Deep learning

Yann LeCun, YoShua Bengio & Geoffrey Hinton

nature 14539

SuYong Jeong

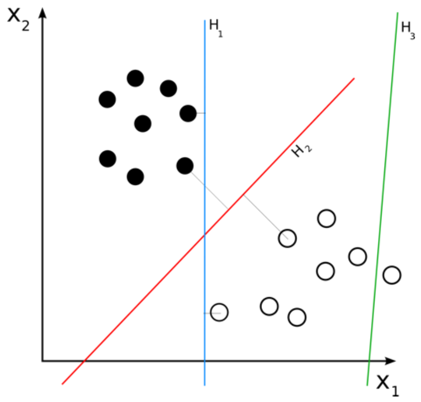
Outline

- ☐ Conventional Machine Learning
- ☐ Deep Learning
- ☐ Convolutional Neural Networks (CNN)
- ☐ Distributed Representation & Language Processing
- ☐ Recurrent Neural Network (RNN)

Conventional Machine Learning

☐ Procedure of machine learning

- Most common form of machine learning is supervised learning
  - Machine is shown an image and produces an output in the form of a vector of scores for each category
  - Measure the error between the output scores and the desired pattern of scores using an obj. function
  - During training process, hand-engineered features are needed to reduce the error



https://en.wikipedia.org/wiki/Machine_learning

Conventional Machine Learning

☐ Shortcoming of conventional machine learning

- Limit on processing natural data in their raw form exists
  - ☐ Considerable domain expertise is needed to design a feature extractor to transforms the raw data into suitable feature vector
- Only carve their input space into very simple regions
  - ☐ Struggling to handle **selectivity-invariance dilemma**

Conventional Machine Learning

- **Struggling to handle selectivity-invariance dilemma**
  - Selectivity: particular minute variations to distinguish
  - Invariance: irrelevant variations to distinguish

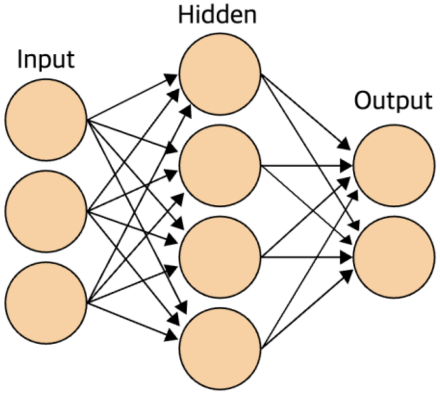
Conventional Machine Learning

- For more powerful classifiers
  - Conventional option is to hand-design good feature extractors
    - It requires a considerable amount of engineering skill and domain expertise
  - But **It can all be avoided if good features can be learned automatically** using a general-purpose learning procedure

Deep Learning

What is deep learning?

- Deep learning is a subset of machine learning that **utilizes deep neural networks**
- It requires very little engineering by hand unlike conventional model and **automatically learn the features from data**
- An architecture of deep learning is a multilayer stack of simple modules
  - Input layer, Hidden layers, and Output layer



https://en.wikipedia.org/wiki/Neural_network_(machine_learning)

Deep Learning

☐ Feedforward neural network architecture

- To go from one layer to the next, a set of units computes a weighted sum of their inputs from the previous layer
- A weighted sum emphasizes important features and suppress unnecessary information

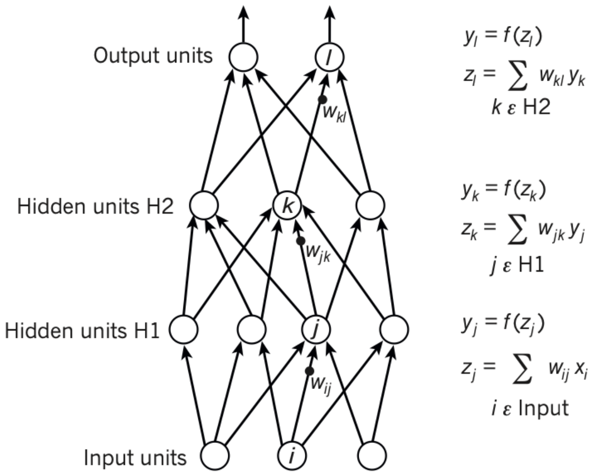


Fig. 1, c

Deep Learning

☐ Activation fuction

- Without activation functions, stacked linear functions remain linear and cannot model nonlinear relationships
- Activation fuction transform the representation at one level into a representation at a higher, slightly more abstract level

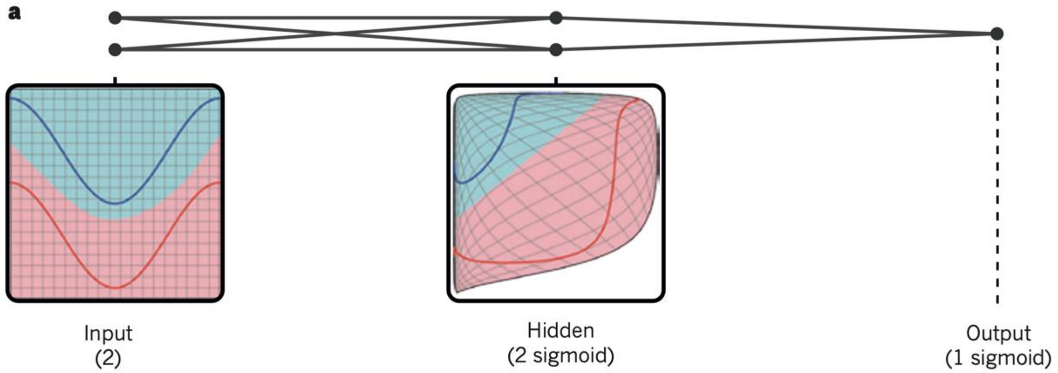


Fig. 1, a

Deep Learning

Backpropagation

- The backpropagation procedure is to compute the gradient of an obj. function with respect to the weights of a multilayer stack
- A deep learning model minimizes the loss function by computing its gradient and updating the weights in the direction that reduces the loss

$$W = W - \alpha \frac{\partial L}{\partial W}$$

Deep Learning

□ Backpropagation

- The gradient of the objective with respect to the input of a module can be computed by working backwards from the gradient with respect to the output of that module

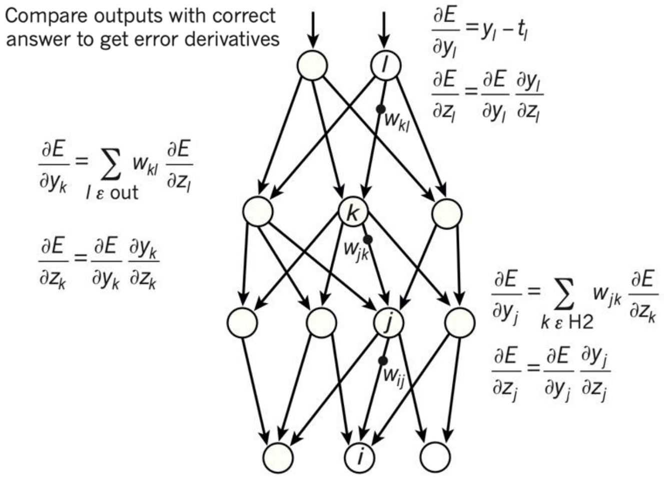


Fig. 1, d

Deep Learning

☐ Backpropagation

- Through backpropagation, the gradient of the error with respect to each module's weight can be computed, **allowing the model to train automatically**

$$Z_k = \sum_{j \in H_1} w_{jk} y_j \rightsquigarrow \frac{\partial Z_k}{\partial w_{jk}} = y_j$$

$$\frac{\partial E}{\partial w_{jk}} = \frac{\partial Z_k}{\partial w_{jk}} \times \frac{\partial E}{\partial Z_k}$$

$$\frac{\partial E}{\partial w_{jk}} = y_j \times \frac{\partial E}{\partial Z_k}$$

