Recent Architectures of Deep Convolutional Neural Networks..

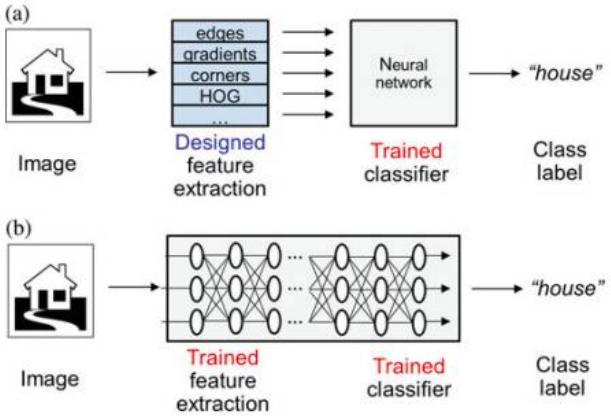


Fig. 1.2 Comparing the (**a**) classical approach to machine learning using hand-designed features to the (**b**) multi-layered representational approach of deep learning operating on raw inputs

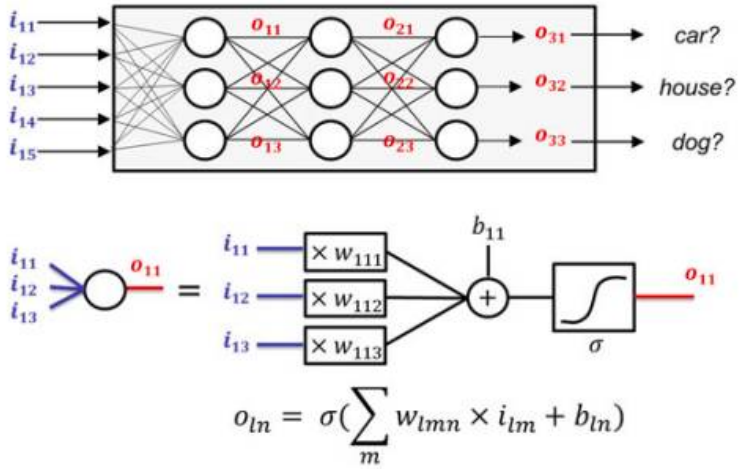


Fig. 1.4 Graphical representation of a simple deep feed-forward neural network

$$O = a\left(\sum_{m=0}^{M} W[m] \times x[m] + B\right)$$

Universal approximation theorem (Hornik et al. 1989) :

a neural network with 1 hidden layer can approximate any continuous function for inputs within a specific range. If the function jumps around or has large gaps, we won't be able to approximate it.

- The training algorithm might get stuck in a local minimum or it could converge to the wrong function when it is overfitting.
- deeper networks suffer from vanishing gradients (Hochreiter et al. 2001)
- In general, the giant networks that might be necessary to learn complex nonlinear functions are infeasible to train sometime.

CNN Literature

- 1989 to date.
- These improvements can be categorized as parameter optimization, regularization, structural reformulation, etc.
- main thrust in CNN performance improvement came from the restructuring of processing units and the designing of new blocks.

CNN Literature

- Point of focus:
  - spatial exploitation
  - Depth
  - multi-path
  - Width
  - feature-map exploitation
  - channel boosting
  - attention-based CNNs.

Spatial Exploitation based CNNs

- CNNs have a large number of parameters and hyper-parameters, such as weights, biases, number of layers, and processing units (neurons), filter size, stride, activation function, learning rate, etc.

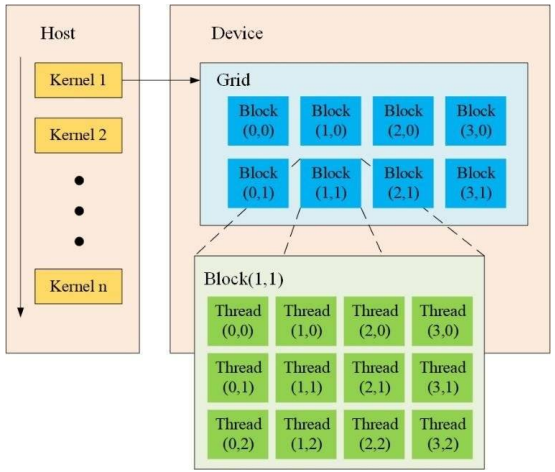
- As convolutional operation considers the neighborhood (locality) of input pixels, therefore different levels of correlation can be explored by using different filter sizes.

- Different sizes of filters encapsulate different levels of granularity; usually, small size filters extract fine-grained and large size extract coarse-grained information.

LeNet

- LeNet was the first CNN architecture, which not only reduced the number of parameters but was able to **learn features from raw pixels automatically**.
- showed state-of-the-art performance on hand digit recognition tasks
- ability to classify digits without being affected by small distortions.
- **LeNet is a feed-forward NN that constitutes of five alternating layers of convolutional and pooling, followed by two fully connected layers**
- exploited the underlying basis of the image that the neighboring pixels are correlated to each other and feature motifs are distributed across the entire image.
- Therefore, convolution with learnable parameters is an effective way to extract similar features at multiple locations with few parameters.
- Learning with sharable parameters

Limitation

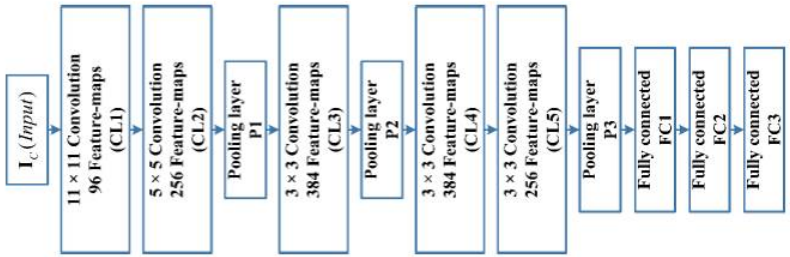


- GPU was not commonly used to speed up training, and even CPUs were slow
- considers each pixel as separate input and applies a transformation on it, which was a substantial computational burden
- CNN was limited to hand digit recognition tasks and didn't perform well to all classes of images.

