Brief Concepts of Deep Learning and Computer Vision

ZIJUN ZHANG

Limitations of Shallow Models

- Shallow Neural Networks: Three-layer Perceptron – One input layer, One hidden layer, and One output layer
- Capability of modeling highly nonlinear and complex process limited
- Limited ability to process natural data in raw form
- The Selectivity-Invariance dilemma
- Backpropagation and Gradient Descent might not be effective to well train perceptron with a large number of hidden layers

The selectivity invariance dilemma

Majorly in image and speech recognition problems, which require model to be insensitive to irrelevant variations of the data.

For example, position, orientation, and illumination of the same object;

Variations in pitch or accent of speech

While being sensitive to minute variations, e.g., white wolf and breed of wolf-like white dog, Samoyed

e.g., Samoyed with different positions could be very different but wolf and samoyed with the same position could be similar

The selectivity variance dilemma

- To achieve the goal, shallow models need a very good feature extractor (A lot of trials and design efforts)
- One that produces representations that are:
  - Selective to aspects of image important for discrimination
  - But irrelevant to irrelevant aspects such as pose of the animal
- Generic features (Gaussian kernels) do not generalize well far from training examples;
- Hand-crafted good feature extractors requires engineering skills and domain expertise;
- Deep learning targets on automatically learning representative features.

Features of Deep Models

- Computational models composed of multiple processing layers
  - To learn representations of data with multiple levels of abstraction
- Dramatically improved state-of-the-art in:
  - Speech recognition, Visual object recognition, Object detection
  - Other domains: Drug discovery, Genomics
- Discovers intricate structure in large data sets
  - Using backpropagation to update parameters
  - Compute representation in each layer from previous layer
- Deep convolutional nets: image proc, video, speech
- Recurrent nets: sequential data, e.g., text, speech

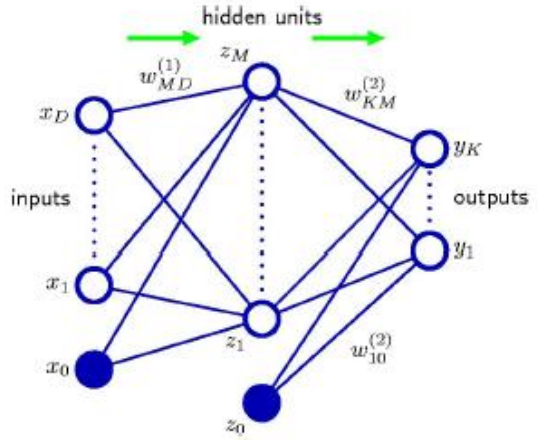
Representation learning

- Methods that allow a machine to be fed with raw data to automatically discover representations needed for detection or classification
- Deep Learning methods are Representation Learning Methods
- Use multiple levels of representation
  - Composing simple but non-linear modules that transform representation at one level (starting with raw input) into a representation a ta higher slightly more abstract level
  - Complex functions can be learned
  - Higher layers of representation amplify aspects of input important for discrimination and suppress irrelevant variations

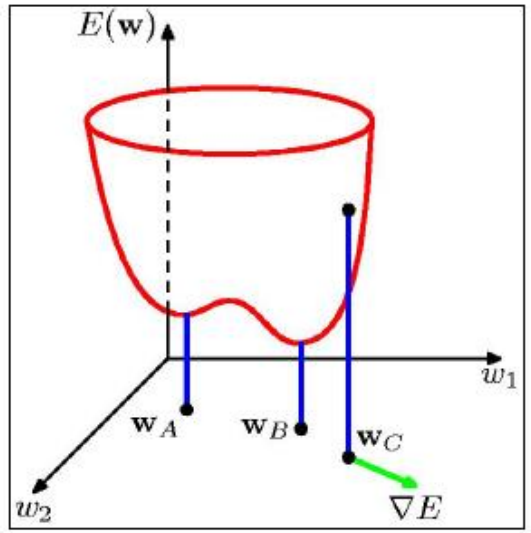
Image Example

- Input is an array of pixel values
  - First stage is presence or absence of edges at particular locations and orientations of image
  - Second layer detects motifs by spotting particular arrangements of edges, regardless of small variations in edge positions
  - Third layer assembles motifs into larger combinations that corresponds to parts of familiar objects
  - Subsequent layers would detect objects as combinations of these parts
- Key aspect of deep learning:
  - These layers of features are not designed by human engineers
  - They are learned from data using a general purpose learning procedure

Gradient Descent



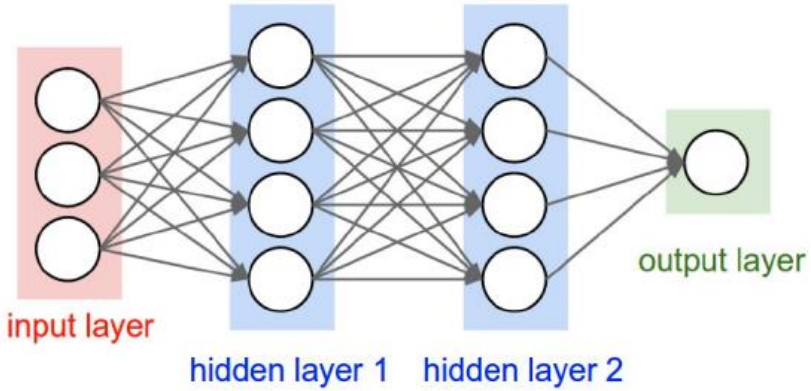
- Objective function, averaged over all training examples is a hilly landscape in high-dimensional space of weight values
- Negative gradient vector indicates direction of steepest descent taking it closer to a minimum
- Goal is to learn the weights w from a labelled set of training samples
- Learning procedure has two stages
  1. Evaluate derivatives of error function $\nabla E(w)$ with respect to weights $w_1,..w_T$
  2. Use derivatives to compute adjustments to weights $w^{(\varsigma+1)} = w^{(\varsigma)} - \eta \nabla E(w^{(\varsigma)})$



Stochastic Gradient Descent

- In deep learning, SGD is most commonly applied
- Consists of showing the input vector for a few examples, computing the outputs and errors.
- Computing the average gradient for those examples, and adjusting the weights accordingly.
- Process repeated for many small sets of examples from the training set until the average of the objective function stops decreasing
- Stochastic because each small set returns a noisy estimate of the average gradient over all examples