Deep Learning

☐ Vanishing gradient in backpropagation

- In each layer, the gradient is multiplied by the gradient of the activation function and the gradient becomes smaller, learning becomes ineffective

$$z_k = \sum_{j \in H1} w_{jk} y_j \rightsquigarrow \frac{\partial z_k}{\partial w_{jk}} = y_j$$

$$f'(z_k) = \frac{\partial y_k}{\partial z_k} = \text{활성화함수의 미분값.}$$

$$\frac{\partial E}{\partial w_{jk}} = \frac{\partial E}{\partial y_k} \cdot \frac{\partial y_k}{\partial z_k} \cdot \frac{\partial z_k}{\partial w_{jk}}$$

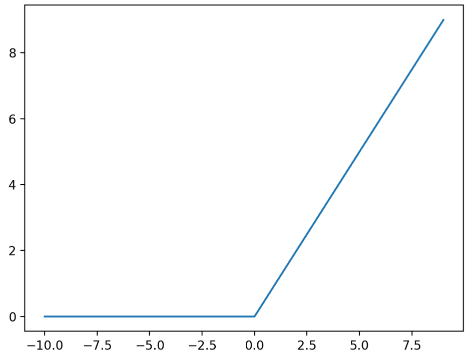
$$= \underline{\frac{\partial E}{\partial y_k} \cdot f'(z_k) \cdot y_j} \rightsquigarrow \frac{\partial E}{\partial w_{jk}} = \frac{\partial E}{\partial y_l} \cdot \underline{f'(z_l) \cdot \prod_{k=l}^{1} f'(z_k) \cdot y_j}$$

Deep Learning

Rectified linear unit

- ReLU is better than other functions such as Tanh, Sigmoid, allowing training of a deep supervised network without unsupervised pre-training

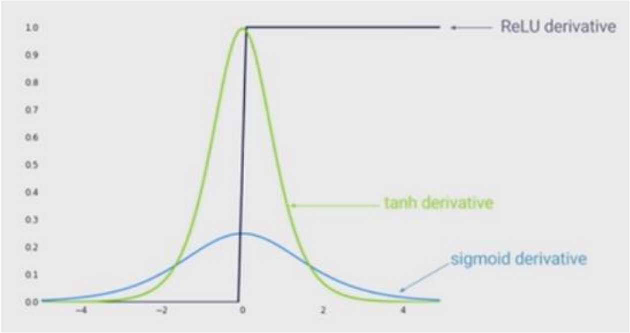
$$\mathrm{ReLU}(x) = x^+ = \max(0, x) = \frac{x + |x|}{2} = \begin{cases} x & \text{if } x > 0, \\ 0 & x \le 0 \end{cases}$$



Deep Learning

☐ Why does ReLU perform better than others?

- ReLU is much simpler computationally
  - ☐ The forward and backward passes through ReLU are both just a simple “if” statement
  - ☐ In comparison, Sigmoid and Tanh requires computing an exponent
- ReLU is more robust in vanishing gradient
  - ☐ Using Sigmoid and Tanh, the vanishing gradient problem occurs, preventing proper learning

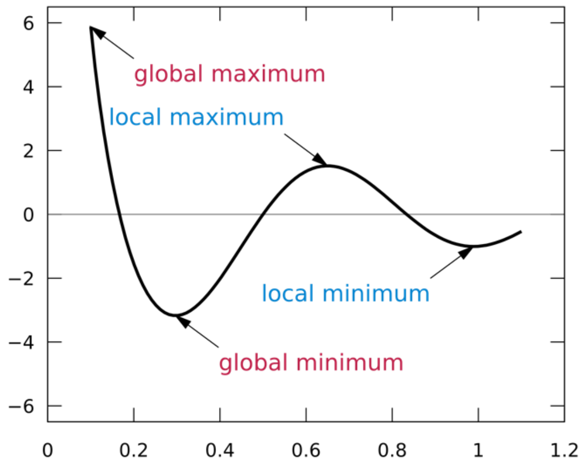


https://www.sciencedirect.com/topics/mathematics/hyperbolic-tangent-function

Deep Learning

□ Criticism for deep learning

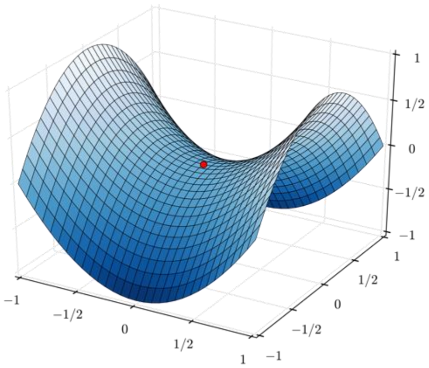
- It was widely thought that learning useful, multistage, feature extractors with little domain expertise was infeasible
- And simple gradient descent would get trapped in poor local minima



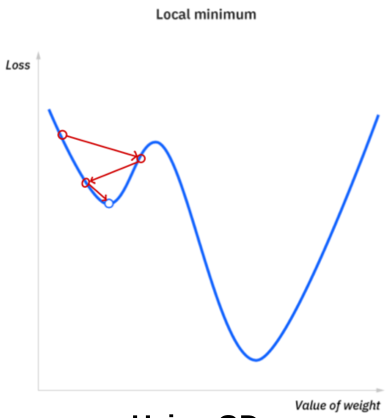
https://en.wikipedia.org/wiki/Maximum_and_minimum

Stochastic Gradient Descent

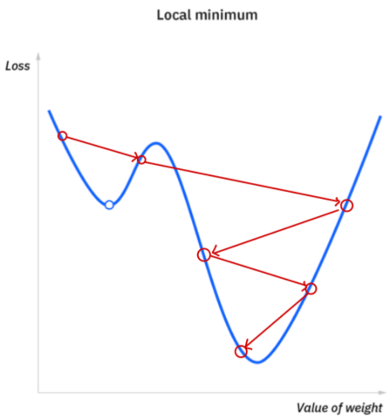
- Stochastic gradient descent (SGD) updates weights using the gradient computed from a randomly selected data sample
- SGD is more robust to saddle points and local minima due to its stochastic nature, allowing it to escape them more easily



