What's the story

Concept Learning - understanding the dynamics of relationship between objects - like person, dog and leash. This is

done by attempting to localize on of hand-holding leash and attached-to-leash part of the image.

What are the challenges in localizing on hand-holding-leash - the small object localization problem.

Learning by use of context. Use other examples.

Introduction

Because of complexity and variability in static images, semantic understanding of an image has remained one of the most difficult problem to solve in computer vision. While human eye is trained to rapidly cull relevant information from an image to build a language level understanding of the scene, such feat has remained elusive artificial intelligence systems. Understanding relationships between objects in an image can help understand the semantic meaning of an image as it provides evidence for a particular situation being present in the image. For example, the ability of an artificial intelligence system to localize on a "hand-holding-leash" relationship in an image provides a fairly good clue that the image most likely has a dog and a person walking the dog and this information can be used to generate semantic understanding of the image.

In this work we focus to localize on hand-holding-leash relationship in dog walking images.

Only prototypical images

Deep learning, CNN

Deep Learning/CNN

What is deep learning/CNN

How is it used for regression

Why could this be a good fit for this problem

Neural networks and its more recent variant deep Convolutional Neural Networks (CNN) have achieved significant improvement in state-of-the-art for classification as well as regression image processing tasks since its resurgence in 2012 when a neural network based architecture, called AlexNet [Krizhevsky et al.], was proposed for image classification of ImageNet dataset. CNN were created to overcome the scaling problem of traditional neural networks. In traditional neural networks neurons in one layer of the network are connected to all the neurons in adjacent layers, which makes it difficult to scale for intelligent image understanding and analysis. A CNN model consists of many layers (Figure 1) and each layer learns features or representations at increasing level of abstraction. Neurons (also called filters or kernels) in each layer of the CNN are connected only to a small region of the input and are three dimensional (Figure 2).

The height and width of the filter determines the region of spatial connectivity of the filter to a layer before. Depth of the filter indicates how many such filters are there in what's called a filter bank. A filter contains real numbered weights. The dot product of these weights with the input region that it's spatially connected to creates output activation map (also called feature map), as depicted in Figure 2 below. A filter convolves (slides) across it's input with a predefined stride. As illustrated in Figure 1, the input is zero padded if the filter does not align with the image at the edges as it convolves. The weight in the filter remains unchanged or is shared as convolutions are performed. The rationale of sharing weights is twofold: first it reduces the number of parameters in the model, and secondly any useful features identified in one part of the image can be re-used everywhere else without having to be independently learned [Murphy 2012 (Machine Learning, section 16.5.1)].

|  |  |
| --- | --- |
| nput  24x24  Convolution  Feature maps  4@20x20  Feature maps  4@10x10  Subsampltng  Feature maps  8@8x8  Convolution  Feature maps  8@4x4  Subsampling  Output  20@1x1  Convolution | Macintosh HD:Users:sharadkumar:Library:Group Containers:UBF8T346G9.Office:msoclip1:01:14B695D6-3ACD-E841-8639-83AE8564CD36.png |
| Figure 1: Explain    *From:* [*https://www.slideshare.net/hanneshapke/introduction-to-convolutional-neural-networks*](https://www.slideshare.net/hanneshapke/introduction-to-convolutional-neural-networks) | Figure 2: Illustrates filter's transformation of the input into output. The figure illustrates two-dimensional input, but usually the input also has a third dimension of depth which is matched by filter's third dimension |

***Forward Pass***

There are mainly three types of components in a CNN layer - convolution, Rectified Linear Unit (ReLU) and pooling - and the layers are arranged in a sequence, as depicted in Figure 1. Each CNN layer transforms one volume of activations into another volume during the forward pass of processing. Filters are mainly involved in computation in the convolution layer and process its input as explained above. ReLU layer introduces non-linearity into the model by applying *max (0,x)* function element-wise to the activations produced by the filters in the convolution layer. Use of *max (0,x)* non-linear function results in accelerated training as compared to logistic or hyperbolic tangent function [Krizhevsky, ImageNet Classification with Deep Convolutional Neural Networks], where the gradient in the saturating part of the activation function graph becomes very small (for very large or very small weights) resulting in sluggish training because of minuscule weight change during back-propagation. Introducing non-linearity with ReLU ensures that the CNN model does not collapse into a large linear model [Murphy 2012 (Machine Learning, section 16.5.1), Hastie et al., Essentials of Statistical Learning (Section 11.3)]. Neural networks can therefore be thought of as a nonlinear generalization of the linear model and by introducing the nonlinear transformation it greatly enlarges the class of linear model [Hastie et al., Essentials of Statistical Learning (Section 11.3)].

