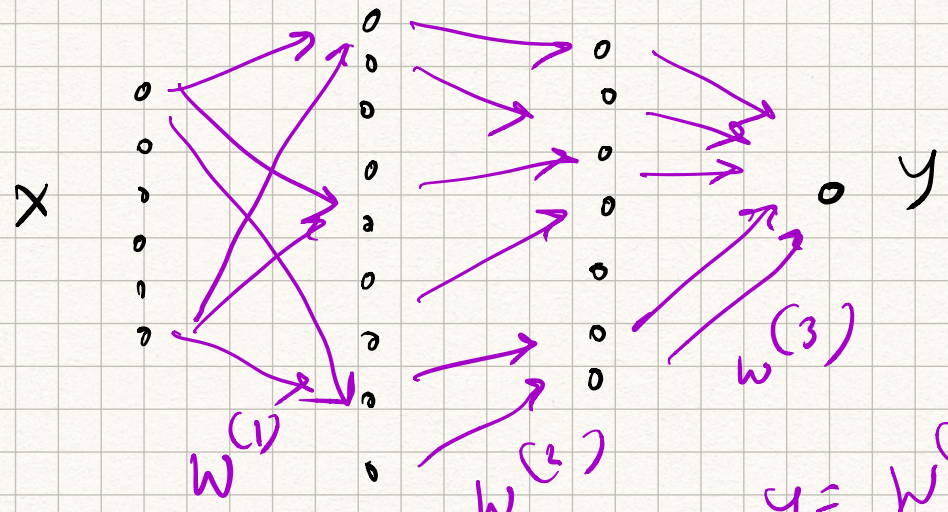
ECE6143 - G4
Intro to ML

- Recap: Neural networks
- Convolutional Neural Nets (CNNs)
- Recurrent Neural Nets (RNNs)



"Fully connected"

$$y = W^{(3)} \sigma\left(W^{(2)} \sigma\left(W^{(1)} x\right)\right)$$

Problems.

- Computational cost.

e.x. Image processing
Input : RGB images of 400×400
4 layers of neurons : 1000 neurons
output : 10 classes.

edges (weights)

$\approx 3 \times 400 \times 400 \times 1000 + 1000 \times 10$

$\approx$ 500 million weights.

Generally: $O(L \cdot d^2)$

- Sample complexity.
  - Requires a very large training dataset.
- Loss of context.
  - all input is flattened into vectors.
  - e.g. images lose spatial information
  - e.g. NLP / speech have long-range temporal dependencies.
- Confounding features.
  - Occlusion
  - Background clutter
  - Illumination
  - Other perturbations

Convolutional Neural Net. (CNNs) // image recognition

- Locality: the model should look at

local regions of your input image.

- Shift-invariance: the model should make the same prediction to the same object no matter where it appears.

Convolutional layer

Convolution

$x[t]$ — input

$w[t]$ — filter / kernel

$$y[t] = (x * w)[t]$$

$$= \sum_{\tau} x[t - \tau]\, w(\tau)$$

"Translate-and-scale".

Not hard to prove:

$$y[t] = \sum_{\tau} x[\tau]\, w[t - \tau].$$

"Flip-and-dot".

- Linear, associative, commutative

- Shift-invariance //

2D-images $x(s,t)$ $w(s,t)$.

$$(x * w)[s,t] = \sum_{\sigma} \sum_{\tau} x[s-\sigma, t-\tau]\, w[\sigma,\tau]$$

Convolution layers

same as convolution, except no flipping.

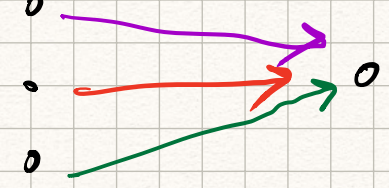
$$z[s,t] = (x * w)[s,t] = \sum_{\sigma=-\Delta}^{\Delta} \sum_{\tau=-\Delta}^{\Delta} x[s+\sigma, t+\tau]\, w[\sigma,\tau]$$

$x[s+\sigma, t+\tau]\; w\;[\sigma,\tau]$.

example $\Delta : 2 \rightarrow$ restrict to $5 \cdot 5 = 25$ indices of input.

$$h[s,t] = \phi(z[s,t])$$

$$h[s,t] = \phi\left((x * w)[s,t]\right)$$



Consequences:

1) Sparsely connected network.

