FML HW10

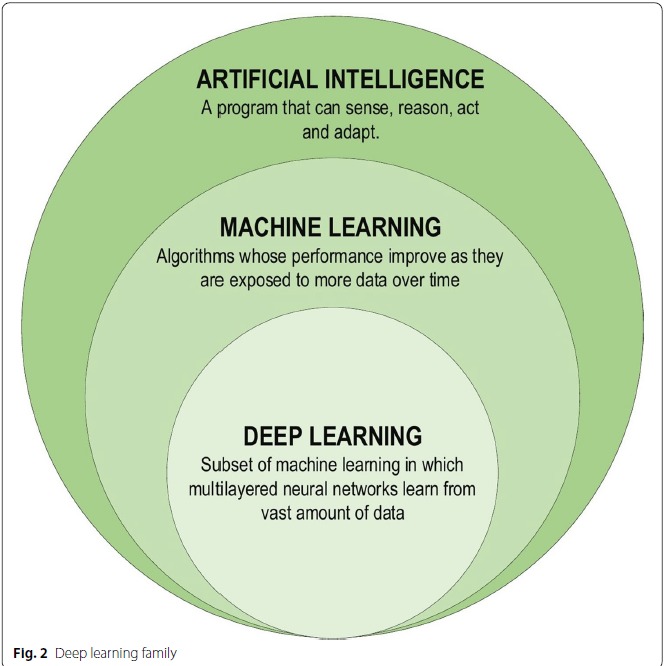
Review of Deep Learning

Jeongmin Hwang

**0. Introduction**

DL is subset of machine learning in which multilayered neural networks learn from vast amount of data

DL is outperforming other ML algorithm and used in many types of data for various purposes.



Deep learning became widely used and outperformed ML algorithm for following reasons

**1) Universal Learning Approach**

**2) Robustness**

**3) Generalization**

**4) Scalability**

**1. Classification of DL approaches**

**1) Deep supervised learning**

Deals with labeled data.

Ex) RNN, CNN, DNN

\* All of data have to be labeled and learning process

\* Simpler than other algorithm while still makes good performance

**2) Deep semi-supervised learning**

Deals with semi-labeled datasets (use both labeled and unlabeled data for learning process)

Can minimize the amount of labeled data

Irrelevant input may result in incorrect decision

**3) Deep unsupervised learning**

Deals with unlabeled data

Algorithm works by learns structure or relationship in input data

Generative networks, dimensionality reduction, clustering are examples of unsupervised learning

**4) Deep reinforcement learning**

Application of deep learning in fields of reinforcement learning (RL)

In RL, agent aims to take an action that maximizes the reward in given environment

**2. DL networks**

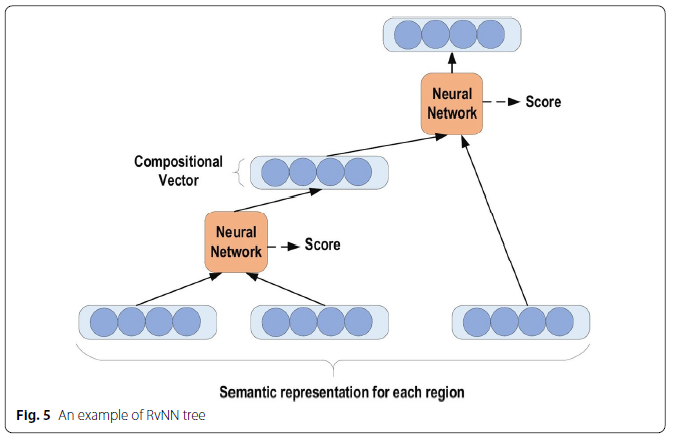
**1) Recursive neural networks (RvNN)**

RvNN achieves predictions in a hierarchical structure also classify the outputs utilizing compositional vectors.

Apply the same set of weights recursively over a structured input.

