[Project #2]

Deep Learning for Image Classification with EM Algorithms

1 INTRODUCTION

Cifar10 data 는 총 60000 개 이미지이며 10 개의 label 을 가지고 있는 데이터입니다. 딥러닝은 black box 모델로 어떠한 근거로 출력 값을 모델이 냈는지 알지 못 합니다. 합성곱 층의 일부인 모델의 앞부분 필터를 가지고 추론이 가능하지만 모델의 뒤로 갈수록 필터들이 학습하고 고수준의 특징을 추출하려는 특성 때문에 형태를 알아보기는 어렵습니다.

요즘 black box 인 딥러닝의 한계를 무너뜨리기 위한 XAI 라는 신경망 연구가 활발히 이뤄지고 있으며, 그 중 일부가 CAM, Grad-CAM 입니다. 본 프로젝트는 Cifar10 data 를 합성곱 신경망을 통해 학습합니다. 또한 필터와 특성맵이 어떤 형태를 가지고 있으며 필터를 지나면서 특성맵이 어떻게 변화하는지 관찰하는데 목표가 있습니다. 또한 CAM 을 실제로 적용해 모델이 어떤 부분을 보고 분류하는 task 를 수행했는지 알고자 하는데 목표가 있습니다.

모델을 학습하고 나면 feature vector 를 얻을 수가 있습니다. 따라서 feature vector 를 T-SNE 를 사용해 2 차원 벡터로 축소하여 벡터들이 어떠한 형태를 띄고 있는지 관측하며 어떤식으로 label 별 분포를 가지고 있는지 확인합니다. 또한 가우시안 혼합 분포 모델을 사용해 같은 분포를 가진 벡터끼리 모이도록 학습하고 이를 시각화하는데 목표가 있습니다.

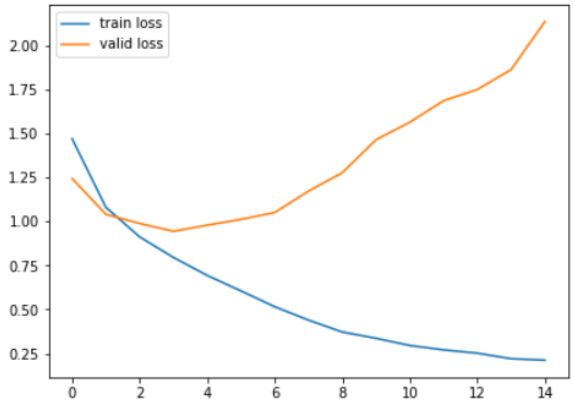
2 MAIN STRUCTURE

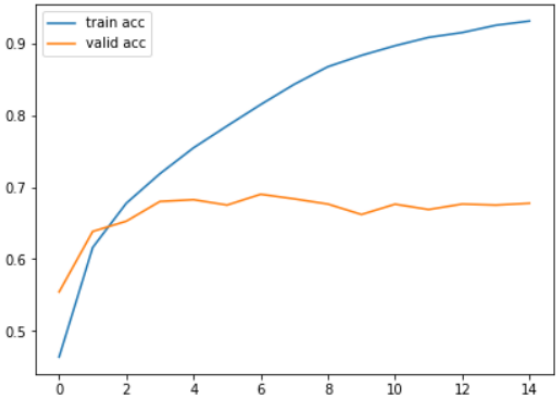
ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

모델은 VGG11 구조를 활용합니다. Cifar10 데이터의 이미지 크기는 32x32 입니다. 기존 VGG11 의 입력 값은 224X224 이고 출력은 7X7 입니다. 이때 출력 값을 똑같이 활용하기 위해 5 번의 MaxPooling layer 를 2 번만 사용했으며 Conv3-256 층 다음과 마지막 Conv3-512 층 다음에 MaxPooling layer 를 설정해 32X32 의 이미지 크기가 7X7 로 축소되도록 했습니다. 또한 fully connected layer 는 기존 VGG11 구조와 동일하게 출력층을 제외한 은닉층 2 개를 사용했으며 은닉노드의 개수만 다르게 설정했습니다. 첫 번째 은닉층은 1000 개 두 번째 은닉층은 512 개 세 번째 은닉층은 label 의 개수 10 개로 설정했습니다. 합성곱 층의 출력인 512 X 7 X 7 을 AdaptiveAvgPool 층을 사용해 512 X 1 X 1 크기로 만들어 fully connected layer 의 입력 값이 되도록 했습니다. 기존 VGG11 과 같이 Flatten 층을 지나 fully connected layer 의 입력이 되도록 하려고 했지만 이렇게 되면 25088 의 입력 값이 되어 연산량이 많아져 오히려 학습에 방해가 될 것이라고 생각했습니다. 활성화 함수는 VGG11 구조와 동일하게 ReLU 함수를 사용했습니다.

3 TRAINING THE NETWORK

모델에 따라 다르지만 VGG 구조는 연산량이 많아 먼저 단순한 모델 구조를 사용해 Dropout, 배치 정규화, l2 regularization Layer , 이미지 전처리 등의 실험을 했습니다.

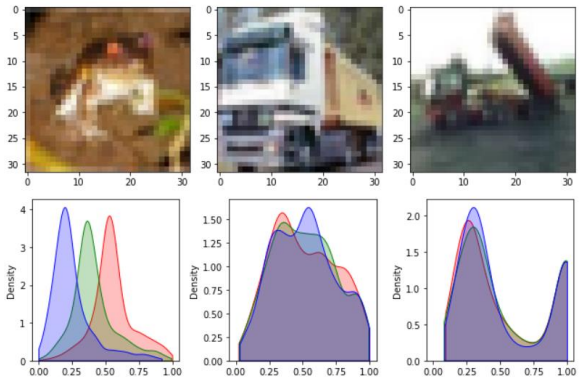


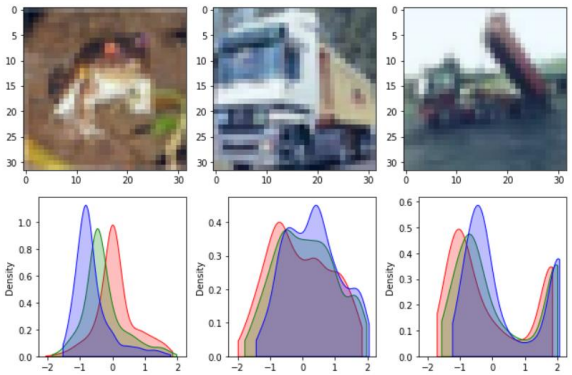


처음 아무것도 추가를 하지 않고 학습을 시켰을 때 loss 와 accuracy 의 시각화입니다.

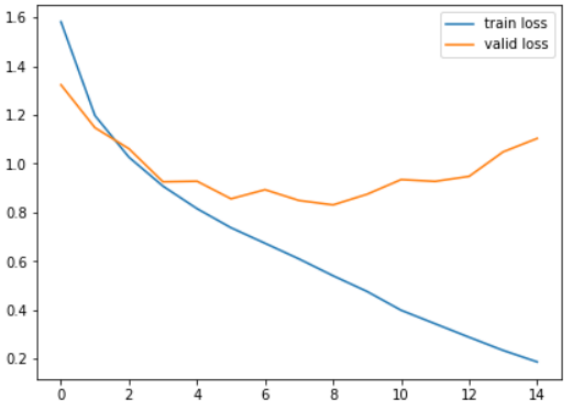
위 그래프와 같이 valid loss 가 증가하는 것을 보면 과대 적합 경향을 보이고 있습니다.