Saddle Point



Using GD



Using SGD

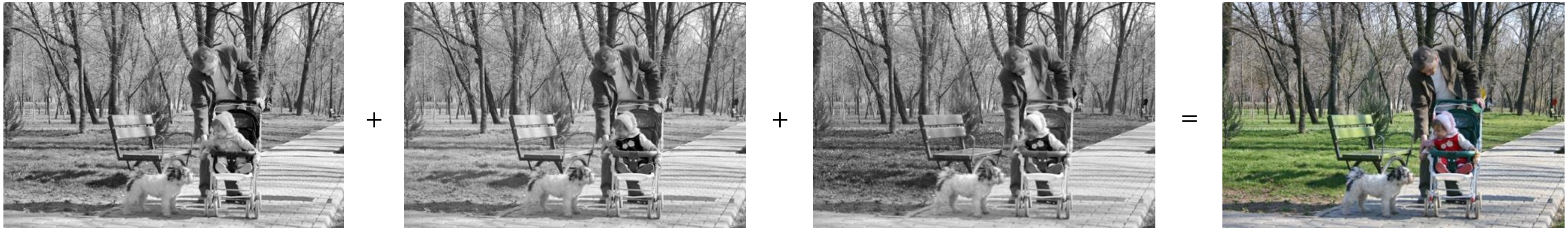
Outline

- ☐ Conventional Machine Learning
- ☐ Deep Learning
- ☐ Convolutional Neural Networks (CNN)
- ☐ Distributed Representation & Language Processing
- ☐ Recurrent Neural Network (RNN)

Convolutional Neural Network

☐ What is CNN?

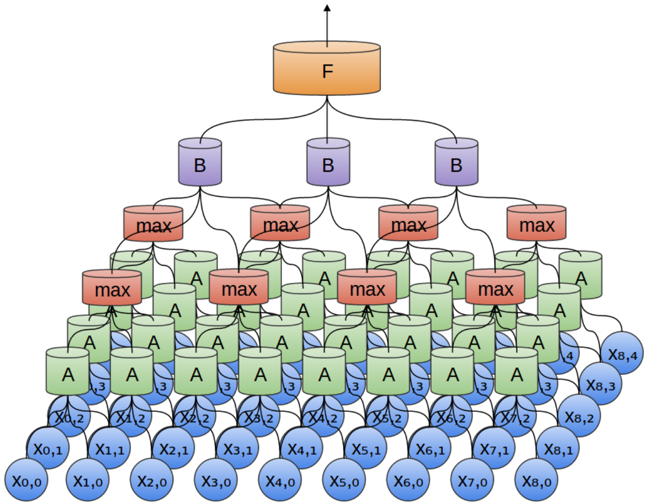
- A Convolutional Neural Network (CNN) is a deep neural network that uses convolution and pooling operations to extract and learn local patterns better
- CNN is designed to process data that come in the form of multiple arrays such as a color image composed of three 2D arrays in the three color channels

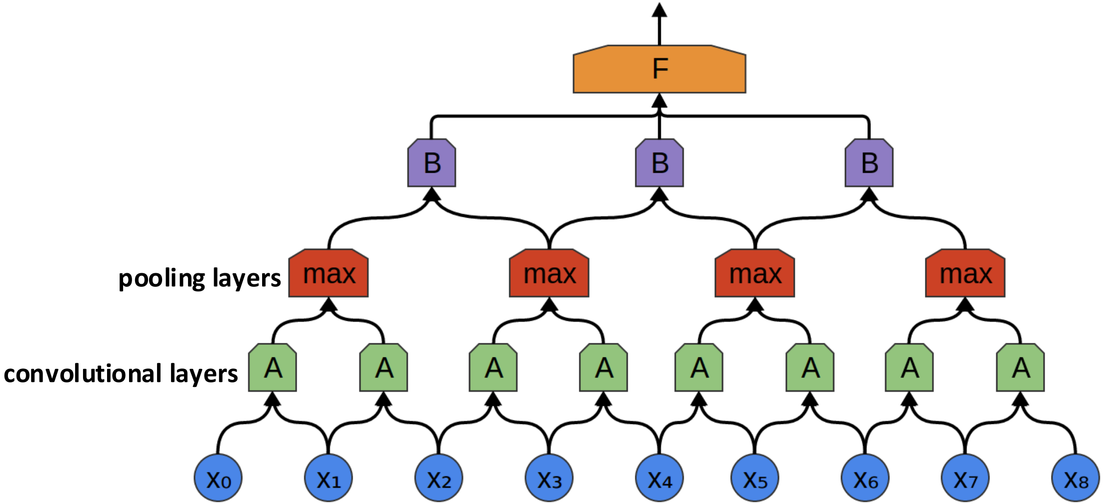


Red   Green   Blue

Convolutional Neural Network

☐ Overall Structure of CNN



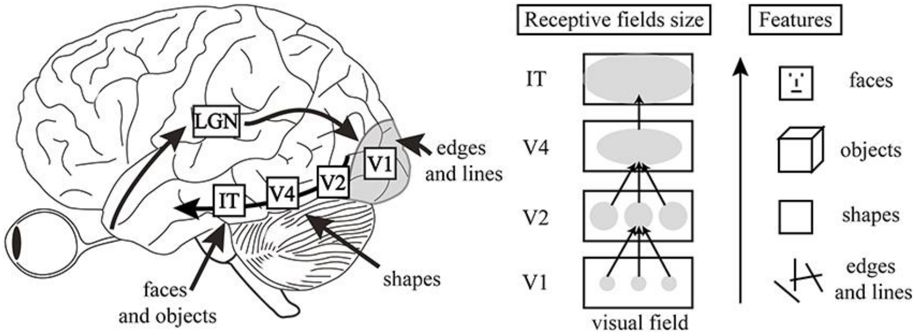


Left: https://colah.github.io/posts/2014-07-Conv-Nets-Modular/
Right: ImageNet Classification with Deep Convolutional Neural Networks, Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton

Convolutional Neural Network

Overall Structure of CNN

- The convolutional and pooling layers in CNNs are directly inspired by the complex cells in visual neuroscience
  - CNNs operate similarly to how the human brain processes visual information in a hierarchical manner

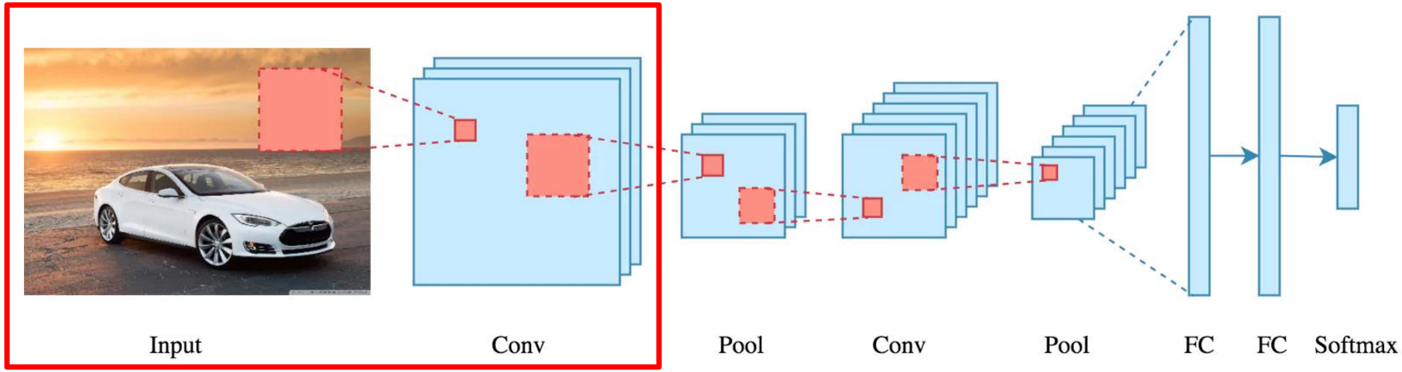


https://www.researchgate.net/figure/Left-A-typical-hierarchical-feedforward-model-where-information-processing-starts-at_fig1_267872860

