|  |
| --- |
| Convolutional Neural Network |
| Case Studies |
| Practical advices for ConvNets |
| Detecting Algorithm |
| Special Application : Face recognition and Neural Style Transfer |

|  |
| --- |
| **Computer Vision**  Computer vision is one of the evolving area in deep learning. There are many applications has been built using CV which we could not think few years before to be possible.  Some of the computer vision problems are as follows:  What triggered CNN in computer vision?  We have described the image recognition problem using 64x64x3 size image.  But in practice it is a very small image. In CV we have very large image size that can even go to 1000x1000x3 pixels.  In this case as depicted we might need to consider 3B parameters. Which can cause overfitting and need huge computational and storage requirement.  To address this problem the concept of convolutional neural network has come.  Which is the primary building block in computer vision field.    Even if we are not dealing with computer vision problem some of the concept we can apply from computer vision field to the other area as well. |
| **Vertical Edge detection**  Let say we have a 6x6 pixel image which is represented below as matrix. We have selected another special matrix of size 3x3 with specific values in each columns (1st all 1s , middle all 0s and last all -1s). This matrix is called filter.  We use this filter to have the convolution with the source image and get a convolved image of size 4x4  This filter is called vertical edge detecting filter. As shown in the picture below if we use this filter on a picture having only bright and dark region then it could detect the transition / edge in the picture.  Like wise we have several types of filters. Which is the backbone of the CNN.  Intuitively we can say that the edge detecting filter reduces the value if there is no transition along vertical side and increased value if there is transition along vertical direction. |
| **More edge detection** |
|  |
| **Padding**  It is a technique of adding extra rows and columns in all side of the image matrix. Those cells are filled with 0s.  Why we need padding?   * As seen in the previous example if we convolve 6x6 image with an 3x3 matrix then the o/p becomes 4x4. So, for deep network the size of the images reduces and becomes very small. * Also, if you see the convolution operation gives more priority to the central cells and less to the corner of edges. This can cause loss of useful information that is distributed over corner or edges     Formula:  O/p size: n+2p-f+1 (n: size of source, p: padding, f: filter size)  **Types of Convolution:**  Valid convolution: if padding is zero  Same convolution: p chosen such that input size equals output size. |
| **Stride Convolution**  Stride in relation to the convolution of CNN signifies the shift of filter from one computation to next both in horizontal and vertical direction.  **Formula**:  O/p size: floor((n+2p-f)/s+1) (n: size of source, p: padding, f: filter size, s: stride) |
| **Convolution over volume / 3D convolution:**   * Like we consider the example of grey scale image what if we want to convolve an RGB image. In that case we need to use the same number of channels for the filter * We can use same filter / different filter per channel as per need * The o/p will be 1D image feature     **Multiple Filters:**   * In CNN architecture it is common to use multiple filters (also called feature detectors) at any convolution level * It helps to extract more features in same step * Since the volume convolution gives 1D o/p. The o/p of the multiple filters convolution gives a multi-dimensional o/p where the #channels / depth of the output would be equals # filters / #feature detectors |
| **One Layer of Convolution:**   * Filters can be assumed as the weights of the logistic regression. * We can add bias term and apply non-linearity (Relu) * The o/p of the Relu would be output of the convolution operations (1 layer) * The number of filters used decides the depth or #channel in the o/p image     **Summary of notation:** |
| **Example of simple ConvNet**    **Types of Layers in ConNet**   * Convolution Layer * Pooling Layer * Fully Connected Layer |
| **Pooling**  Pooling is the technique of reducing the feature (o/p image of conv layer) size.  So that it would be computationally more efficient  There are different types of pooling (Max, Min , Avg). However, most common type of pooling in modern CNN architecture in Max pooling  In case of #channels more than 1 pooling is applied at each layer independently  Example: |
| **CNN Architecture Example: LeNet – 5** |
| **Why look at case studies?**   * In CV some of the CNN architecture which worked for one problem or task can be useful for other areas as well * Even if we are not working on computer vision problem the knowledge of idea from the case studies can help in solving problems in other areas/ develop applications for other areas   Some of the classical CNN architecture to read:   * LeNet – 5 * AlexNet * VGG   Advanced Architecture:   * ResNet (Residual Network)   **LeNet – 5**     * In paper sigmoid/tanh activation was used. Because use of Relu as non-linear activation was not common that time * Average pooling is used which is also not used much in modern era     **AlexNet**   * The architecture is like LeNet however it is more deeper and training 60 M parameters * Relu is used as activation function * Trained in multiple GPUs (due to computational constraint back then) * Local response normalization was used which is later not used much as was not very helpful     **VGG – 16**     * Architecture is simple as it follows same filters and pooling across * Much bigger than AlexNet as it is training 138M parameters * **VGG – 16** name is due to layer count is 16. There is another variant VGG -19 |
| **ResNet**   * It is very difficult to train very deep network due to vanishing or exploding gradient problem * ResNet uses residual block which helps it to train very deep network * There are some additional connection in the architecture which feeds input to the deeper layer before activation function as stated below        * In traditional deep network the training error goes up instead of going down with increase of layers which theoretically should go down * But in ResNet the error goes down with increase of layers due to use of residual blocks |
| **Why ResNet work?**   * The skip connection is added so that the network will learn identity function if there is no additional learning * In the illustration below if we apply weight and bias decay then in deep network the weights and bias become very small or nearly zero even then it will output the activation from skip connection * Generally, we see that the convolution in ResNet is used same convolution so that there is no size mismatch between skip connection activation and the current linear function o/p * If there is size mismatch, then it can be passed through another layer to have the same dimension |
| **Network in Network / 1x1 Convolution**   * If the source is of one channel, then it might appear we are multiplying one scalar value to all cells * However, if we consider one volume having multi-dimension then it make sense. Because it consider one slice form the volume and apply linear and Relu function * So this can be added to add non-linearity into system      * It can even be used to shrink / expand the size of the o/p |
| **InceptionNet**   * It uses all types of filters for convolutions and stacks them all to let network learn which representation is better * Computationally very expensive due to 5x5 convolution having o/p 28x28x128 * Alternet approach can be taken to achieve the same result |
| **Transfer Learning**   * Transfer learning is a way of training deep neural network where we take one pre-trained model and just train some high level layers of it to adapt this for some other application * It is very useful as we don’t need to train a very big network with lots of dataset that needs resources, time * If we have very less amount of data for our problem and we find one model which work on similar kind of problem we can follow the trasfer learning * In some cases only the classification layer is removed and FC are trained, whereas in other cases we might train some end layers keeping initial layers intact * Or if we have resources and dataset available we can train entire network |
| **Data Augmentation** |
| **State of Computer Vision**   * Less availability of data needs more Hand Engineering * Still there are not enough data in field of computer vision |
| **Tips for doing well in benchmark/winning competition**   * Ensembling : Training multiple network in parallel and taking average of their o/p. This kind of technique might be helpful for winning competition but never used in production * Multi-crop at test time: Data augmentation by cropping multiple times the same image. |
| **Detecting Algorithms**  **Object Localization**  Object localization is the technique by which we can detect the object as well as the location of the object in the image. Ideally the object is enclosed by a bounding box to denote the location of the object.  So far, we have seen techniques to classify an object in a picture. Generally, there is one image in the centre of the picture.  Image classification |