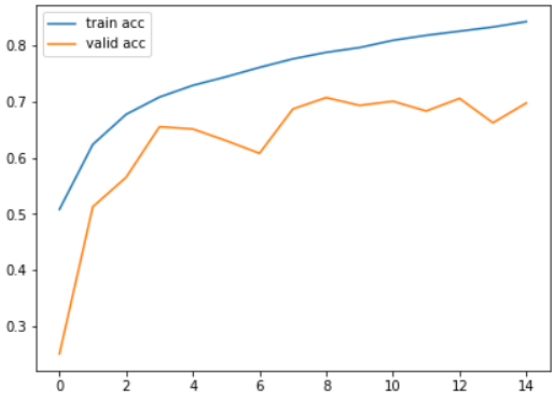
따라서 과대 적합을 줄이기 위해 배치 정규화를 적용해 실험했습니다.





배치 정규화를 적용하기 전과 배치 정규화를 적용했을 때의 분포입니다.

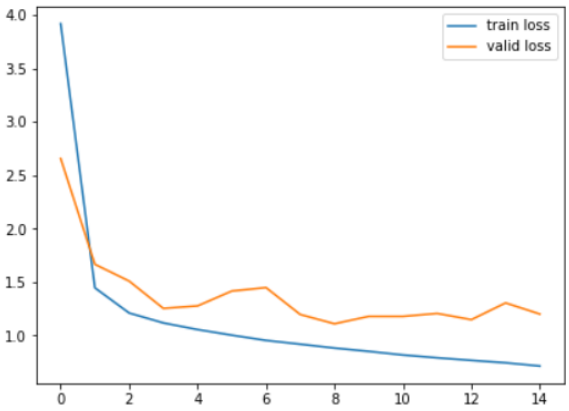


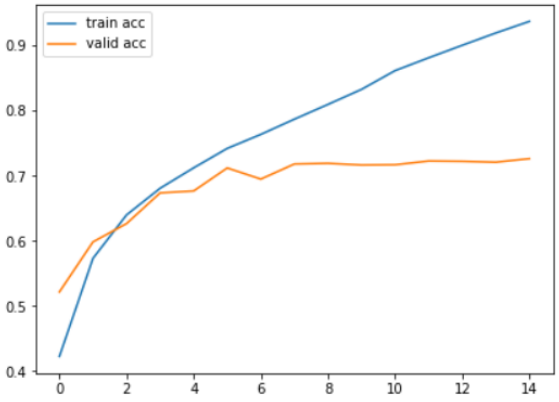


배치 정규화를 적용했을 때 loss 와 accuracy 시각화입니다.

Batch Normalization 을 적용했을 때 과대 적합이 줄어든 것으로 보입니다.

하지만 여전히 과대 적합 경향이 있어 l2 regularization 을 추가했습니다.

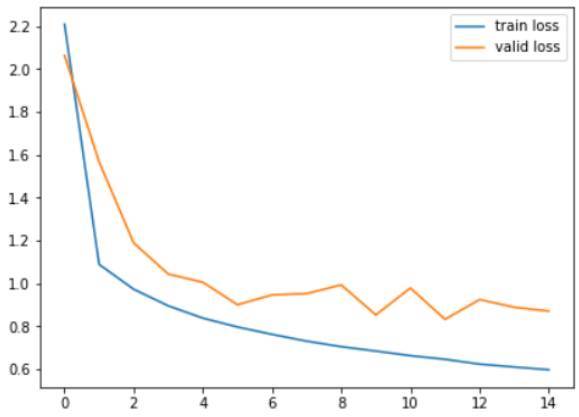


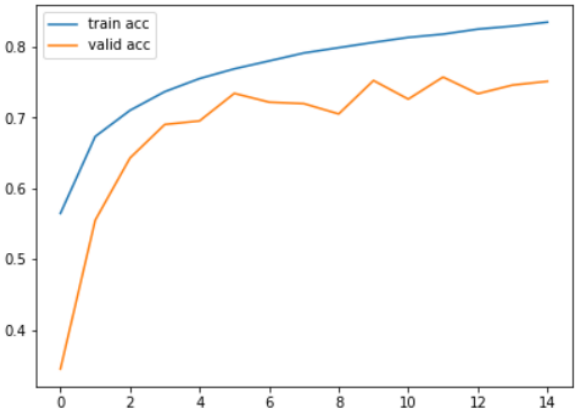


L2 regularization 을 적용했을 때 loss 와 accuracy 의 시각화입니다.

배치정규화를 적용했을 때 보다 과대 적합이 줄어든 것으로 보입니다.

하지만 여전히 train 데이터의 정확도와 valid 데이터의 정확도 차이가 많이 나므로 data augmentation 을 추가로 적용했습니다.





data augmentation 을 적용했을 때 train 정확도와 valid 정확도의 간격이 줄어든 것으로 보입니다.

먼저 단순한 모델을 통해 어떤 방법을 적용해야 할지 실험을 했습니다. 이를 바탕으로 VGG 구조로 적용했습니다.

VGG 구조로는 5 가지 모델을 사용해 학습했습니다.

첫 번째 모델은 일반 모델입니다.

두 번째 모델은 배치 정규화를 적용한 모델입니다.

세 번째 모델은 배치 정규화와 Dropout 을 적용한 모델입니다.

네 번째 모델은 l2 regularization 을 적용한 모델입니다.

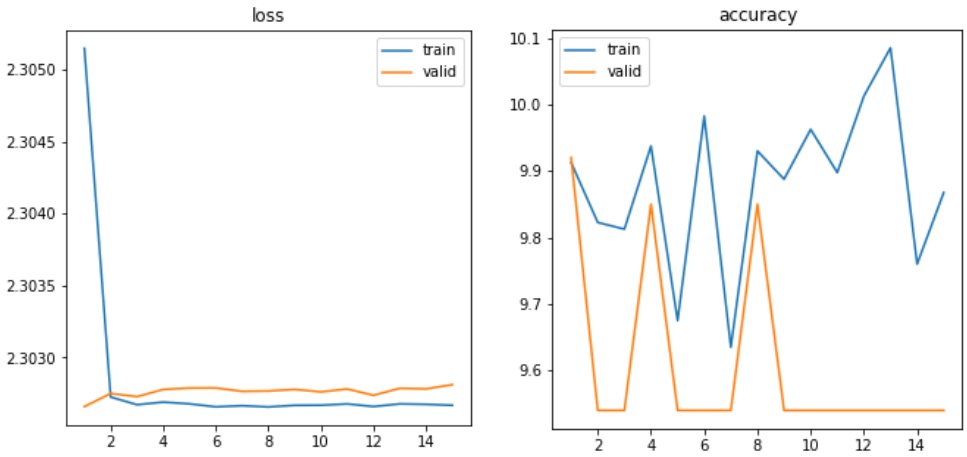
다섯 번째 모델은 이미지 전처리를 적용한 모델입니다.

총 60000 개 데이터를 train 데이터 40000 개 valid 데이터 10000 개 test 데이터 10000 개로 분할을 했으며 각 데이터는 R,G,B 차원에 맞는 평균과 분산으로 정규화를 했습니다.

모델을 학습할 때는 Adam 옵티마이저를 사용했습니다. learning rate 는 0.001, 0.01, 0.001 과 0.01 을 번갈아가는 learning rate scheduler 등 세가지 경우를 실험한 후 학습이 가장 안정적인 0.001 을 선택했습니다. Epoch 는 15 번을 기준으로 valid 데이터의 성능이 좋아지는 경향이 없어 15Epoch 만을 학습했습니다. 또한 배치정규화를 사용하기 때문에 배치사이즈는 128 로 큰 값으로 설정했습니다..