GPU advantage:

- **A CPU consists of four to eight CPU cores, while the GPU consists of hundreds of smaller cores.**
- **massively parallel architecture**

AlexNet



Basic layout of AlexNet architecture showing its five convolution and three fully connected layers.

- AlexNet is considered as the first deep CNN architecture, which showed groundbreaking results for image classification and recognition tasks
- In AlexNet, depth was extended from 5 (LeNet) to 8 layers to make CNN applicable for diverse categories of images.
- depth improves generalization for different resolutions of images but, the main drawback associated with an increase in depth is overfitting.

Overfitting:Bias and Variance

Bias is the algorithm's tendency to consistently learn the wrong thing by not taking into account all the information in the data.

Variance is **an error from sensitivity to small fluctuations in the training set**

- *underfitting: High Bias and High Variance*
- Overfitting occurs if **the model or algorithm shows low bias but high variance**

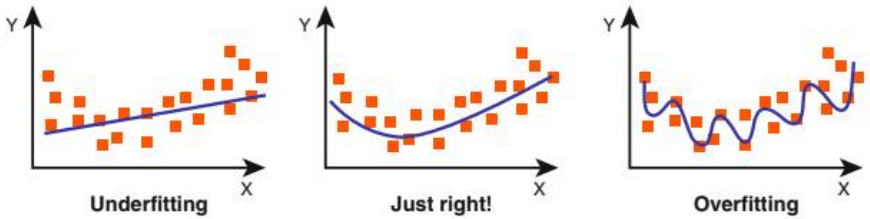
AlexNet Features

- **ReLU** was employed as a non-saturating activation function to improve the convergence rate by alleviating the problem of **vanishing gradient** to some extent .
- Overlapping subsampling and local response normalization were also applied to improve the generalization by reducing overfitting.

- use of large size filters (11x11 and 5x5) at the initial layers, compared to previously proposed networks.
- Due to the efficient learning approach of AlexNet, it has significant importance in the new generation of CNNs and has started a new era of research in the architectural advancements of CNNs.

Regularization

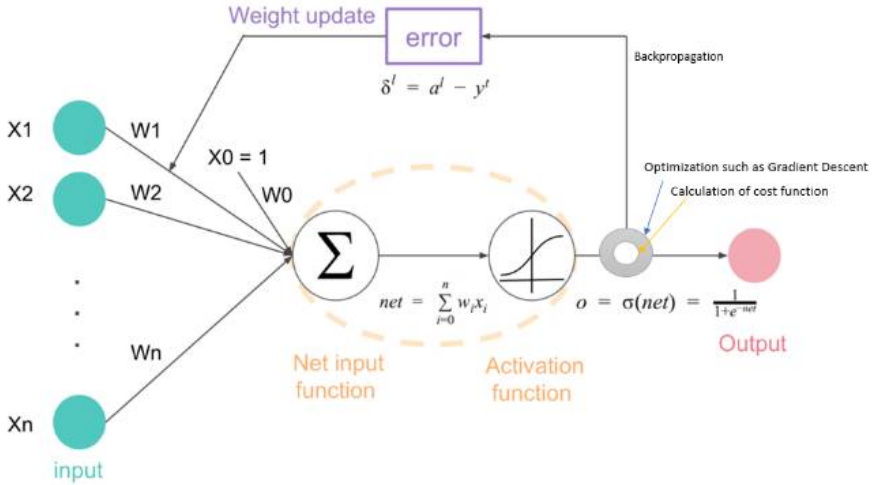
A network is overfitting, when it performs well on its training data, but not on new unseen inputs.



Solutions:

- Data augmentation : fake data is generated and added to the training set.
  - not for many other tasks such as natural language processing.
- Parameter norm penalties : a penalty is added to the loss function to optimally limit the capacity of a model.
- Dropout : some connections are randomly put to zero every iteration.
- k-Fold Cross-Validation: Cross-validation is a resampling procedure used to evaluate machine learning models on a limited data sample.