Deep Learning Architecture



- Multilayer stack of simple modules
- All modules (or most) subject to :
  - Learning
  - Non-linear input-output mappings
- Each module transforms input to improve both selectivity and invariance of the representation
- With depth of 5 to 20 layers can implement extremely intricate functions of input
  - Sensitive to minute details: Distinguish Samoyeds from white wolves
  - Insensitive to irrelevant variations: Background, pose, lighting, surrounding objects

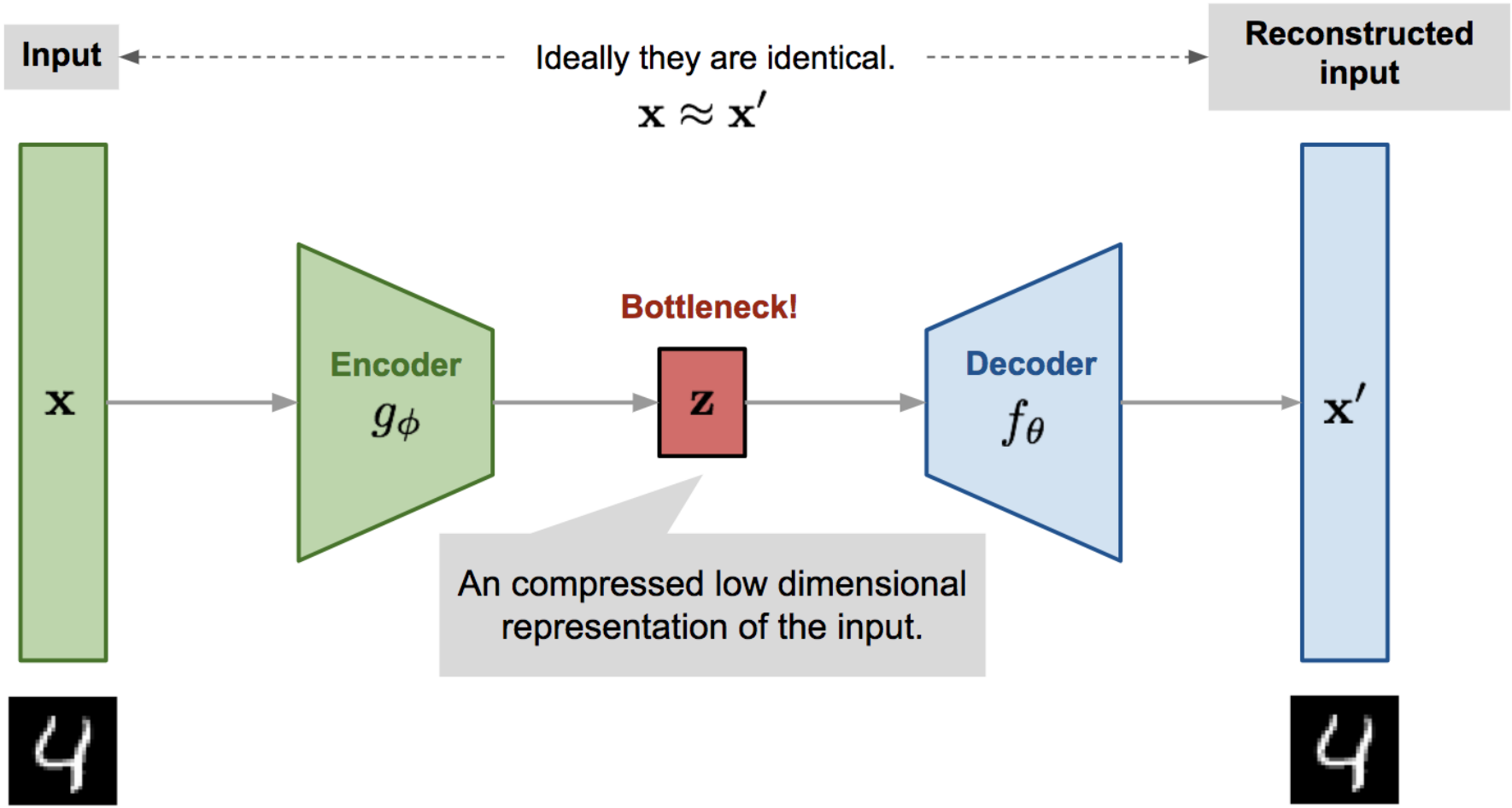
Local minima and saddle points

- In low dimensional problem, we worry about GD got trapped in local minima
- In high dimensional problem, we worry about saddle points
- Local minima is rarely a problem with large networks
  - Regardless of initial conditions, system always reaches solutions of similar quality
  - Landscape is packed combinatorially large number of saddle points where gradient is zero
  - Almost all have similar values of objective function

Two-stage Semi-supervised Learning

- Pre-training with an unsupervised learning scheme to develop hidden layers stacked in the deep network
  - Autoencoders, Deep autoencoders
- Fine-tuning by adding the output layer on top of the network, which is a supervised learning
  - Regular BP

Autoencoders



Convolutional Neural Networks

- Type of deep forward network
- Much easier to train and generalized much better than networks with full connectivity between adjacent layers
- Designed to process data coming in the form of multiple arrays
  - E.g., a color image composed of three 2-d arrays of pixel intensities in three color channels
- Many data modalities are in the form of multiple arrays:
  - 1-d for signals and sequences, including language
  - 2-d for images and audio spectrograms
  - 3-d for video or volumetric images

Four key ideas behind ConvNets

Take advantages of properties of natural singals:

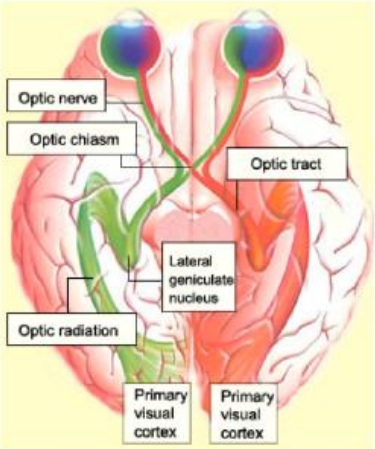
- Local connections
- Shared weights
- Pooling
- Use of many layers