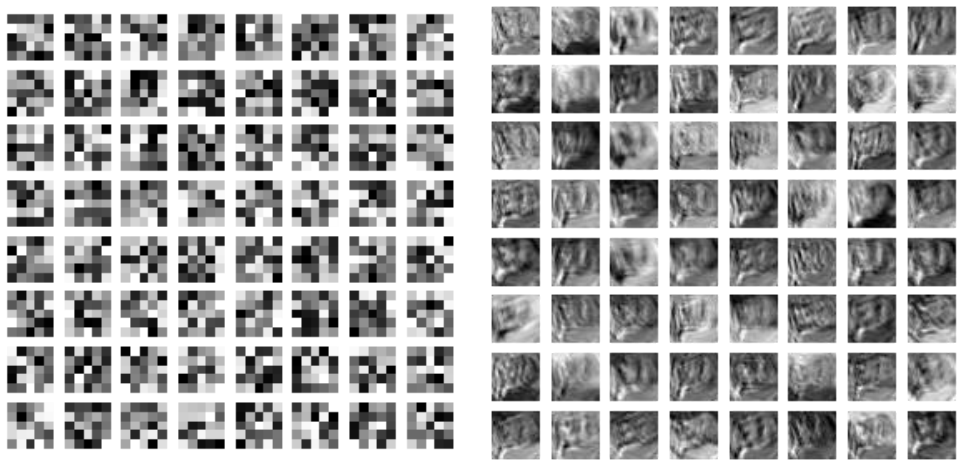
4 EXPERIMENTAL RESULTS

4.1 기존 모델 시각화



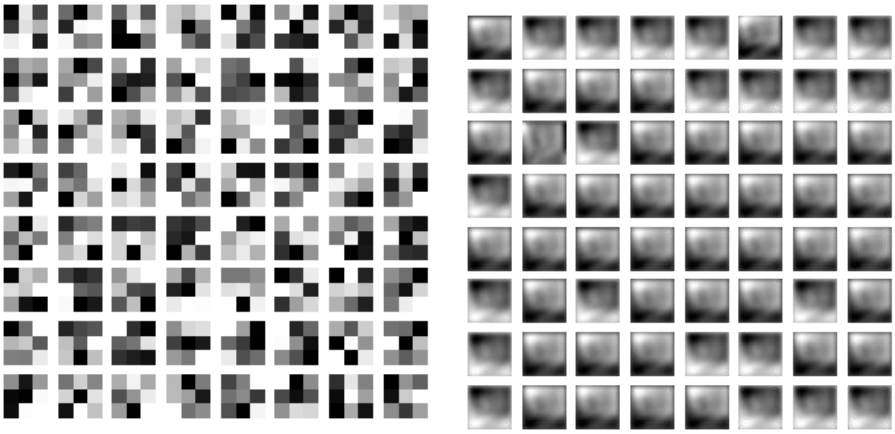
VGG11 구조를 활용해 Cifar10 데이터를 학습했을 때 loss 값과 정확도의 시각화입니다.

VGG11 구조도 VGG11, 13, 16, 19 모델에 비해서는 적은 가중치지만 복잡한 모델이므로 학습을 제대로 하지 못하는 모습을 보이고 있습니다. 정확도 측면으로 볼 때 10% 주변을 맴돌고 있는 것으로 보아 특정 label 로 찍는 것과 같아 보입니다.



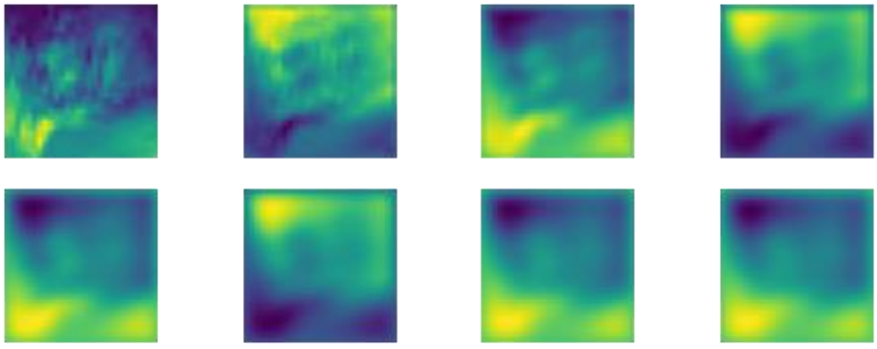
학습되지 않은 필터와 그 때 필터를 지난 특성맵 입니다.

학습되지 않아 필터가 어떤 특정 부분을 추출하고 있는 것처럼 보이지 않습니다.



합성곱 층의 마지막 필터와 이를 지난 특성맵의 시각화입니다.

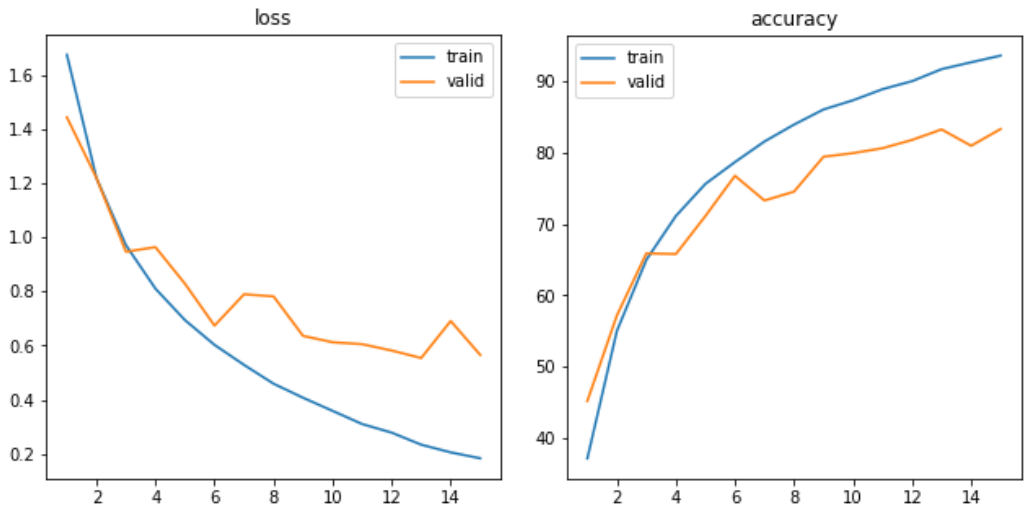
학습이 되지 않아 필터가 뚜렷한 특징을 보이지 않으며 따라서 필터를 지난 특성맵이 제대로된 특징을 추출하지 못하는 것으로 보입니다.



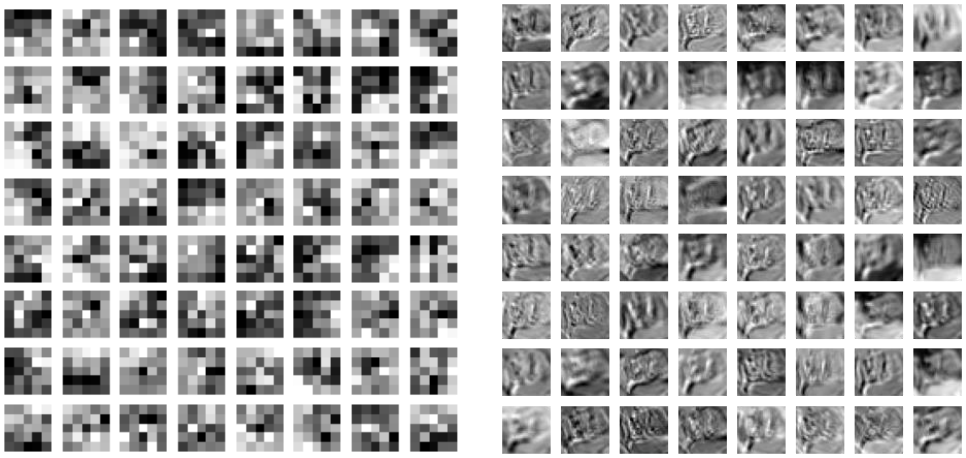
한 이미지가 모든 필터를 지난 특성맵의 시각화입니다. 학습이 잘 되지 않아 필터에서 다양하고 특정 특징들을 추출하지 못하는 것으로 보입니다.