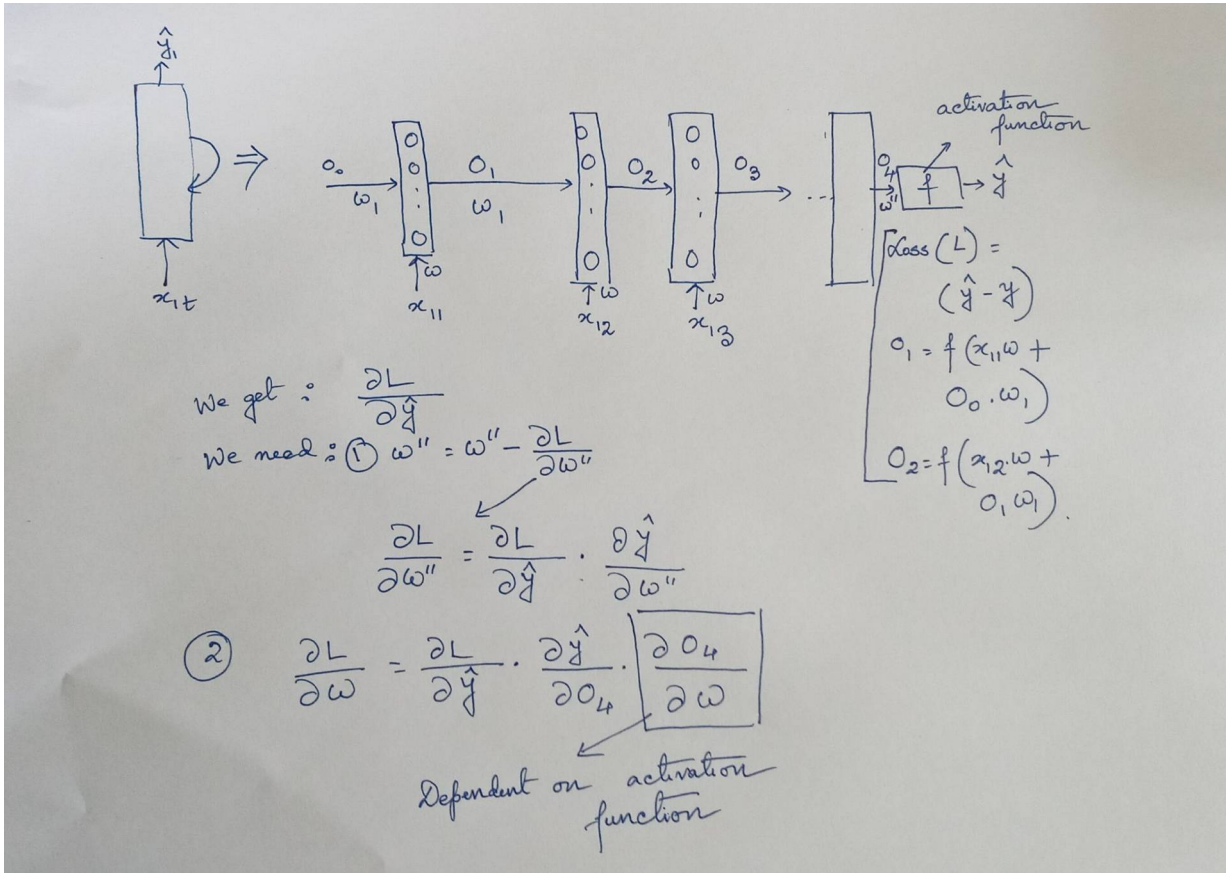
Vanishing Gradient Problem



Deep networks are hard to train because of the notorious vanishing gradient problem — as the gradient is back-propagated to earlier layers, repeated multiplication may make the gradient infinitively small.

As a result, as the network goes deeper, its performance gets saturated or even starts degrading rapidly.

Vanishing Gradient Problem



Other adjustments in AlexNet

-use of large size filters (11x11 and 5x5) at the initial layers, compared to previously proposed networks

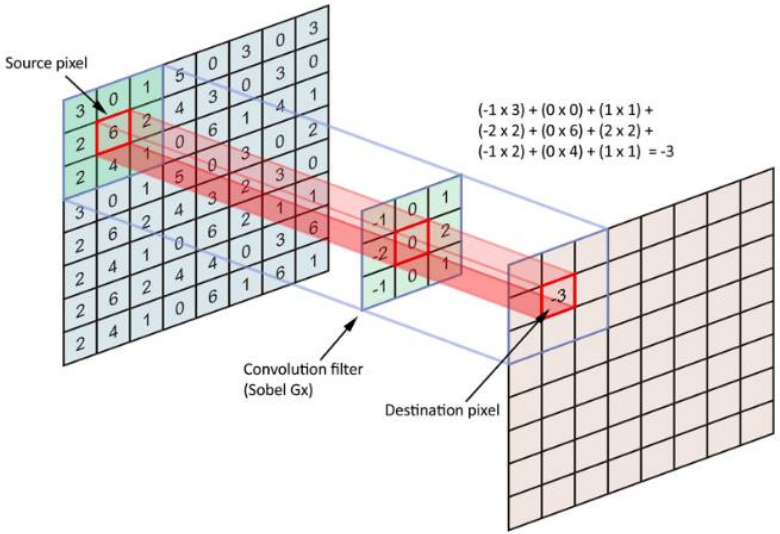
Due to the efficient learning approach of AlexNet, it has significant importance in the new generation of CNNs and has started a new era of research in the architectural advancements of CNNs.

Choice of filter size*

- *In a convolution, a convolution filter slides over all the pixels of the image taking their dot product [linear combination of the pixels weighted by the convolutional filter extracts some kind of feature from the image]
- Most of the useful features in an image are usually local and it makes sense to take few local pixels at a time to apply convolutions.
- Most of these useful features may be found in more than one place in an image: slide
- single kernel all over the image to extract same feature in different parts of the image
- Small kernel instead of a fully connected network:
  - benefit from weight sharing and reduction in computational costs. *

https://towardsdatascience.com/deciding-optimal-filter-size-for-cnns-d6f7b56f9363

Why smaller size and odd number?



- we are trying to encode the pixels in the neighborhood of anchor/source pixel
- source pixel is the anchor point at which the kernel is centered and we are encoding all the neighboring pixels, including the anchor/source pixel
- .Odd sized kernel is symmetrically shaped (not symmetric in kernel values)

- Asymmetric kernels lead to distortion/ errors

ZfNet - 2013

- learning mechanism of CNN, before 2013, was based mainly on hit-and-trial, without knowing the exact reason behind the improvement.
- In 2013, Zeiler and Fergus proposed an interesting multilayer Deconvolutional NN (DeconvNet), which got famous as ZfNet.
- ZfNet was developed to visualize network performance quantitatively. [
- The idea of the visualization of network activity was to monitor CNN performance by interpreting neuron’s activation.]
- DeconvNet works in the same manner as the forward pass CNN but reverses the order of convolutional and pooling operation.
- This reverse mapping projects the output of the convolutional layer back to visually perceptible image patterns, consequently gives the neuron-level interpretation of the internal feature representation learned at each layer.

