크게 발생하게 된다. 특히 SVHN 데이터셋의 경우 MNIST 데이터셋과의 도메인 격차가 매우 크기 때문에 비지도 도메인 적응 방법으로도 해결하기 어렵다[1].

실험 및 평가에 사용할 딥러닝 모델을 이미지넷에서 사전 학습된 ResNet50[21] 모델로 선정하였다. 규모가 작은 데이터셋의 경우 SGD 옵티마이저를 사용했으며, 이 때 모멘텀은 0.9, 학습율은 0.01로 설정하였다. 배치 사이즈는 128, λ_{reg}는 0.5로 설정하였다.

Table 1. Results of MNIST->MNIST-M

학습데이터	방법	성능
S	Source (S)	54.6
S + T	DAN[23]	76.9
S + T	DANN[1]	77.4
S + T	DIRT-T[30]	98.9
S	Ours w/o feature regularization	79.7
S	Ours	80.9
T	Target (T)	95.5

Table 2. Results of MNIST->SVHN

학습데이터	방법	성능
S	Source (S)	28.3
S + T	DANN[1]	35.7
S + T	DRCN[31]	40.1
S + T	DIRT-T[30]	54.5
S	Ours w/o feature regularization	50.5
S	Ours	52.2
T	Target (T)	93.4

Table 1은 MNIST->MNIST-M 실험 결과를 정리하였으며 Table 2는 MNIST->SVHN 실험 결과를 정리하였

다. Table 1과 Table 2의 Source (S)는 자기 정규화 과정 없이 MNIST 데이터로 학습한 모델을 바로 각각의 시험 데이터에 적용했을 때의 결과이며 각 실험의 하한선(Lower bound)이다. Target (T)는 각 실험의 시험 데이터셋을 학습 데이터와 검증 데이터셋으로 나눈 후 학습 데이터에 대해 학습한 모델을 검증 데이터셋에 적용한 결과이며 각 실험의 상한선(Upper bound)이다. 제안하는 알고리즘의 성능 비교를 위해 도메인 적응 알고리즘의 실험 결과를 인용하였다. 제안하는 알고리즘의 성능은 10 에폭(epoch)동안 학습을 진행한 모델의 시험 결과이며 도메인 적응 알고리즘의 실험 결과는 각 알고리즘의 논문으로부터 인용하였다. 도메인 적응 알고리즘의 경우 시험 데이터와 동일한 도메인의 비지도 데이터를 학습 시 사용할 수 있는 반면 제안하는 알고리즘은 오로지 학습 도메인의 데이터만을 사용한다.

두 실험의 결과를 통해 자기 정규화 과정으로 모델을 학습한 경우 매개 데이터의 도움 없이 학습 도메인에 대한 모델의 편향을 방지할 수 있음을 알 수 있다. 또한 픽셀 정규화 과정만으로도 모델의 학습 데이터 편향을 방지할 수 있지만 특징 정규화 과정을 통해 시험 데이터셋에서의 모델 성능을 추가적으로 향상시킬 수 있음을 확인할 수 있다. 도메인 적응 알고리즘과의 비교를 통해 제안하는 알고리즘이 시험 도메인 데이터를 학습 시 활용하여 직접 학습 방향을 가이드하는 도메인 적응 알고리즘과 유사한 성능 향상을 성취했음을 확인할 수 있다. Table 1과 Table 2의 자기 정규화 모델의 성능을 측정하기 위해 사용된 모델은 같은 모델로 한 번의 자기 정규화 학습으로 다양한 도메인 변화에 강인한 모델을 얻을 수 있음을 확인하였다.

4.3 규모가 큰 데이터셋

규모가 큰 데이터셋으로는 Visda2017-C 데이터셋을 사용하였다. Visda2017-C 데이터셋은 비지도 도메인 적응 알고리즘 평가를 위해 만들어진 데이터셋이며, 3D CAD 영상으로 이루어진 학습 데이터와 실 영상으로 이루어진 시험 데이터로 이루어져있다. 데이터의 양도 이전의 숫자 분류 데이터셋에 비해 많을 뿐만 아니라, 학습 데이터와 시험 데이터간의 도메인 격차도 더 크다. 규모가 큰 데이터셋에서도 이미지넷에서 사전 학습된 ResNet50 모델로 실험 및 평가를 진행하였다.

