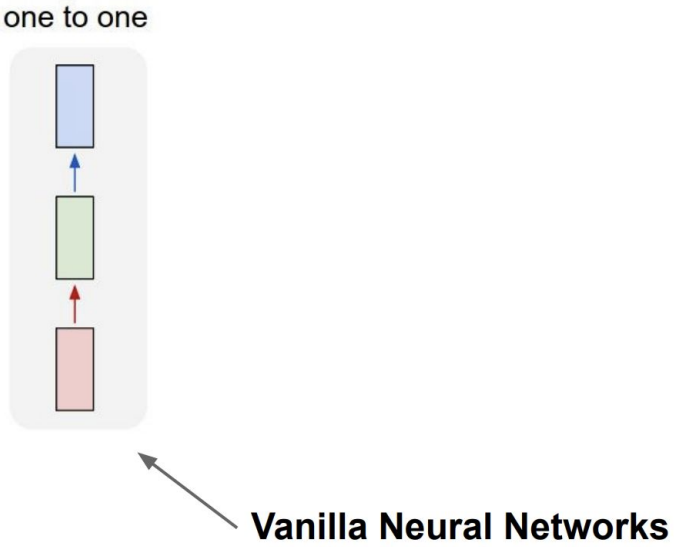
Week 9
RNN, LSTM,
and Semi-supervised Learning

Content

- Recurrent Neural Network (RNN)
- Long Short-term Memory (LSTM)
- Attention Mechanism
- Recap: Supervised Learning vs. Unsupervised Learning
- Semi-supervised Learning
- Example: RNN, LSTM in python

Recurrent Neural Network (RNN)

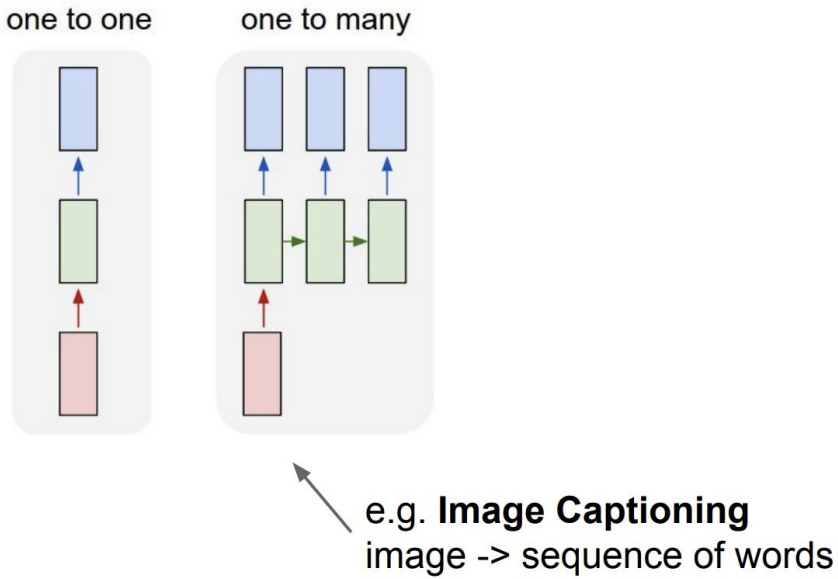
- **“Vanilla” Neural Network**



Li, R. (2022). Lecture 10: Self-supervised learning. [Lecture slides]. CS231n: Convolutional Neural Networks for Visual Recognition.

Recurrent Neural Network (RNN)

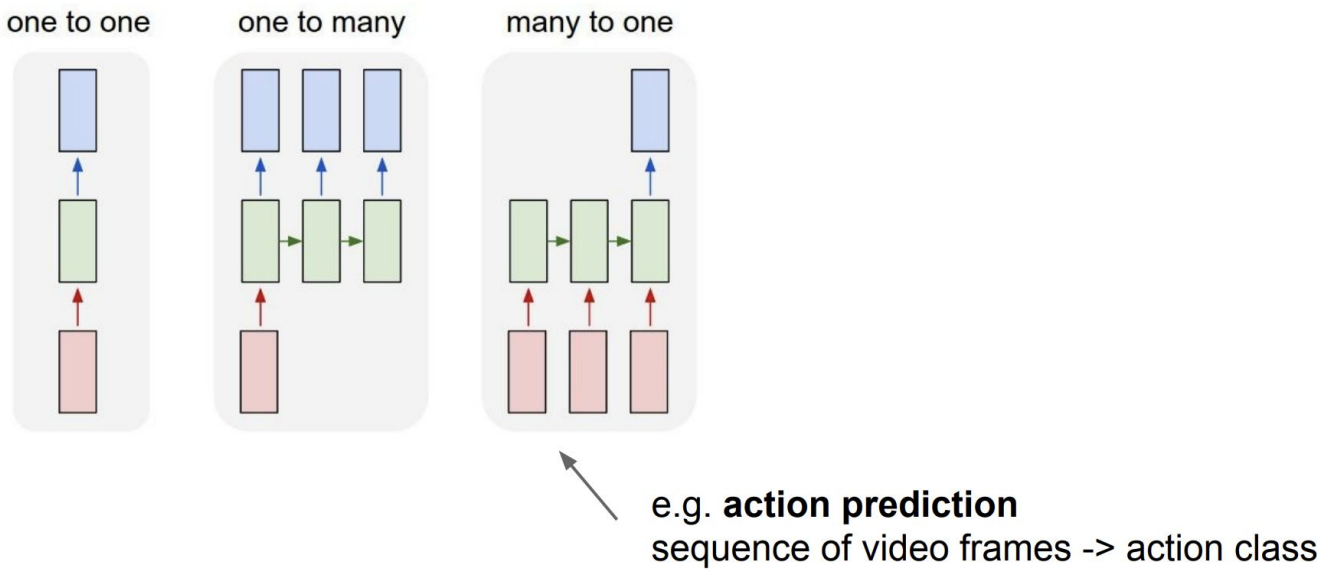
- **Recurrent Neural Networks: Process Sequences**



Li, R. (2022). Lecture 10: Self-supervised learning. [Lecture slides]. CS231n: Convolutional Neural Networks for Visual Recognition.

Recurrent Neural Network (RNN)

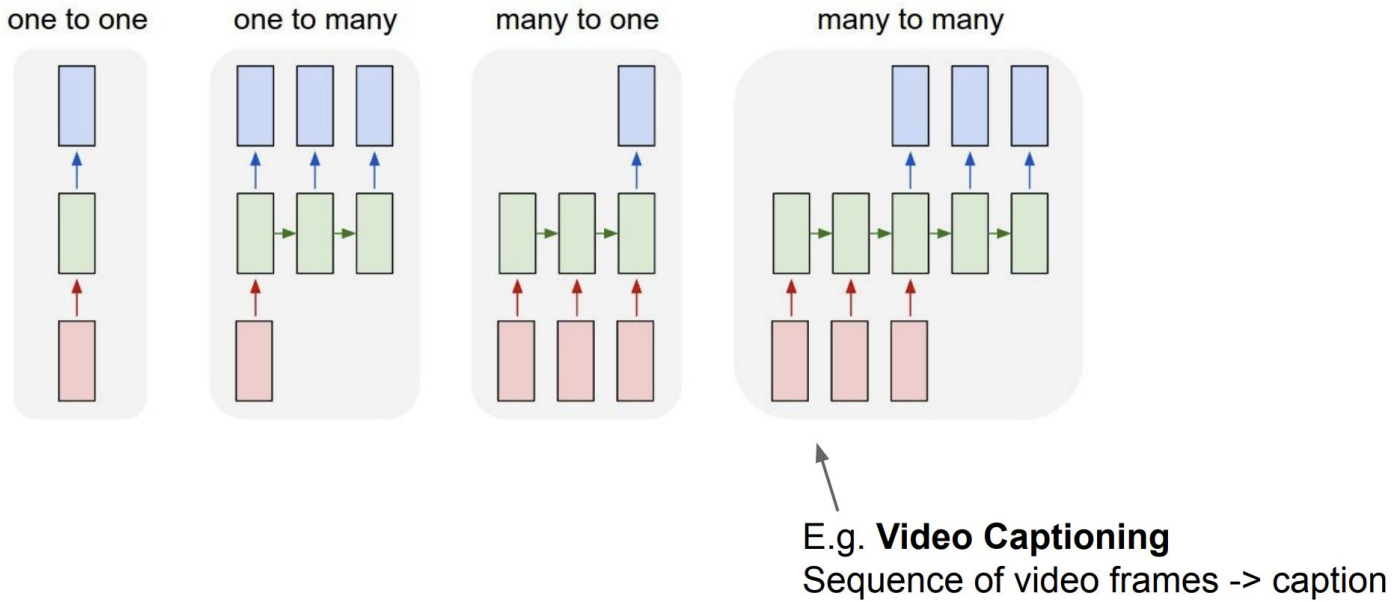
- **Recurrent Neural Networks: Process Sequences**



Li, R. (2022). Lecture 10: Self-supervised learning. [Lecture slides]. CS231n: Convolutional Neural Networks for Visual Recognition.

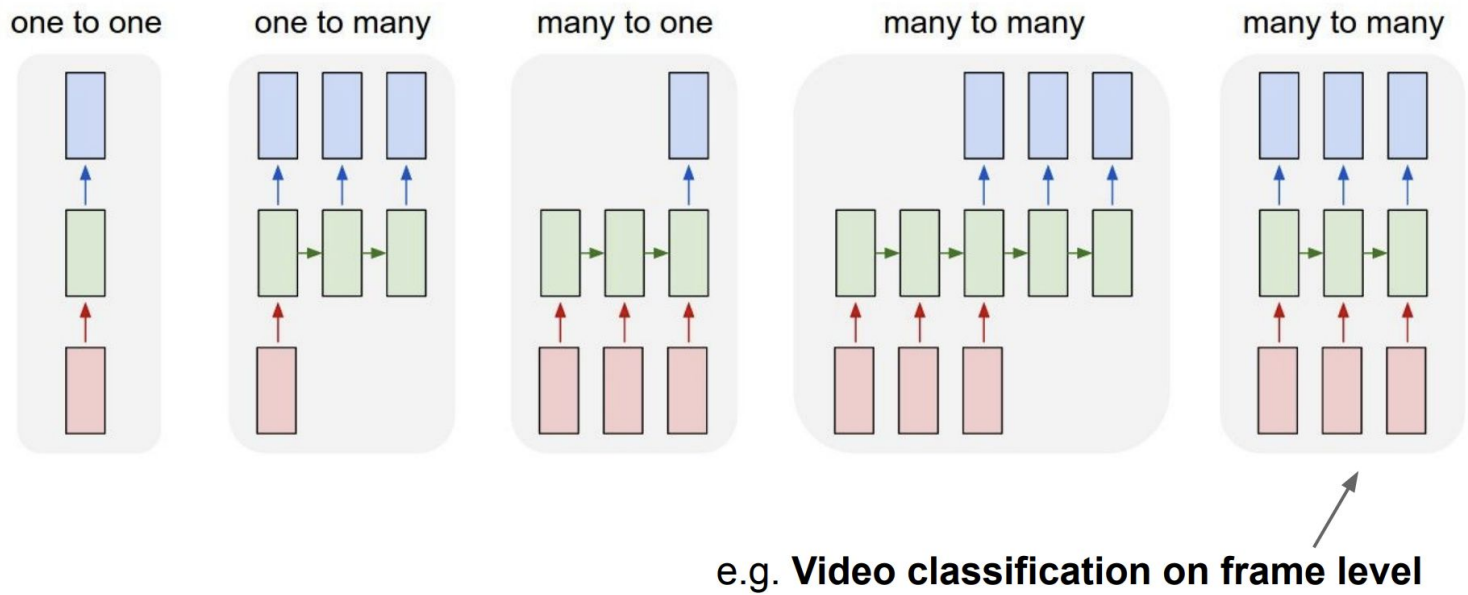
Recurrent Neural Network (RNN)

- **Recurrent Neural Networks: Process Sequences**



Recurrent Neural Network (RNN)

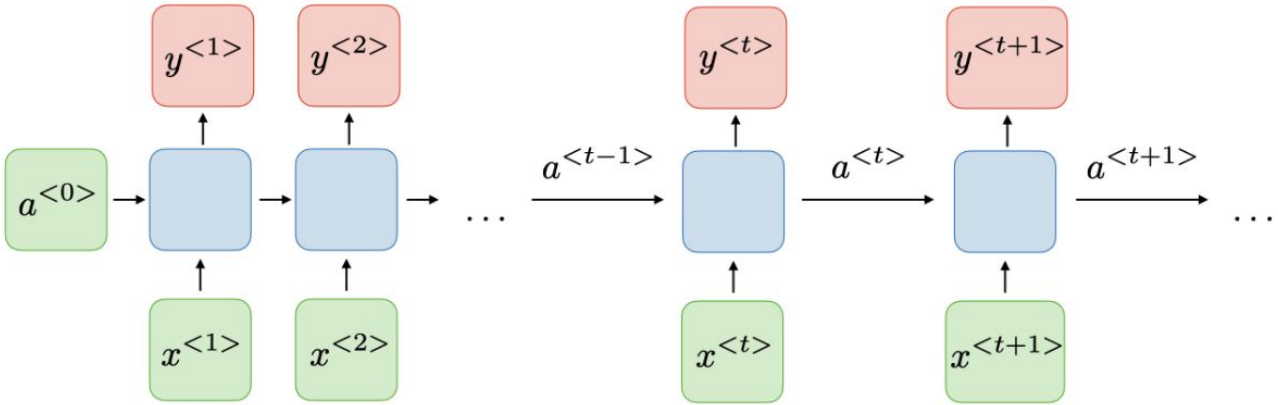
- **Recurrent Neural Networks: Process Sequences**



Li, R. (2022). Lecture 10: Self-supervised learning. [Lecture slides]. CS231n: Convolutional Neural Networks for Visual Recognition.