Ex) Recursive neural networks(RvNN), Recursive auto-associative memory(RAMM)

RvNN is mainly used in NLP



**2) Recurrent neural networks**

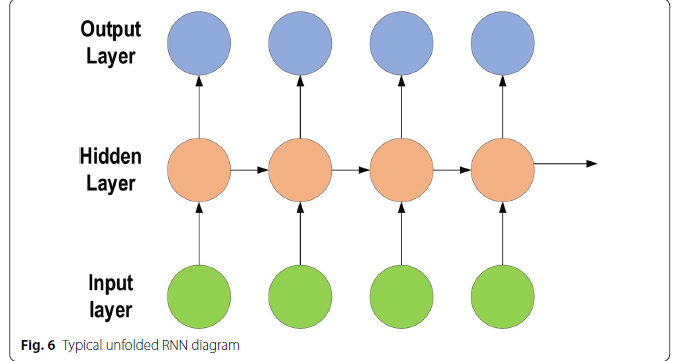
Deals with sequential data (ex) daily prices)

RNN (vanilla RNN) are vulnerable to exploding gradient and vanishing gradient problem

**Cf) Exploding and vanishing gradient problem**

As process of backpropagation proceeds, the gradient may get smaller and smaller as it comes from output layer (Think about how chain rule applies in backpropagation). This results in weights to change by only small amount which may cause gradient descent to never converges to the optimum level. This is called vanishing gradient problem. On the other hand, if the gradient gets bigger and bigger in backpropagation process, it may result in gradient descent to diverge from the optimum level. This is called exploding gradient problem

To address this issue, more advanced RNN such as LSTM and GRU were invented



**3. Basic of CNN**

CNN are applied in many different fields and mainly used for image recognition. It’s one of the most famous and commonly used algorithms in deep learning.

**1) Architectures of CNN**

**\* Input**

The input x of each layer in CNN model are organized in three dimensions:

Height, width, depth (m\*m\*r) – Assume that height = width (can be different)

Depth is channel number(ex) RGB image’s depth is 3)

**\* Convolution layer**

Calculates a dot product between its input and weights as in EQ1

****

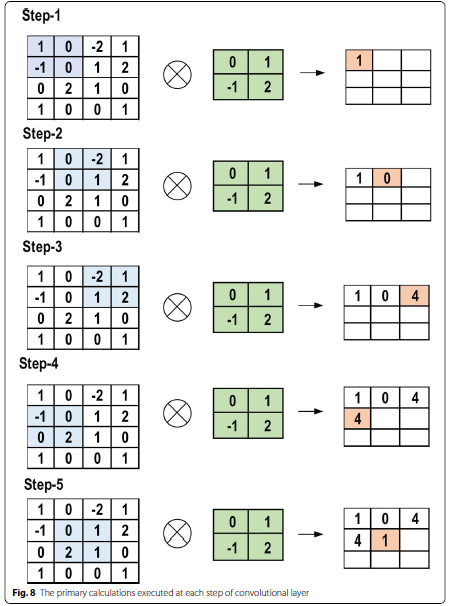
**\* Kernel**

Kernel is filer applied to layer of Convolution layer.

It also has three dimension (n\*n\*q) – Assume that height = width (can be different)

n < m and q <= r

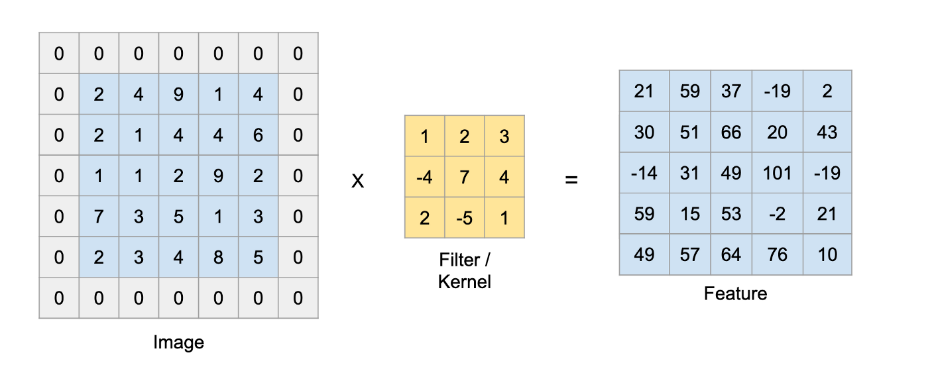
Using Kernel changes the shape of input data

