MLP 모델을 위한 Mixup 알고리즘 기반의 Data Augmentation에 관한 연구

현선영*, 김필송*, 황성연**, 하영국**†
*건국대학교 인공지능학과
**건국대학교 컴퓨터공학과

mss0061@naver.com, ggsks@naver.com, wiw100@naver.com, ygha@konkuk.ac.kr

A Study on Data Augmentation based on Mixup Algorithm for MLP Model

Sun-young Hyun*, Pil-song Kim*, Seong-yeon Hwang**, Young-guk Ha**†
*Dept. of Artificial Intelligence, Konkuk University
**Dept. of Computer Science and Engineering, Konkuk University

요 약

본 논문에서는 CNN 모델에서 학습에 사용할 이미지 데이터를 늘리기 위해 사용되는 Mixup 알고리즘을 MLP 모델에 사용하는 데이터셋에 적용하여 data augmentation 효과를 얻을 수 있는 지에 대한 테스트를 수행했다. 테스트 결과 MLP 모델에 사용할 데이터셋에도 Mixup 알고리즘으로 data augmentation 효과를 기대할 수 있음을 보여준다.

1. 서론

최근 인공지능 신경망의 연구에서는 학습 데이터가 부족한 경우 이를 극복하기 위해 data augmentation방법을 주로 사용한다. 다양한 data augmentation방법이 존재하지만 대부분의 방법은 이미지 데이터셋에 적용되는 방법들이다. 이러한 상황에서 MLP(Multi-Layer Perceptron) 모델에 사용할 데이터셋에 data augmentation을 적용하기 위하여 이미지 데이터에 적용되는 방법을 MLP 모델에 사용할 데이터셋에 적용하는 연구를 진행하였다. 여기서 MLP 모델에 사용할 데이터셋이란 int, float, category형 등의 데이터로 이루어진 데이터셋을 말한다.

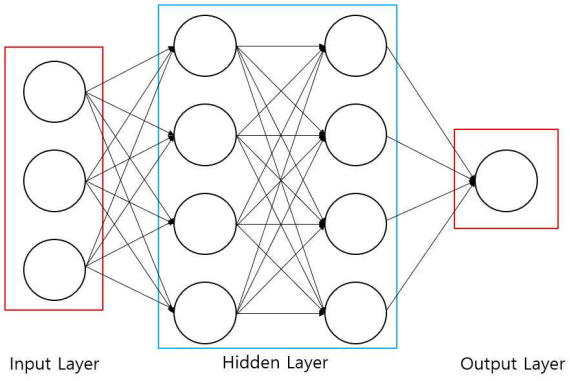
본 연구에서는 Mixup 기법을 MLP 구조에 적용하여 data augmentation 효과를 확인하고자 한다.

2. 관련연구

2.1 MLP(Multi-Layer Perceptron) 모델

MLP 모델은 DNN 모델들의 기초가 되는 모델로서 인간의 신경세포의 다수의 입력으로부터 하나의 결과를 출력하는 것과 유사한 알고리즘을 갖는 퍼셉트론(Perceptron)을 여러 층으로 쌓은 모델이다. 퍼셉트론은 입력 값에 대하여 각각의 입력에 해당하는 가중치를 곱하고 그 곱들의 전체 합이 임계치를 넘으면 1을 출력신호로 내보내고 그렇지 않을 경우 0을 출력하는 인공신경망의 기초가 되는 모델이다.

MLP 모델은 이러한 퍼셉트론으로 구성된 입력층, 은닉층, 출력층 등 셋 이상의 노드계층으로 구성된다. 입력층과 출력층은 각각 값을 입력받고 출력하는 층으로 각각 한 개씩 가지고 은닉층은 최소 한 층 이상을 가진다[1].



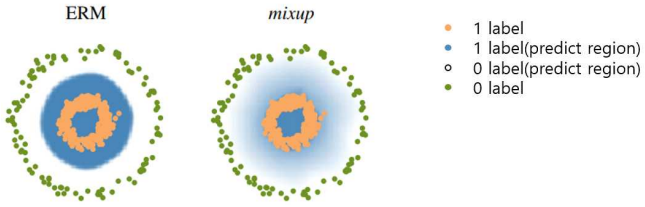
(그림 1) MLP 모델

† 교신저자

2.2. Mixup Algorithm

Mixup 알고리즘은 주로 지도학습에 사용되는 레이블링된 이미지를 augmentation하는 방식으로 한 장의 이미지가 아닌 두 장의 이미지를 사용하는 것이 특징이다. 두 장의 이미지가 있을 때 두 이미지를 합성하여 새로운 이미지를 생성하는 것이다[2].