Recurrent Neural Network (RNN)

- **Architecture of a traditional RNN**
  - Recurrent neural networks, also known as RNNs, are a class of neural networks that allow previous outputs to be used as inputs while having hidden states. They are typically as follows:



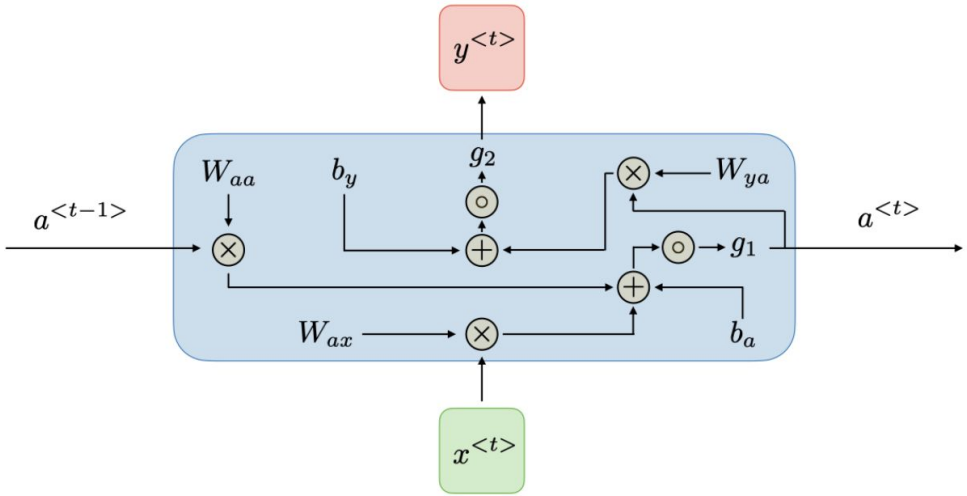
Recurrent Neural Network (RNN)

- **Architecture of a traditional RNN**

For each timestep t, the activation $a^{<t>}$ and the output $y^{<t>}$ are expressed as follows:

$$a^{<t>} = g_1(W_{aa}a^{<t-1>} + W_{ax}x^{<t>} + b_a) \quad \text{and} \quad y^{<t>} = g_2(W_{ya}a^{<t>} + b_y)$$

where $W_{ax}, W_{aa}, W_{ya}, b_a, b_y$ are coefficients that are shared temporally and g_1, g_2 activation functions.



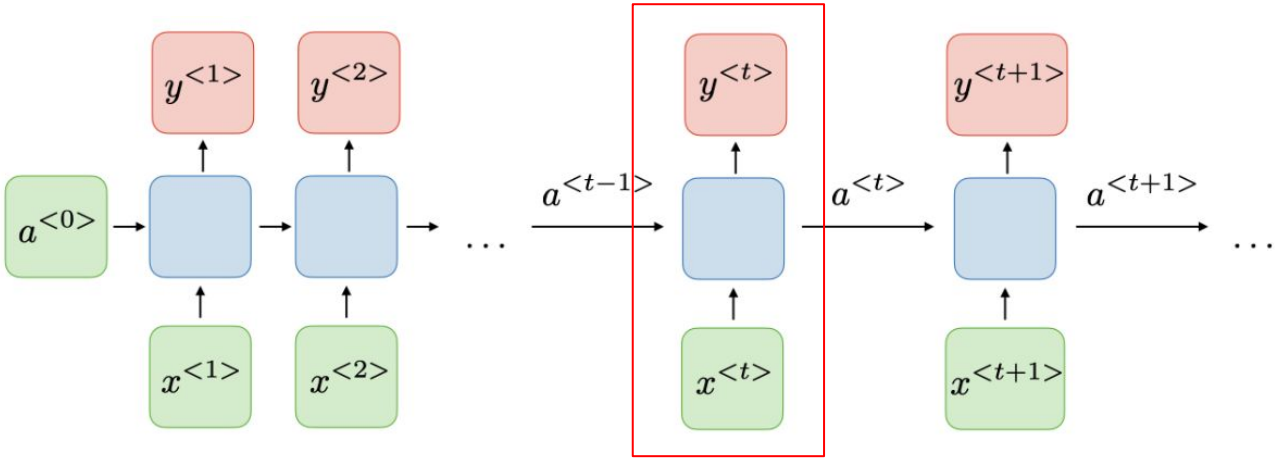
Recurrent Neural Network (RNN)

- **The pros and cons of a typical RNN architecture**
  - **Pros**
    - Possibility of processing input of any length
    - Model size not increasing with size of input
    - Computation takes into account historical information
    - Weights are shared across time
  - **Cons**
    - Computation being slow
    - Difficulty of accessing information from a long time ago
    - Cannot consider any future input for the current state

Recurrent Neural Network (RNN)

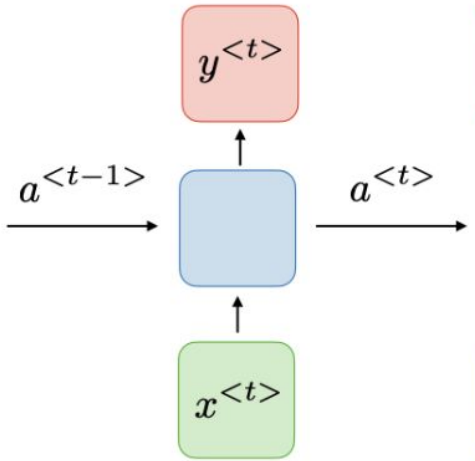
- **Architecture of a traditional RNN**
  - Recurrent neural networks, also known as RNNs, are a class of neural networks that allow previous outputs to be used as inputs while having hidden states. They are typically as follows:

Key idea: RNNs have an "**internal state**" that is updated as a sequence is processed



Recurrent Neural Network (RNN)

- **RNN hidden state update**
  - We can process a sequence of vectors x by applying a recurrence formula at every time step:



$$h_t = f_W(h_{t-1}, x_t)$$

new state

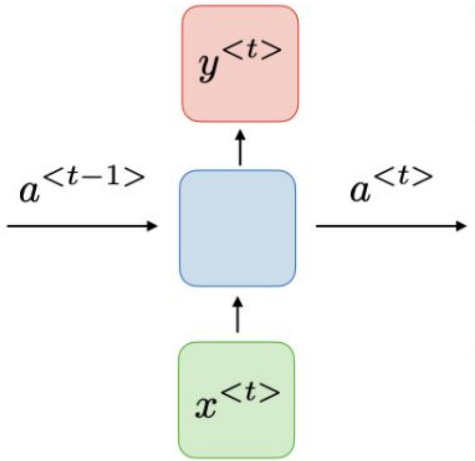
some function with parameters W

old state

input vector at some time step

Recurrent Neural Network (RNN)

- **RNN hidden state update**
  - We can process a sequence of vectors x by applying a recurrence formula at every time step:



$$y_t = f_{W_{hy}}(h_t)$$

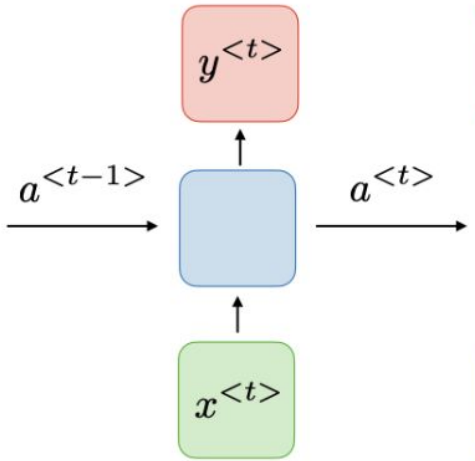
output

new state

another function
with parameters W_o

Recurrent Neural Network (RNN)

- **RNN hidden state update**
  - We can process a sequence of vectors x by applying a recurrence formula at every time step:

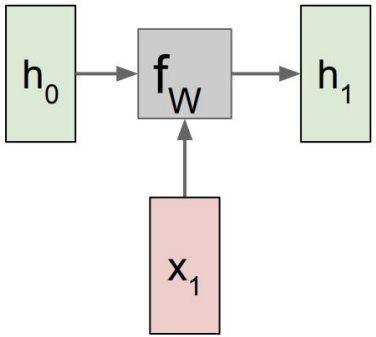


Note: the same function and the same set of parameters are used at every time step.

$$h_t = f_W(h_{t-1}, x_t)$$

Recurrent Neural Network (RNN)

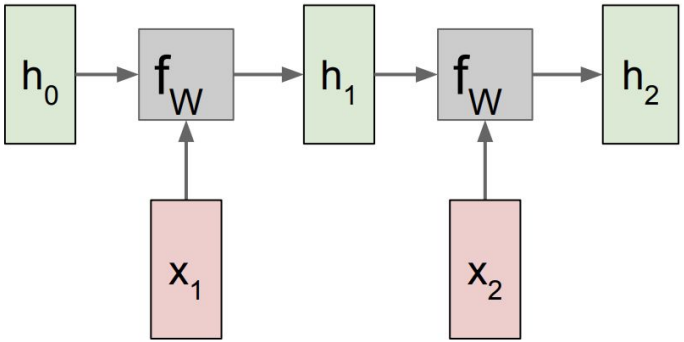
- **RNN: Computational Graph**



Li, R. (2022). Lecture 10: Self-supervised learning. [Lecture slides]. CS231n: Convolutional Neural Networks for Visual Recognition.

Recurrent Neural Network (RNN)

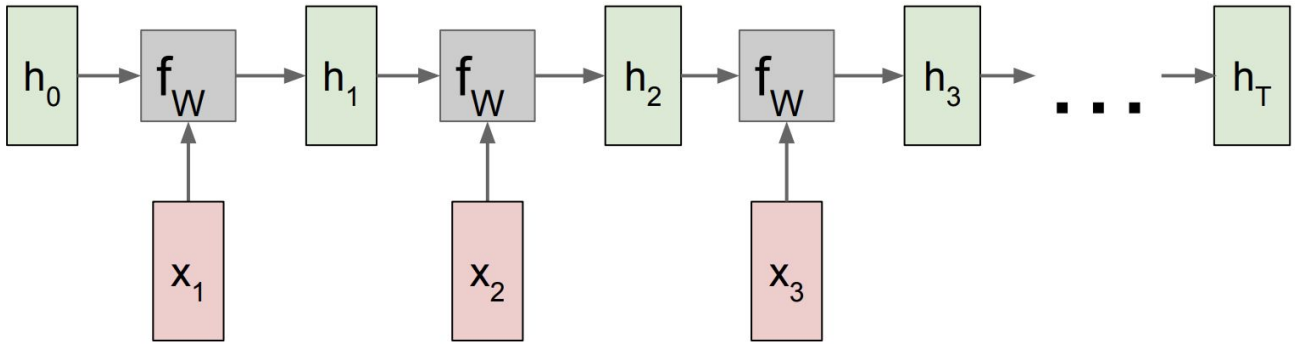
- **RNN: Computational Graph**



Li, R. (2022). Lecture 10: Self-supervised learning. [Lecture slides]. CS231n: Convolutional Neural Networks for Visual Recognition.

Recurrent Neural Network (RNN)

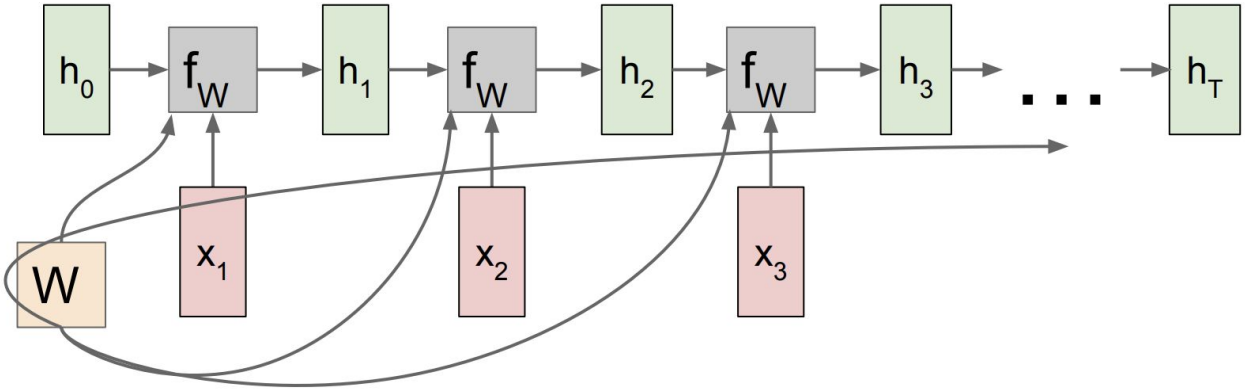
- **RNN: Computational Graph**



Li, R. (2022). Lecture 10: Self-supervised learning. [Lecture slides]. CS231n: Convolutional Neural Networks for Visual Recognition.

Recurrent Neural Network (RNN)

- **RNN: Computational Graph**

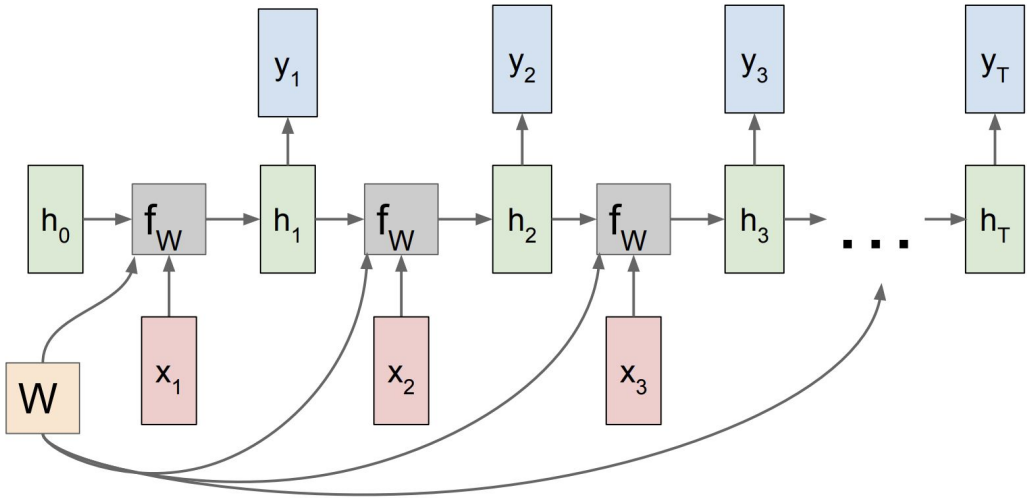


Re-use the same weight matrix at every time-step

Li, R. (2022). Lecture 10: Self-supervised learning. [Lecture slides]. CS231n: Convolutional Neural Networks for Visual Recognition.