Activations from the ReLU layer are subsampled by the pooling layer. As shown in Figure3, pooling layer subsamples the activations spatially and is usually done either by averaging or by computing the max over a small window of activations produced by the ReLU layer. Subsampling reduces the number of parameters in the model and hence the amount of computation in the network, it also results in small shift invariance as the image activations propagate through the network. Because of the subsampling, there's a progressive reduction in size of activations, as shown in Figure 1.

|  |
| --- |
| 5 9  12 7  Average Pooling  •aan  21 12  18 10  Max Pooling |
| Figure 3: Explain    *From:* <https://medium.com/@Aj.Cheng/convolutional-neural-network-d9f69e473feb> |

***Back Propagation***

Weights in the filters are the parameters of a CNN model and model fitting involves learning these weights using back-propagation as forward and backward passes are made over the network. Back-propagation is a optimization algorithm where the goal is to find a set of value for model parameters that generalizes the learning in such a way that the model achieves superior classification or regression accuracy on test dataset. This goal is achieved by minimizing the error on training dataset by using gradient descent (or its variant) method for optimization. Validation dataset complements the training dataset to find training hyper-parameters that generalizes learning. Stochastic gradient descent is one of the most effective techniques for back-propagation where gradient of the cost function and chain rule of derivatives are used to update model weights iteratively in minibatches. Use of minibatches in stochastic GD accelerates the training and also helps to obtain unbiased estimate of the gradient by taking the average gradient on a minibatch, assuming examples in minibatches are drawn in identically and independently distributed (IID) fashion [Goodfellow et al. (Deep Learning, 2016, Section 8.3)].

Challenges of overfitting: regularization, momentum, dropout

Weight initialization

Papers used and explored

Little Landmark - Singh

VGG-16 paper

Alexnet paper

Deconvnet paper - Fergus

Dropout - Hinton

Visualizing and Understanding Convolutional Networks - Fergus

ROC Paper

Learning Invariant Feature Hierarchies - Yann LeCun (<http://yann.lecun.com/exdb/publis/pdf/lecun-eccv-12.pdf>)

ROC Curve?

Related work

RCNN for small objects

Improving Small Object Proposals for Company Logo Detection

A CLOSER LOOK: SMALL OBJECT DETECTION IN FASTER R-CNN

Main Paper

Explain architecture/model

Why was max pooling not used

Explain with help of pictures how the coordinates are localized in three steps

Explain preprocessing

Explain how out\_locs values are created - using calculation from Stanford's notes

Performance of version with distortion and without distortion. Right now the training results I have are all with distortion. Answer the question why are distortions done.

Explain Performance Metric

Performance Metrics for Evaluating Object and Human Detection and Tracking Systems

My implementation of the model

Tensorflow/Python

My Models and Experiments

Discuss what transfer learning is

Different layers of the network learn different types of abstractions (explain more). Murphy pp 995 - explain transfer learning using this

Discuss what VGG-16 model is and how was transfer learning incorporated in the model

Discuss Results

It took 27 hours to achieve same result with transfer learning

Training the entire model vs. fine tuning

Explain two models

First for localizing hand-holding-leash

Second for extending the model for identifying two small objects in one training

Explain the vanishing gradient problem that happened in getNormalizedLocationWeightsFast() function and how it was solved by using normalization. Overlapping Pooling may have solved this problem (ImageNet Classification with Deep Convolutional Neural Networks, section 3.4)

Explain the problem of over-fitting

that it may be because the context around hand holding leash is not sufficient for localizing

It may be because of not enough data and transfer learning did not help in this case

Use pictures to show where latent coordinates and the final coordinates fall during training, which is pretty close

Discussed ideas used to solve the over-fitting - different hyper-parameters (momentum, weight decay and dropouts)

dropout also has the amazing property that it prevents co-adaptation of feature detectors which improves a network’s ability to generalize (<https://dmm613.wordpress.com/2014/10/15/intriguing-properties-of-neural-networks/>)

Discuss about use of different types of optimizers - Adam and SGD

In this case fine tuning is done to the columns. The assumption is that VGG 16 model is generalized enough and does not have overfitting problem. Because of this the weights in the VGG-16 model are not changed.

Miscellaneous

Include discussion about bias and variance and explore if that can be used in explaining something