Convolutional Neural Network

☐ Convolutional Layers

- Units in a convolutional layer are grouped into feature maps
- Each unit in a feature map is connected to a small local patch in the feature maps of the previous layer

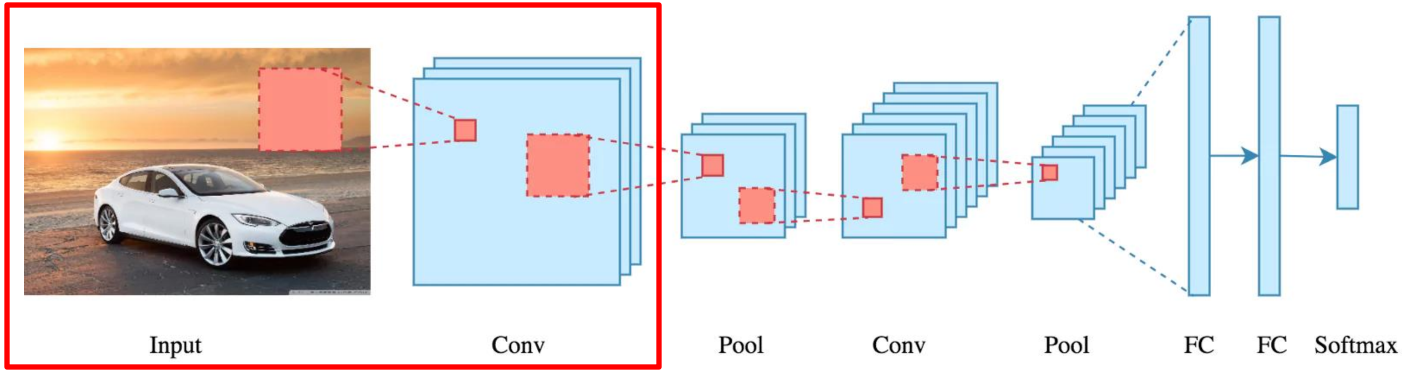


https://www.analyticsvidhya.com/blog/2021/05/convolutional-neural-networks-cnn/

Convolutional Neural Network

☐ Convolutional Layers

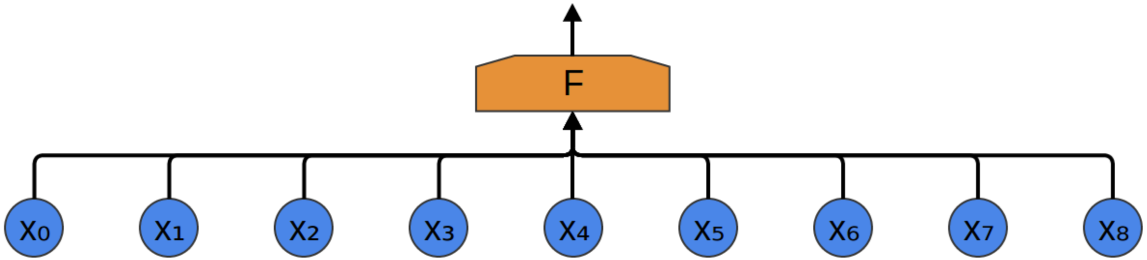
- This connection is established through a set of weights known as a filter bank
- All units in a feature map share the same filter bank and different feature maps in a layer use different filter banks



https://www.analyticsvidhya.com/blog/2021/05/convolutional-neural-networks-cnn/

Convolutional Neural Network

- **Why do CNNs use local patches to process images?**
  - Preserving the spatial structure of the image
    - Fully connected layers flatten the image into a single vector, losing the spatial structure of the pixels
    - CNNs apply local feature extraction using convolutional filters, preserving edges, textures, and shapes

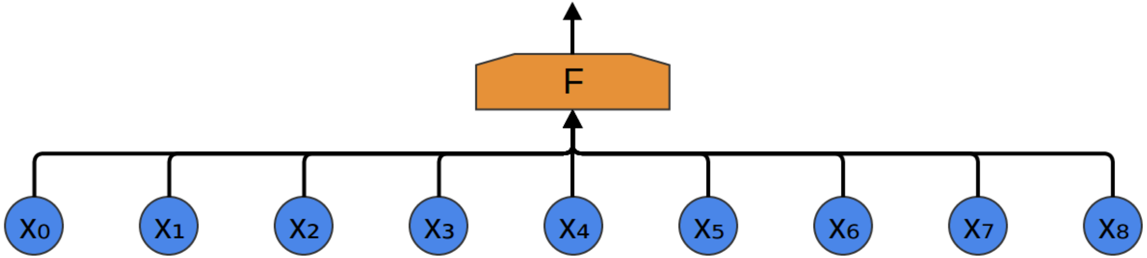


https://colah.github.io/posts/2014-07-Conv-Nets-Modular/

Convolutional Neural Network

☐ Why do CNNs use local patches to process images?

- Reducing the number of parameters for efficient learning
  - ☐ In a fully connected layer, each neuron is connected to all input pixels, requiring a unique weight for each connection. For a 100×100 image, one neuron needs 10,000 weights
  - ☐ In contrast, CNNs use small filters that slide over the image, sharing the same weights across different regions



https://colah.github.io/posts/2014-07-Conv-Nets-Modular/

Convolutional Neural Network

Why do CNNs share the filter bank?

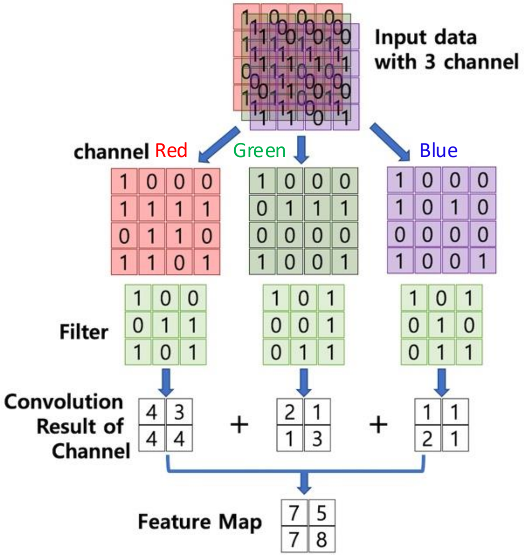
- In array data such as images, local groups of values are often highly correlated
- The local statistics of images and other signals are invariant to location
  - For example, if a motif can appear in one part of the image, it could appear anywhere