****

**\* Padding**

Padding is used to avoid change in shape of the data due to usage of kernel

Use same padding makes the shape of input data unchanged.



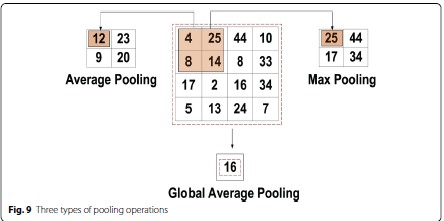
**\* Pooling layer**

Sub-sampling in Convolution layer

Pooling function is applied to an adjacent area of size p\*p

Max or Average is widely used for pooling function

Handles overfitting issue and accelerates the training process

****

**\* Activation Function**

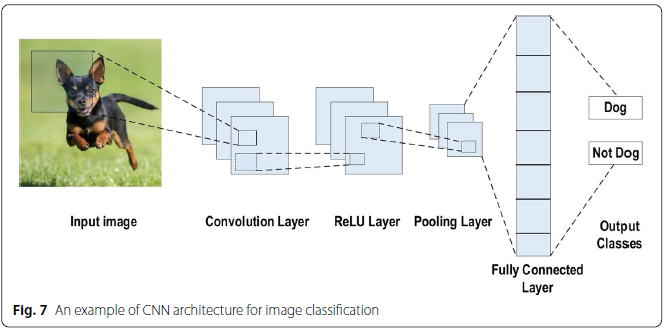
Just like other DL algorithm, activation function can be used between layers

Ex) Sigmoid, Tanh, ReLU, Leaky ReLU, etc.

**\* Fully connected layer**

It receives abstracted data generated by processing of convolution layer, pooling layer, etc.

Outputs are calculated

****

**\* Loss Function**

① **Cross-Entropy or Softmax Loss function** (Used in classification problem)

② **Euclidean Loss function** (Used in regression)

It’s also called as MSE (mean squared error)

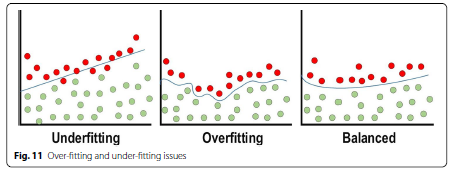
③ **Hinge Loss function**

Maximum-margin based classification (Used in SVM)



**2) Improving performance of CNN**

**\* Regularization to CNN**



**① Dropout**

Proportion of neurons are randomly dropped out during training (not used for both backpropagation and forward propagation)