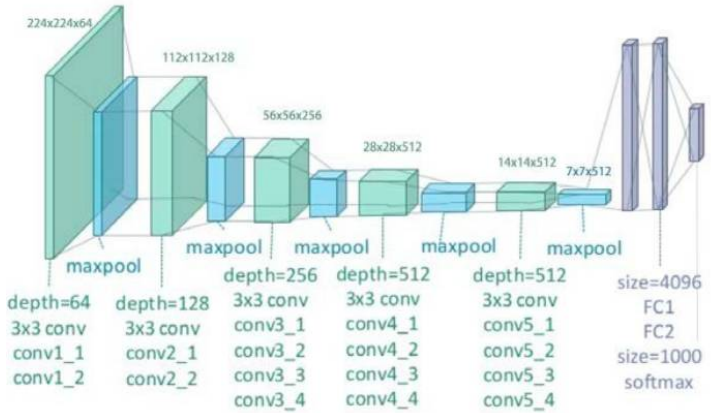
ZfNet - 2013

- idea of feature visualization proposed by ZfNet was experimentally validated on AlexNet using DeconvNet,:

  Few observations:

- which showed that only a few neurons were active.
- In contrast, other neurons were dead (inactive) in the first and second layers of the network.
- maximized the learning of CNN by reducing both the filter size and stride to retain the maximum number of features in the first two convolutional layers. This readjustment in CNN topology resulted in performance improvement, which suggested that features visualization can be used for the identification of design shortcomings and for timely adjustment of parameters.

VGG-2015*

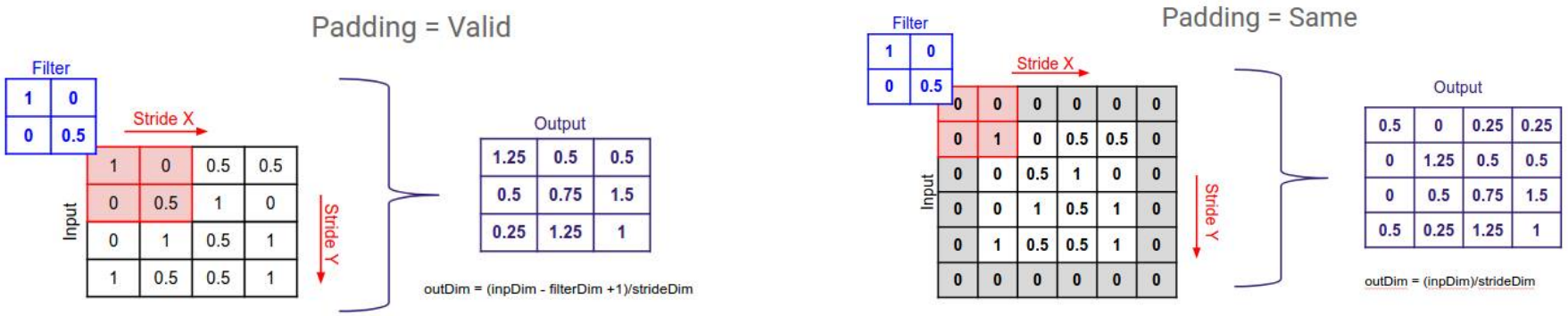


- The successful use of CNNs in image recognition tasks has accelerated the research in architectural design.
- Simonyan et al. proposed a simple and effective design principle for CNN architectures: VGG
- modular in layers pattern VGG was made 19 layers deep compared to AlexNet and ZfNet to simulate the relation of depth with the representational capacity of the network
- *https://arxiv.org/abs/1409.1556v6

VGG Improvements

- **ZfNet suggested that small size filters can improve the performance of the CNNs.**
- **Based on these findings, VGG replaced the 11x11 and 5x5 filters with a stack of 3x3 filters layer and experimentally demonstrated that concurrent placement of small size (3x3) filters could induce the effect of the large size filter (5x5 and 7x7).**
- Soft-Max: Softmax is **a mathematical function that converts a vector of numbers into a vector of probabilities [suitable** multi-class classification tasks**]**
- For the tuning of the network, max-pooling is placed after the convolutional layer, while padding was performed to maintain the spatial resolution

Why Padding is important?



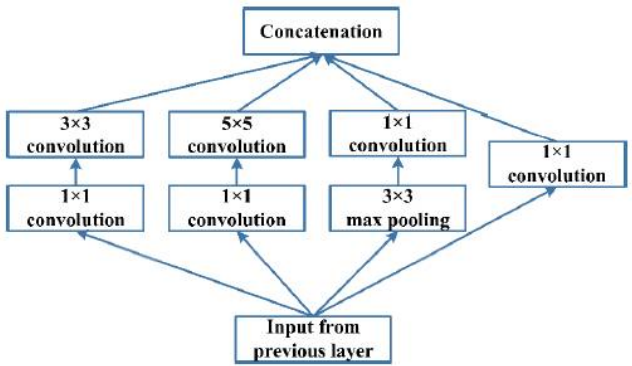
1. **Maintained the dimensions of the input**: It's easier to design networks if we preserve the height and width
2. Without padding, reduction in volume size would reduce too quickly.
3. Padding actually **improves performance by keeping information at the borders**.