https://en.wikipedia.org/wiki/Samoyed_dog

Convolutional Neural Network

☐ Creating feature maps of images

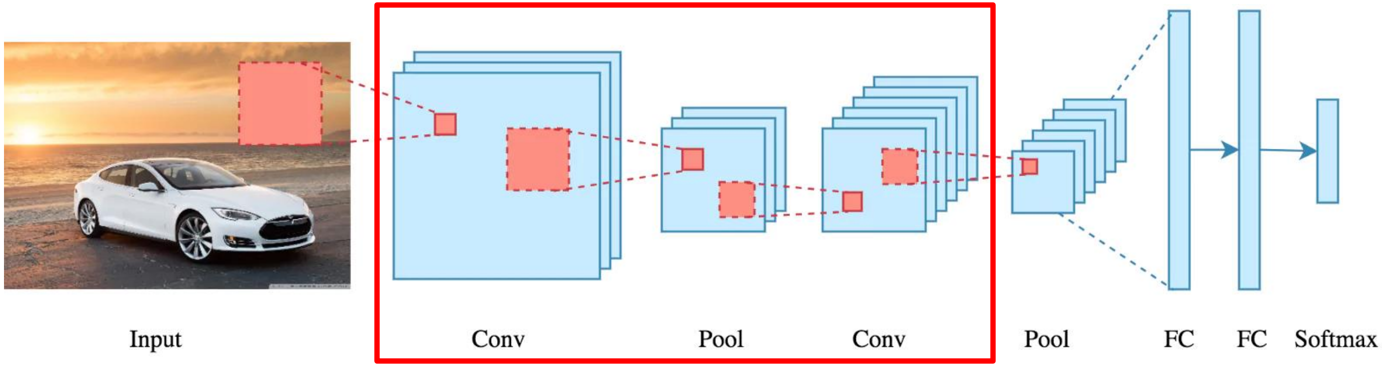
- Each channel process inputs with a different filter
- The outputs from each channel are aggregated to form a feature map for the local patch



Convolutional Neural Network

Pooling Layers

- The result of this local weighted sum is passed through a non-linearity such as a ReLU
- And it is passed to the pooling layer

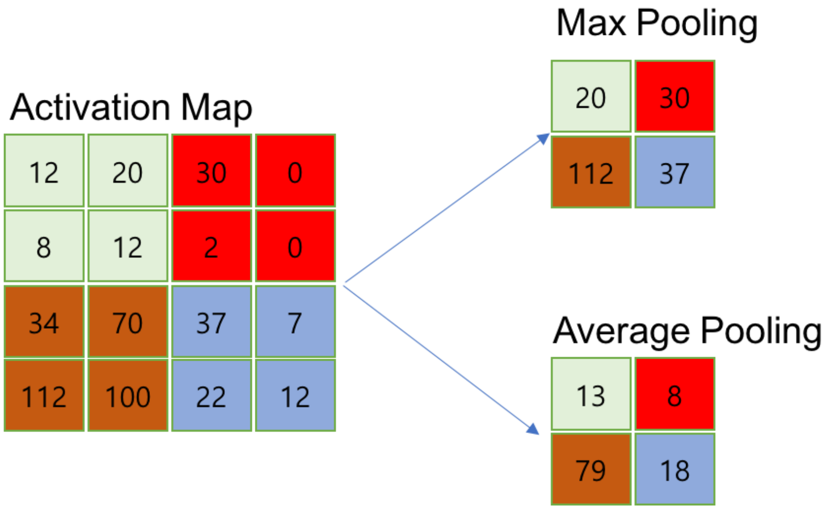


https://www.analyticsvidhya.com/blog/2021/05/convolutional-neural-networks-cnn/

Convolutional Neural Network

☐ Pooling Layers

- The role of the pooling layer is **to merge semantically similar features into one**



http://taewan.kim/post/cnn/

Convolutional Neural Network

□ Pooling Layers

- The relative positions of the features forming a motif can vary somewhat, detecting the motif can be done by coarse-graining the position of each feature

Left: https://www.reddit.com/r/samoyeds/comments/zuhlq3/do_you_regularly_trim_your_samoyeds_ear_fluff/
Right: https://www.shinee-pet.com/st/what-if-a-dog-has-different-ears%EF%BC%9F/

Outline

- ☐ Conventional Machine Learning
- ☐ Deep Learning
- ☐ Convolutional Neural Networks (CNN)
- ☐ **Distributed Representations & Language Processing**
- ☐ **Recurrent Neural Network (RNN)**

Distributed Representations

☐ What is distributed representation?

- Distributed representation represent data as high-dimensional vectors, capturing meaningful relationships by **representing each entity as a combination of multiple features**

King = [0.9, 0.1, 0.6, etc.]

Queen = [0.9, 0.9, 0.8, etc.]

Distributed Representations

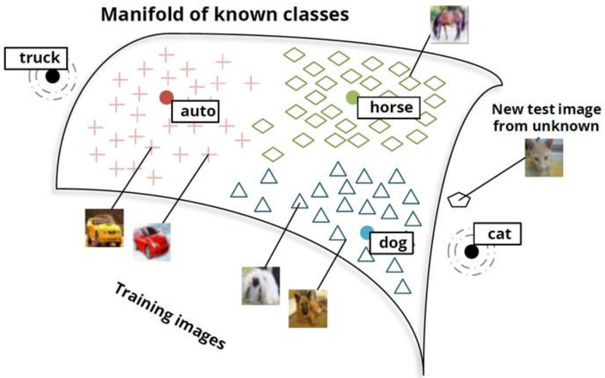
☐ Exponential Advantages

- Learning distributed representations enable generalization to new combinations of the values of learned features beyond those seen during training
- For example, the model wasn't trained to classify cats, it can classify cats using existing distributed representations

Dog Image = ["4 legs", "has fur", "small eyes"]

Car Image = ["4 wheels", "metal body", "has an engine"]

New Image = ["4 legs", "has fur", "big eyes"]



ImageNet Classification with Deep Convolutional Neural Networks, Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton

Distributed Representations

☐ Exponential Advantages

- Composing layers of representation in a deep net brings the potential for another exponential advantage with the depth

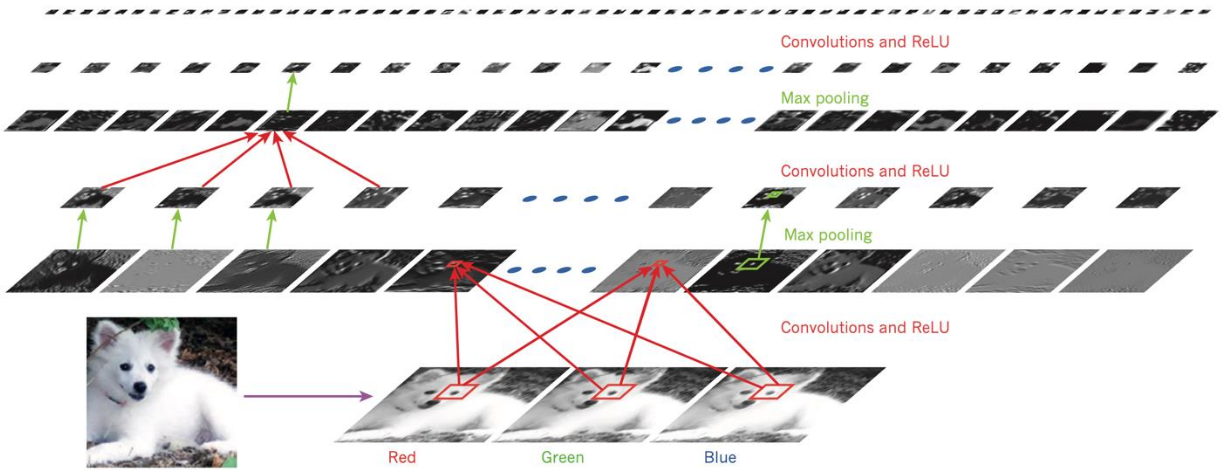


Fig. 2

Language Processing

- ☐ **N-grams**
  - The approach was based on counting frequencies of occurrences of short symbol sequences of length up to N