.

4.2 배치정규화 모델 시각화

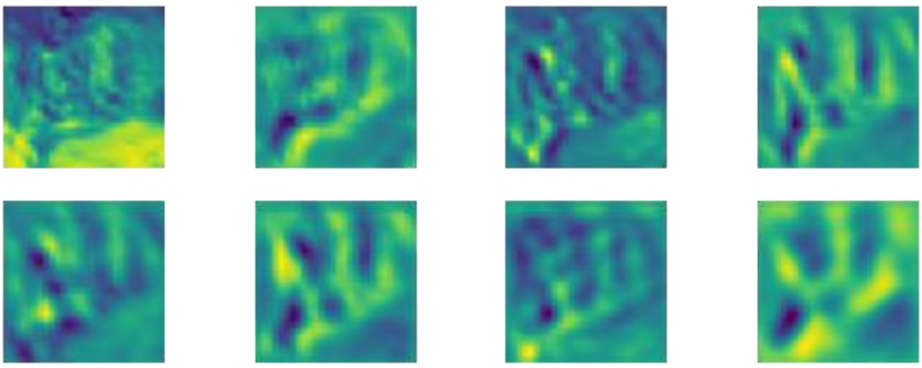


기존 VGG11 구조를 그대로 사용하면 학습이 전혀 되지 않아 배치 정규화를 사용했습니다. Train loss 와 valid loss 가 줄어드는 것을 볼 수 있으며 정확도 역시 train 데이터는 약 90% valid 데이터는 약 80%를 보여주고 있습니다. 약간의 과대 적합 되는 경향이 있는 것으로 보입니다.

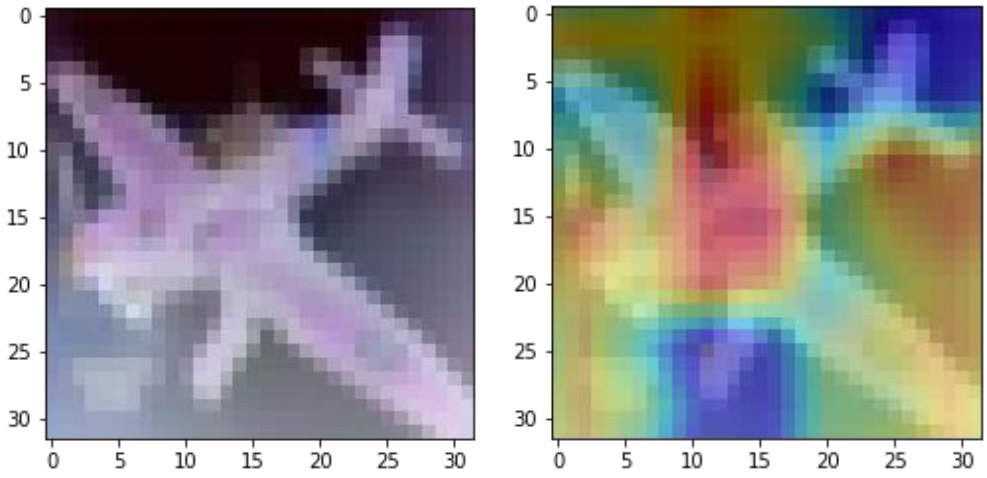


배치정규화를 적용했을 때 첫 번째 필터와 특성맵을 시각화 한 모습입니다. 학습이 되지 않았을 때보다 필터가 뚜렷해진 것을 볼 수 있으며 특성맵의 경우도 다양한 특징과 저수준 정보를 추출하고 있는 것을 볼 수 있습니다.

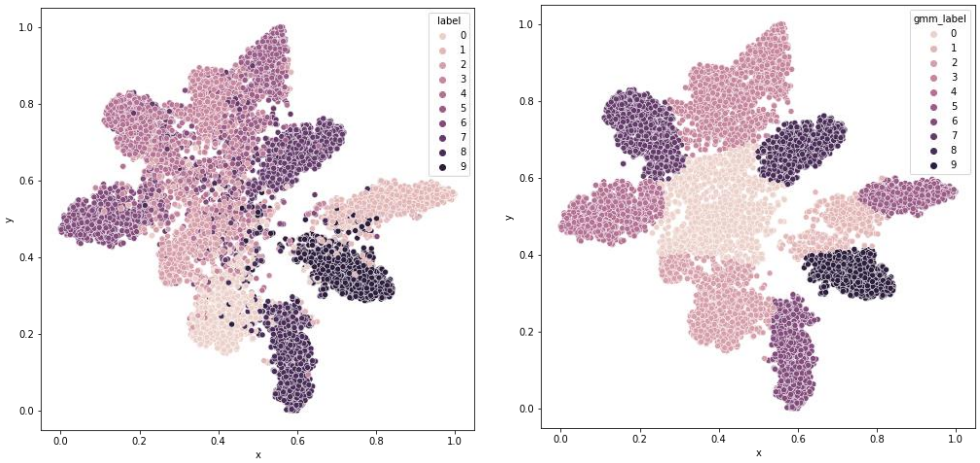
마지막 합성곱 층의 필터와 특성맵의 시각화입니다. 학습이 진행될수록 필터의 특징이 뚜렷해지며 특성맵 또한 흐릿하게 보이며 많은 고수준의 다양한 특성들이 추출되는 것을 볼 수 있습니다.



이미지가 필터를 거치면서 보다 고수준의 특징이 추출되는 것으로 보입니다.

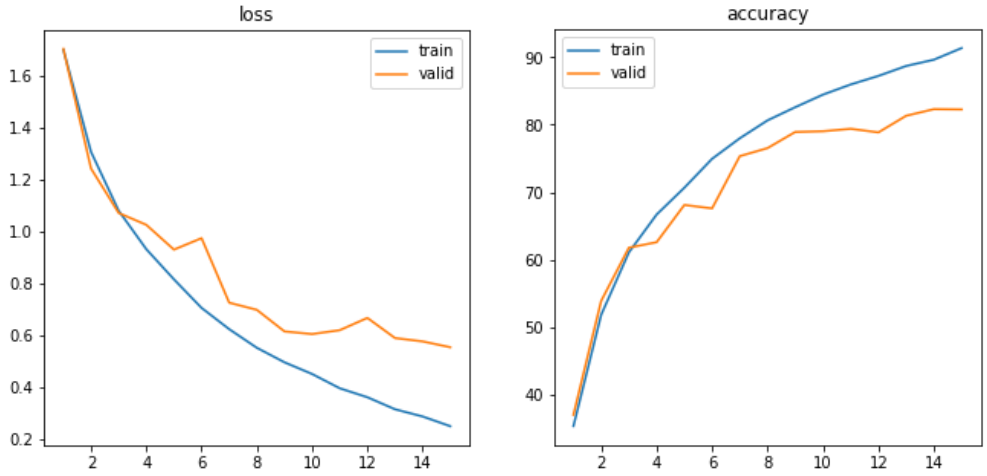


배치 정규화를 사용했을 때 모델은 비행기의 가운데 부분을 보고 판단한 것으로 보입니다.