Recurrent Neural Network (RNN)

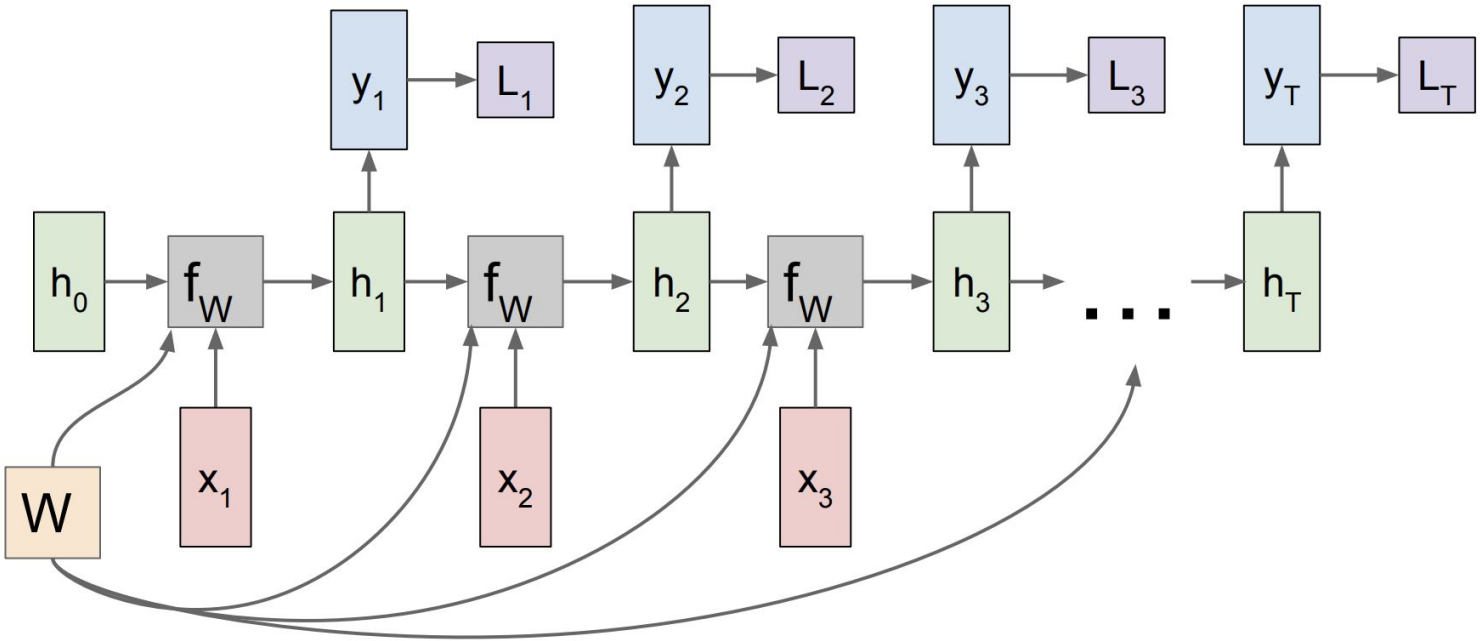
- **RNN: Computational Graph: Many to Many**



Li, R. (2022). Lecture 10: Self-supervised learning. [Lecture slides]. CS231n: Convolutional Neural Networks for Visual Recognition.

Recurrent Neural Network (RNN)

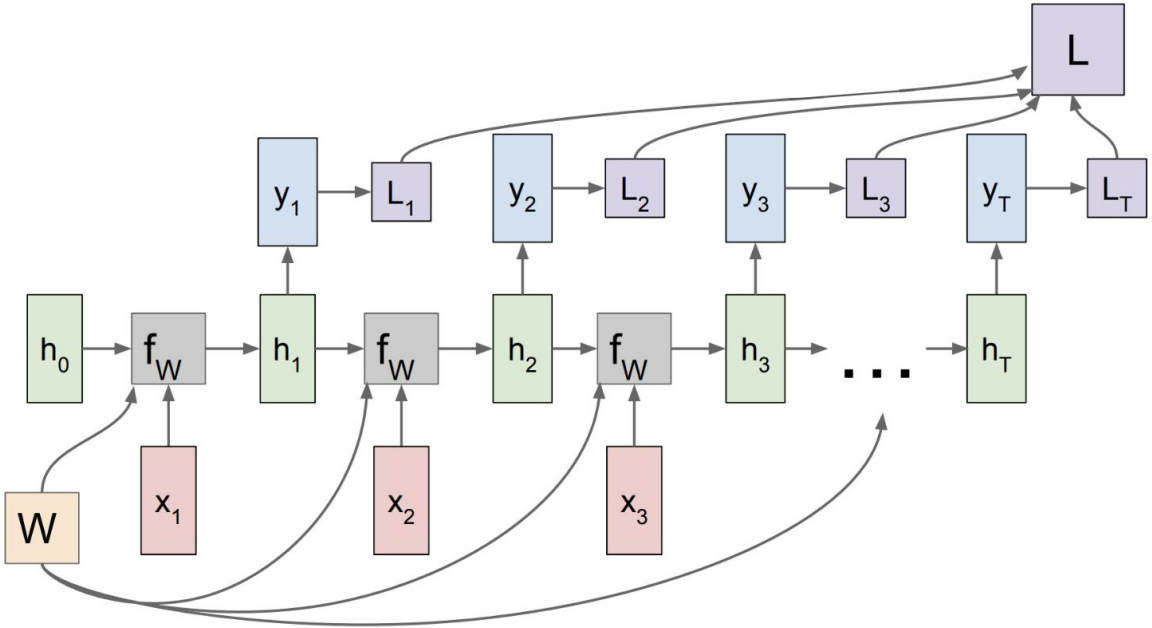
- **RNN: Computational Graph: Many to Many**



Li, R. (2022). Lecture 10: Self-supervised learning. [Lecture slides]. CS231n: Convolutional Neural Networks for Visual Recognition.

Recurrent Neural Network (RNN)

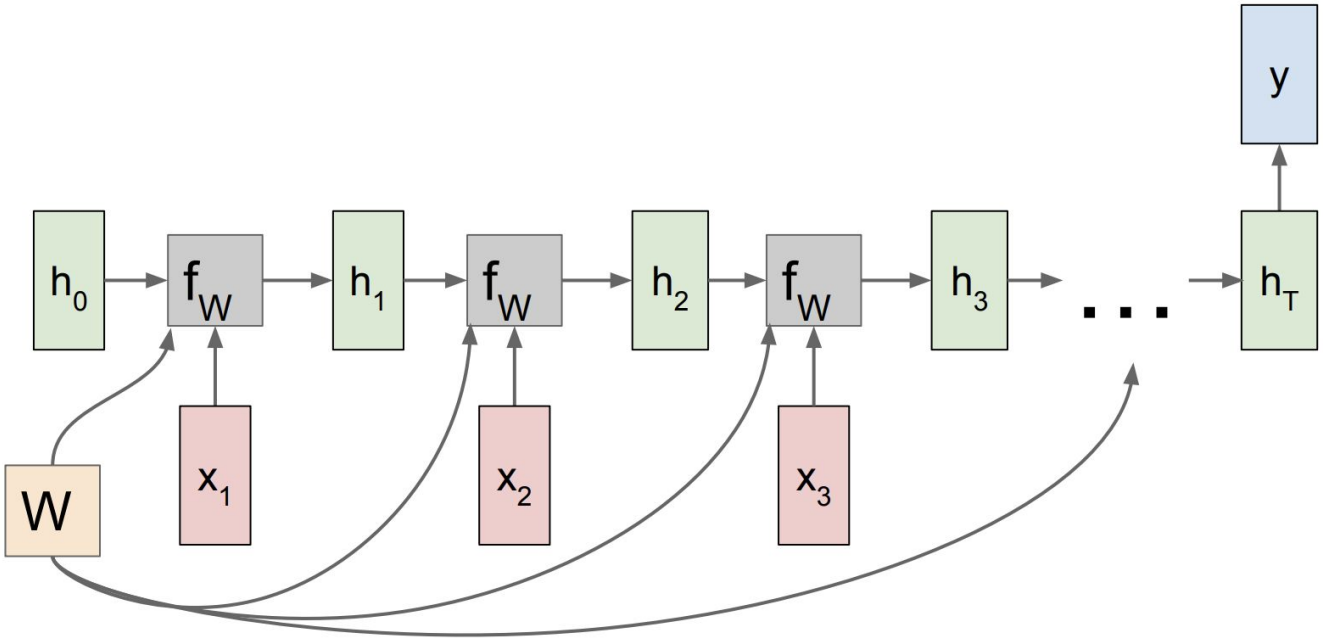
- **RNN: Computational Graph: Many to Many**



Li, R. (2022). Lecture 10: Self-supervised learning. [Lecture slides]. CS231n: Convolutional Neural Networks for Visual Recognition.

Recurrent Neural Network (RNN)

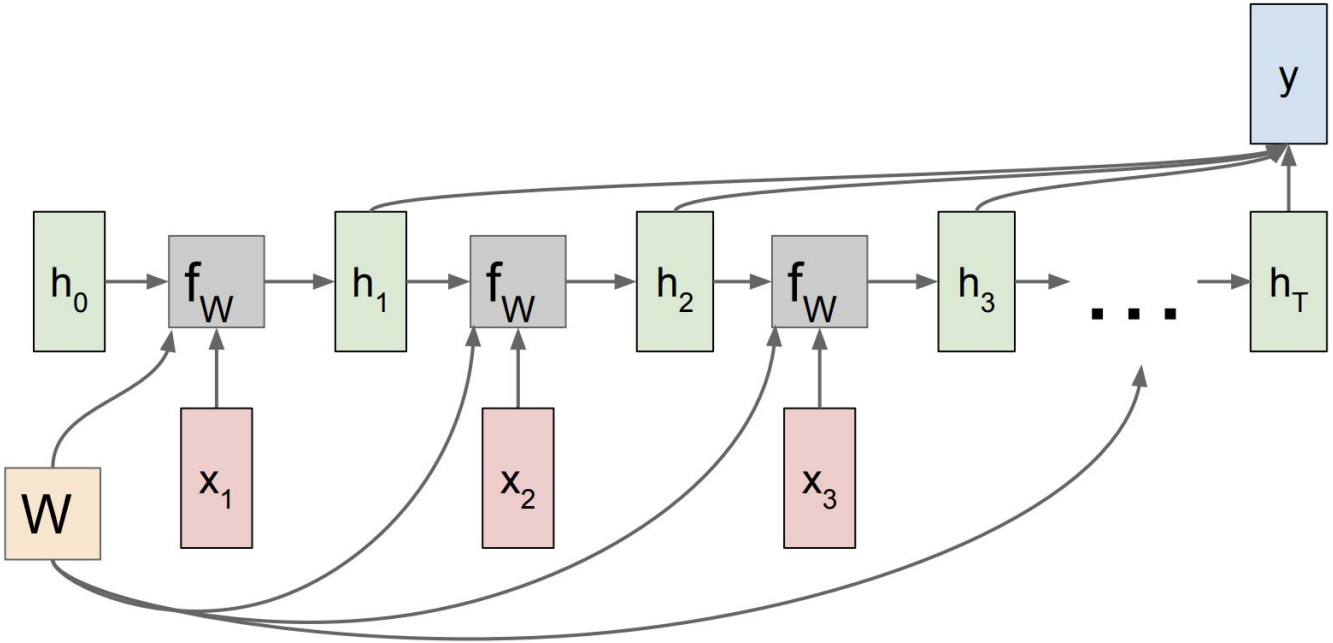
- **RNN: Computational Graph: Many to One**



Li, R. (2022). Lecture 10: Self-supervised learning. [Lecture slides]. CS231n: Convolutional Neural Networks for Visual Recognition.

Recurrent Neural Network (RNN)

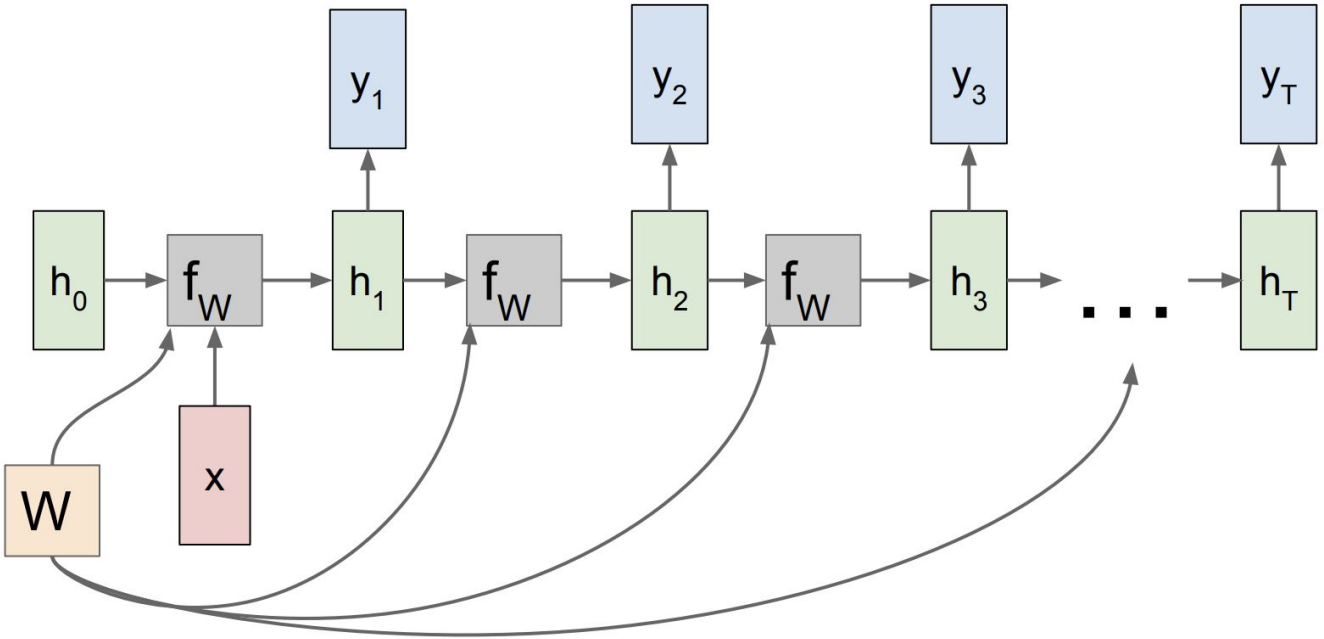
- **RNN: Computational Graph: Many to One**



Li, R. (2022). Lecture 10: Self-supervised learning. [Lecture slides]. CS231n: Convolutional Neural Networks for Visual Recognition.