**② Drop weights**

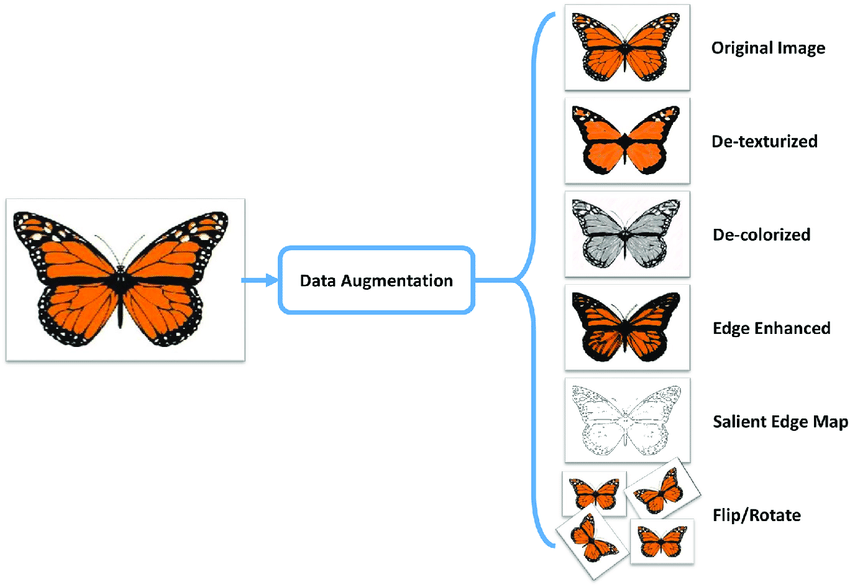
Neurons weights are dropped out instead of dropping entire neuron

**③ Data augmentation**

Generate new data based on the data we have. This results in increase in size of dataset which helps address overfitting issue.

Commonly used technique for data augmentations are as follows

Mirroring, Random Cropping, Rotation, Shearing,



**④ Batch Normalization**

Normalize layers’ input by applying centering and scaling

Make training faster and more stable.

**\* Optimizer selection**

① **Batch gradient descent**

Update gradient using gradient descent using all training data

② **Stochastic gradient descent**

Unlike Batch gradient descent it only uses parts of data to compute gradient

③ **Mini-batch Gradient descent**

Calculate set of gradients using mini-batches, and use average of gradients to update the gradient.

④ **Momentum**

Determines the next update as linear combination of gradient and the previous update

⑤ **Adam**

Running averages of both the gradients and second moments of gradients are used.

⑥ **Other optimizers**

AdaGrad, RMSProp, etc.

**4. (Advanced) CNN architectures**

**1) AlexNet**

AlexNet is highly respected as it achieved innovative results in fields of image recognition and classification.

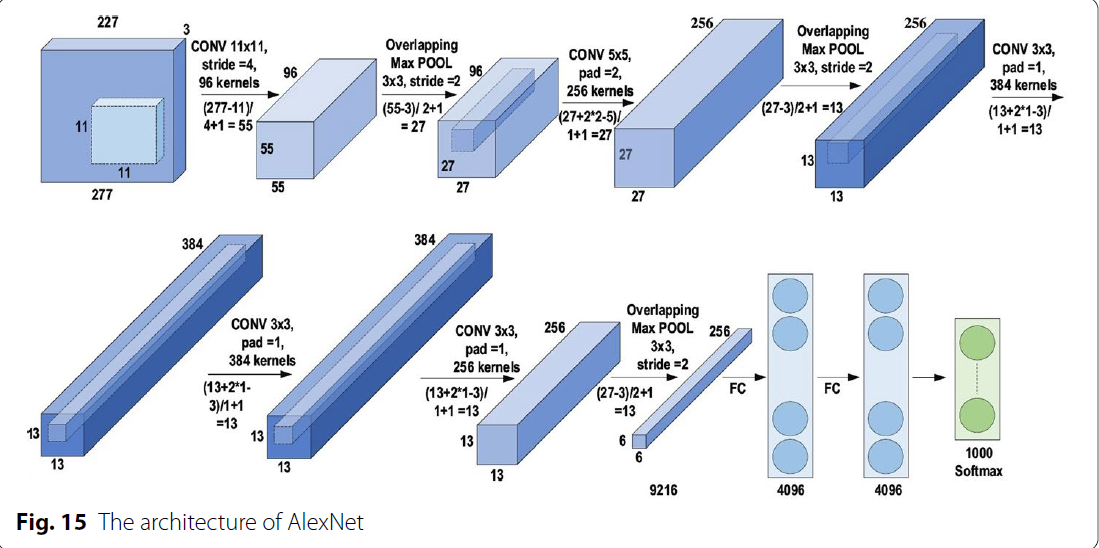
\* Two GPUs were used in parallel to train AlexNet

\* The number of feature extraction stage is increased to seven

\* To ensure robustness of model, model used several transformational units throughout the training stage

\* To address, gradient vanishing problem, ReLU was used.