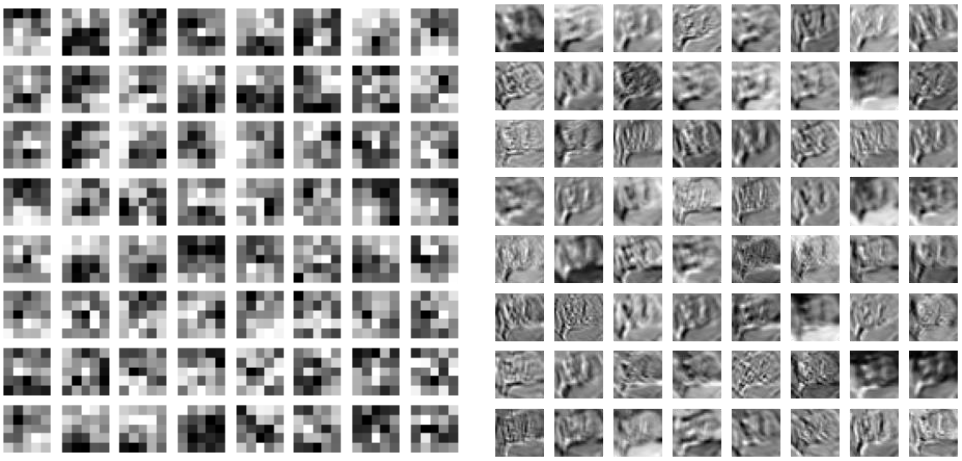


일반적으로 학습이 잘 진행되었을 경우 학습된 가중치를 거친 출력 값의 벡터는 label 에 따라 잘 군집화 되는 것으로 보입니다.

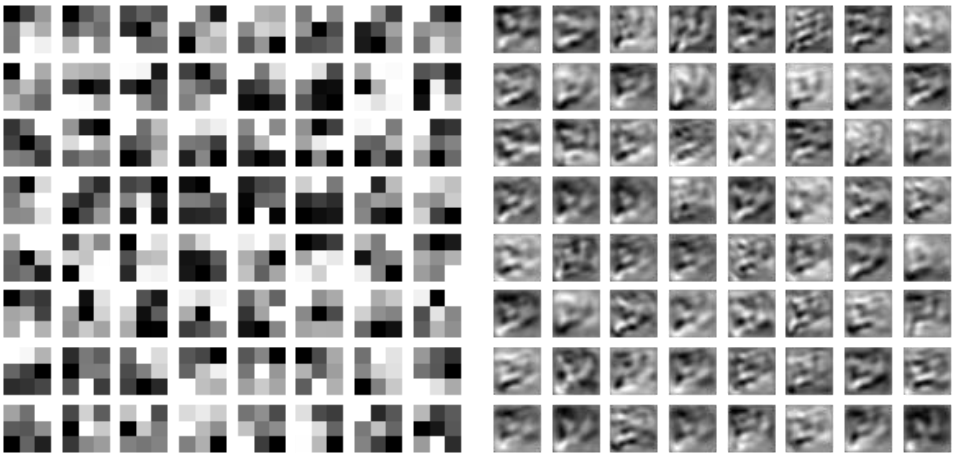
4.3 배치정규화 & 드롭아웃 모델 시각화



배치정규화를 적용했을 때 Epoch 가 증가함에 따라 과대적합이 보여 MaxPooling layer 이후 0.2 의 비율로 노드를 끄는 드롭 아웃을 적용해서 실험했습니다. 드롭아웃을 적용했을 때 valid loss 와 valid accuracy 를 보면 배치정규화를 했을 때보다 과대 적합의 경향이 줄어드는 것으로 보입니다.

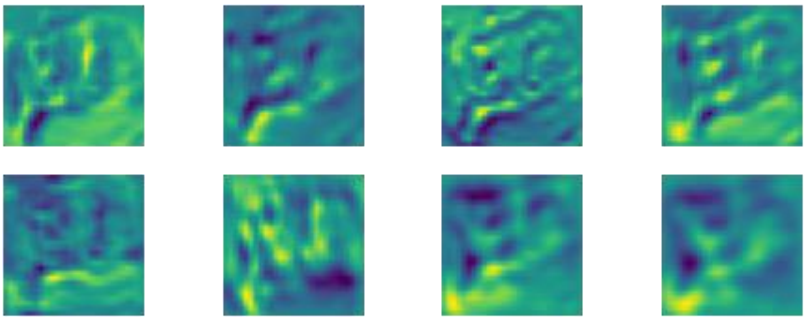


드롭 아웃을 적용했을 때 첫 번째 필터와 특성맵의 시각화입니다. 첫 번째 필터를 지났을 때는 저수준의 특성이 추출되는 것으로 보입니다.

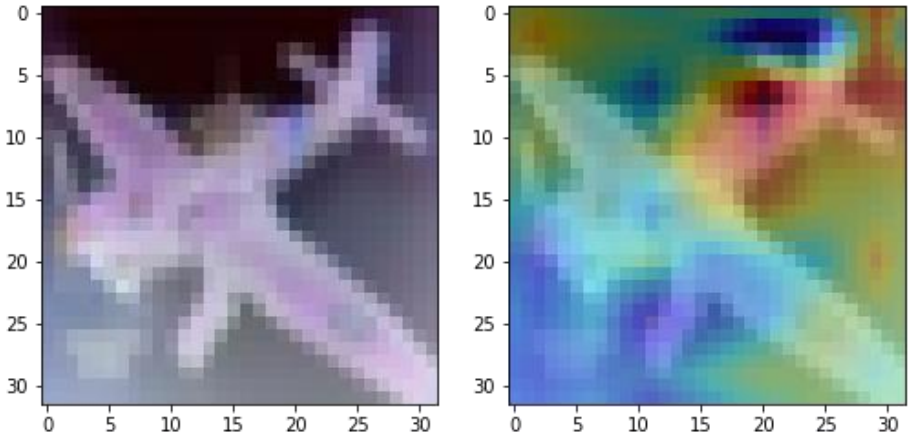


배치 정규화와 드롭 아웃을 적용했을 때 마지막 필터와 이를 지난 특성맵의 시각화입니다.

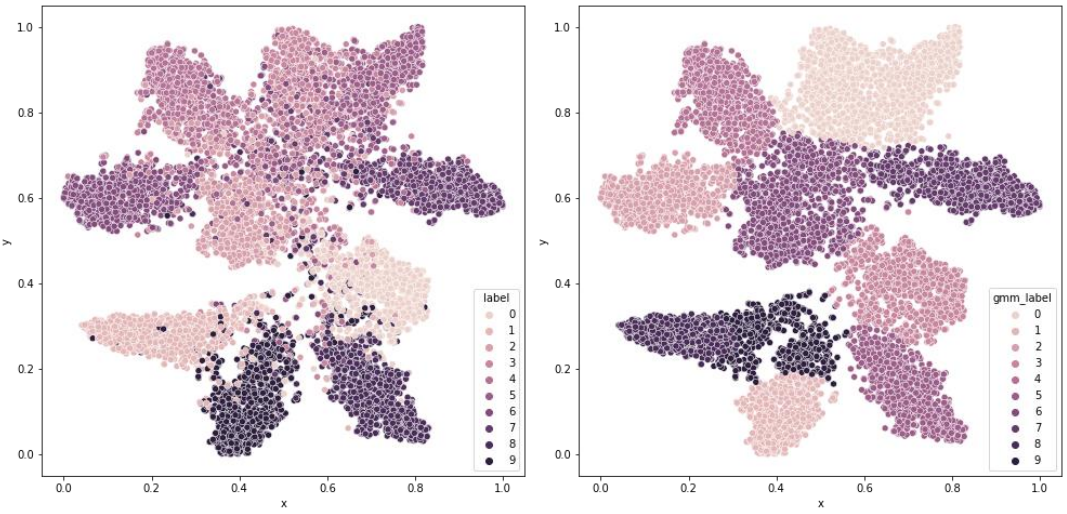
고수준의 특징을 추출하는 것으로 보입니다.



첫 번째 층을 지났을 때는 필터가 모든 부분을 보며 수직필터, 수평필터 등 저수준의 특징을 추출하는 것으로 보이며 합성곱 층을 지나면서 특정 부분인 고수준 특징을 추출하는 것으로 보입니다.