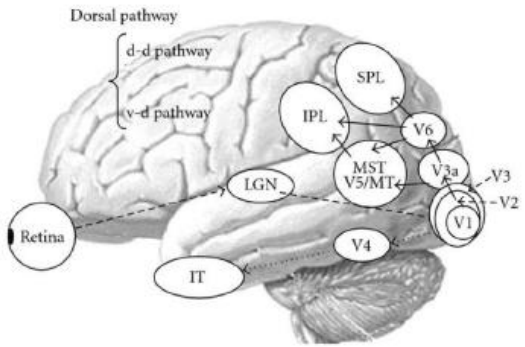
Exploitation of Local Properties

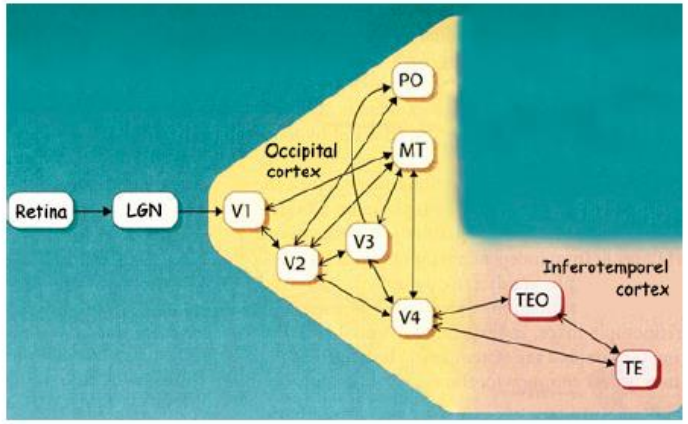
- Network should exhibit invariance to translation, scaling, and elastic deformations
  - A large training set can take care of this
- Shallow NN ignores a key property of images
  - Nearby pixels are more strongly correlated than distant ones
  - Modern computer vision methods exploit this property
- Information can be merged at later stages to get higher order features and about whole image

CNN inspired by Visual Neuroscinece

- Architecture similar to LGN-v1-v2-v4-IT hierarchy in visual cortex ventral pathway
  - LGN: lateral geniculate nucleus receives input from retina
  - 30 different areas of visual cortex: V1 and V2 are principal
  - Infero-Temporal cortex performs object recognition