$$P(\text{mat} \mid \text{I saw a cat on a}) = P(\text{mat} \mid \text{cat on a}) = \frac{N(\text{cat on a mat})}{N(\text{cat on a})}$$

https://lena-voita.github.io/nlp_course/language_modeling.html

Language Processing

☐ N-grams

- N-grams treat each word as an atomic unit
  - ☐ They cannot generalize across semantically related sequences of words
  - ☐ If one wants to model the joint distribution of 10 consecutive words with V of size 100,000, there are potentially 100000^{10} - 1 free parameters -> **curse of dimensionality**

Before

P(I saw a cat on a mat) =

P(I)
· P(saw | I)
· P(a | I saw)
· P(cat | I saw a)
· P(on | I saw a cat)
· P(a | I saw a cat on)
· P(mat | I saw a cat on a)

After (3-gram)

P(I saw a cat on a mat) =

P(I) → P(I)
· P(saw | I) → · P(saw | I)
· P(a | I saw) → · P(a | I saw)
· P(cat | ~~I~~ saw a) → · P(cat | saw a)
· P(on | ~~I saw~~ a cat) → · P(on | a cat)
· P(a | ~~I saw a~~ cat on) → · P(a | cat on)
· P(mat | ~~I saw a cat~~ on a) → · P(mat | on a)

ignore use

Language Processing

☐ Language Model with Neural Network

- Each word in the context is presented to the network as a one-of-N vector
- In the first layer, each word creates a different pattern of activations, or word vectors

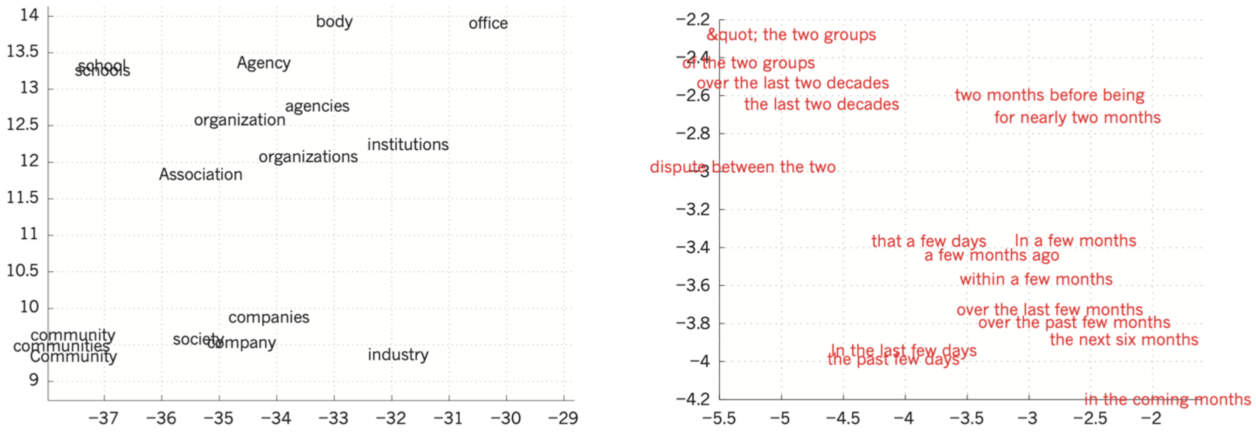
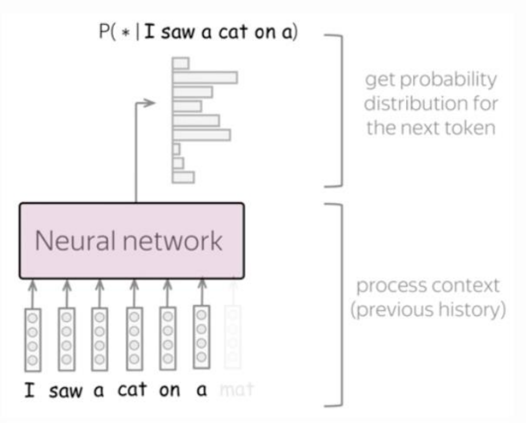


Fig. 4

Language Processing

Language Model with Neural Network

- The other layers of the network learn to convert the input word vectors into an output word vector for the predicted next word
- It can be used to predict the probability for any word to appear as the next word

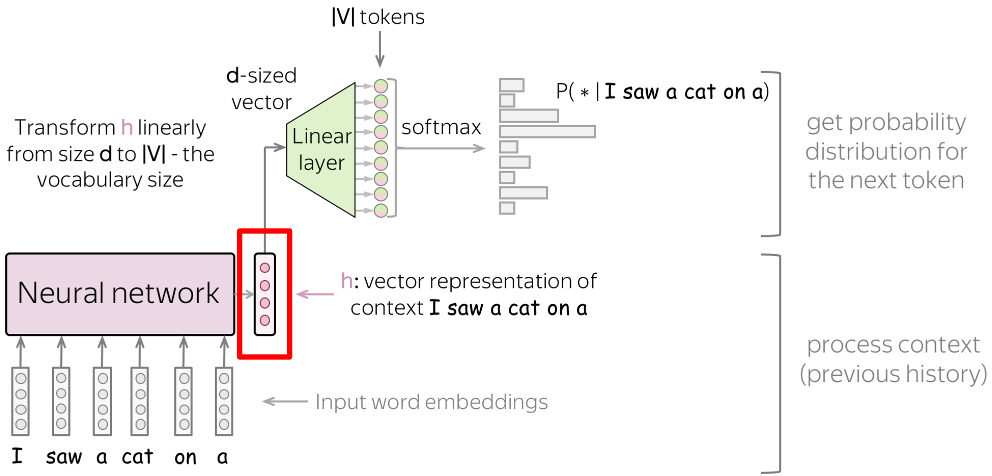


https://lena-voita.github.io/nlp_course/language_modeling.html

Language Processing

☐ Language Model with Neural Network

- The network learns **word vectors that contain many active components** each of which can be interpreted as a separate feature of the word



https://lena-voita.github.io/nlp_course/language_modeling.html

Recurrent Neural Network

What is RNN?