Recurrent Neural Network (RNN)

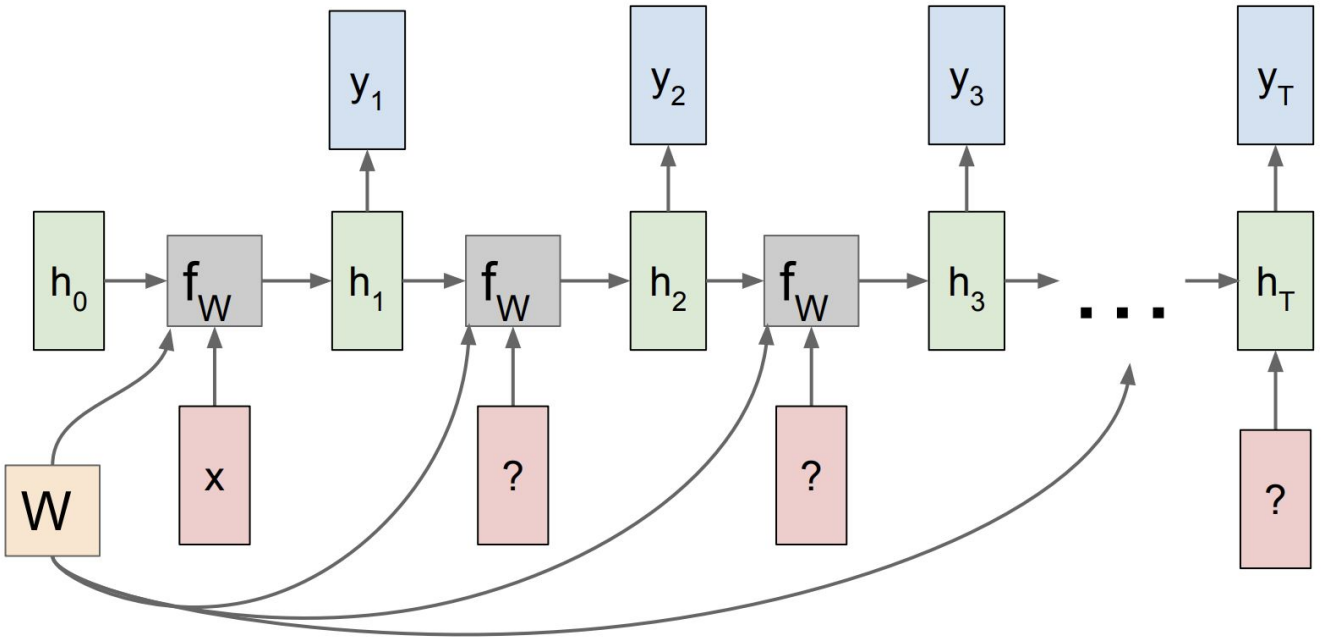
- **RNN: Computational Graph: One to Many**



Li, R. (2022). Lecture 10: Self-supervised learning. [Lecture slides]. CS231n: Convolutional Neural Networks for Visual Recognition.

Recurrent Neural Network (RNN)

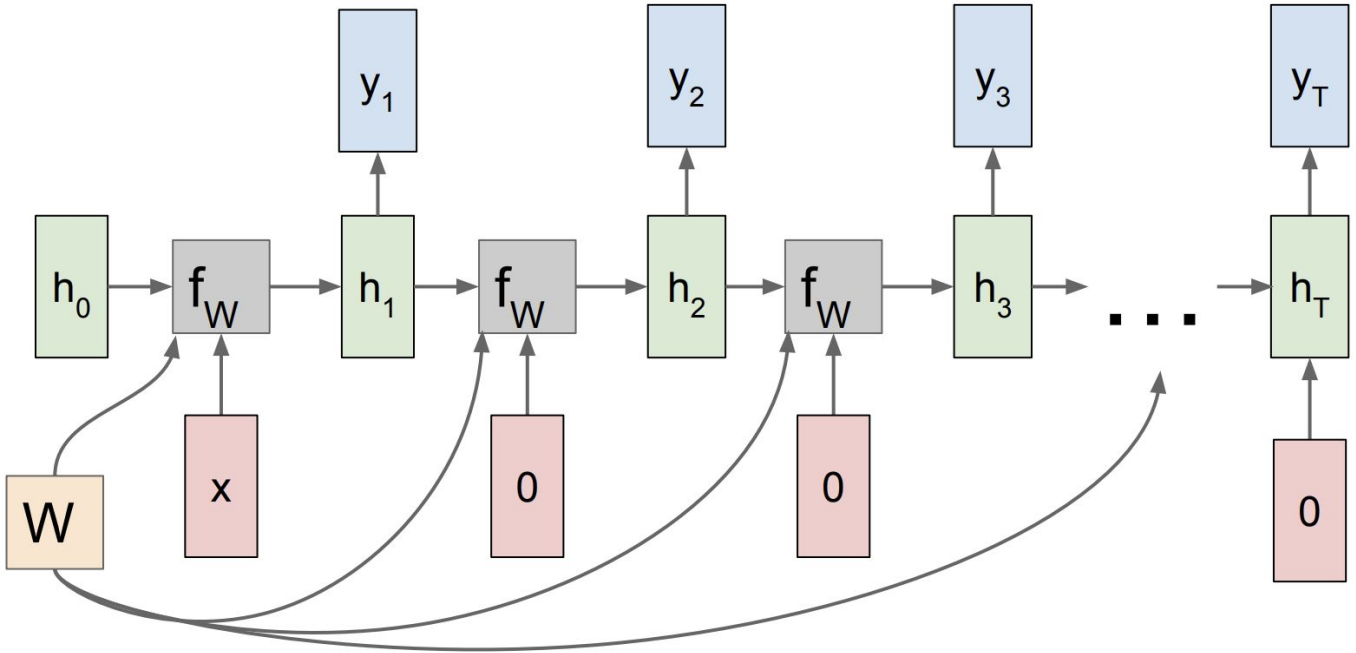
- **RNN: Computational Graph: One to Many**



Li, R. (2022). Lecture 10: Self-supervised learning. [Lecture slides]. CS231n: Convolutional Neural Networks for Visual Recognition.

Recurrent Neural Network (RNN)

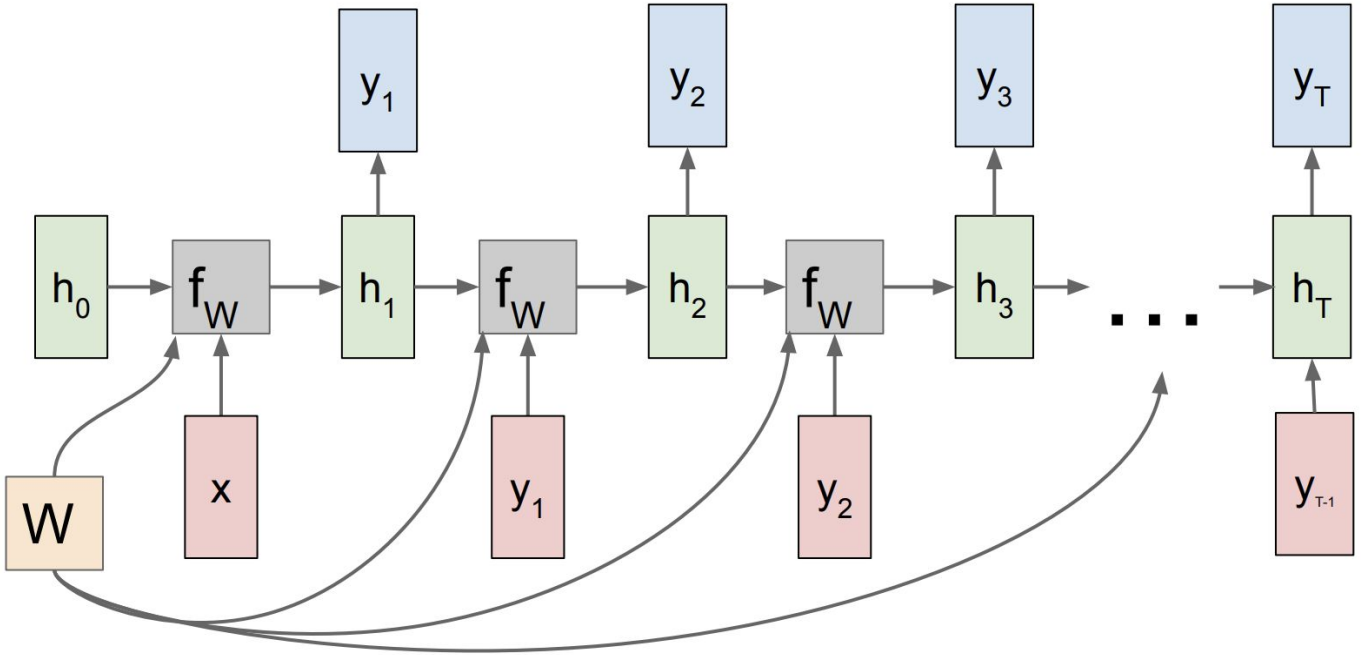
- **RNN: Computational Graph: One to Many**



Li, R. (2022). Lecture 10: Self-supervised learning. [Lecture slides]. CS231n: Convolutional Neural Networks for Visual Recognition.

Recurrent Neural Network (RNN)

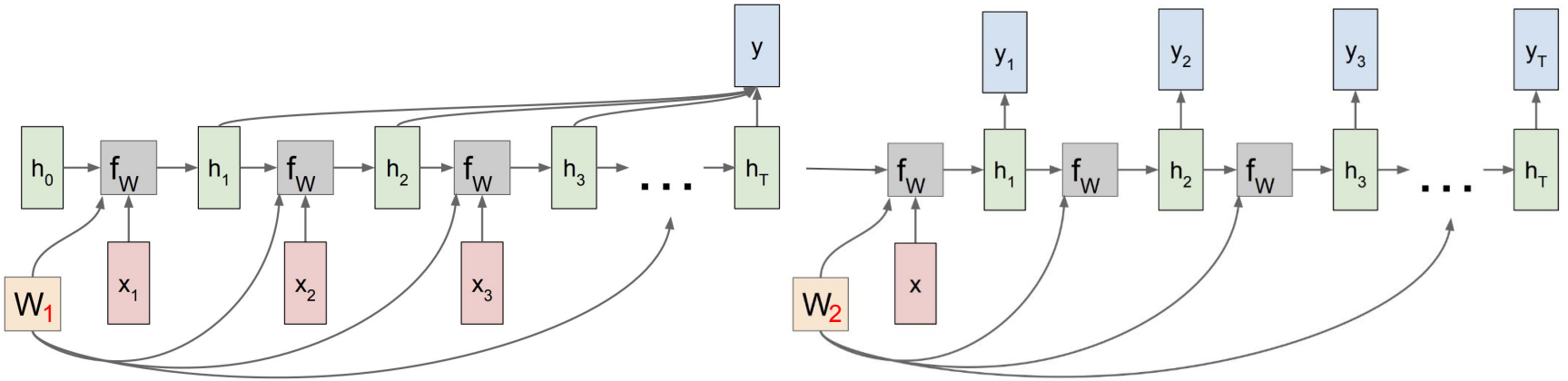
- **RNN: Computational Graph: One to Many**



Li, R. (2022). Lecture 10: Self-supervised learning. [Lecture slides]. CS231n: Convolutional Neural Networks for Visual Recognition.

Recurrent Neural Network (RNN)

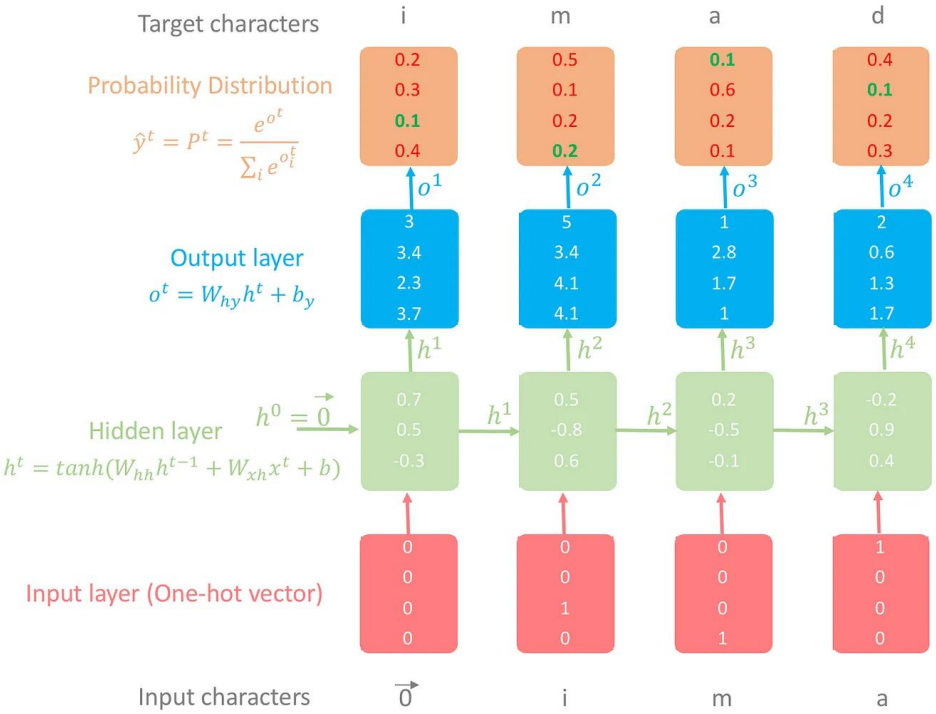
- **RNN: Computational Graph: Many-to-one + one-to-many**



Li, R. (2022). Lecture 10: Self-supervised learning. [Lecture slides]. CS231n: Convolutional Neural Networks for Visual Recognition.