이러한 방식으로 데이터를 augmentation할 경우 두 클래스 간의 decision boundary가 더 부드럽게 되어 overfitting이 덜 발생하게 된다는 장점이 있다. (그림 2)의 좌측과 같이 기존 ERM(Empirical risk Minimization)방식을 사용하여 학습 시킬 경우 각 클래스의 경계에 있는 값을 한쪽 클래스로 분류하지만 Mixup을 적용한 데이터셋으로 학습시킬 경우 우측 그림과 같이 경계가 흐려져 중간에 있는 값에 대하여 좀 더 유연하게 추론하게 되는 것이다.



(그림 2) ERM과 Mixup 비교[2]

3. MLP Mixup

Mixup 알고리즘은 이미지 데이터에 적용하는 방법이다. 이 방법이 이미지 뿐만 아니라 MLP 모델에서 사용하는 N차원 데이터에 적용 가능하다면 MLP 모델에도 적용할 수 있다. 따라서 다음의 수식을 사용하였다.

$$\widetilde{x_d} = \lambda x_{di} + (1-\lambda)x_{dj} \quad (1)$$

$$\tilde{y} = \lambda y_i + (1-\lambda)y_j \quad (2)$$

$\widetilde{x_d}$는 새롭게 생성된 d차원의 값이고 $\tilde{y}$는 새롭게 생성된 label 값이다. x_{di}와 x_{dj}는 각각 두 데이터 i, j input data의 d차원의 값이고 y_i와 y_j는 one-hot label encoding된 값들이다. λ는 임의의 변수 α를 두 매개변수로 하는 beta 분포로부터 추출한 0부터 1사이의 값으로 두 data의 혼합 비율을 나타낸다.($\lambda \in [0,1]$)

위 수식으로 2차원의 이미지를 n차원으로 확장할 경우 차원에 영향을 받는 변수가 없기 때문에 차원에 관계없이 데이터 augmentation이 가능하다.

```
x : data_x, y : data_y(one-hot vectors),
onehot_x, x̃ : matrix(only have index)
for col in x.columns :
    if dtype(x[col]) == str or dtype(x[col]) == "category" :
        temp_x = one_hot(x[col])
    else :
        temp_x = x[col]
    onehot_x = merge(onehot_x, temp_x)
train_x = onehot_x, train_y = y
for i in range(n) :
    for j in range(i, n) :
        λ = random_beta_distribution(α)
        x̃ = mixup(onehot_x[i], onehot_x[j], λ)
        ỹ = mixup(y[i], y[j], λ)
        train_x = concat(train_x, x̃)
        train_y = concat(train_y, ỹ)
```

(그림 3) 알고리즘

실험에 사용된 알고리즘은 (그림 3)과 같다. one-hot 함수는 입력받은 str 혹은 category형 데이터에 one-hot encoding을 적용하고 반환해주는 함수이고, random_beta_distribution 함수는 입력받은 α를 두 매개변수로 하는 beta 분포를 따르는 값 중 임의의 값을 추출해주는 함수다. mixup 함수는 입력받은 세 매개변수를 각각 수식 (1)(x_{di}, x_{dj}, λ) (2)(y_i, y_j, λ)에 대입하여 데이터를 생성하는 함수이다.

4. 실험

4.1 실험 환경

본 연구는 AMD Ryzen Threadripper 2950X CPU, 128GB 메모리, RTX 3090 GPU 그리고 Ubuntu 18.04 LTS OS가 탑재된 서버에서 진행되었다. 신경망 모델에는 BN, Dropout 등의 기법이 적용되었고 사용된 하이퍼파라미터는 <표 1>과 같다. 하이퍼파라미터의 경우 성능 최적화가 아닌 augmentation의 적용 여부에 따른 차이 비교를 중점으로 결정된 값이다.