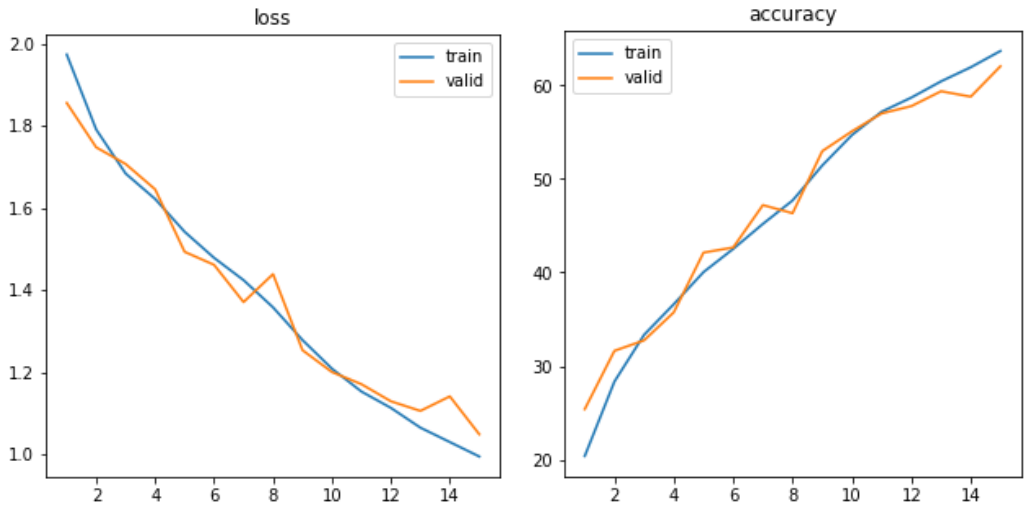


배치 정규화와 드롭 아웃을 적용했을 때 비행기의 뒷부분을 보고 비행기라고 판단한 것으로 보입니다.



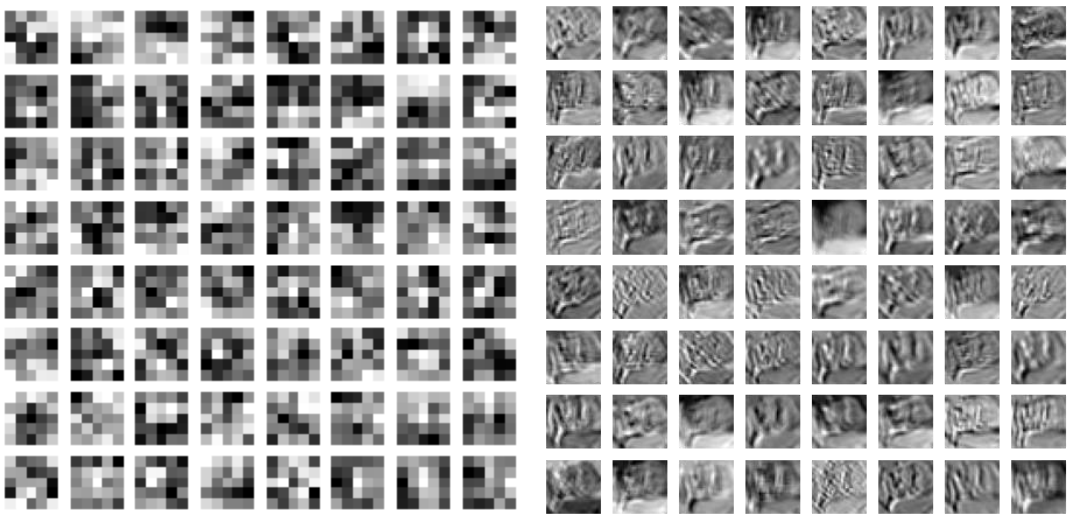
모델의 fully connected layer 의 두번 째 층의 512 차원의 벡터를 추출 해 시각화 했을 때 어느정도 군집이 잘 이뤄진 것으로 보입니다. GMM 을 적용했을 때는 실제 label 과는 다른 형태를 보이고 있습니다.

4.4 L2 REGURIZATION

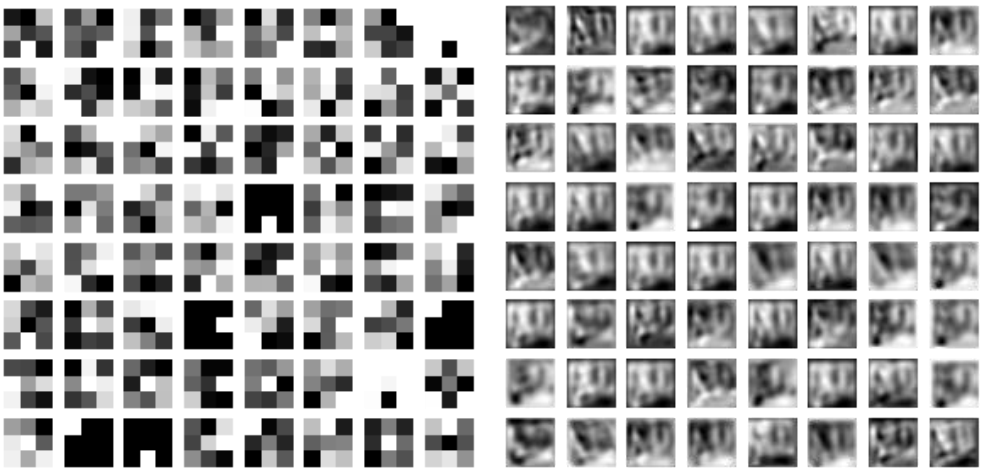


배치 정규화와 비교해 L2 regularization 이 어떻게 학습하는지 확인하고자 l2 regularization 을 적용했습니다.

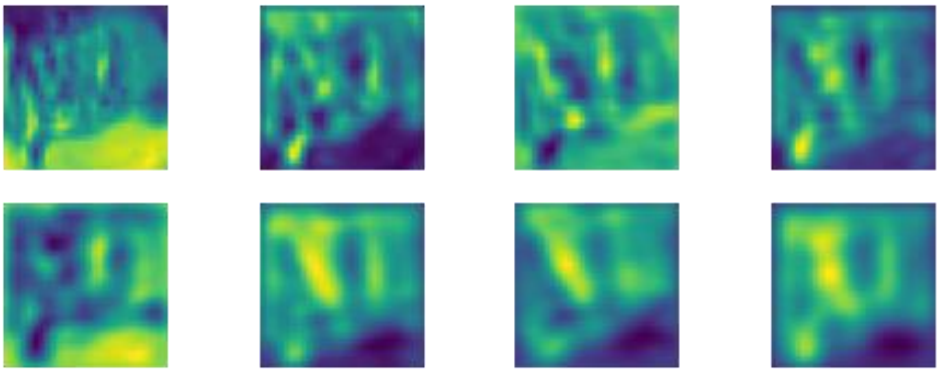
가중치의 크기를 줄이기 때문에 train loss 와 train 정확도를 보면 과소 적합되는 경향이 있지만 valid 데이터의 loss 와 정확도는 train 데이터와 비슷한 형태를 띄므로 오히려 과대 적합을 막는데는 더 효율적으로 보입니다.