- VGG showed good results both for image classification and localization problems.
- VGG was at 2nd place in the 2014-ILSVRC competition but, got fame due to its simplicity, homogenous topology, and increased depth.

- The main **limitation** associated with VGG was the use of 138 million parameters, which make it computationally expensive and difficult to deploy it on low resource systems.

Winner of 2014-ILSVRC competition : **GoogleNet**

GoogleNet-Inception Block



Basic architecture of the inception block showing the split, transform, and merge concept.

- also known as Inception-V1
- objective of the GoogleNet architecture was to achieve high accuracy with a reduced computational cost.
- incorporates multi-scale convolutional transformations using **split, transform and merge idea**.
- conventional convolutional layers are replaced in small blocks similar to the idea of substituting each layer with micro NN as proposed in Network in Network (NIN) architecture (Lin et al. 2013).
- Inception block encapsulates filters of different sizes (1x1, 3x3, and 5x5)

*https://machinelearningmastery.com/introduction-to-1x1-convolutions-to-reduce-the-complexity-of-convolutional-neural-networks/

Depth based CNN

Deep CNN architectures are based on the assumption that with the increase in depth, the network can better approximate the target function with a number of nonlinear mappings and more enriched feature hierarchies.

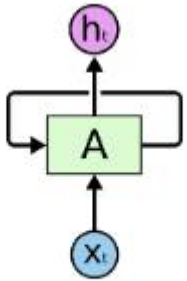
Theoretical studies have shown that deep networks can represent certain classes of function more efficiently than shallow architectures

Limitation: Slow training and convergence speed

Highway Networks

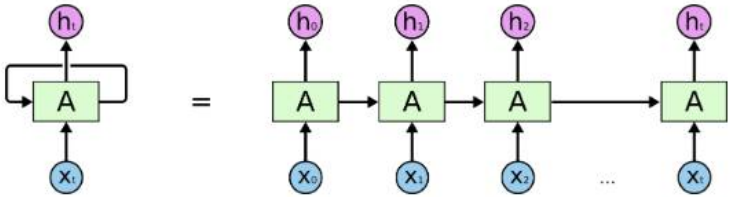
- In machine learning, a **highway network** is an approach to optimizing networks and increasing their depth.
- Highway Networks are also categorized as multi-path based CNN architectures.
- 50-layers showed a better convergence [GoogleNet: 22layers]
- Highway networks use learned gating mechanisms to regulate information flow, inspired by Long Short-Term Memory (LSTM) recurrent neural networks.
- The gating mechanisms allow neural networks to have paths for information to follow across different layers ("information highways").[1][2]
- Highway networks have been used as part of text sequence labeling and speech recognition tasks

Recurrent Neural Networks



Recurrent Neural Networks have loops.

- Humans don’t start their thinking from scratch every second.
- we understand each word based on your understanding of previous words.
- Traditional neural networks can’t d classify what kind of event is happening at every point in a movie. It’s unclear how a traditional neural network could use its reasoning about previous events.
- Applications: speech recognition, language modeling, translation, image captioning



An unrolled recurrent neural network.

- a chunk of neural network, A looks at some input x_t and outputs a value h_t

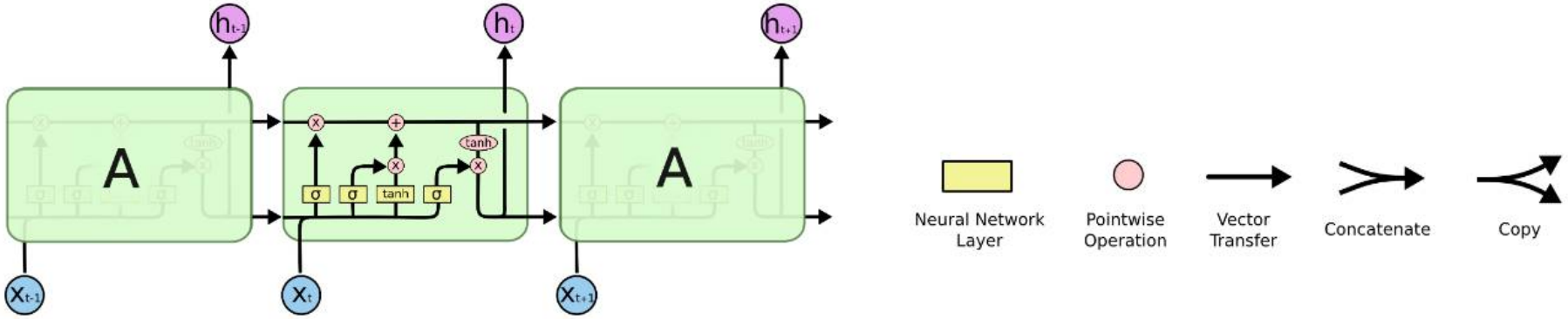
- A loop allows information to be passed from one step of the network to the next.
- This chain-like nature reveals that recurrent neural networks are intimately related to sequences and lists.

- “I grew up in France… I speak fluent *French*.” Recent information suggests that the next word is probably the name of a language,
- RNNs are absolutely capable of handling such “long-term dependencies

LSTM Networks*

Long Short Term Memory networks – usually just called “LSTMs” – are a special kind of RNN, capable of learning long-term dependencies.

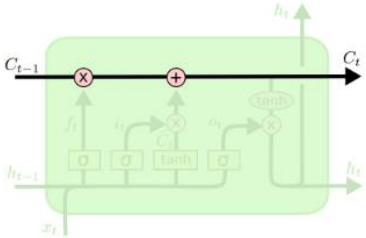
LSTMs are explicitly designed to avoid the long-term dependency problem.