- RNN is a type of neural network that handles sequential inputs, such as speech and language by utilizing previous inputs

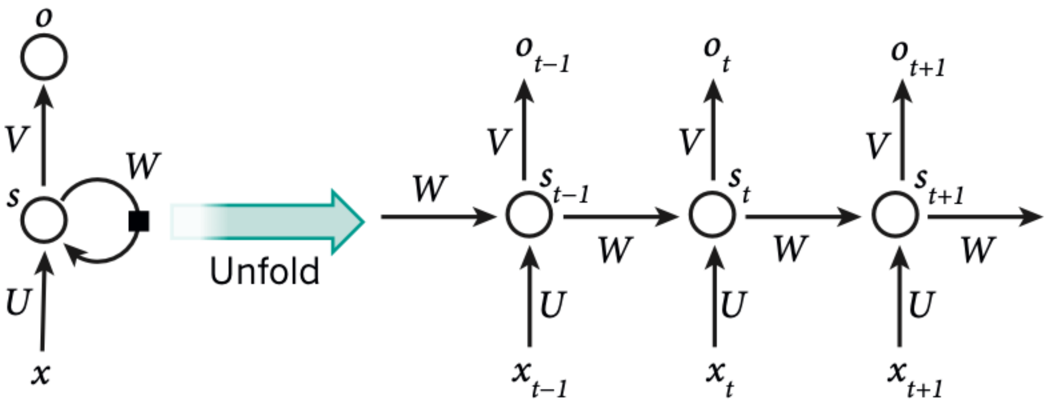
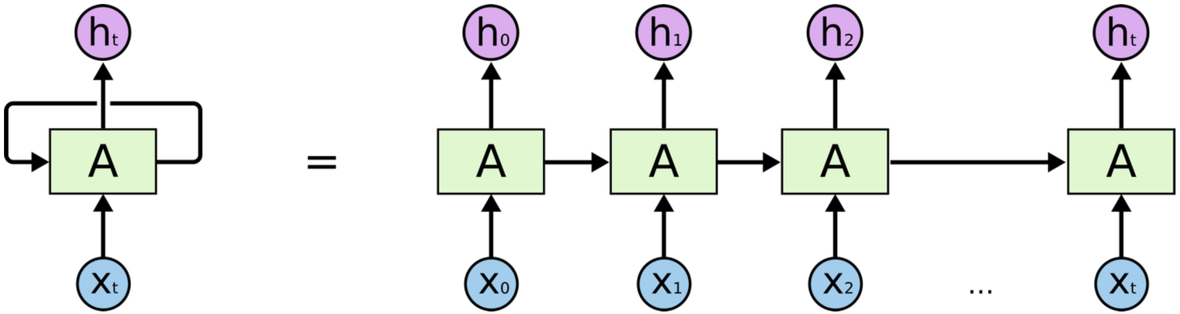


Fig. 5

Recurrent Neural Network

☐ What is RNN?

- RNNs process an input sequence one element at a time, maintaining in their hidden units a 'state vector'
- State vector is that implicitly contains information about the history of all the past elements of the sequence

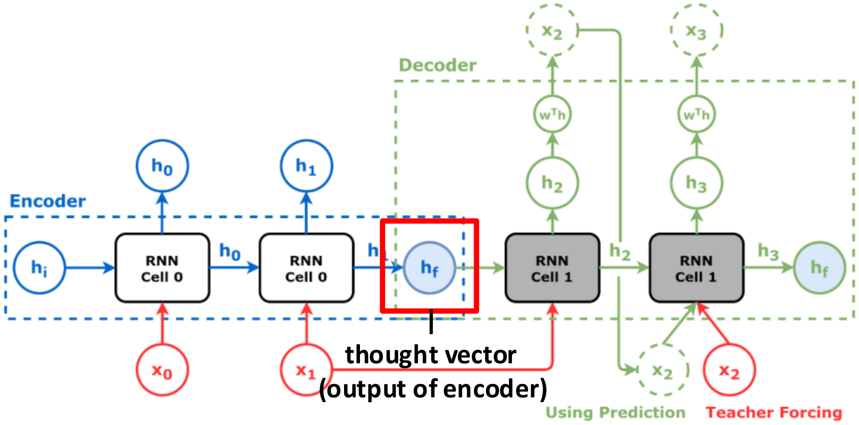


https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Recurrent Neural Network

☐ Language Model with RNN

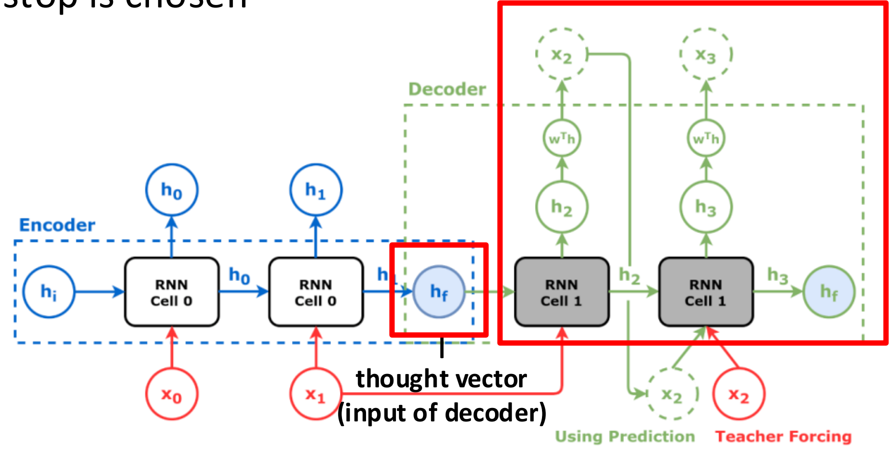
- RNNs have been found to be very good at predicting the next character in the text or the next word in a sequence but they can also be used for more complex tasks
- An English ‘encoder’ network can be trained so that the final state vector of its hidden units is a good representation of the thought expressed by the sentence



Recurrent Neural Network

☐ Language Model with RNN

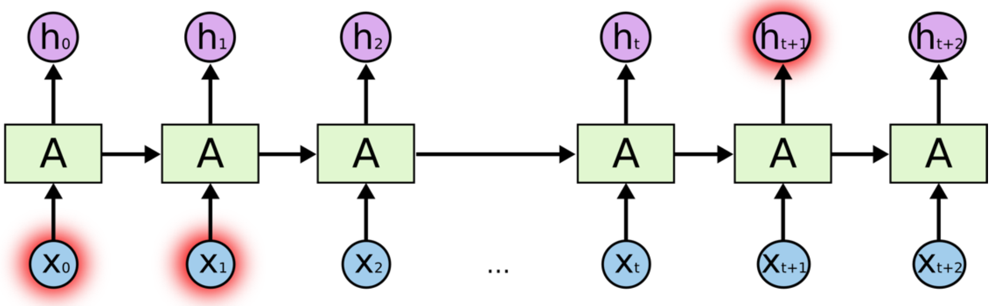
- Thought vector can then be used as the initial hidden state of French 'decoder' network, which outputs a probability distribution for the first word of the French translation
- And it will then output a probability distribution for the second word of the translation and so on until a full stop is chosen



Recurrent Neural Network

☐ Problem of RNN

- RNNs are very powerful dynamic systems, but training them has proved to be problematic because the backpropagated gradients either grow or shrink at each time step
- Over many time steps they either explode or vanish



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

Recurrent Neural Network

☐ Why do these problems occur?