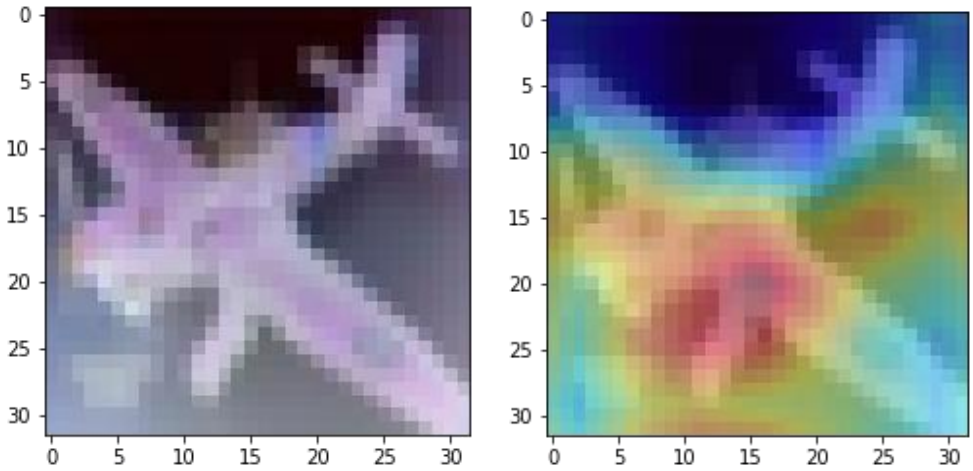
첫 번째 필터와 특성맵의 시각화입니다. 초반에는 저수준 특징을 추출하는 것으로 보입니다



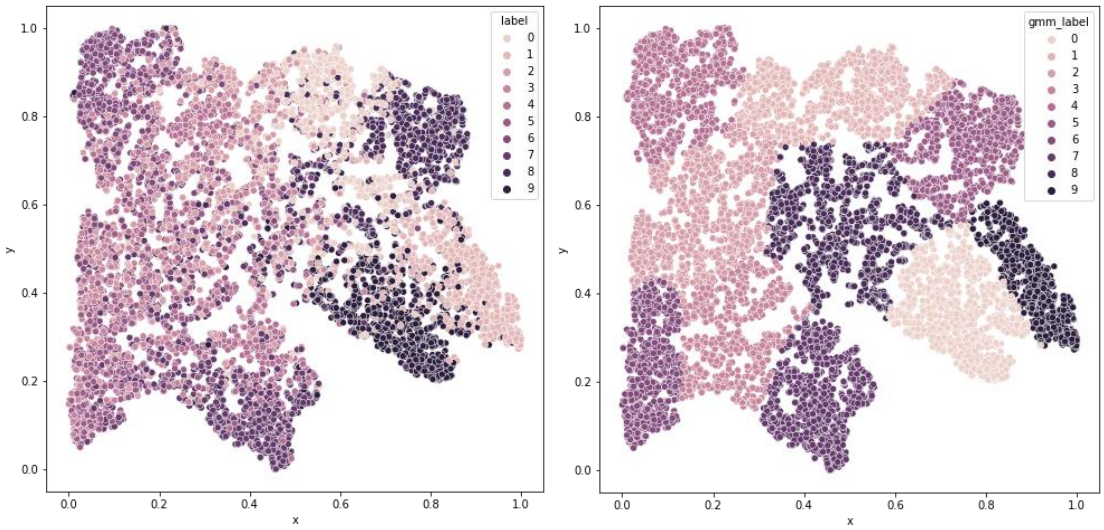
마지막 합성곱의 필터와 이를 지난 특성맵의 시각화입니다. 특성맵을 보면 고수준 특징을 추출하는 것으로 보입니다.



앞 layer 를 지날 때는 어떤 윤곽선의 저수준 정보를 추출하는 것으로 보이며 과소 적합이 되어 다양한 특성을 추출하는 것으로 보이지는 않습니다.



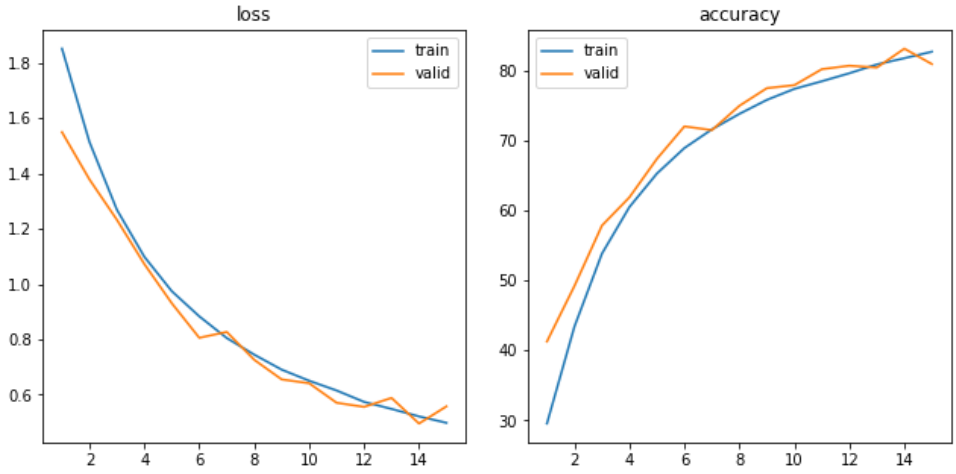
비행기의 왼쪽 날개 부분의 가중치가 높은 것으로 보입니다.



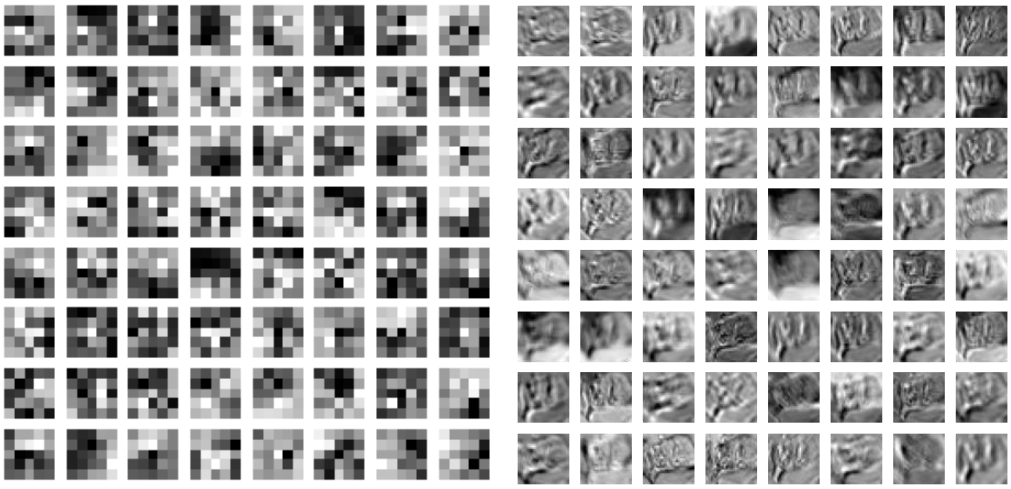
과소적합이 되는 경향을 보여 추출한 벡터가 다른 모델과 비교해 일부분만이 label 별로 잘 분류가 되는 것으로 보입니다. GMM 을 적용했을 때는 label 과는 다르게 군집되는 것으로 보입니다.

4.5 이미지 전처리 모델 시각화

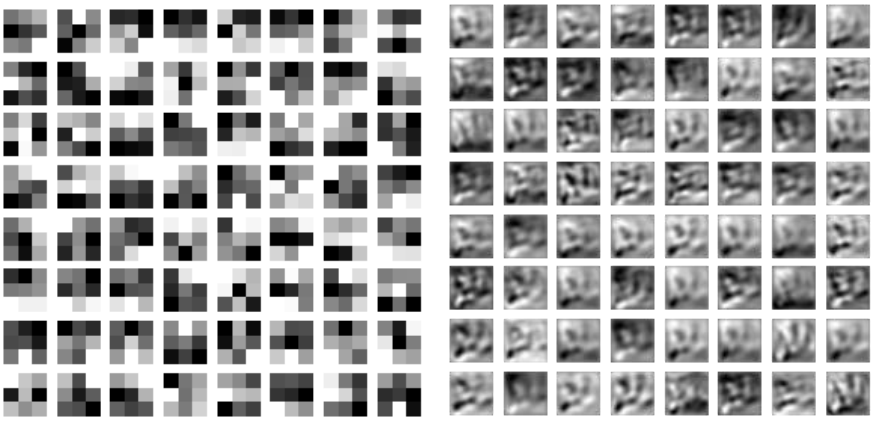
모델을 학습할 때 같은 이미지를 다양한 형태로 학습하면 더 다양한 표현을 학습하고 과대적합도 방지할 것으로 기대해 이미지 전처리를 진행해 실험했습니다. 4.1 의 일반적인 모델이 학습을 잘 하지 못하는데 이미지 변화까지 주면 학습하기가 더 어려울 것 같아 배치정규화와 함께 사용했습니다. 이미지를 전처리한 방법은 총 5 가지로 1. 상하반전 2. 회전 3. 아핀 변환 4. 일부분 crop 5. 색 변경 을 사용했습니다.