The repeating module in an LSTM contains four interacting layers.

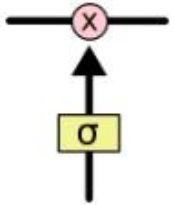
https://colah.github.io/posts/2015-08-Understanding-LSTMs/

LSTM Networks



The key to LSTMs is the cell state.

The LSTM does have the ability to remove or add information to the cell state, carefully regulated by structures called gates.

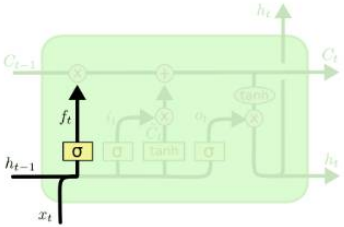


Gates are a way to optionally let information through.

-composed out of a sigmoid neural net layer and a pointwise multiplication operation.

sigmoid layer outputs numbers between zero and one, describing how much of each component should be let through. A value of zero means “let nothing through,” while a value of one means “let everything through!”

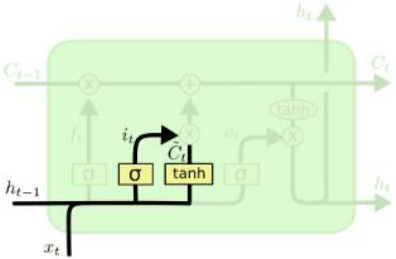
Example



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

- language model trying to predict the next word based on all the previous ones.

- It looks at h_{t-1} and x_t and outputs a number between 0 and 1



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

- The next step is to decide what new information we're going to store in the cell state. This has two parts. First, a sigmoid layer called the "input gate layer" decides which values we'll update. Next, a tanh layer creates a vector of new candidate values, C_t C~t, that could be added to the state.

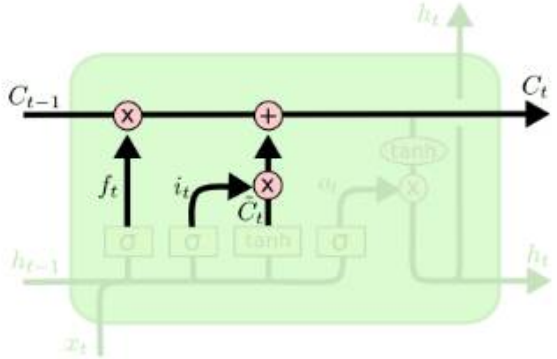
- In the next step, we'll combine these two to create an update to the state.
- 'Tanh' is the wide variety of sigmoid function

Example

It's now time to update the old cell state, C_{t-1}, into the new cell state C_t. The previous steps already decided what to do, we just need to actually do it.

We multiply the old state by f_t, forgetting the things we decided to forget earlier. Then we add $i_t * \tilde{C}_t$. This is the new candidate values, scaled by how much we decided to update each state value.

In the case of the language model, this is where we'd actually drop the information about the old subject's gender and add the new information, as we decided in the previous steps.

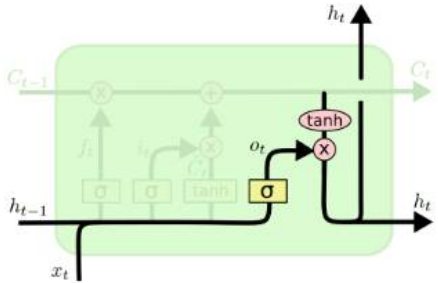


$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Example

Finally, we need to decide what we're going to output. This output will be based on our cell state, but will be a filtered version. First, we run a sigmoid layer which decides what parts of the cell state we're going to output. Then, we put the cell state through tanh (to push the values to be between -1 and 1) and multiply it by the output of the sigmoid gate, so that we only output the parts we decided to.

For the language model example, since it just saw a subject, it might want to output information relevant to a verb, in case that's what is coming next. For example, it might output whether the subject is singular or plural, so that we know what form a verb should be conjugated into if that's what follows next.



$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Multi-Path based CNNs

- Vanishing gradient problem not only results in higher test error but also higher training error (Pascanu et al. 2012; Dong et al. 2016; Dauphin et al. 2017).
- For training deep networks, the concept of multi-path or cross-layer connectivity was proposed (Srivastava et al. 2015a; Larsson et al. 2016; Huang et al. 2017; Kuen et al. 2018).
- Multiple paths or shortcut connections can systematically connect one layer to another by skipping some intermediate layers to allow the specialized flow of information across the layers (Mao et al. 2016; Tong et al. 2017).
- Cross-layer connectivity partitions the network into several blocks. These paths also try to solve the vanishing gradient problem by making gradient accessible to lower layers.

Highway Networks Revisited

The model has two gates in addition to the $\mathbf{H(W_H, x)}$ gate: the transform gate $\mathbf{T(W_T, x)}$ and the carry gate $\mathbf{C(W_C, x)}$. Those two last gates are non-linear transfer functions (by convention Sigmoid function). The $\mathbf{H(W_H, x)}$ function can be any desired transfer function.

The carry gate is defined as $\mathbf{C(W_C, x) = 1 - T(W_T, x)}$. While the transform gate is just a gate with a sigmoid transfer function.