Visda2017-C 데이터셋에 대한 실험의 경우 SGD 옵티마이저를 사용했으며, 모멘텀은 0.9, 학습율은 0.0001로 설정하였다. 학습 시 배치 사이즈는 128, λ_{reg}는 0.5로 설정하였다. Visda2017-C 데이터셋에서의 실험 결과는 Table 3과 같다. Table 3의 자기 정규화 실험 결과는 규모가 작은 데이터셋에 대한 실험과 같이 10 에폭동안 학습한 모델의 성능결과이다. Visda2017-C 데이터셋 역시 도메인 적응 분야의 방법론들과의 비교를 통해 자기 정규화를 통해 학습된 모델의 성능을 분석하였다.

Table 3. Results on Visda2017-C using ResNet50

학습데이터	방법	성능
S	Source only	44.5 (45.6)
S + T	DAN[23]	53.0
S + T	RTN[25]	53.6
S + T	DANN[1]	55.0
S + T	DTA[26]	76.2
S	Ours w/o feature regularization	55.7
S	Ours	59.1

Table 4. Results on Visda2017-C using ResNet101

학습데이터	방법	성능
S	Source only	45.3 (50.8)
S + T	DAN[23]	61.1
S + T	DANN[1]	57.4
S + T	DTA[6]	81.5
S	Ours w/o feature regularization	56.7
S	Ours	60.8

Table 3의 DAN[23], RTN[25], DANN[1], DTA[26]의 결과는 Lee[26]로부터 인용하였다. 비교의 편의를 위해 Lee[26]의 Source only 결과를 괄호 안에 표기하였다. Table 3의 결과를 통해서 어떠한 시험 도메인 데이터의 접근 없이 자기 정규화로 학습된 모델이 학습 시 시험 도메인의 데이터와 함께 학습한 비지도 도메인

적응 분야의 기법들과 유사한 성능을 낼 수 있음을 알 수 있다. 또한 특징 정규화를 하지 않고 학습한 모델과의 비교를 통해 두 정규화 모두 성능 향상에 도움이 됨을 알 수 있다. 자기 정규화 방법이 특정 네트워크에 국한되지 않음을 증명하기 위해 이미지넷에서 사전학습된 ResNet101[21] 모델을 이용하여 Visda2017 데이터셋에서 추가적으로 실험을 진행하였다. Table 4는 ResNet101 모델을 이용한 실험결과를 비지도 도메인 적응 분야의 기법들과 비교한 표이다. Table 4의 DAN[23], DANN[1], DTA[26]의 결과는 Table 3과 같이 Lee[26]로부터 인용하였으며, Table 4의 자기 정규화 실험은 Table 3의 실험과 동일한 환경에서 실행하였다.

Fig. 5와 Fig. 6은 각각 숫자 분류 데이터셋(green: mnist, blue: mnist-m, red: svhn)과 Visda2017 데이터셋(green: train, blue: test)에 대해 자기 정규화를 통해 학습한 모델과 단순히 학습 데이터로 재학습(fine-tuning)한 모델을 t-SNE[32]를 통해 분석한 그림이다. 두 그림 모두 왼쪽은 재학습한 모델의 결과를, 오른쪽은 자기 정규화를 통해 학습한 모델의 결과를 담고 있다. t-SNE[32] 분석을 통해 자기 정규화를 통해 학습한 모델은 단순 재학습한 모델에 비해 학습 도메인과 다른 도메인의 데이터에 대해서도 판별적인(discriminative) 특징 벡터를 추출할 수 있다는 점을 확인할 수 있다.