Recurrent Neural Network (RNN)

- **Illustrative example of character-level language model using RNN**



Recurrent Neural Network (RNN)

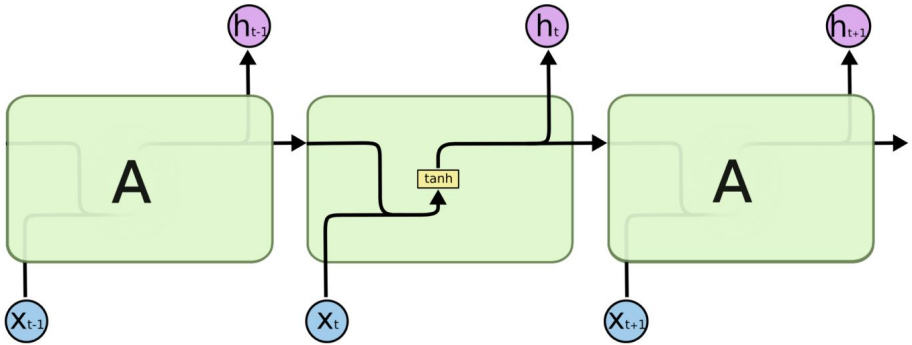
- **Character-Level Language Model in Python**

See Notebook:TODO

  - Dataset: census-derived-all-first.txt
    - 5,163 names: 4,275 male names, 1,219 female names, and 331 names that can be both female and male names.
  - Training
    - The RNN architecture we'll be using to train the character-level language model is called many to many where time steps of the input T_x= time steps of the output T_y.
  - Forward propagation
  - Backpropagation
  - Sampling
  - Fitting the model

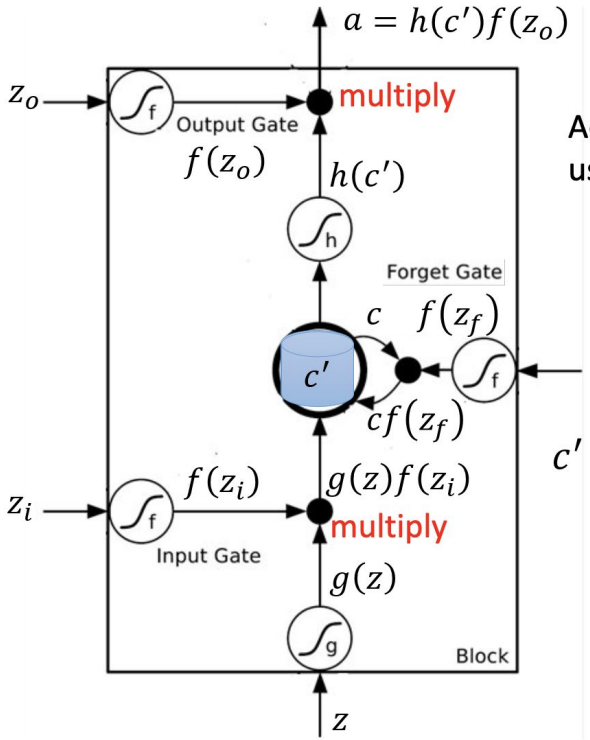
Long Short-term Memory (LSTM)

- **Long Short Term Memory networks**
  - Long Short Term Memory networks – usually just called “LSTMs” – are a special kind of RNN, capable of learning long-term dependencies. *Hochreiter & Schmidhuber (1997)*
  - All recurrent neural networks have the form of a chain of repeating modules of neural network. In standard RNNs, this repeating module will have a very simple structure, such as a single tanh layer.



The repeating module in a standard RNN contains a single layer

Long Short-term Memory (LSTM)



Activation function f is usually a sigmoid function

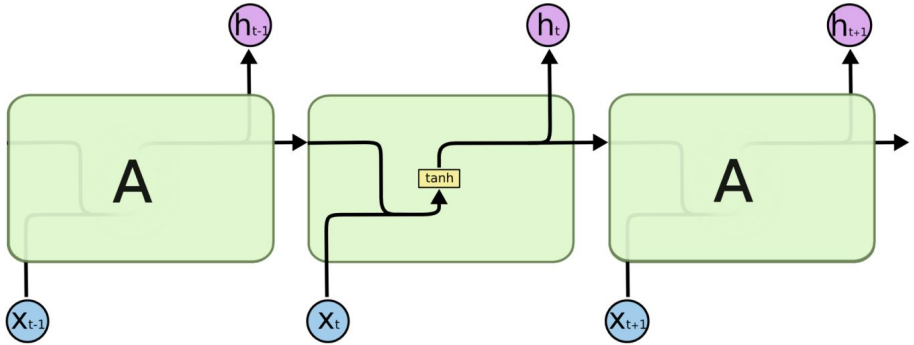
Between 0 and 1

Mimic open and close gate

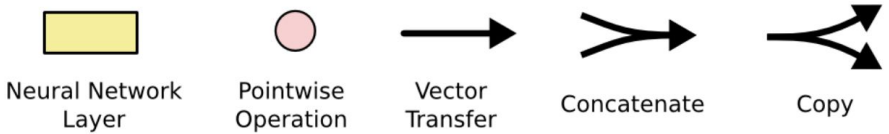
$$c' = g(z)f(z_i) + cf(z_f)$$

Long Short-term Memory (LSTM)

- **LSTM have the same chain like structure except for the repeating module.**

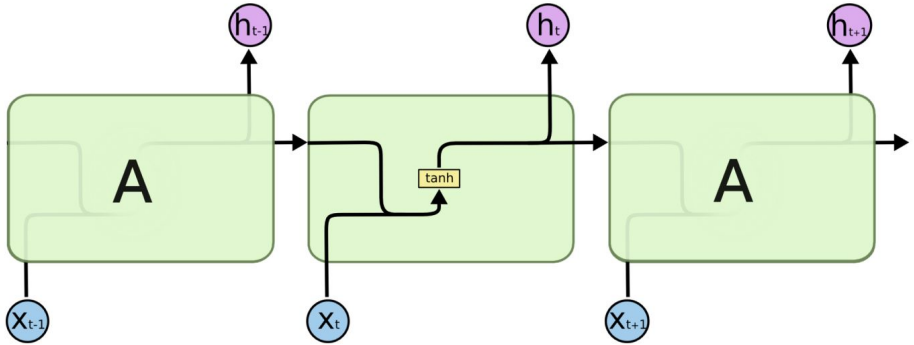


The repeating module in a standard RNN contains a single layer

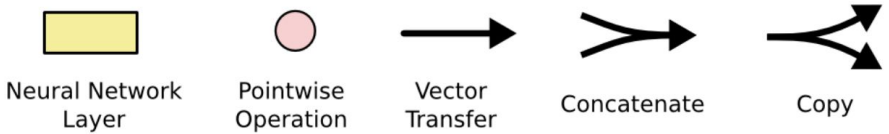


Long Short-term Memory (LSTM)

- **LSTM have the same chain like structure except for the repeating module.**

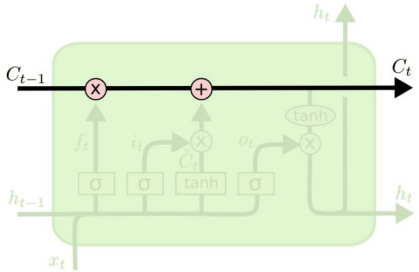


The repeating module in a standard RNN contains a single layer

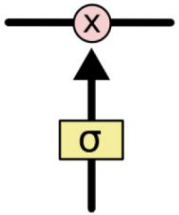


Long Short-term Memory (LSTM)

- The core idea behind LSTMs is the **cell** state.



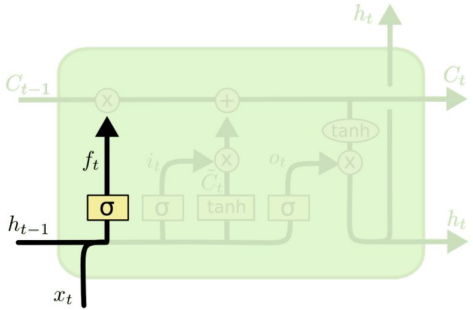
- The LSTM has the ability to **remove** or **add** information to the cell state : thanks to gates.



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

Long Short-term Memory (LSTM)

- **Step-by-Step LSTM Walk Through**
  - Step 1: Decide what information to throw away from the cell state, forget layer.

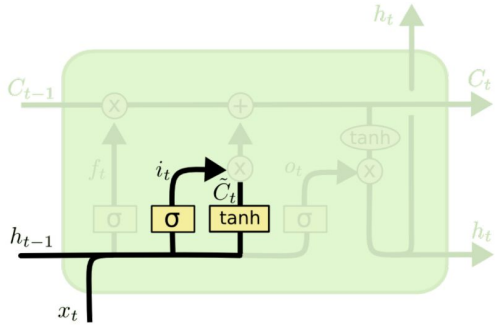


$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

  - 1 represents “completely keep this”
  - 0 represents “completely get rid of this.”

Long Short-term Memory (LSTM)

- **Step-by-Step LSTM Walk Through**
  - Step 2: Decide what new information we're going to store in the cell state



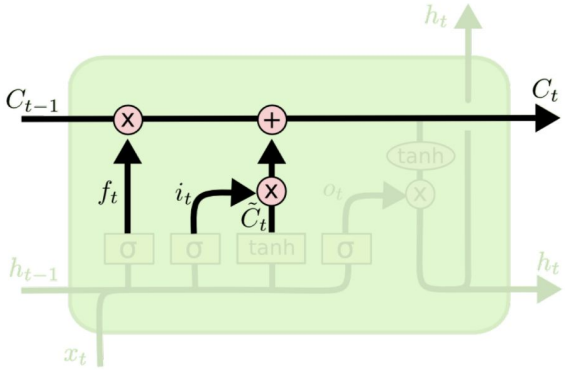
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

- **Input gate layer :** decides which values we will update
- **Tanh layer :** creates a vector of new candidate values

Example : "I grew up in France… I speak fluent French."

Long Short-term Memory (LSTM)

- **Step-by-Step LSTM Walk Through**
  - Step 3: Update the cell state

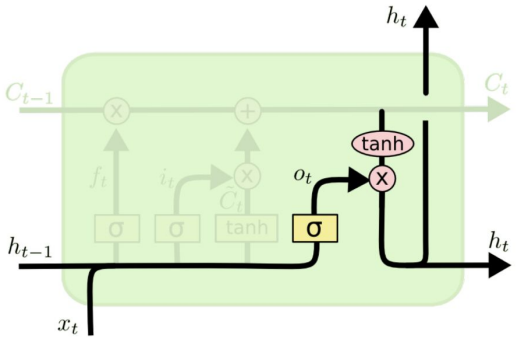


$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Example : “I grew up in France… I speak fluent French.”

Long Short-term Memory (LSTM)

- **Step-by-Step LSTM Walk Through**
  - Step 4: Decide what is the output



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

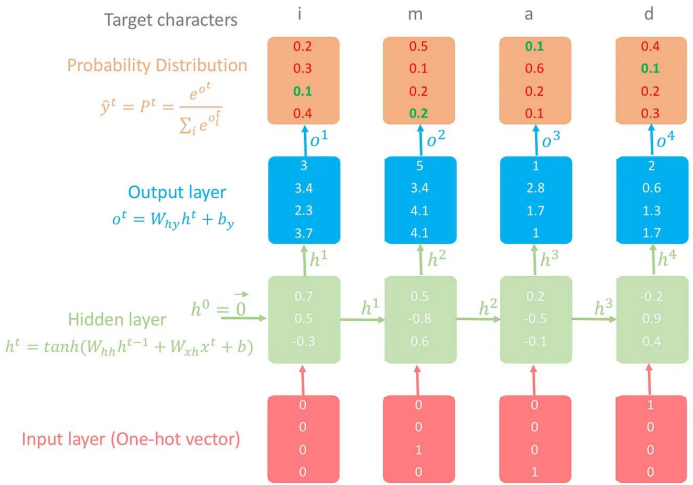
$$h_t = o_t * \tanh\left(C_t\right)$$

Example : “I grew up in France… I speak fluent French.” (Use this example to explain functions in LSTM)

Long Short-term Memory (LSTM)

- **TODO: Illustration**

  - Step 4:

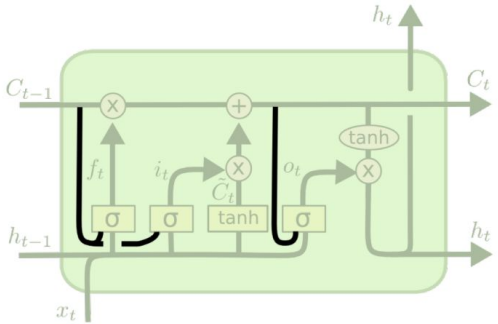


Example : “I grew up in France… I speak fluent French.”