Structure [edit]

The structure of a hidden layer follows the equation:

$$y = H(x, W_H) \cdot T(x, W_T) + x \cdot C(x, W_C) = H(x, W_H) \cdot T(x, W_T) + x \cdot (1 - T(x, W_T))$$

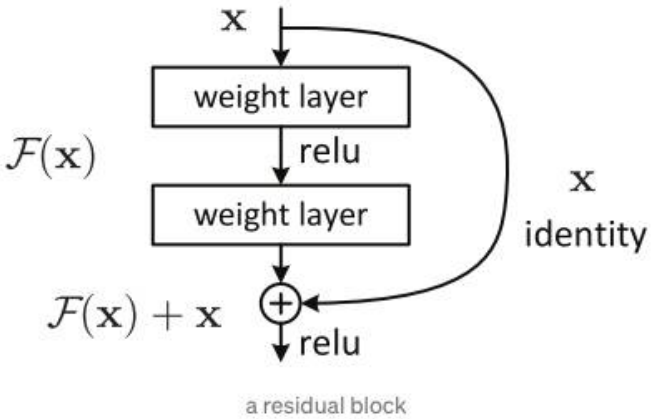
The advantage of a Highway Network over the common deep neural networks is that solves or partially prevents the Vanishing gradient problem, thus leading to easier to optimize neural networks.

ResNet -2015

- ResNet can train up to hundreds or even thousands of layers and still achieves compelling performance.

- concept of residual learning
- placed under the Multi-Path based CNNs
- ResNet proposed 152-layers deep CNN, which won the 2015-ILSVRC competition:
  - 20 and 8 times deeper than AlexNet and VGG, respectively,
  - showed less computational complexity than previously proposed networks (Krizhevsky et al. 2012; Simonyan and Zisserman 2015).
  - ResNet with 50/101/152 layers has less error on image classification task than 34 layers plain Net.
  - Moreover, ResNet gained a 28% improvement on the famous image recognition benchmark dataset named COCO (Lin et al. 2014).

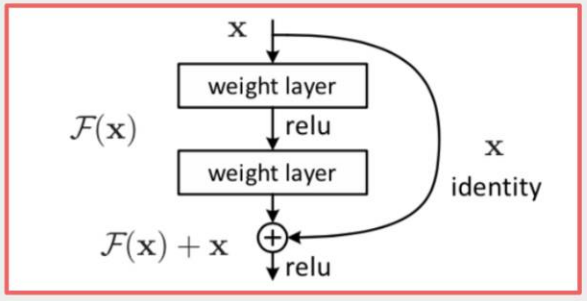
Residual learning*



a residual block

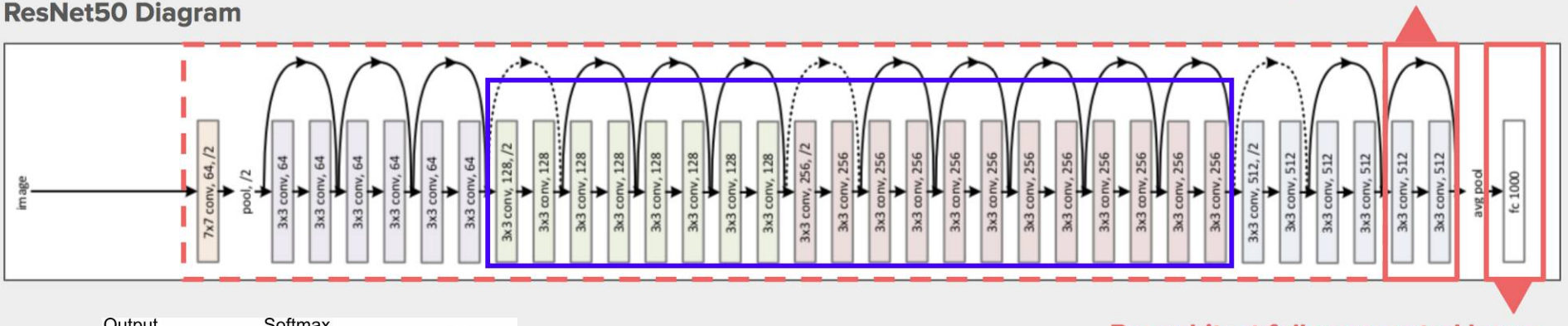
- introducing a so-called “identity shortcut connection” that skips one or more layers.
- deeper model should not produce a training error higher than its shallower counterparts.
- gradients can flow through the shortcut connections to any other earlier layer unimpededly.

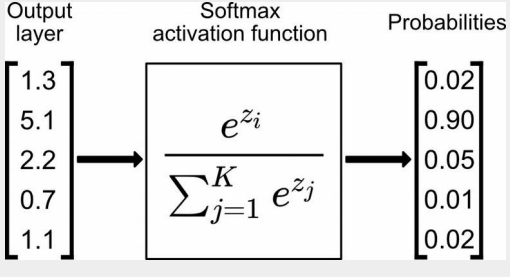
Retrain ResNet50



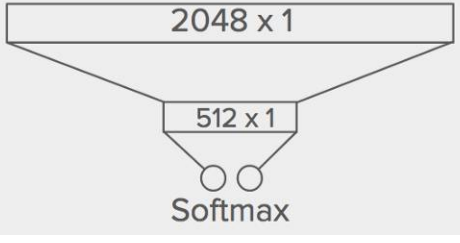
Residual Learning Block

ResNet50 Diagram





Re-architect fully-connected layers



Few Combinations

Inception-V3**:**

- **1x1 convolutional operation was used, which maps the input data into 3 or 4 separate spaces that are smaller than the original input space, and then maps all correlations in these smaller 3D spaces, via regular (3x3 or 5x5) convolutions**

Inception-ResNet:

- combined the power of residual learning and inception block
- filter concatenation was replaced by the residual connection.

