Visualizing and Understanding Convolutional Networks

1 Citation

Zeiler, Matthew D., and Rob Fergus. "Visualizing and understanding convolutional networks." European conference on computer vision. Springer, Cham, 2014.

`https://arxiv.org/pdf/1311.2901.pdf`

2 Abstract

AlexNet set state of the art on ImageNet, but it's hard to see what the filters have learned and how to improve the model. We show how a deconvulution network can help us visualize the problems and train a better AlexNet that sets the state of the art on Caltech-101 and Caltech-256.

3 Introduction

We make a technique that shows how what images each filter in AlexNet responds to. You can use this to visualize filters as the model trains. You can also occlude parts of the image and see how the filters behave. We then improve AlexNet and show that it serves as a good feature extractor for other datasets.

Prior work on visualizing convnets have used gradient descent to maximize a unit's activations, but doesn't show what the unit is invariant to. Other work uses a Hessian hessian technique to help see invariance, but it uses an unrealistic quadratic approximation.

4 Approach

Recall that AlexNet turns a 224×224 RGB image into a probability distribution over 1000 classes by using 5 conv layers (with ReLU and occasionally max-pooling), 3 fully-connected layers, and a softmax.

Consider the final conv layer. To "undo" it, we unpool, rectify, and then deconvolve - let's consider each of these. Max-pooling is noninvertile, but we approximate an inverse by outputting the input with the maximum value at its original location (other values are set to zero). To get the original location of the maximum value, we need the pooling layer to track this (called a switch) and give it to the unpooling layer. Rectification is just a ReLU. Deconvolution is simply convolution with the transposed filters.

Thus, for each conv layer we can make a deconv layer. Thus, by feeding an input image through the conv layers and then the deconv layers, we can visualize the result as an image.

5 Training Details

We use AlexNet - 5 conv layers (with ReLU and max-pooling), 3 fully-connected layers, and a 1000-way softmax. We use the same cropping/resizing technique as in the AlexNet paper. We train with SGD +

momentum with learning rate annealing and dropout. We re-normalize first-layer weights if their RMS values exceeds 10^{-1}. We train for 70 epochs, which takes 12 days on GTX580 GPU.

6 Convnet Visualization

We look at the top images that trigger activation in each filter of each layer. The second layer detects corners and edges and the fifth layer detects things like keyboards - so features are hierarchical.

Looking at the visualization after each epoch, we see the first few layers converge quickly while the deeper layers take a longer time. Filters seem invariant to translation and scaling, but not to rotation.

Visualizing the first and second layers show that we have little diversity in features (so we'll reduce feature size) and aliasing (so we'll decrease stride). Doing this improves accuracy.

We also occlude different parts of the image and see how the activation changes. This tells us what exactly in the image the filter is sensitive to. To see what parts of the image correspond to what parts of the activation, we do the following. We take the filter that responds to dog faces and take 5 pictures of dog faces. We then occlude part of the image (e.g. eyes, nose) and compute the difference between the regular and occluded version $\boldsymbol{\epsilon}_i^l = \mathbf{x}_i^l - \tilde{\mathbf{x}}_l^i$ where l is the layer and $i \in \{1...5\}$ is the image index. We then measure the consistency as $\triangle_l = \sum_{i,j=1,i\neq j} \mathcal{H}(\text{sign}(\boldsymbol{\epsilon}_i^l), \text{sign}(\boldsymbol{\epsilon}_j^l))$ (lower value means greater consistency) where $\mathcal{H}$ is Hamming distance.

7 Experiments

Our modified AlexNet (includes the improvements we made based on the visualization) sets state of the art on ImageNet. Removing two fully-connected layers or the middle two conv layers doesn't hurt error too much (but don't remove both).

We then use our modified AlexNet as a feature extractor and train a softmax classifier for other datasets. We set state of the art on Caltech-101 and Caltech-256. We also do quite well on PASCAL VOC.

We consider the effect of using a subset of the layers when computing features for the feature extractors. The more layers you add (up to the full 7 layers we consider), the better the classification accuracy.

8 Discussion

We demonstrated a deconvolution approach for visualizing activation of filters. This guided us to make improvements to AlexNet, which sets state of the art on ImageNet and acts as a good feature extractor for other classification tasks.