$$\frac{\partial L}{\partial W_h} = \sum_{t=1}^{T} \frac{\partial L}{\partial h_T} \cdot \frac{\partial h_T}{\partial W_h} \rightarrow \text{chain rule} \rightsquigarrow \sum_{t=1}^{T} \frac{\partial L}{\partial h_T} \cdot \frac{\partial h_T}{\partial h_t} \cdot \frac{\partial h_t}{\partial W_h}$$

$$\rightsquigarrow \sum_{t=1}^{T} \frac{\partial L}{\partial h_T} \cdot \prod_{k=t}^{T} \frac{\partial h_k}{\partial h_{k-1}} \cdot \frac{\partial h_t}{\partial W_h}$$

$$\frac{\partial h_T}{\partial h_t} = \frac{\partial h_T}{\partial h_{T-1}} \cdot \frac{\partial h_{T-1}}{\partial h_{T-2}} \cdots \frac{\partial h_{t+1}}{\partial h_t}$$

$$= \prod_{k=t}^{T} \frac{\partial h_k}{\partial h_{k-1}}$$

대입

$$\Downarrow$$

$$\frac{\partial L}{\partial W_h} = \sum_{t=1}^{T} \frac{\partial L}{\partial h_T} \cdot \prod_{k=t}^{T} \frac{\partial h_k}{\partial h_{k-1}} \cdot \frac{\partial h_t}{\partial W_h}$$

hidden state $= h_t = f(z)$

$$= f(W_h \cdot h_{t-1} + W_x \cdot x_t + b)$$

$$\frac{\partial h_t}{\partial h_{t-1}} = f'(z_t) \cdot \frac{\partial z_t}{\partial h_{t-1}}$$

$$z_t = W_h \cdot h_{t-1} + W_x \cdot x_t + b$$

$$\frac{\partial z_t}{\partial h_{t-1}} = W_h$$

z_t, $\frac{\partial z_t}{\partial h_{t-1}}$ 대입 $\rightarrow$ $\frac{\partial h_t}{\partial h_{t-1}} = f'(W_h \cdot h_{t-1} + W_x \cdot x_t + b) \cdot W_h$

Recurrent Neural Network

☐ Why do these problems occur?

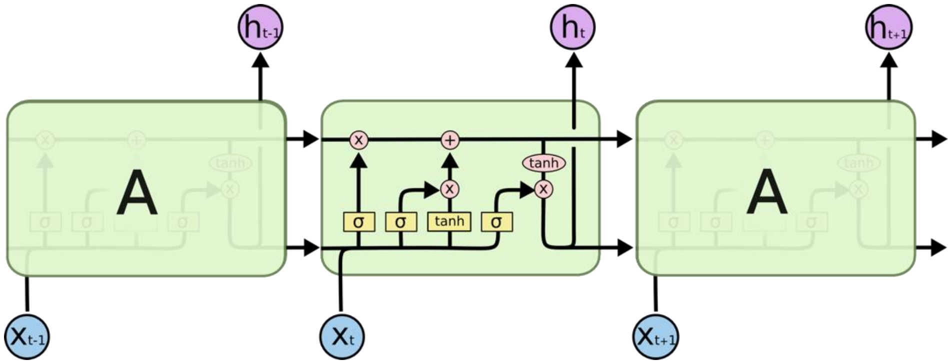
- Repeated multiplication of W_h during backpropagation can cause vanishing gradients, making learning ineffective, or exploding gradients, leading to unstable training

$$\frac{\partial L}{\partial W_h} = \sum_{t=1}^{T} \frac{\partial L}{\partial h_T} \prod_{k=t}^{T} \frac{\partial h_k}{\partial h_{k-1}} \frac{\partial h_t}{\partial W_h} \qquad \frac{\partial h_t}{\partial h_{t-1}} = f'(W_h h_{t-1} + W_x x_t + b) \cdot W_h$$

Recurrent Neural Network

☐ RNN with an Explicit Memory

- **Long short-term memory (LSTM)** networks are a special kind of RNN, capable of learning long-term dependencies
- A special unit called the memory cell acts like an accumulator or a gated leaky neuron

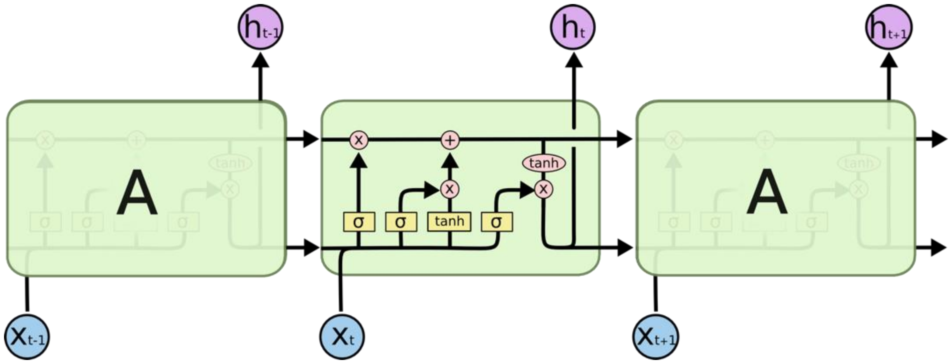


https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Recurrent Neural Network

☐ RNN with an Explicit Memory

- LSTM unit has a connection to itself at the next time step that has a weight of one, so **it copies its own real-valued state** and **accumulates the external signal**
- And this self-connection is multiplicatively gated by another unit that **learns when to clear the content of the memory**



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

The Future of Deep Learning

Unsupervised Learning

- Unsupervised learning to become far more important in the longer term
- Human learning is largely unsupervised, not by being told the name of every object

The Future of Deep Learning

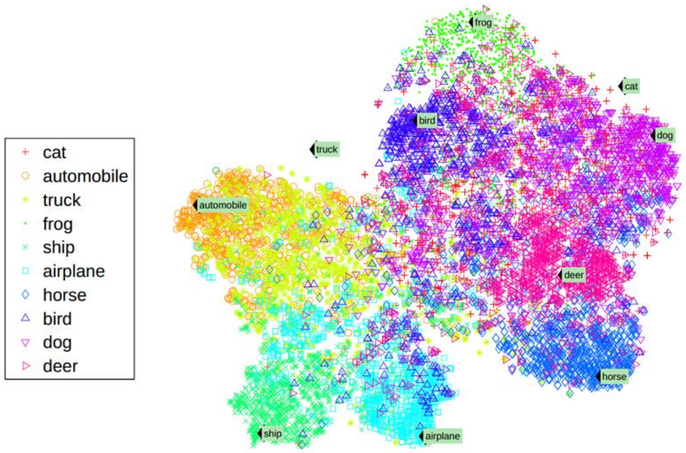
□ Reinforcement Learning and Complex Reasoning

- Much of the future progress in vision to come from systems that are trained end-to-end and combine CNNs with RNNs that use reinforcement learning to decide where to look
- Ultimately, major progress in artificial intelligence will come about through systems that combine representation learning with complex reasoning

The Future of Deep Learning

☐ New Paradigms with Distributed Representation

- New paradigms are needed to replace rule-based manipulation of symbolic expressions by operations on large vectors



ImageNet Classification with Deep Convolutional Neural Networks, Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton

Conclusion

- **Deep learning** brought significant innovation to artificial intelligence research by enabling multi-layer neural networks to **learn complex data representations automatically**

- **CNNs** demonstrated remarkable performance in image and video processing, while **RNNs** have excelled in handling sequential data such as text and speech

- Future advancements in AI are expected to **integrate deep learning with reinforcement learning**, **utilizing advanced reasoning capabilities** to build more powerful AI systems

Thank you