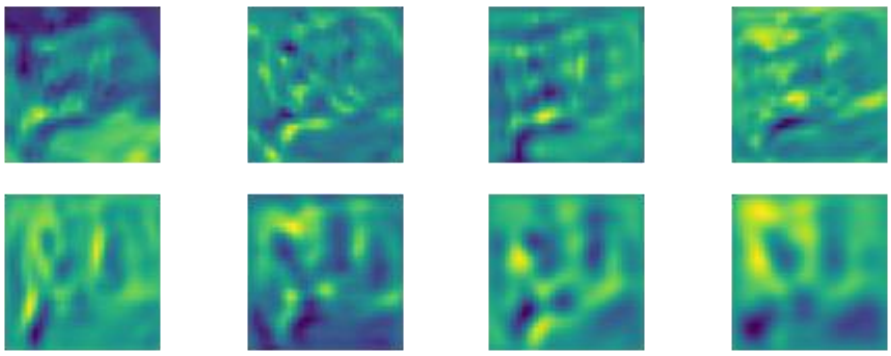
이미지를 전처리하고 모델을 학습하니 배치 정규화를 사용했을 때 보다 과대 적합이 줄어든 것으로 보이지만 과소 적합 되는 경향도 보입니다. 이러한 이유는 학습할 때는 전처리가 진행되면서 계속 바뀌는 이미지를 학습하기 때문입니다.



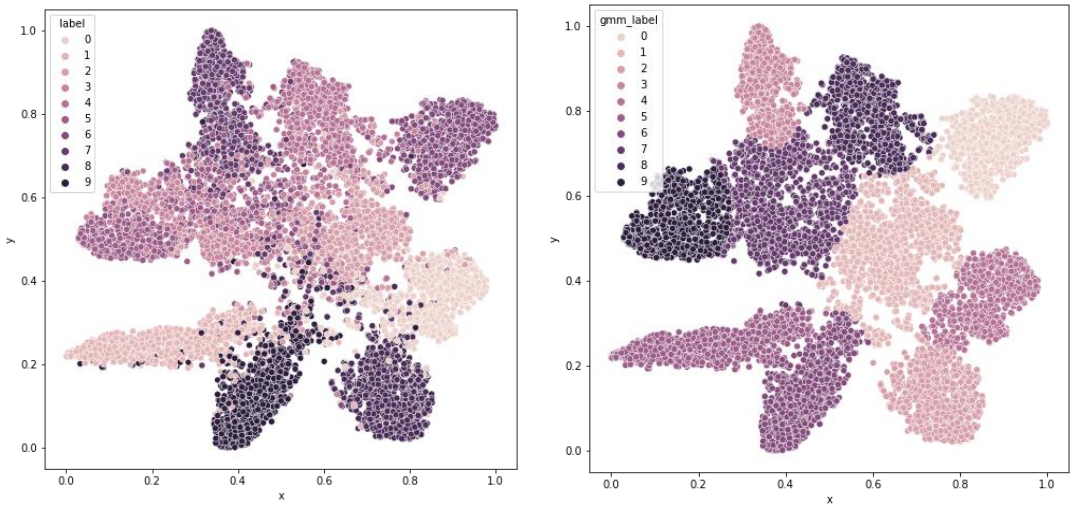
이미지 전처리를 했을 때 첫 번째 필터와 합성곱층을 지난 특성맵의 시각화입니다. 저수준의 특성을 추출하는 것으로 보입니다.



마지막 합성곱 층의 필터와 이를 지난 특성맵입니다. 여러 필터를 지날수록 고수준의 특징을 추출하는 것으로 보이지만 과소 적합으로 인해 다양한 특성을 추출하지 못 하는 것으로 보입니다.

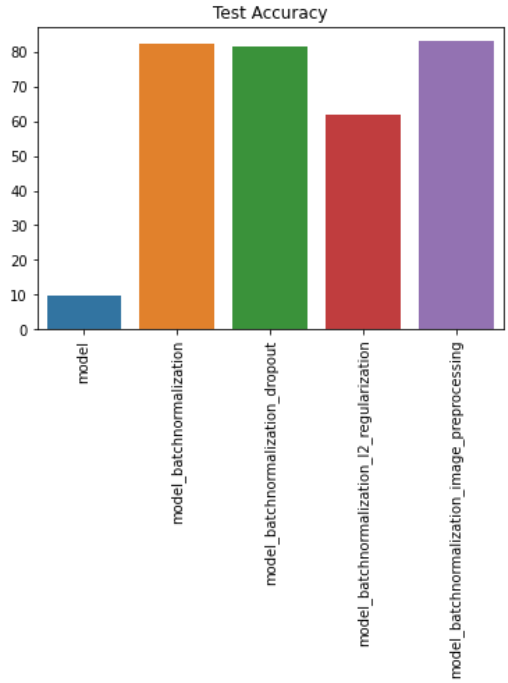


모델의 layer 를 지나면서 필터를 지난 특성맵이 고수준 특징을 추출하므로 형태를 알아보기가 어렵습니다.



이미지 전처리를 진행했을 때 label 별로 벡터들이 군집화가 잘 이뤄진 것으로 보입니다.

5 Discussion and Conclusion



학습한 모델을 Test 데이터를 이용해 예측한 정확도입니다. 배치 정규화, 배치 정규화와 드롭아웃, 배치 정규화와 이미지 전처리를 적용했을 때 80%의 비슷한 정확도를 보여주고 있습니다.

Cifar10 데이터를 활용해 CNN 기법을 적용해보고 실험했습니다. 실험을 통해 알고자 했던 것은 어떤 방법을 사용했을 때 과대적합을 막고 학습을 효율적으로 하는지에 대한 것이었습니다. L2

regularization 은 과대적합을 막는 방법으로는 좋아 보입니다. 하지만 과소적합이 되고 이를 해결하기 위해 더 복잡한 모델을 사용하거나 학습 횟수를 증가시키는 것은 연산량의 증가로 이어지기 때문에 효율적인 학습 방법이 아닌 것으로 보입니다. 따라서 배치 정규화를 먼저 적용하고 이때 과대적합이 되었다면 이미지 전처리 혹은 드롭 아웃을 추가하는 것이 학습 속도가 빠르며 과대적합을 효율적으로 막는 방법으로 보입니다.

각 모델별로 이미지의 어떤 부분을 보고 label 을 분류했는지 확인하고자 fully connected layer 의 벡터를 추출하고 T-SNE 을 사용해 2 차원으로 축소해 확인했습니다. 또한 CAM 기법을 사용해 모델이 label 을 분류할 때 어떤 특징을 참고하는지 보고자 가중치를 활용했습니다. 모델이 학습을 어떻게 했는지에 따라 조금씩 가중치를 두고 있는 부분이 다른 것으로 보였습니다. 하지만 대부분 비행기의 날개, 몸통, 뒷부분을 보고 label 을 분류하는 것으로 보였습니다.

모델을 학습할 때 데이터에 맞는 모델 구조, 활성화 함수, 가중치 초기화, 다양한 regularization 방법들을 적용해야 하는 것으로 보입니다.