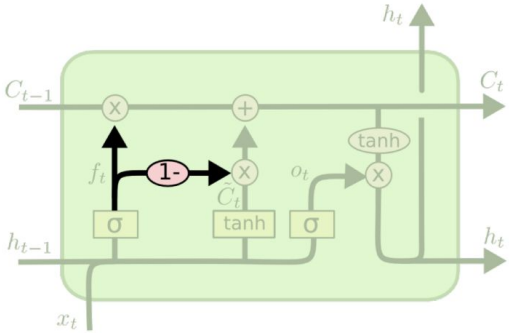
Example : “I grew up in France… I speak fluent French.” (Use this example to explain functions in LSTM)

Long Short-term Memory (LSTM)

- **Variants of LSTM (what is the input, gates, output, functions?)**



$$f_t = \sigma\left(W_f \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] \; + \; b_f\right)$$
$$i_t = \sigma\left(W_i \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] \; + \; b_i\right)$$
$$o_t = \sigma\left(W_o \cdot [\boldsymbol{C_t}, h_{t-1}, x_t] \; + \; b_o\right)$$



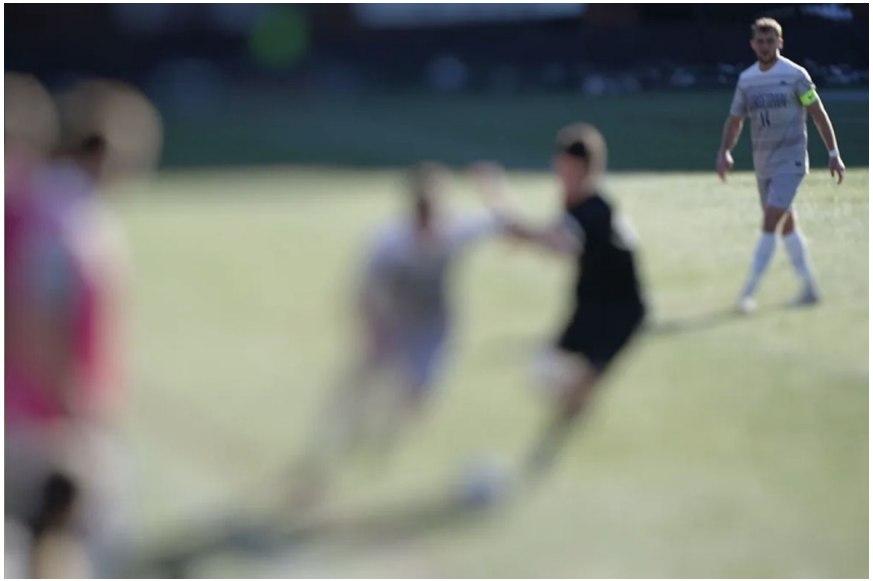
$$C_t = f_t * C_{t-1} + (\mathbf{1} - \boldsymbol{f_t}) * \tilde{C}_t$$

Attention Mechanism

- **What exactly is the attention mechanism?**
    - Look at the image below, what is the color of the soccer ball? Also, which Georgetown player, the guys in white, is wearing the captaincy band?

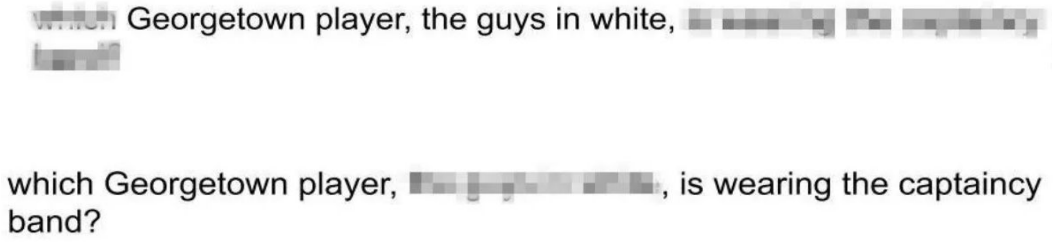
Attention Mechanism

- **What exactly is the attention mechanism?**
  - When you were trying to figure out answers to the questions above, did your mind do this weird thing where it focused on only part of the image?



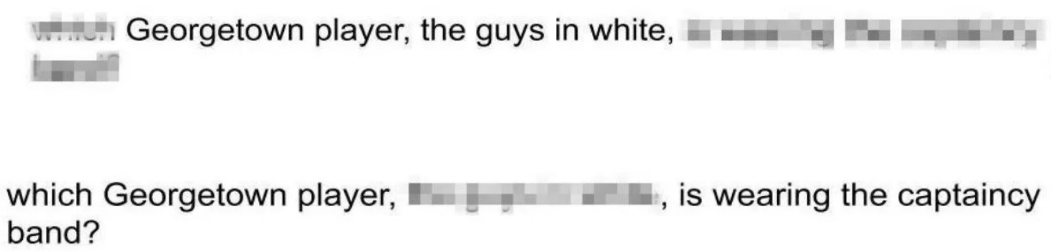
Attention Mechanism

- **What exactly is the attention mechanism?**
  - Also when you were reading the sentence above, did your mind start associating different words together, ignoring certain phrases at times to simplify the meaning?



Attention Mechanism

- **What exactly is the attention mechanism?**
  - Also when you were reading the sentence above, did your mind start associating different words together, ignoring certain phrases at times to simplify the meaning?



Q: What happened?

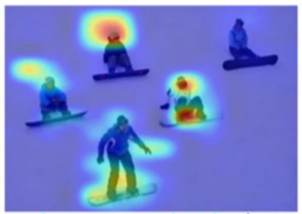
A: You were ‘focusing’ on a smaller part of the whole thing because you knew the rest of the image/sentence was not useful to you at that particular moment.

Attention Mechanism

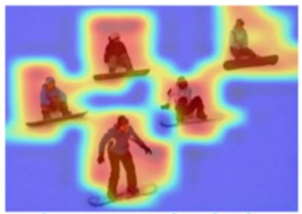
- **What exactly is the attention mechanism?**
  - In psychology, attention is the cognitive process of selectively concentrating on one or a few things while ignoring others.
  - Inspired by the workings of the human mind, researchers in Deep Learning have sought to replicate this behavior through what is called the ‘attention mechanism.’
  - In simple terms, the attention mechanism allows the model to focus on a specific part of the input while disregarding the rest.

A stop sign is on a road with a mountain in the background.


Q: how many snowboarders in formation in the snow, four is sitting? **A**: **5**


how many snowboarders in formation in the snow , four is sitting ?


how many snowboarders in formation in the snow , four is sitting ?


how many snowboarders in formation in the snow , four is sitting ?

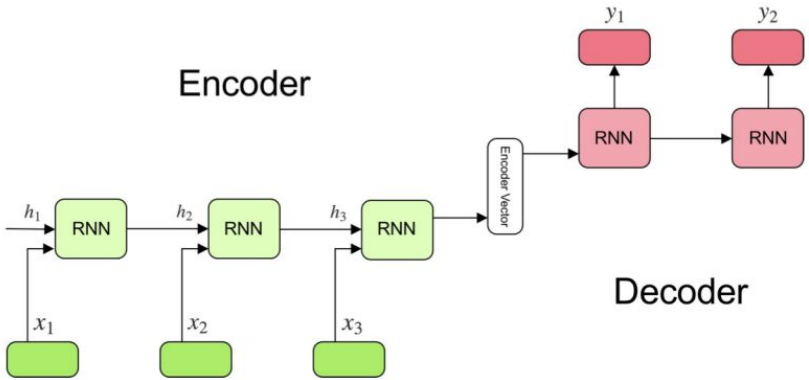
Soydaner, D. Attention mechanism in neural networks: where it comes and where it goes. *Neural Comput & Applic* **34**, 13371–13385 (2022). https://doi.org/10.1007/s00521-022-07366-3

Attention Mechanism

- **What exactly is the attention mechanism?**
  - The attention mechanism emerged as an improvement over the encoder decoderbased neural machine translation system in natural language processing (NLP). Later, this mechanism, or its variants, was used in other applications, including computer vision, speech processing, etc.
  - Before attention, neural machine translation was based on encoder decoder RNN/LSTM (Seq2Seq models). Both encoder and decoder are stacks of LSTM/RNN units. It works in the two following steps:
    - The **encoder** LSTM is used to process the entire input sentence and encode it into a context vector,
    - The **decoder** LSTM or RNN units produce the words in a sentence one after another

Attention Mechanism

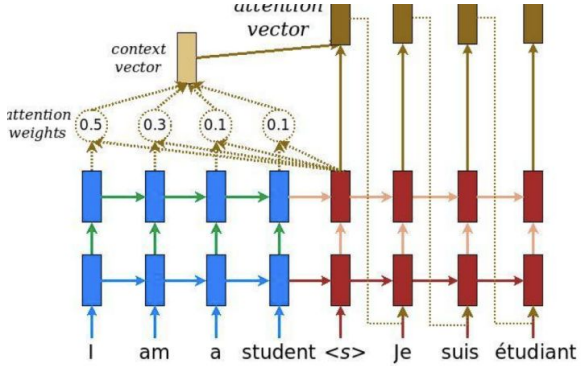
- **What exactly is the attention mechanism?**
  - It works in the two following steps:
    - The **encoder** LSTM is used to process the entire input sentence and encode it into a **context vector,**
    - The decoder LSTM or RNN units produce the words in a **sentence** one after another



Attention Mechanism

- **Why do we need attention mechanism?**
  - The main drawback of this approach : If the encoder makes a bad summary, the translation will also be bad !
  - Long-range dependency problem of RNN/LSTMs : the encoder creates a bad summary when it tries to understand longer sentences.
  - So is there any way we can keep all the relevant information in the input sentences intact while creating the context vector?

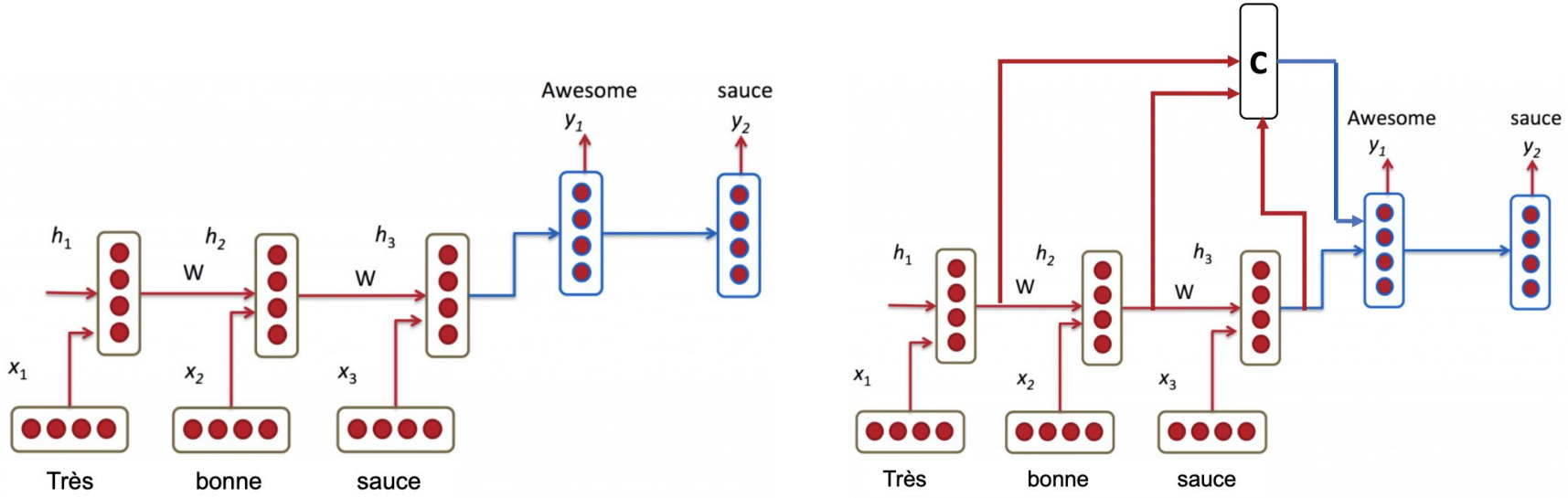
Attention mechanism !
Attention Weights
Explain that it's important



Attention mechanism applied to encoder-decoder

Attention Mechanism

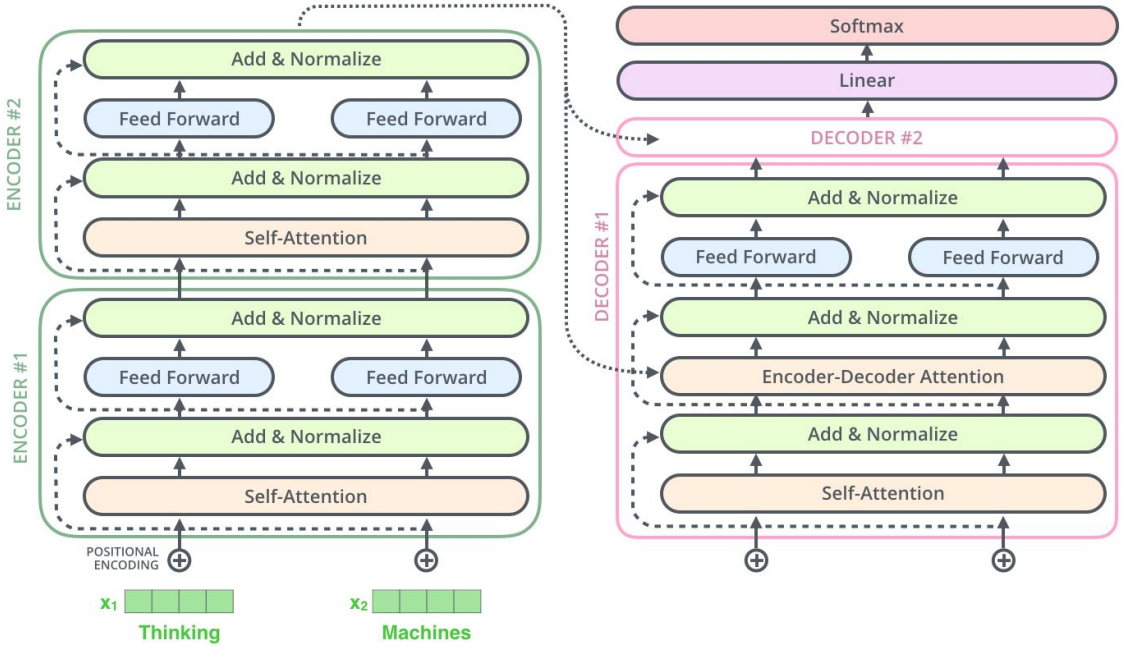
- **How the attention mechanism work ?**