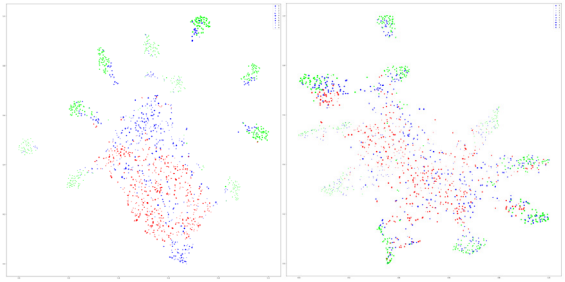

Fig. 5. t-SNE analysis of digit datasets

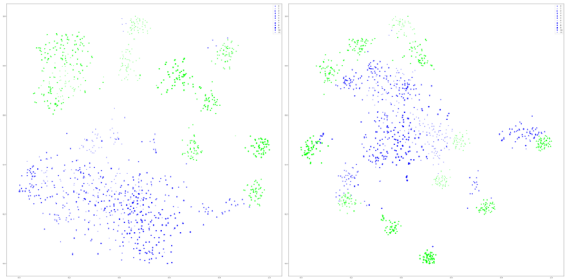

Fig. 6. t-SNE analysis of visda datasets

5. 결 론

본 연구는 시험 데이터와 학습 데이터간의 도메인 차이로 인해 딥러닝 모델의 성능 하락을 해결하기 위한 방법을 연구하였다. 위의 문제를 해결하는 대부분의 사전 연구들은 학습 데이터와 시험 데이터간의 간극을 이어주는 매개 데이터를 통해 딥러닝 모델 성능 하락을 방지하였다. 하지만 현실에서는 매개 데이터를 수집하는 데에 많은 시간과 노력이 들어가거나 불가능한 경우가 빈번하다. 이를 해결하기 위해, 본 연구는 모델이 학습 과정에서 학습 데이터에 쉽게 편향되지 않도록 하는 자기 정규화 과정을 제안하였다. 자기 정규화 과정에는 픽셀 정규화와 특징 정규화로 구성되어 있으며, 픽셀 정규화 과정에서 학습 데이터의 공간 정보를 유지하기 위해 Unpooled AdaIN 알고리즘을 제안하였다. 자기 정규화 과정의 성능을 평가하기 위해 다양한 크기의 데이터셋에서 시험을 수행했으며, 실제로 자기 정규화 과정을 통해 학습한 모델이 매개 데이터를 이용한 사전 연구들의 성과와 유사하거나 보다 우수한 성능을 내는 것을 확인하였다.

References

[1] Y. Ganin, V. Lempitsky, “Unsupervised Domain Adaptation by Backpropagation,” International Conference on Machine Learning, pp. 1180-1189, 2015.

[2] J. Deng, W. Dong, R. Socher, L. Li, K. Li, L. Fei-Fei, “Imagenet: A Large-Scale Hierarchical Image Database,” IEEE Conference on Computer Vision and Pattern Recognition, pp. 248-255, 2009.

[3] R. Geirhos, P. Rubisch, C. Michaelis, M. Bethge, F. A. Wichmann, W. Brendel, “ImageNet-Trained CNNs are Biased Towards Texture; Increasing Shape Bias Improves Accuracy and Robustness,” arXiv Preprint arXiv:1811.12231, 2018.

[4] X. Peng, B. Usman, N. Kaushik, D. Wang, J. Hoffman, K. Saenko, “Visda: A Synthetic-to-Real Benchmark for Visual Domain Adaptation,” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 2021-2026, 2018.

[5] E. Tzeng, J. Hoffman, K. Saenko, T. Darrell, “Adversarial Discriminative Domain Adaptation,” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7167-7176, 2017.

[6] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, “Generative Adversarial Nets,” Advances in neural Information Processing Systems, pp. 2627-2680, 2014.

[7] K. Bousmalis, N. Siberman, D. Dohan, D. Erhan, D. Krishnan, “Unsupervised Pixel-Level Domain Adaptation with Generative Adversarial Networks,” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3722-3731, 2017.

[8] J. Hoffman, E. Tzeng, T. Park, J. Y. Zhu, P. Isola, K. Saenko, A. A. Efros, T. Darrell, “Cycada: Cycle-Consistent Adversarial Domain Adaptation,” International Conference on Machine Learning, pp. 1989-1998, 2018.

[9] S. Motiian, Q. Jones, S. Iranmanesh, G. Doretto, “Few-Shot Adversarial Domain Adaptation,” Advances in Neural Information Processing Systems, pp. 6670-6680, 2017.

[10] X. Xu, X. Zhou, R. Venkatesan, G. Swaminathan, O. Majumber, “d-sne: Domain Adaptation Using Stochastic Neighbor Embedding,” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2497-2506, 2019.

