Proposed Method

In short, our method can be characterized as CutMix/MixUp Augmentations, with an EfficientNet-B7 (66M) backbone along with other training techniques. In this section, we present a brief description of the architectures and methods and the application with respect to the present context. We have used Timm's open-source library to train the model as it incorporates almost all the required latest resources and techniques to improve the accuracy of the model.

CutMix / Mixup Augmentation

Cutmix is one of the augmentation methods that outperforms the state-of-the-art augmentation strategies on CIFAR and ImageNet classification tasks. The main idea behind cutmix / mixup augmentation is to replace a patch from another image instead of just erasing the pixels from an image. The ground truth labels are also proportionately mixed according to the area of the pixels. CutMix now enjoys the property that there is no uninformative pixel during training, making training efficient while retaining the advantages of regional dropout to attend to non-discriminative parts of objects. Mixup trains a neural network on convex combinations of pairs of examples and their labels. By doing so, mixup regularizes the neural network to favor simple linear behavior in-between training examples. We borrow the insights from these augmentations and try to improve the regularization of the network

EfficientNet

We have decided to use state-of-the-art ImageNet classification architecture EfficientNet. This model is a result of neural architecture search with carefully balancing of network depth, width, and resolution. In our experiments first six models from this list didn't generalize well, and then we went for EfficientNet-B7 as it matched the size description of the competition as well as gave us good accuracy on the validation data.

Overview

This approach only trains EfficientNet on all labeled data.
We use the same data augmentation strategy during training time only:
1. resize to 384 × 384 pixels,
2. random horizontal flip and vertical flip,
3. Random erase and crop percentage of 0.9.
4. Cutmix and Mixup Augmentation

Model Ensembling

We tried ensembling two EfficientNet-B6 models trained on 2 splits of the training dataset. But, the results were not comparable to the single EfficientNet-B7 model as it was able to generalize better on the dataset during our experimentation.

Experiments

Training

We implemented our models in PyTorch and used 'SAME' padding that is available in TensorFlow as it helped improve the accuracy better than the PyTorch-based padding technique. We trained the network with Adam optimizer with 0.001. learning rate and batch size of 16. We used the cross-entropy loss as our loss function. We also use the exponential moving average of the model weights to smoothen it across several epochs. Dropout and dropblock parameters were also used to train the model to improve the regularization of model. We trained our models on 2 Nvidia Tesla RTX6000 GPUs with 24 Gb memory each, which allows us to use 16 × 2 images and accumulate the batches over 8 steps. We noticed that a smaller batch size leads to a decrease in target accuracy.

Testing

During testing, we have achieved an accuracy of 48.49 on the test data which is an increase of 23% over the baseline model given.

References :-

1. Cutmix Augmentation - https://arxiv.org/pdf/1905.04899.pdf
2. Mixup Augmentation - https://arxiv.org/pdf/1710.09412.pdf
3. Dropblock - https://arxiv.org/pdf/1810.12890.pdf
4. Dropout - https://jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf
5. EfficientNet - https://arxiv.org/pdf/1905.11946.pdf