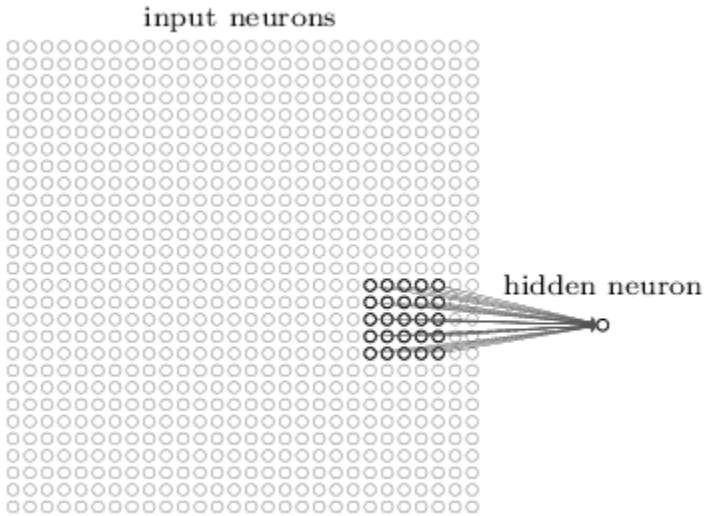






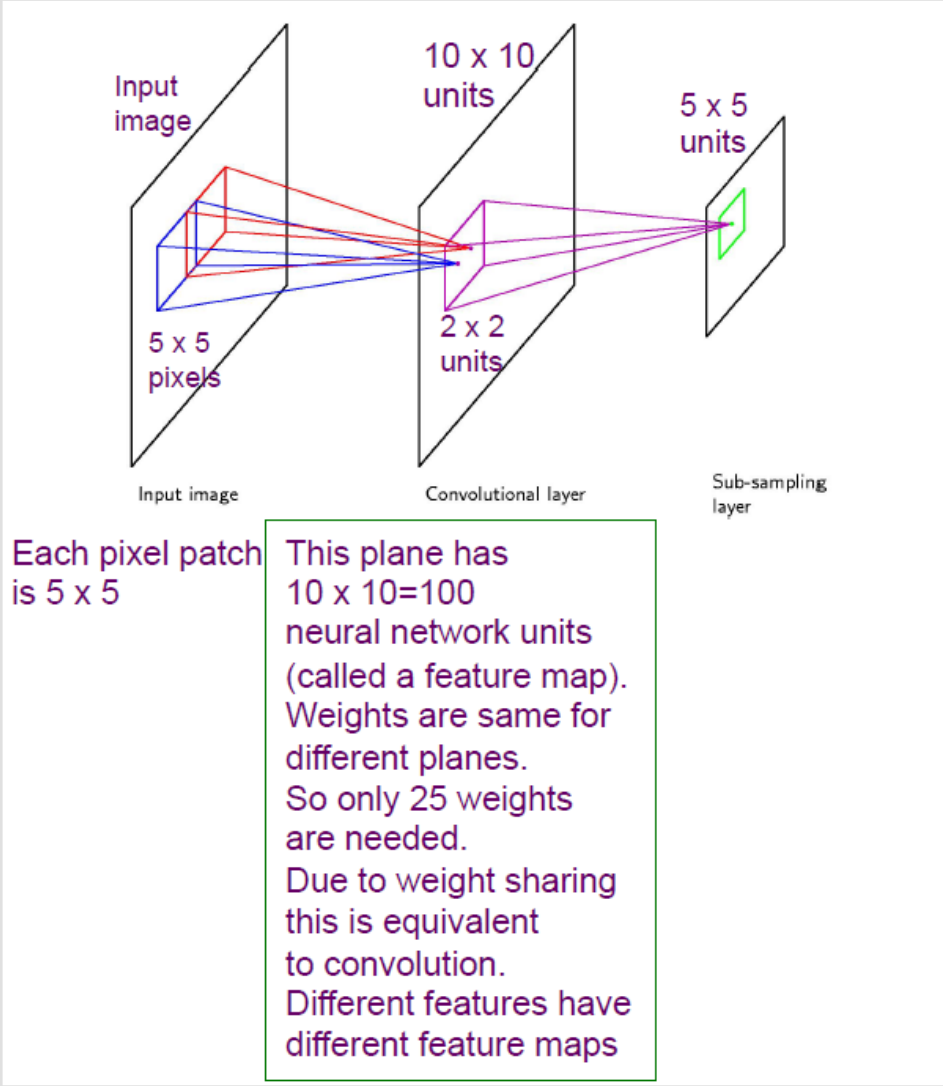
Three mechanisms of CNN

- Local receptive fields
- Subsampling -> Pooling
- Weight sharing



Convolution and Subsampling

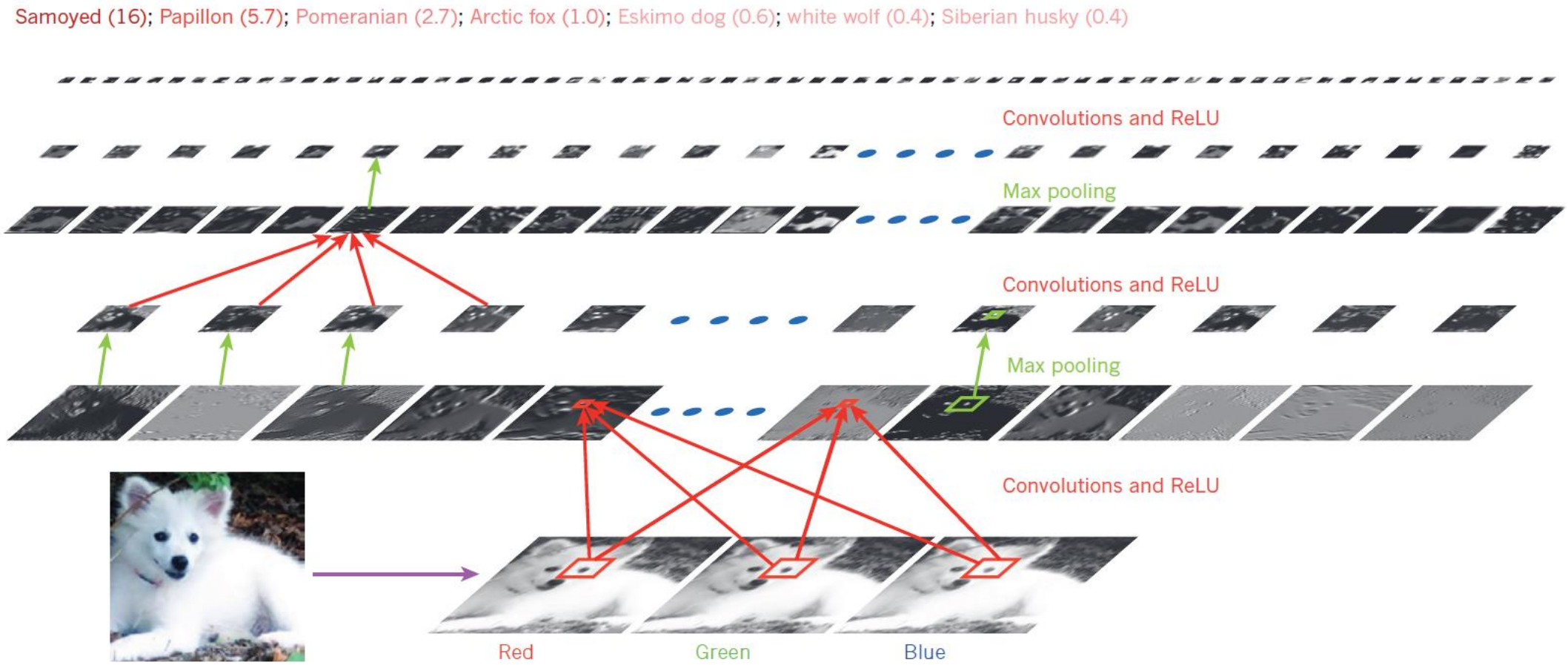
- Instead of treating input to a fully connected network
- Two layers of Neural networks are used
  1. Layer of convolutional units which consider overlapping regions
  2. Layer of subsampling units
- Several feature maps and subsampling
  - Gradual reduction of spatial resolution compensated by increasing no. of features
- Final layer has softmax output
- Whole network trained using backpropagation



Soft weight sharing

- Reducing the complexity of a network
- Encouraging groups of weights to have similar values
- Only applicable when form of the network can be specified in advance
- Division of weights into groups, mean weight value for each group and spread of values are determined during the learning process

Example


Samoyed (16); Papillon (5.7); Pomeranian (2.7); Arctic fox (1.0); Eskimo dog (0.6); white wolf (0.4); Siberian husky (0.4)
Convolutions and ReLU
Max pooling
Convolutions and ReLU
Max pooling
Convolutions and ReLU
Red
Green
Blue

Architecture of a typical CNN

- Structured as a series of stages
- First few stages are composed of two types of layers and a non-linearity:
  1. Convolutional layer: To detect local conjunctions of features from previous layer
  2. Non-linearity: ReLU
  3. Pooling layer: To merge semantically similar features into one

A convolutional layer unit

- Organized in feature maps
- Each unit connected to local patches in feature maps of previous layer through weights (called a filter bank)
- Result is passed through a ReLU

A pooling layer unit

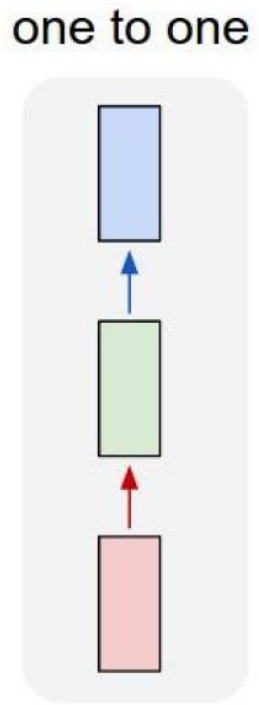
- Computes maximum of a local patch of units in one feature map
- Neighboring pooling units take input from patches that are shifted by more than one row or column
  - Thereby reducing the dimension of the representation
  - Creating invariance to small shifts and distortions

Backpropagation of Gradients through CNN

- As simple as through a regular deep network
- Allow all weights in all filter banks to be trained