[11] D. Li, Y. Yang, Y. Z., T. M. Hospedales, “Deeper Borader and Artier Domain Generalization,” Proceedings of the IEEE International Conference on Computer Vision, pp. 5542-5550, 2017.

[12] Y. Balaji, S. Sankaranarayanan, R. Chellappa, “Metareg: Towards Domain Generalization Using Meta-Regularization,” Advances in Neural Information Processing Systems, pp. 998-1008, 2018.

[13] L. A. Gatys, A. S. Ecker, M. Bethge, “Image Style Transfer Using Convolutional Neural Networks,” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2414-2423 2016.

[14] J. Johnson, A. Alahi, L. Fei-Fei, “Perceptual Losses for Real-Time Style Transfer and Super-Resolution,” European Conference on Computer Vision, pp. 694-711, 2016.

[15] X. Huang, S. Belongie, “Arbitrary Style Transfer in Real-Time with Adapive Instance Normalization,” Proceedings of the IEEE International Conference on Computer Vision, pp. 1501-1510, 2017.

[16] Y. Li, C. Fang, J. Yang, Z. Wang, X. Lu, M. H. Yang, “Universal Style Transfer via Feature Transforms,” Advances in Neural Information Processing Systems, pp. 386-396, 2017.

[17] M. D. Zeiler, R. Fergus, “Visualizing and Understanding Convolutional Networks,” European Conference on Computer Vision, pp. 813-833, 2014.

[18] T. Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Piotr, C. L. Zitnick, “Microsoft Coco: Common Objects in Context,” European Conference on Computer Vision, pp. 740-755, 2014.

[19] K. Simonyan, A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” arXiv preprint arXiv:1409.1556, 2014.

[20] P. Arbelaez, M. Maire, C. Fowlkes, J. Malik, “Contour Detection and Hierarchical Image Segmentation,” IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 33, No. 5, pp. 898-916, 2010.

[21] K. He, X. Zhang, S. Ren, J. Sun, “Deep Residual Learning for Image Recognition,” Proceeding of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770-778, 2016.

[22] B. Sun, K. Saenko, “Deep Coral: Correlation Alignment for Deep Domain Adaptaion,” European Conference on Computer Vision, pp. 443-450, 2016.

[23] M. Long, Y. Cao, J. Wang, M. Jordan, “Learning Transferable Features with Deep Adaptation Networks,” International Conference on Machine Learning, pp. 97-105, 2015.

[24] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, Y. A. Ng, “Reading Digits in Natural Images with Unsupervised Feature Learning,” NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011, 2011.

[25] M. Long, H. Zhu, J. Wang, M. I. Jordan, “Unsupervised Domain Adaptation with Residual

Transfer Networks," Advances in Neural Information Processing Systems, pp. 136-144, 2016.

[26] S. Lee, D. Kim, S. G. Jeong, "Drop to Adapt: Learning Discriminative Features for Unsupervised Domain Adaptation," Proceedings of the IEEE International Conference on Computer Vision, pp. 91-100, 2019.

[27] A) Nichol, Kiri, 2016. "Painter by numbers, wikiart," www.kaggle.com/c/painter-by-numbers/data (accessed by 2016).

[28] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, "Gradient-based Learning Applined to Document Recognition," Proceddings of the IEEE, Vol. 86, No. 11, pp. 2278-2324, 1998.

[29] D. P. Kingma, J. L. Ba, "Adam: A Method for Stochastic Optimization," Proceedings of the International Conference on Learning Representations, arXiv:1412.6980, 2015.

[30] R. Shu, H. H. Bui, H. Narui, & S. Ermon, "A Dirt-t Approach to Unsupervised Domain Adaptation," Proceedings of the International Conference on Learning Representations, arXiv:1802.08735, 2018.

[31] M. Ghifary, W. B. Kleijn, M. Zhang, D. Balduzzi, W. Li, "Deep Reconstruction-Classification Networks for Unsupervised Domain Adaptation," In European Conference on Computer Vision, pp. 597-613, Springer, 2016.

[32] L. Van der Maaten, G. Hinton, "Visualizing Data Using t-SNE," Journal of Machine Learning Research, 9(11), 2008.