Seq2seq model without and with attention mechanism

Attention Mechanism

- **How the attention mechanism work? Example: input → encode → decode → output**



A small attention-based Transformer network

Recap: Supervised Learning vs. Unsupervised Learning

The category of a ML algorithm

- **Supervised Learning**

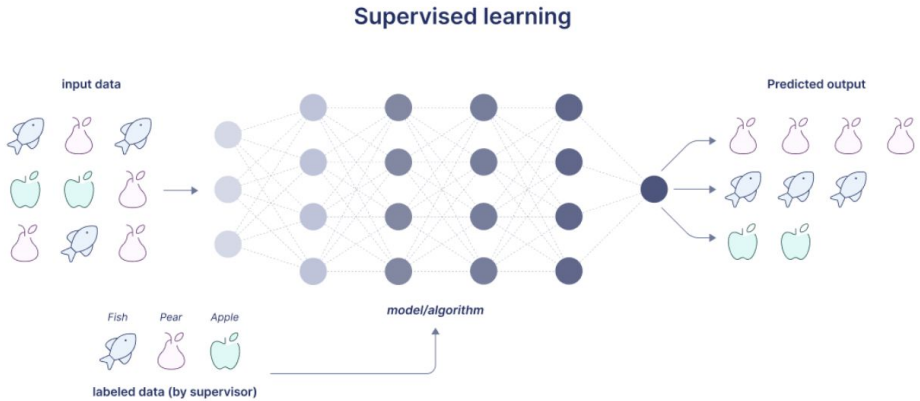
  In supervised learning, an algorithm learns from labeled training data to make predictions or decisions without explicit programming.
  - Regression
  - Classification
- **Unsupervised Learning**

  Unlike supervised learning, where labeled data guides the model, unsupervised learning uses unlabeled data. The goal is to explore the data and uncover hidden patterns, trends, and relationships.
  - Clustering
  - Association (We have learned on week 3's lecture)
  - Dimensionality Reduction
- **Reinforcement Learning (RL)**

  RL focuses on how an intelligent agent should take actions in a dynamic environment to maximize cumulative reward. Unlike supervised learning, it doesn't rely on labeled input/output pairs or explicit corrections for sub-optimal actions. Instead, it balances exploration (uncharted territory) and exploitation (current knowledge) to achieve long-term rewards, even with incomplete or delayed feedback.

Recap: Supervised Learning vs. Unsupervised Learning

- **Supervised Learning vs. Unsupervised Learning**

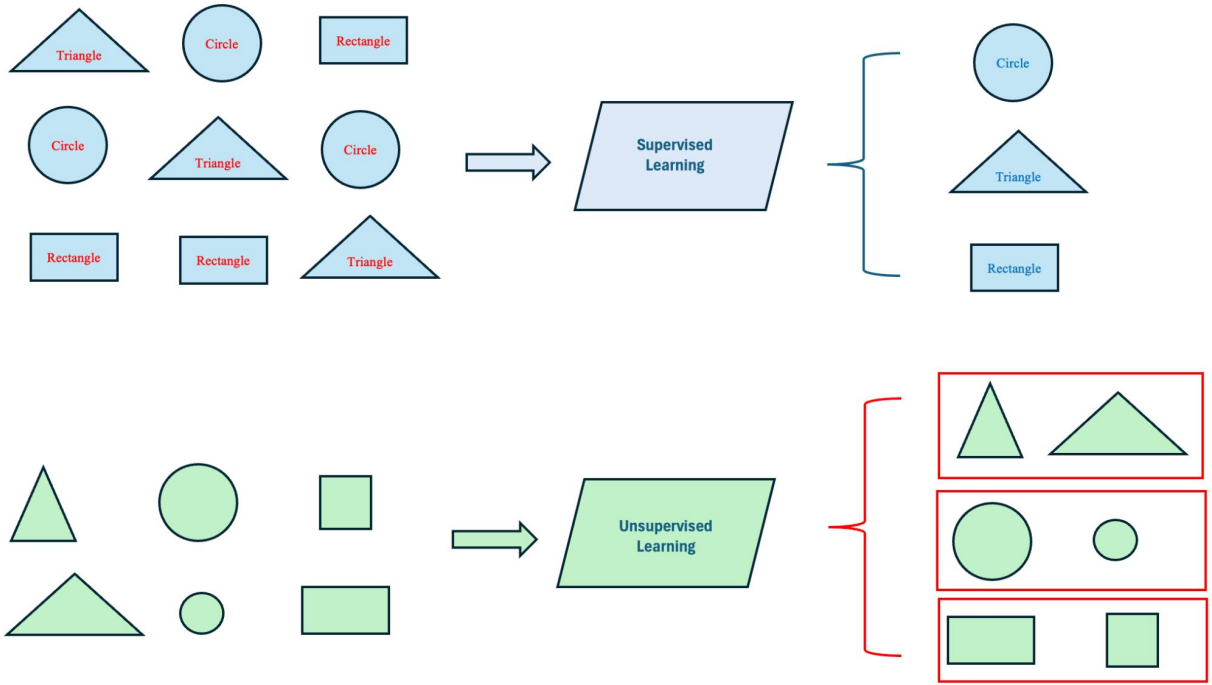


Recap: Supervised Learning vs. Unsupervised Learning

	Supervised learning	Unsupervised learning
Data	Uses labeled data with known answers or outputs	Processes unlabeled data. There are no predefined answers (i.e., no desired output is given)
Goals	To make a prediction (e.g., the future value of a house) or a classification (e.g., correctly identify spam emails)	To explore and discover patterns, structures, or relationships in large volumes of data
General tasks	Classification, regression	Clustering, dimensionality reduction, association learning
Can be applied to	Sentiment analysis, stock market prediction, house price estimation	Medical image analysis, product recommendations, fraud detection
Human supervision	Requires human intervention to provide labeled data for training	Does not require human intervention/explicit guidance
Accuracy	Tends to have higher accuracy because it learns from labeled examples with known answers	Accuracy evaluation is harder and more subjective because there are no correct answers

Recap: Supervised Learning vs. Unsupervised Learning

- **Data & Goals**

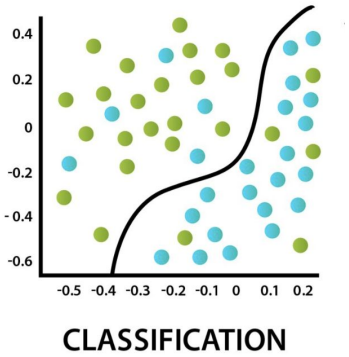


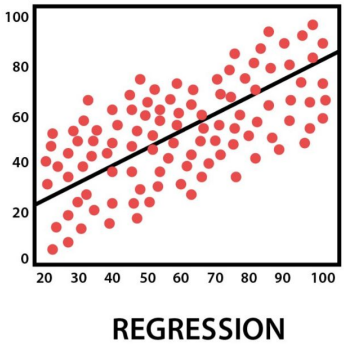
Recap: Supervised Learning vs. Unsupervised Learning

- **Supervised Learning vs. Unsupervised Learning**
  - Methods
    - Supervised Learning
      - Classification
      - Regression
    - **Unsupervised Learning**
      - Clustering
      - Dimensionality Reduction
      - Association Learning

Recap: Supervised Learning vs. Unsupervised Learning

- **Supervised Learning vs. Unsupervised Learning: Methods**
  - Supervised Learning
    - **Classification**
      - Classification is used when our goal is to categorize data into predefined classes or labels. It's like sorting things into different buckets based on their characteristics.
      - Algorithms
        - Logistic Regression
        - Decision Trees
        - Random Forests
        - Support Vector Machines (SVM)
    - **Regression**
      - Regression models predict continuous values (real numbers). They're ideal for problems where the outcome is not discrete but lies on a continuous scale.
      - Algorithms
        - Linear Regression
        - Polynomial Regression
        - Support Vector Regression (SVR)



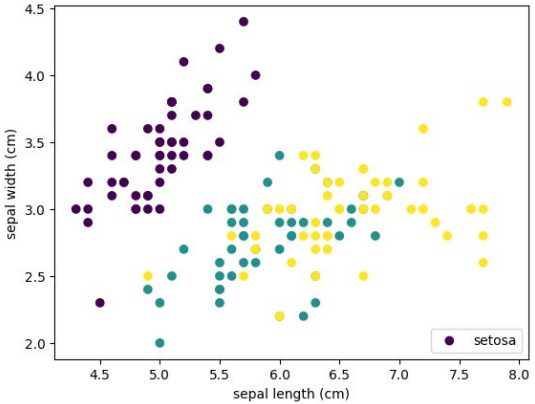


Recap: Supervised Learning vs. Unsupervised Learning

- **Supervised Learning vs. Unsupervised Learning: Methods**
  - **Unsupervised Learning**
    - **Clustering**
      - Clustering aims to group similar data points together based on their features. It helps discover inherent structures within the data.
      - Algorithms
        - K-Means
        - Hierarchical Clustering
        - DBSCAN
        - Gaussian Mixture Models (GMM)
    - **Dimensionality Reduction**
      - Reducing the number of features while preserving essential information. It helps visualize and analyze high-dimensional data.
      - Algorithms
        - **PCA** (Principal Component Analysis)
        - Autoencoders
    - **Association Learning**
      - Identifying interesting relationships or patterns among items in transactional data (e.g., market basket analysis).
      - Examples: Market Basket Analysis, Recommendation Systems

Recap: Supervised Learning vs. Unsupervised Learning

- **Example in Python**
  - Iris flower data set
    - https://en.wikipedia.org/wiki/Iris_flower_data_set
    - **Description**:
      - The Iris dataset contains measurements of sepal and petal lengths for three species of iris flowers: **Setosa**, **Versicolor**, and **Virginica**.
      - Each sample has four features: Sepal Length, Sepal Width, Petal Length, and Petal Width.
      - It's a multi-class classification dataset with a total of 150 samples (50 samples per class).
    - **Attributes**:
      - Sepal Length
      - Sepal Width
      - Petal Length
      - Petal Width
    - **Species**:
      - Setosa
      - Versicolor
      - Virginica



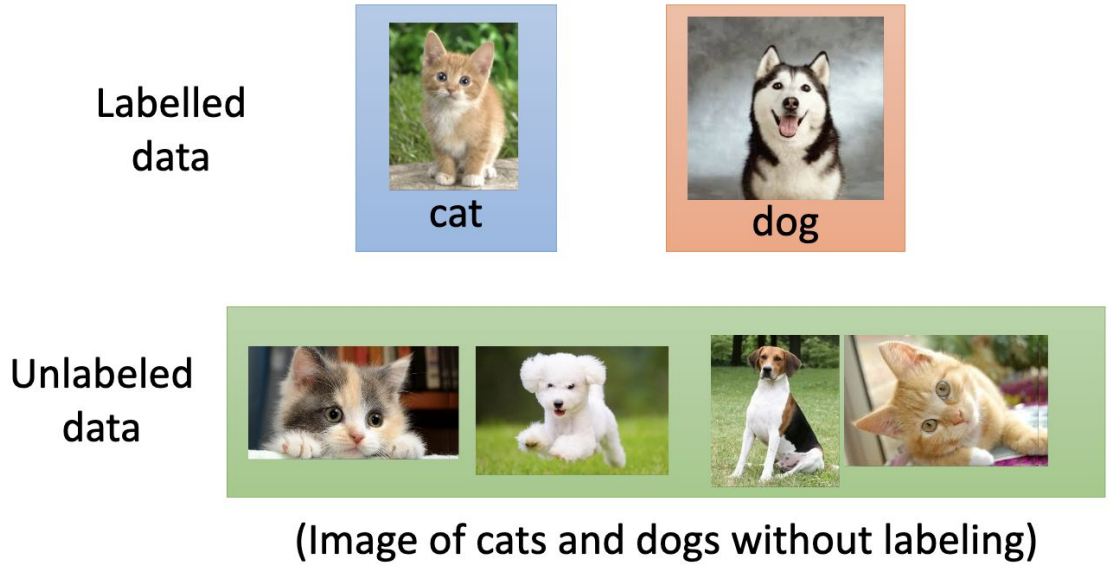

Iris virginica


Iris versicolor


Iris setosa

Semi-supervised Learning

- **Introduction**



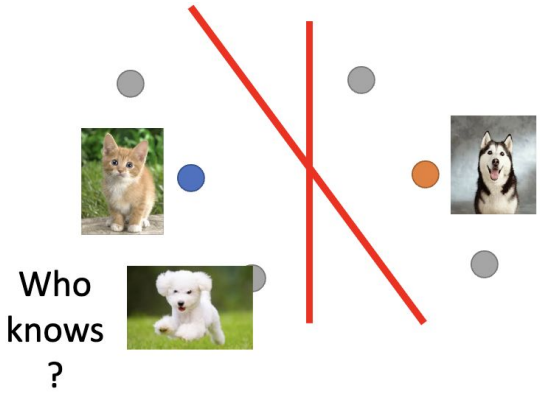
Semi-supervised Learning

- **Introduction**

  - Supervised learning: $\{(x^r, \hat{y}^r)\}_{r=1}^{R}$
    - E.g. x^r: image, $\hat{y}^r$: class labels
  - Semi-supervised learning: $\{(x^r, \hat{y}^r)\}_{r=1}^{R}$, $\{x^u\}_{u=R}^{R+U}$
    - A set of unlabeled data, usually U >> R
    - *Transductive learning*: unlabeled data is the testing data
    - *Inductive learning*: unlabeled data is not the testing data
  - Why semi-supervised learning?
    - Collecting data is easy, but collecting "labelled" data is expensive
    - We do semi-supervised learning in our lives

Semi-supervised Learning

- **Why semi-supervised learning helps?**



The distribution of the unlabeled data tell us **something**.