2) Weight sharing.

In practice, we have D input channels.
F output channels.

Input $x_1, x_2, \cdots, x_D$.

output $z_1, z_2, \cdots, z_F$

$$z_i = \sum_j x_j * w_{ij}$$

$$h_i = \phi(z_i).$$

Computational benefit:

$$(2\Delta+1)^2 \cdot D \cdot F$$

$$\approx O(\Delta^2 \cdot D \cdot F).$$

Eg.

$(2\Delta+1)^2 = 25 \cdot 3 \cdot 10$

$\approx \leq 1000$

- LeNet-5
- AlexNet (2012)
- VGG Net (2014)
- GoogleNet (2014)

[- Residual networks].

Training : Backpropagation.

Recurrent Neural Networks (Time series/ NLP- analysis).

NLP
- Document retrieval
- Speech to text
- Language translation
- Sentiment prediction

- Short range
- long range dependencies.

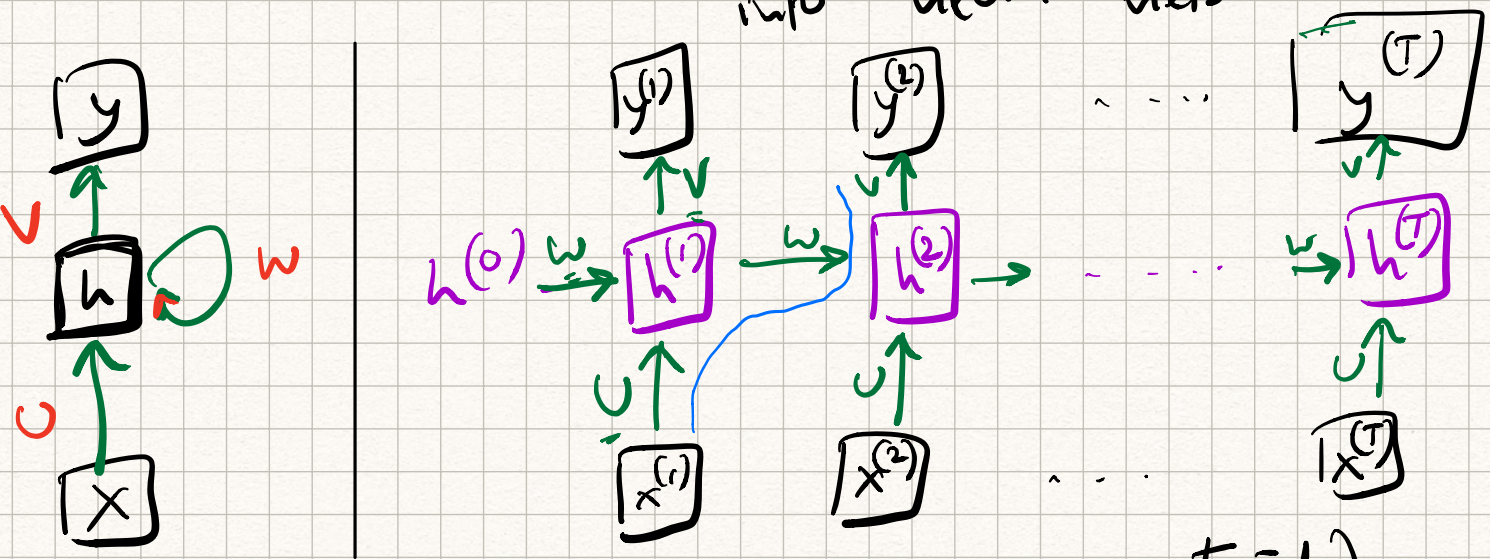
Classically: Markov models.

$$P(\{w_1, w_2, \ldots, w_d\})$$

$$= P(w_1)\, P(w_2 \mid w_1)\, P(w_3 \mid w_2) \cdots P(w_d \mid w_{d-1}).$$

$$P(w_3 \mid (w_1, w_2))$$

Recurrent neural nets: introduce feedback into neural nets.



$$h^t = \phi(U x^t + W h^{t-1}) \quad //$$

$$y^t = V h^t$$

. Observe: U, V, W don't change over time.

∴ Weight sharing.

Examples:

- Basic RNN (h is single layer network).

- GRU (Gated recurrent units)
- LSTM (Long-short term memory).
- Attention networks.