DenseNet* -2017

- Consider a single image x_0 that is passed through a convolutional network:
  - The network comprises L layers, each of which implements a non-linear transformation H_l(·), where l indexes the layer.
  - H_l(·) can be a composite function of operations such as Batch Normalization (BN) , rectified linear units (ReLU) , Pooling , or Convolution (Conv).
- Advantage of ResNets: gradient can flow directly through the identity function from later layers to the earlier layers.
- identity function and the output of H_l are combined by summation, which may impede the information flow in the network.

*https://arxiv.org/pdf/1608.06993v5.pdf

DenseNet

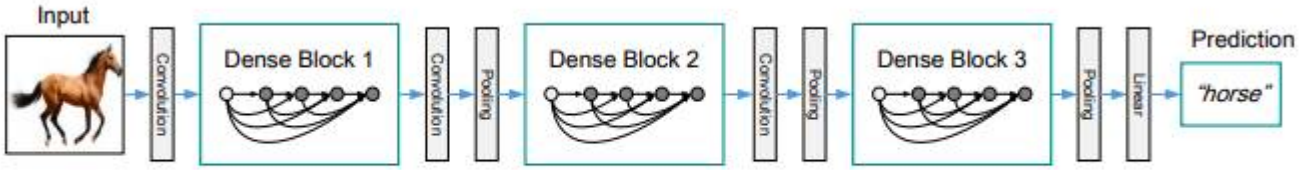


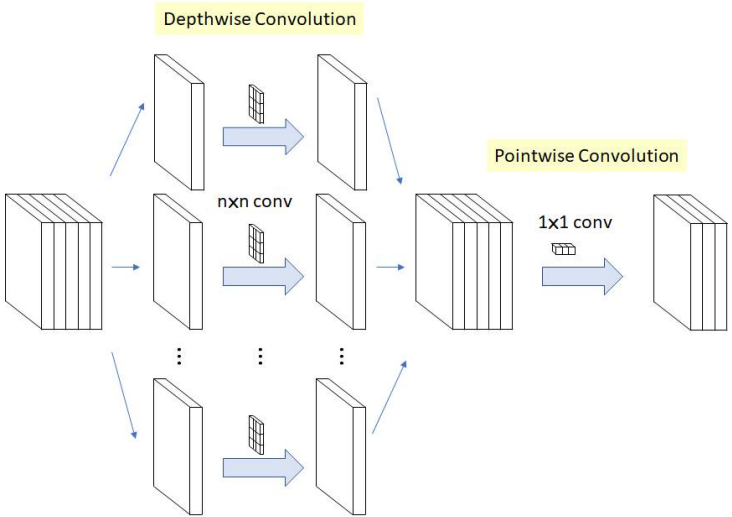
Figure 2: A deep DenseNet with three dense blocks. The layers between two adjacent blocks are referred to as transition layers and change feature-map sizes via convolution and pooling.

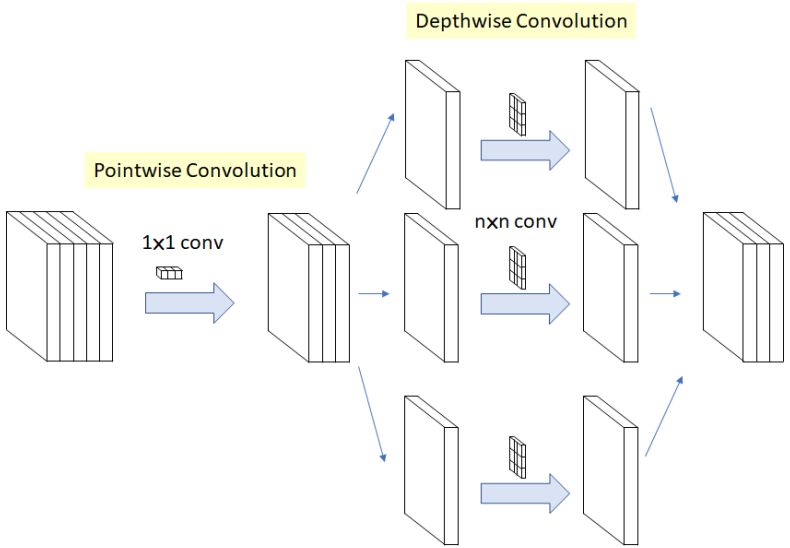
- Main Idea: Concatenating feature-maps learned by different layers increases variation in the input of subsequent layers and improves efficiency.
- To further improve the information flow between layers: a different connectivity pattern to introduce direct connections from any layer to all subsequent layers.

Figure Ref: *https://arxiv.org/pdf/1608.06993v5.pdf

Width Based CNN

[focus of research shifted from deep and narrow architecture towards thin and wide architectures.]





The Modified Depthwise Separable Convolution used as an Inception Module in Xception, so called “extreme” version of Inception module (n=3 here)

1. **Depthwise convolution** is the **channel-wise n×n spatial convolution**. [For 5 channels : 5 n×n spatial convolution.]
2. **Pointwise convolution** actually is the **1×1 convolution** to change the dimension.

Xception -2016

- **Useful in Detecting Face Swaps in Videos**
- Shortcuts between Convolution blocks as in ResNet
- **Mobilenets, based on Xception (from Google): SotA in terms of statistical efficiency, require the least amount of weights and computations to achieve a usable accuracy**

*https://arxiv.org/pdf/1610.02357.pdf

*

https://www.analyticsvidhya.com/blog/2018/04/this-deep-learning-algorithm-detects-f

Variations

- Attention based CNN
- Feature-Map (Channel*FMap*) Exploitation based CNNs etc