Note: Usually with some assumptions

Semi-supervised Learning

- **Why semi-supervised learning helps?**
  - Semi-supervised Learning for **Generative** Model
  - Low-density Separation Assumption
  - Smoothness Assumption
  - Better Representation

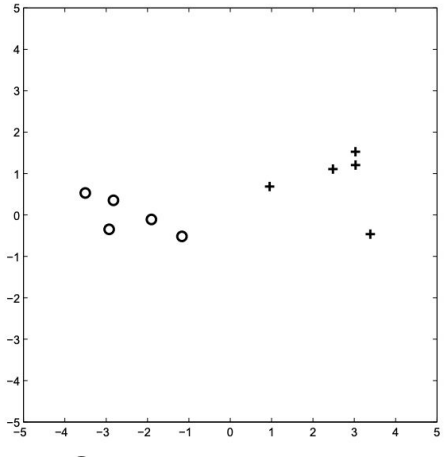
Semi-supervised Learning

- **Semi-supervised Learning for Generative Model**
  - Generative models learn the underlying data distribution and can generate new data points similar to the training data.
  - In a semi-supervised setting, these models benefit from combining labeled and unlabeled data.
  - **Unlabeled data provides structure:** Unlabeled examples help generative models learn a better data distribution, which improves overall performance.
  - **Labeled data improves accuracy:** A small amount of labeled data is used to guide the generative process, ensuring that the model generates meaningful outputs related to the task at hand.
  - **Regularization effect:** The use of both labeled and unlabeled data prevents overfitting on the small labeled dataset by learning from more diverse unlabeled data points.

Semi-supervised Learning

- **A simple example of generative models**

Labeled data (Xi , Yi):



Assuming each class has a Gaussian distribution, what is the decision boundary?

Semi-supervised Learning

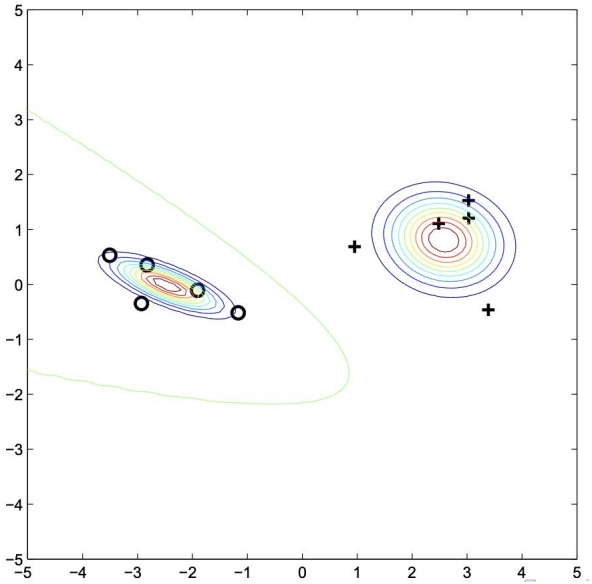
- **A simple example of generative models**
  - Model parameters: $\theta = \{w_1, w_2, \mu_1, \mu_2, \Sigma_1, \Sigma_2\}$
  - Mixture of Gaussian distributions (GMM):

$$\begin{aligned} p(x, y|\theta) &= p(y|\theta)p(x|y, \theta) \\ &= w_y \mathcal{N}(x; \mu_y, \Sigma_y) \end{aligned}$$

  - Classification: $p(y|x, \theta) = \frac{p(x,y|\theta)}{\sum_{y'} p(x,y'|\theta)}$

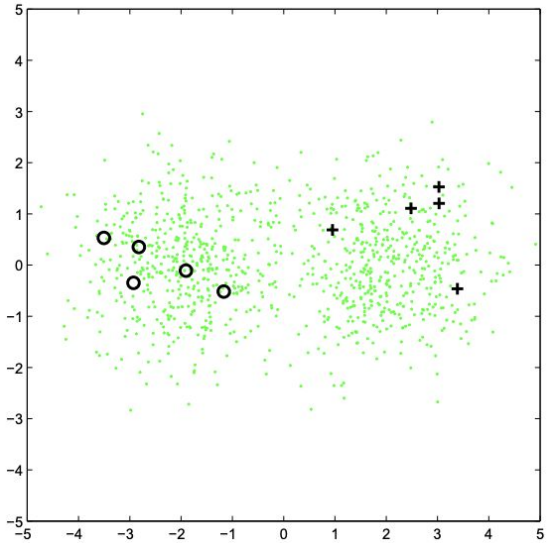
Semi-supervised Learning

- **A simple example of generative models**
  - The most likely model, and its decision boundary:



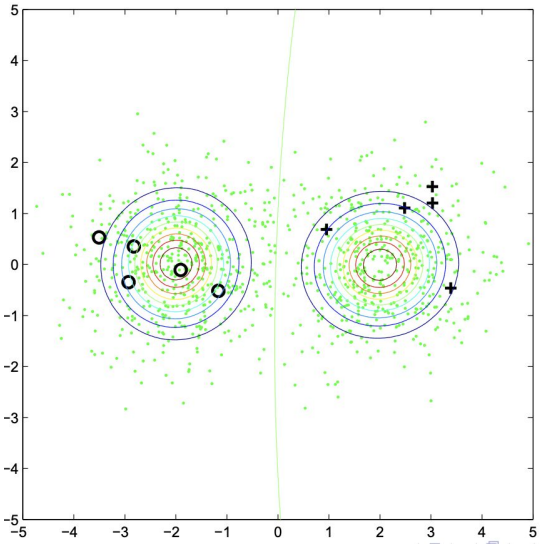
Semi-supervised Learning

- **A simple example of generative models**
  - Adding unlabeled data:



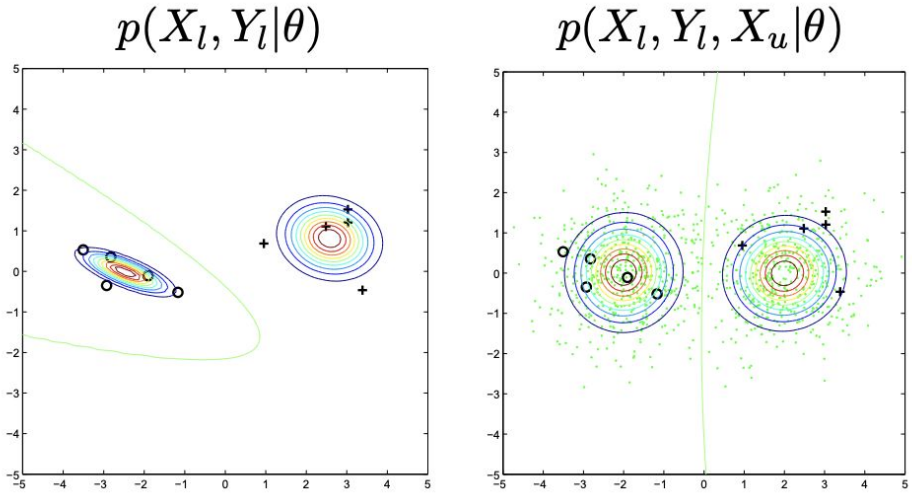
Semi-supervised Learning

- **A simple example of generative models**
  - With unlabeled data, the most likely model and its decision boundary:



Semi-supervised Learning

- **A simple example of generative models**
  - They are different because they maximize different quantities.

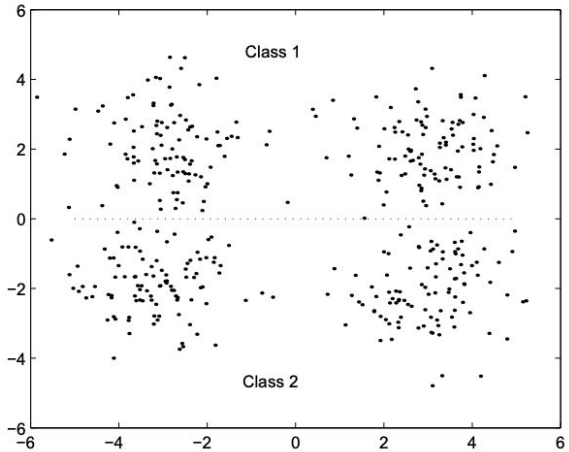


Semi-supervised Learning

- **Generative model for semi-supervised learning**
  - **Assumption**
    - The full generative model p(X, Y |θ).
- Generative model for semi-supervised learning:
  - quantity of interest: $p(X_l, Y_l, X_u|\theta) = \sum_{Y_u} p(X_l, Y_l, X_u, Y_u|\theta)$
  - Find the maximum likelihood estimate (MLE) of θ, the maximum a posteriori (MAP) estimate, or be Bayesian

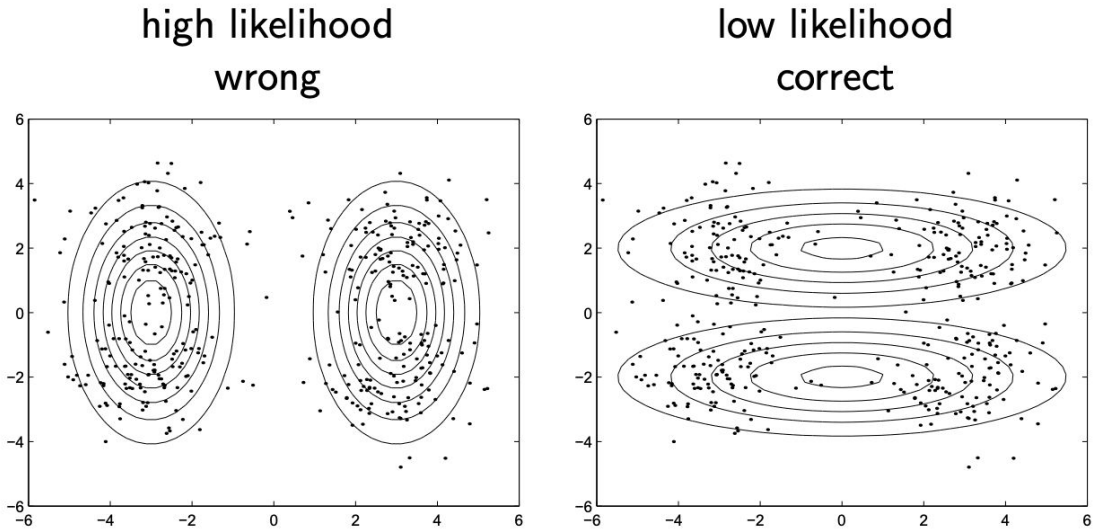
Semi-supervised Learning

- **Advantages and disadvantages of generative models**
  - **Advantages**
    - Clear, well-studied probabilistic framework
    - Can be extremely effective, if the model is close to correct
  - **Disadvantages**
    - Often difficult to verify the correctness of the model
    - Model identifiability
    - EM local optima
    - Unlabeled data may hurt if generative model is wrong
      - For example, classifying text by topic vs. by genre.



Semi-supervised Learning

- **Unlabeled data may hurt semi-supervised learning**
  - If the generative model is wrong:



Example: RNN, LSTM in python

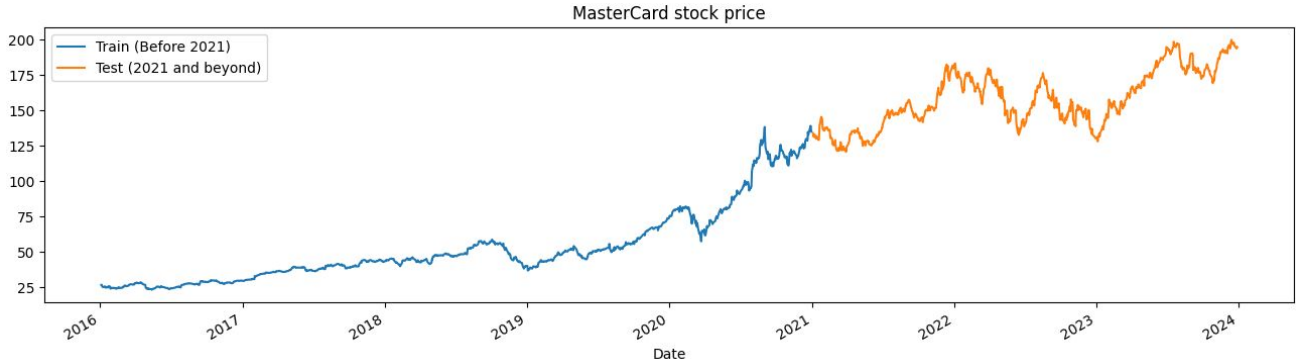
- **Example: RNN for Sentiment Analysis on IMDB Dataset (movie review)**

Layer (type)	Output Shape	Param #
embedding_1 (Embedding)	(32, 500, 32)	320,000
simple_rnn_1 (SimpleRNN)	(32, 32)	2,080
dense_1 (Dense)	(32, 1)	33

	review	label
0	? this film was just brilliant casting locatio...	1
1	? big hair big boobs bad music and a giant saf...	0
2	? this has to be one of the worst films of the...	0
3	? the ? ? at storytelling the traditional sort...	1
4	? worst mistake of my life br br i picked this...	0

Example: RNN, LSTM in python

- **Example: MasterCard Stock Price Prediction by using LSTM**



Layer (type)	Output Shape	Param #
lstm_1 (LSTM)	(None, 125)	63,500
dense_9 (Dense)	(None, 1)	126

Example: RNN, LSTM in python

- **Example: MasterCard Stock Price Prediction by using LSTM**