\* Local response normalization and overlapping subsampling were used to decrease overfitting.



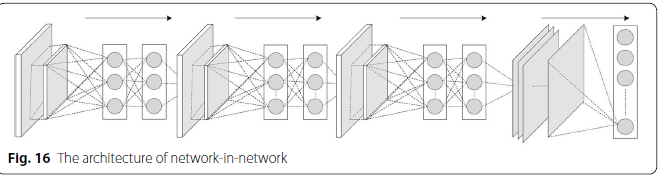
**2) Network-in-network**

Used multiple layers of perception convolution.

(used 1\*1 filter which supports the addition of extra nonlinearity in the networks)

Used regularization technique to address the issue of increasing network depth

Instead of FC layer, Global Average Pooling is used

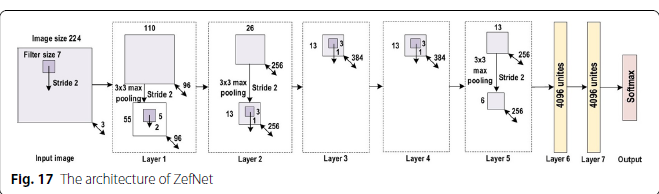


**3) ZefNet**

a multilayer de-convolutional neural network

ZefNet visualized hidden layers in order to monitor the performance of CNN by visualizing network activity.

ZefNet has huge impact on improvement in performance of other CNN networks

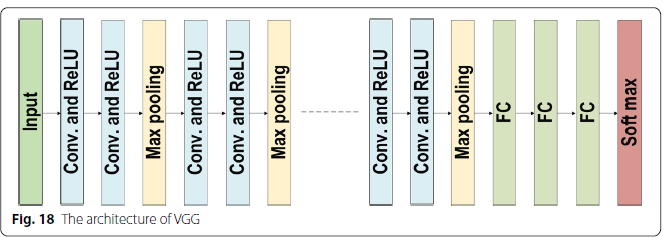


**4) Visual Geometry group(VCG)**

VCG is an easy and efficient design principle for CNN which was proposed by Simonyan and Zisserman.

It established a research trend for working with small-size filters in CNN by finding that working with small size filter can yield the same efficient result with advantage of reducing computational complication.

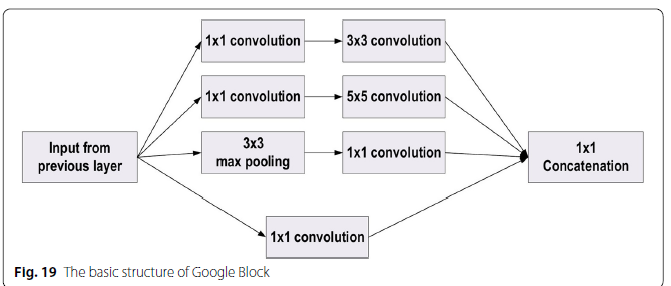
Inserted 1\*1 convolutions in the middle of convolutional layer to regulates the network complexity



**5) GoogLeNet**

Proposed Inception block concept in CNN context to achieve high-level accuracy with decreased computational cost

Basic structure of Google Block is shown below



This block uses filters of different sizes to capture channel information together with spatial information

The common convolutional layer of GoogLeNet is substituted by small blocks using the same concept of network- in-network (NIN) architecture which replaced each layer with a micro-neural network.

It regulates the computation by inserting a 1\*1 convolutional filter, as a bottleneck layer, ahead of using large-size kernel.

It decreased cost by neglecting the irrelevant channels.

The additional regularity factors used included the employment of RMSProp as optimizer and batch normalization