Recurrent Neural Networks

- Vanilla Neural Network



RNN: Process Sequneces

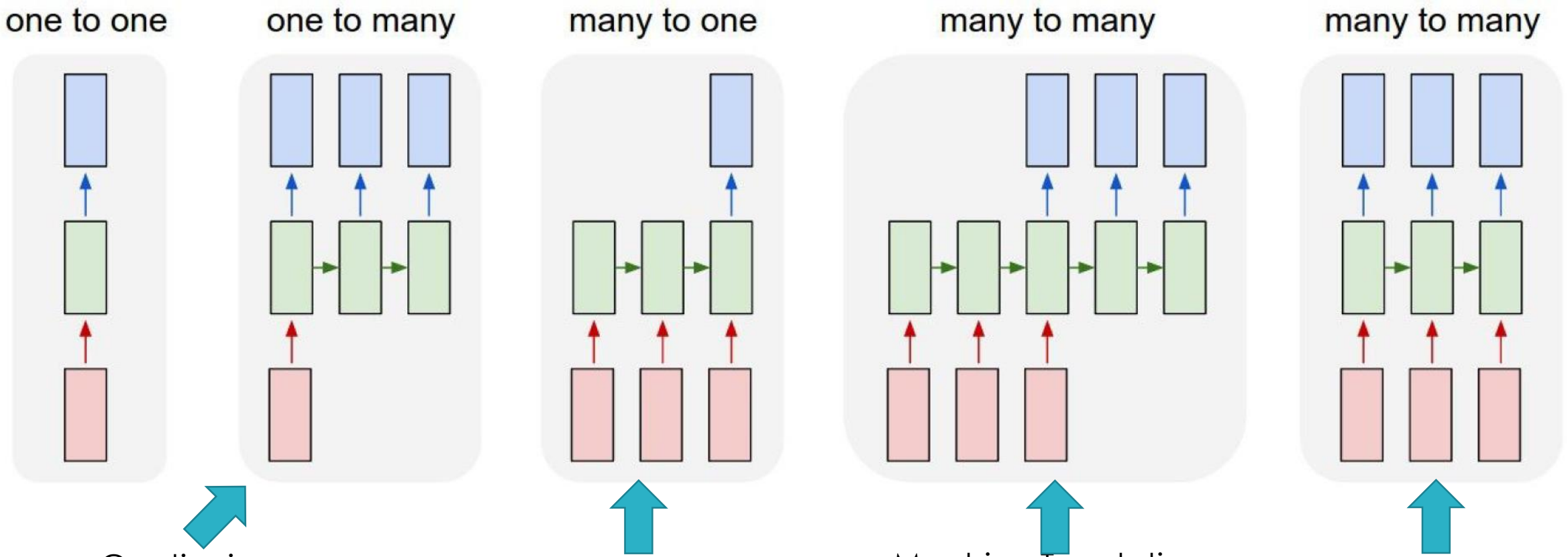


Image Captioning
Image->Sequence of objects

Sentiment classification
Sequence of words->
Sentiment

Machine Translation
Sequence of objects
->sequence of objects

Video classification

RNN

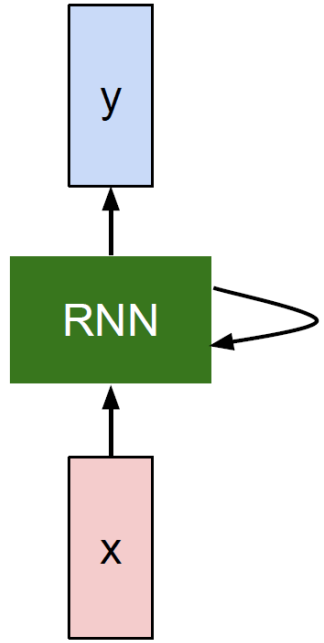
We want to build up such model which process a sequence x by applying a recurrence formula

$h_t = f_w(h_{t-1}, X_t)$

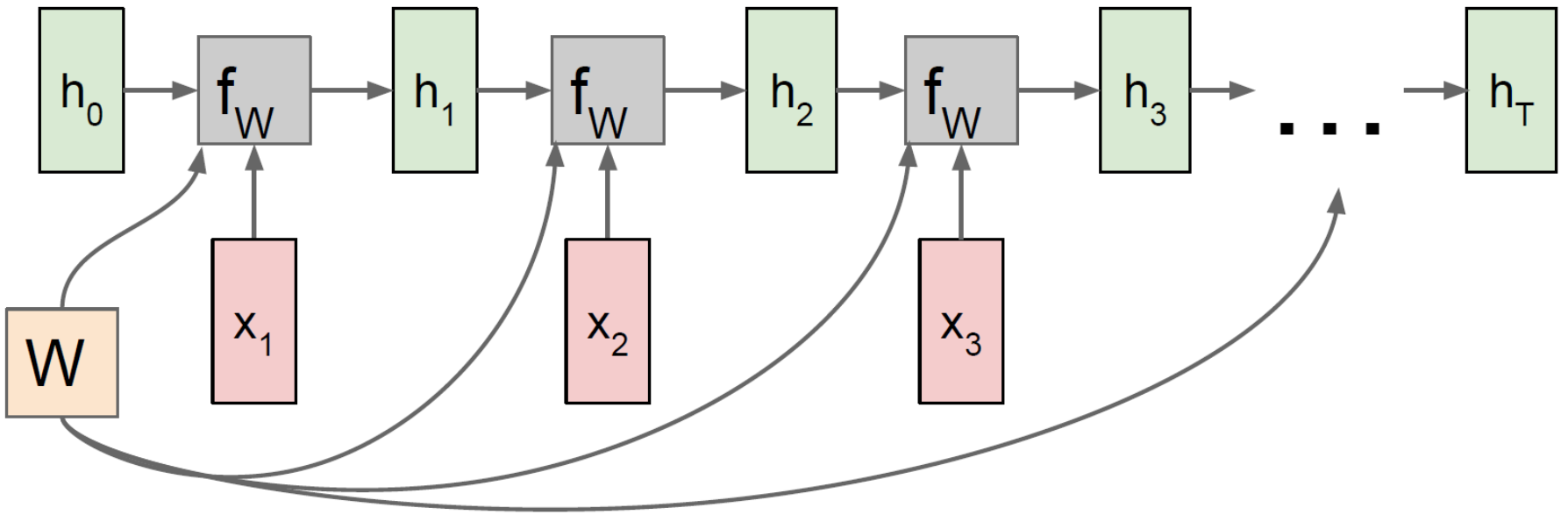
E.g.:

$h_t = \tanh(w_{hh}h_{t-1}+w_{xh}x_t)$

$y_t = w_{hy}h_t$

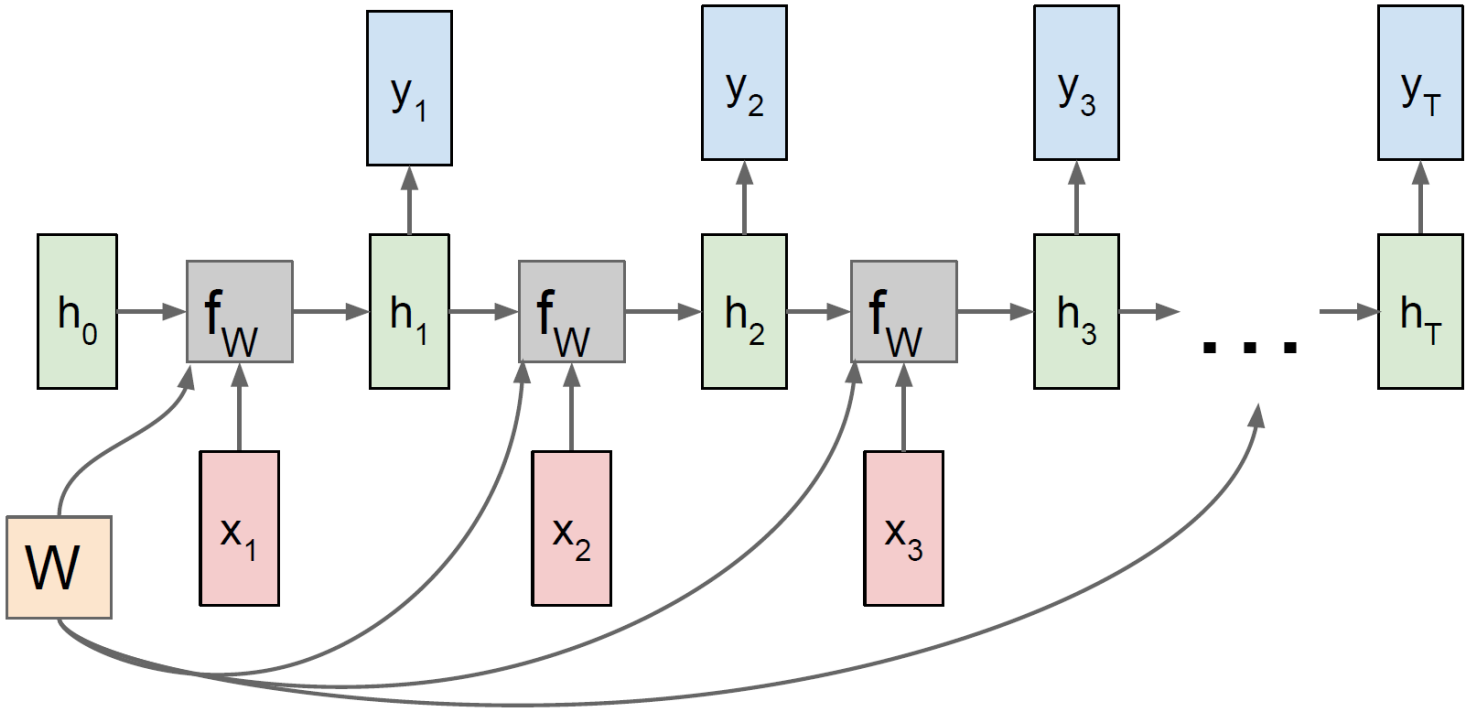


RNN: Computational Graph



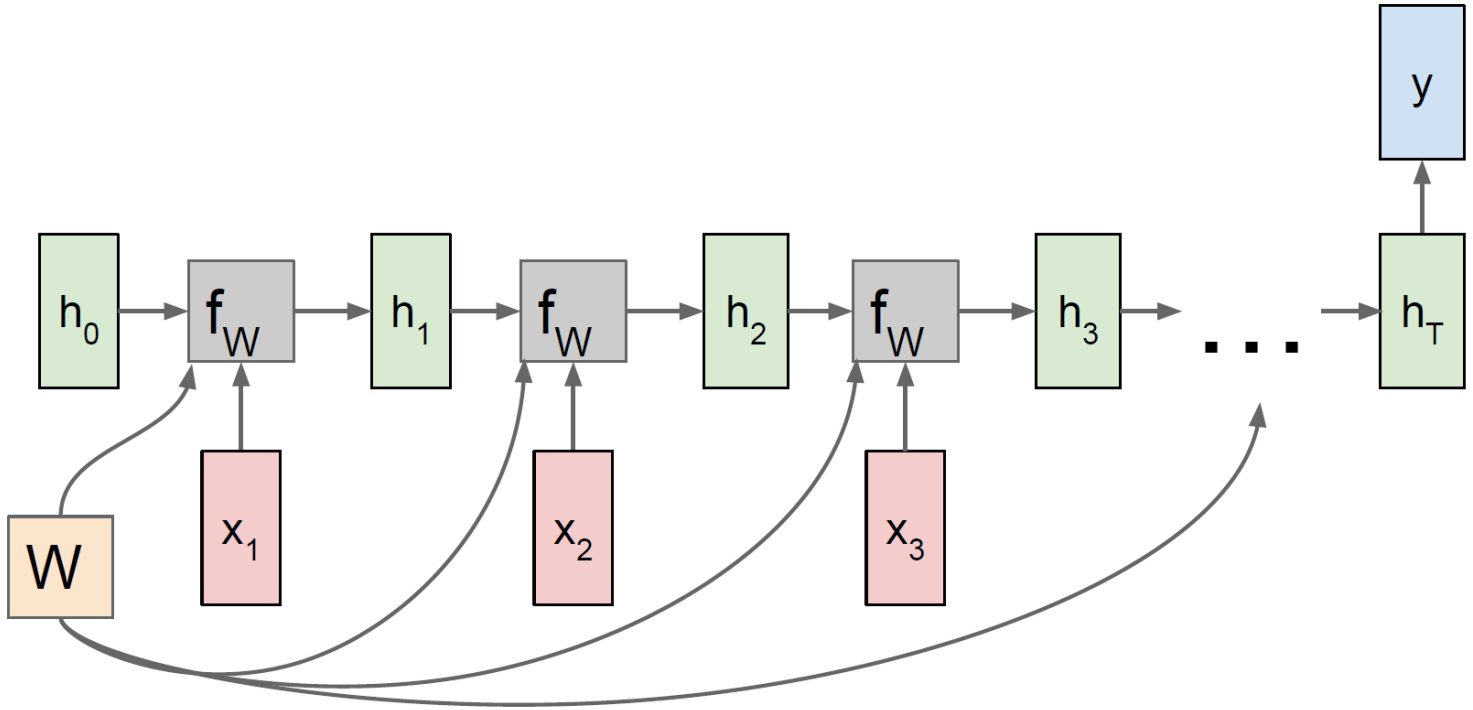
RNN: Computational Graph

- Many to Many



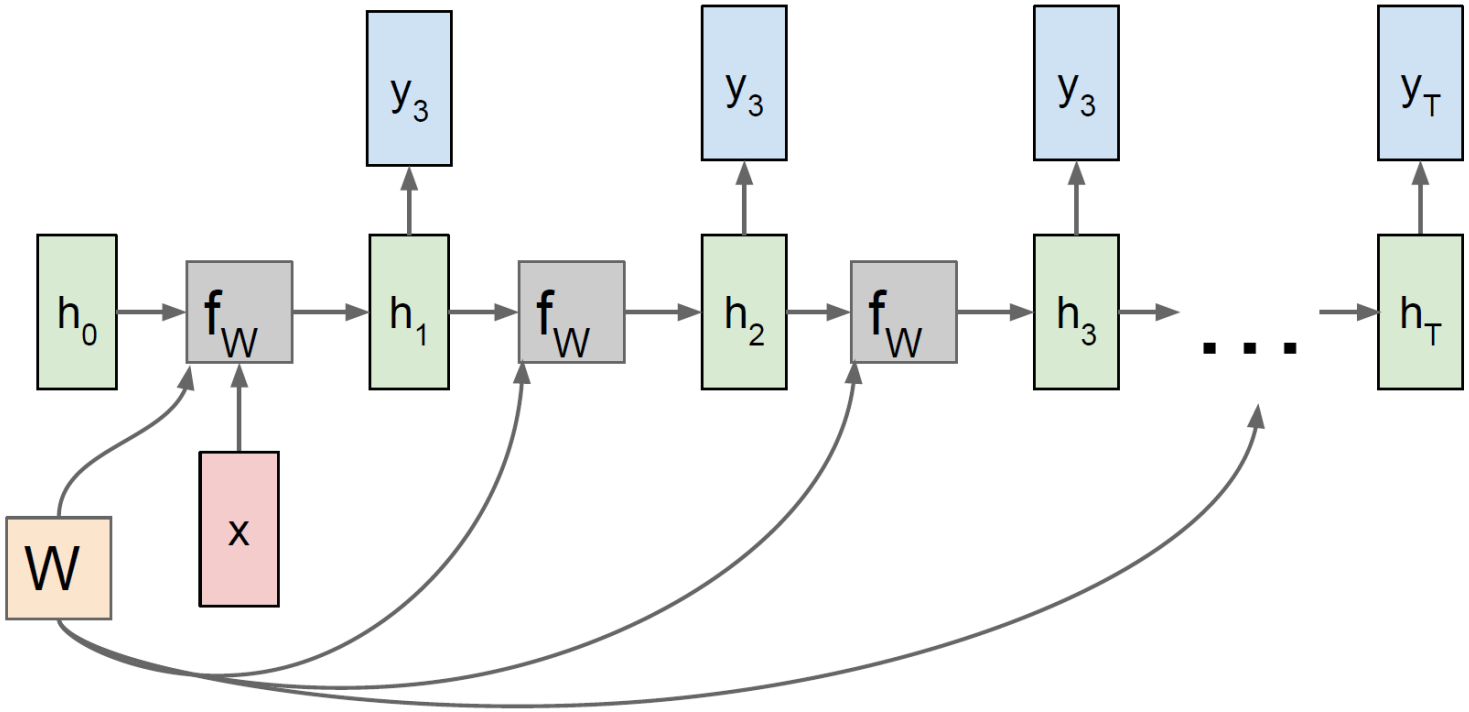
RNN: Computational Graph

- Many-to-one



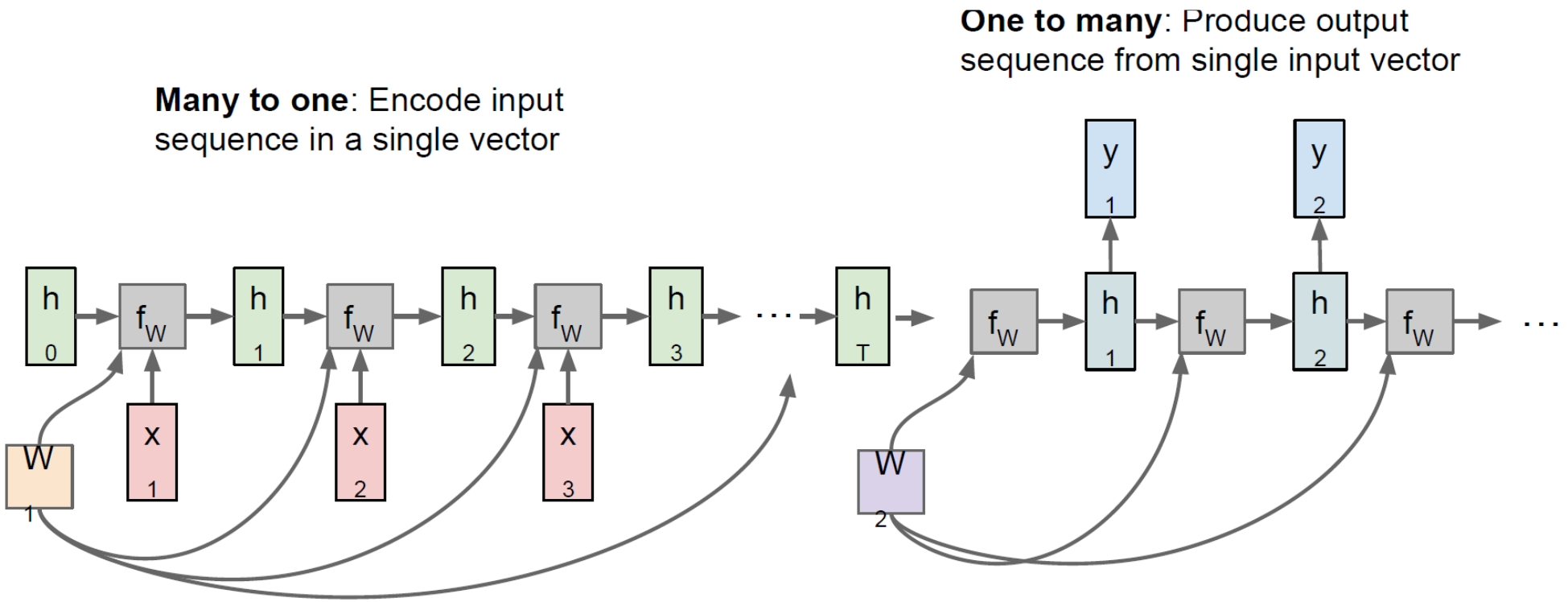
RNN: Computational Graph

- One-to-Many



RNN: Computational Graph

- Many-to-one+One-to-Many



RNN Variants

Disadvantages of Vanilla RNN

- Gradient vanishing and exploding problems
- Cannot process very long sequence if activation function is tanh
- It is very unstable if activation function is ReLU
- Difficult to be stacked into very deep model
- Cannot handle very long-term dependencies