<표 1> Hyperparameter

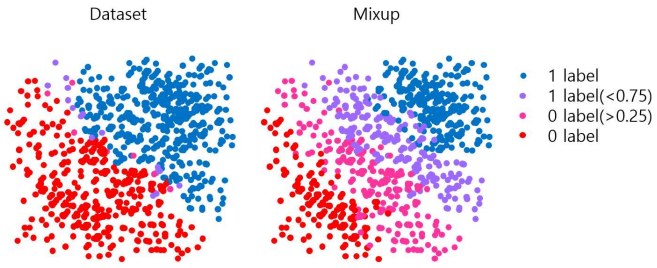
Model Hyperparameter	
Hidden Layer(10)	Input, 30, 20, 30, 40, 40, 20, 30, 20, 20, 10, output
Activate Function	ReLU
Learning Rate	0.001
Dropout	0.3
Loss Function	Cross Entropy Loss
Optimizer	Adam

4.2 실험 데이터

실험에 사용된 데이터는 classification을 위한 데이터셋 중 UCI Machine Learning Repository의 adult 데이터셋[4]을 사용하였다. adult 데이터셋은 1994년도에 미국에서 진행된 인구 조사 DB를 기반으로 연 소득 $50K 초과여부를 추론하는 데이터셋이다. 이 데이터 중 3000개를 이용하여 30000개의 데이터를 augmentation했고 실 데이터 33000개를 사용하여 성능을 비교하였다. 또한 α값에 따른 성능의 차이도 확인하였다.

4.3 실험 결과

데이터에 Mixup 알고리즘을 적용하여 학습시킨 후 출력 값에 따른 입력 데이터의 분포는 (그림 4)와 같다. image augmentation때와 마찬가지로 기존의 레이블 간의 경계에 있는 값에 대하여 조금 더 유연하게 추론 하는 것을 확인할 수 있다.



(그림 4) 데이터 분포 비교

모델의 성능에 대한 결과는 아래 <표 2>와 같다. data augmentation을 하지 않은 case에 대하여 최대 9%의 차이를 보이고 augmentation을 적용하지 않은 33000개의 데이터와 Mixup 알고리즘을 적용한 33000개의 데이터의 결과 α값에 따라 Accuracy가 0.5% ~ 8% 차이를 보였다. 따라서 augmentation 효과를 기대할 수 있음을 확인하였다.

<표 2> 실험 결과(Average Accuracy)

Data	Accuracy(%)
Data 3000	68.33
Data 3000 + 30000 (α=0.1)	71.18
Data 3000 + 30000 (α=0.2)	73.64
Data 3000 + 30000 (α=0.4)	76.75
Data 3000 + 30000 (α=4)	73.79
Data 3000 + 30000 (α=8)	77.15
Data 3000 + 30000 (α=32)	69.72
Data 33000	77.69

실험 결과를 종합하면 2차원의 방법이었던 Mixup 알고리즘은 n차원으로 확장하여 적용이 가능하다고 볼 수 있다.

5. 결론 및 향후 연구

이 연구를 통하여 image dataset에 대한 data augmentation 방법이 n차원 data로의 확장이 가능하다면 MLP 모델을 위한 데이터셋에도 적용할 수 있다는 것을 발견하였다.

즉, int, float, category형 데이터셋을 사용하여 MLP 구조의 인공신경망을 학습하고자 할 때, 학습에 사용할 데이터를 충분히 얻지 못하였거나 데이터가 더 필요한 경우가 발생한다면, 기존의 고안되었던 방법 뿐만 아니라 특정한 조건을 만족하는 다른 데이터형을 위한 data augmentation 알고리즘을 적용하여도 좋은 결과를 기대할 수 있음을 알 수 있었다.

향후 연구에서는 Mixup 알고리즘이 아닌 다른 augmentation방법을 적용한 경우에도 충분한 augmentation 효과 기대 가능 여부와 더 높은 기대치를 가지는 방법을 중심으로 연구해보고자 한다.

사사문구

본 연구는 과학기술정보통신부 및 정보통신기획평가원의 대학ICT연구센터육성지원사업의 연구결과로 수행되었음
(IITP-2021-2016-0-00465)

참고문헌

[1] Marvin Minsky, Seymour Papert, “Perceptrons: an introduction to computational geometry” Expanded Ed, Cambridge Ma, 1987

[2] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, David Lopez-Paz, “mixup: Beyond Empirical Risk Minimization”, ICLR 2018, Vancouver

[3] Michiel Hazewinkel, “Encyclopedia of Mathematics”, Kluwer Academic Publishers, 2002

[4] UCI Machine Learning Repository, Adult Data Set, https://archive.ics.uci.edu/ml/datasets/adult

[5] I.T. Jolliffe, “Principal Component Analysis, Second Edition”, Springer, 2002

[6] Sam Roweis, Geoffrey Hinton, “Stochastic neighbor embedding”, NIPS 2002, Vancouber