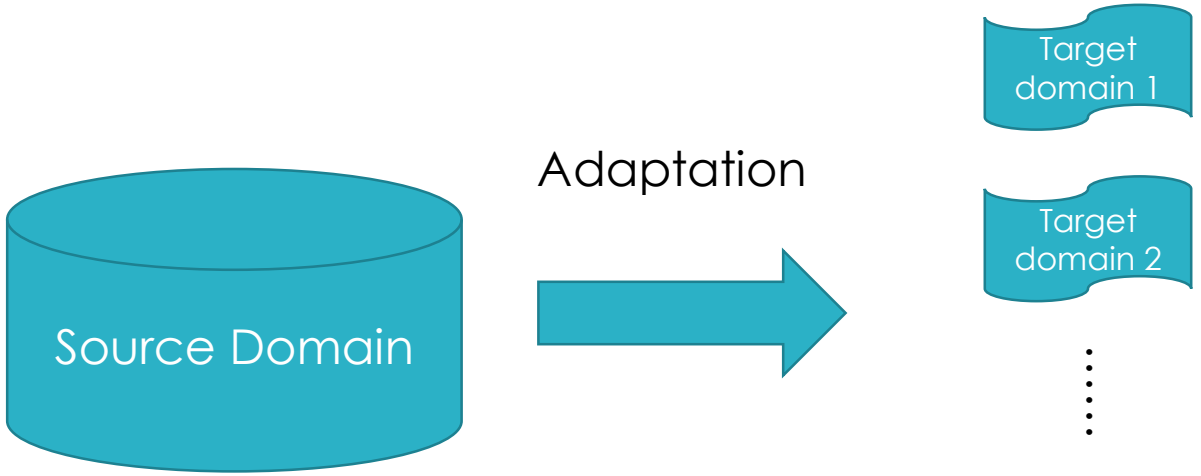
Variants of RNN: LSTM and GRU

Transfer Learning

- Deep neural networks train from scratch every time is time-consuming and wasting efforts
- Can we have a backbone model very generalized, developed from a large volume of data of a high diversity (library: ImageNet)
- Adapt backbone model into different specific applications by modifying network structure via a limited volume of data of those applications

Transfer Learning

New paradigm of organizing data



(x^s, y^s)-> training and validation

$(x^t, y^t)_k$-> fine-tuning and testing

Transfer Learning

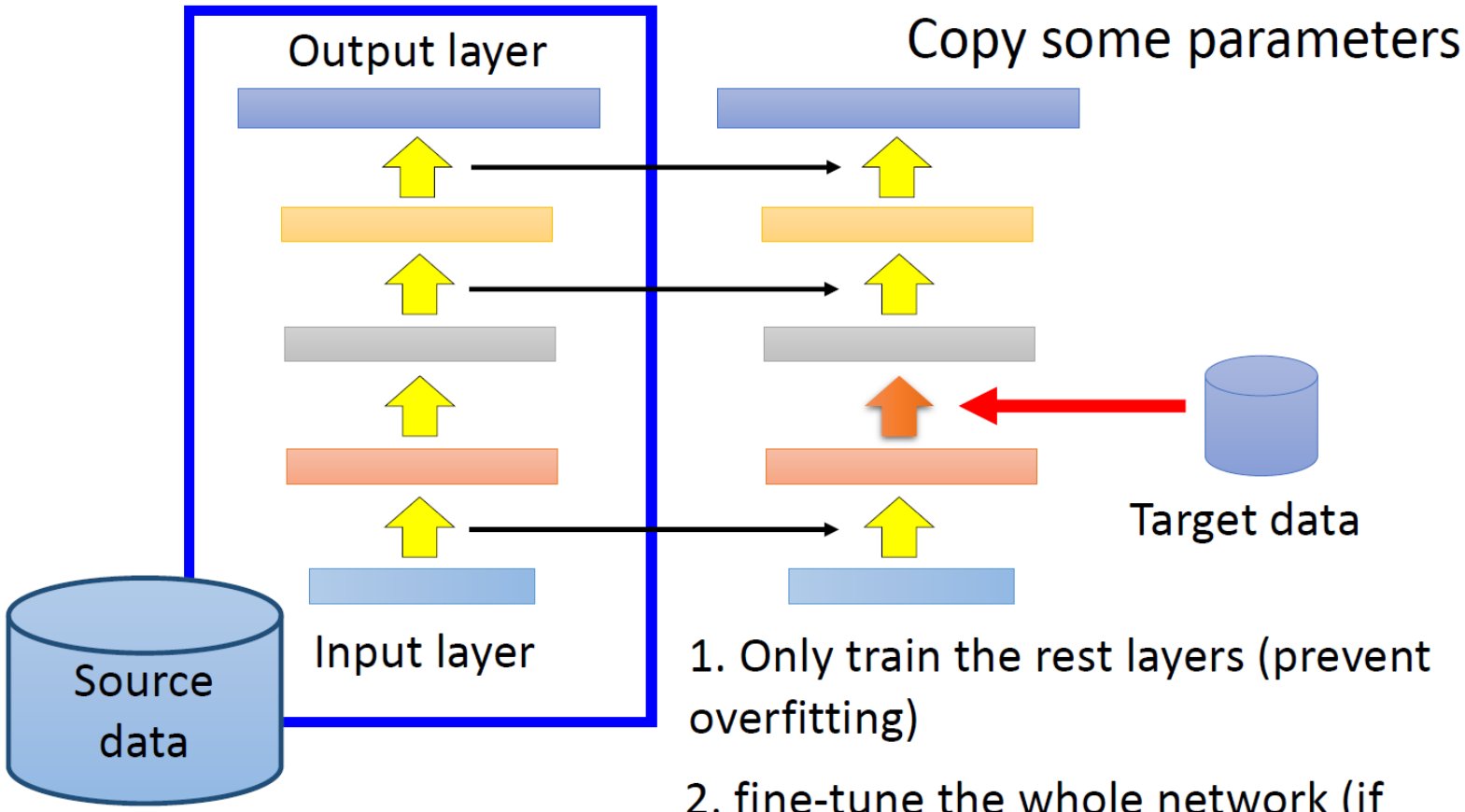
Strategy 1

Use backbone W and b as an initialization, update W and b based on target domains

Transfer Learning

Strategy 2

Layer transfer:



1. Only train the rest layers (prevent overfitting)

2. fine-tune the whole network (if there is sufficient data)

Transfer Learning

Layer transfer, which layer to be transferred?

Empirical,

Image processing: first few layers

Speech recognition: last few layers

Transfer Learning

Feature representation transfer:

Using source and target domain feature spaces to identify a latent representation which offers a generalization of relationships of latent variables mapped from various source and target domains