# CS540 Introduction to Artificial Intelligence

## Lecture 10

Young Wu

Based on lecture slides by Jerry Zhu, Yingyu Liang, and Charles Dyer

June 23, 2022

# Special Bayesian Network for Sequences

## Motivation

- A sequence of features $X_1, X_2, ...$ can be modeled by a Markov Chain but they are not observable.
- A sequence of labels $Y_1, Y_2, ...$ depends only on the current hidden features and they are observable.
- This type of Bayesian Network is called a Hidden Markov Model.

# HMM Applications Part 1

## Motivation

- Weather prediction.
- Hidden states: $X_1, X_2, ...$ are weather that is not observable by a person staying at home (sunny, cloudy, rainy).
- Observable states: $Y_1, Y_2, ...$ are Badger Herald newspaper reports of the condition (dry, dryish, damp, soggy).
- Speech recognition.
- Hidden states: $X_1, X_2, ...$ are words.
- Observable states: $Y_1, Y_2, ...$ are acoustic features.

# HMM Applications Part 2

## Motivation

- Stock or bond prediction.
- Hidden states: $X_1, X_2, \ldots$ are information about the company (profitability, risk measures).
- Observable states: $Y_1, Y_2, \ldots$ are stock or bond prices.
- Speech synthesis: Chatbox.
- Hidden states: $X_1, X_2, \ldots$ are context or part of speech.
- Observable states: $Y_1, Y_2, \ldots$ are words.

# Other HMM Applications

## Motivation

- Machine translation.
- Handwriting recognition.
- Gene prediction.
- Traffic control.

# Hidden Markov Model Diagram

## Motivation

# Transition and Likelihood Matrices

## Motivation

- An initial distribution vector and two-state transition matrices are used to represent a hidden Markov model.

1. Initial state vector: $\pi$.

$$\pi_i = \mathbb{P}\left\{X_1 = i\right\}, i \in 1, 2, ..., |X|$$

2. State transition matrix: $A$.

$$A_{ij} = \mathbb{P}\left\{X_t = j | X_{t-1} = i\right\}, i, j \in 1, 2, ..., |X|$$

3. Observation Likelihood matrix (or output probability distribution): $B$.

$$B_{ij} = \mathbb{P}\left\{Y_t = i | X_t = j\right\}, i \in 1, 2, ..., |Y|, j \in 1, 2, ..., |X|$$

# Evaluation and Training

## Motivation

- There are three main tasks associated with an HMM.

1. Evaluation problem: finding the probability of an observed sequence given an HMM: $y_1, y_2, ...$
2. Decoding problem: finding the most probable hidden sequence given the observed sequence: $x_1, x_2, ...$
3. Learning problem: finding the most probable HMM given an observed sequence: $\pi, A, B, ...$

# Expectation-Maximization Algorithm

## Description

- Start with a random guess of $\pi, A, B$.
- Compute the forward probabilities: the joint probability of an observed sequence and its hidden state.
- Compute the backward probabilities: the probability of an observed sequence given its hidden state.
- Update the model $\pi, A, B$ using Bayes rule.
- Repeat until convergence.
- Sometimes, it is called the Baum-Welch Algorithm.

# Hidden Markov Model Example 1

## Definition

# Hidden Markov Model Example 1 Computations

## Definition

# Hidden Markov Model Example 2

## Definition

# Hidden Markov Model Example 2 Computations

## Definition

# Hidden Markov Model Example 3

## Definition

# Hidden Markov Model Example 3 Computations

## Definition

# Dynamic System

## Motivation

- The hidden units are used as the hidden states.
- They are related by the same function over time.

$$h_{t+1} = f(h_t, w)$$
$$h_{t+2} = f(h_{t+1}, w)$$
$$h_{t+3} = f(h_{t+2}, w)$$
$$\dots$$

# Dynamic System with Input

## Motivation

- The input units can also drive the dynamics of the system.
- They are still related by the same function over time.

$$h_{t+1} = f\left(h_t, x_{t+1}, w\right)$$
$$h_{t+2} = f\left(h_{t+1}, x_{t+2}, w\right)$$
$$h_{t+3} = f\left(h_{t+2}, x_{t+3}, w\right)$$
$$\ldots$$

# Dynamic System with Output

## Motivation

- The output units only depend on the hidden states.

$$y_{t+1} = f\left(h_{t+1}\right)$$
$$y_{t+2} = f\left(h_{t+2}\right)$$
$$y_{t+3} = f\left(h_{t+3}\right)$$
$$\ldots$$

# Dynamic System Diagram

## Motivation

# Recurrent Neural Network Structure Diagram

## Motivation

# Activation Functions

## Definition

- The hidden layer activation function can be the tanh activation, and the output layer activation function can be the softmax function.

$$z_t^{(x)} = W^{(x)} x_t + W^{(h)} a_{t-1}^{(x)} + b^{(x)}$$

$$a_t^{(x)} = g\left(z_t^{(x)}\right), g\left(\boxed{\cdot}\right) = \tanh\left(\boxed{\cdot}\right)$$

$$z_t^{(y)} = W^{(y)} a_t^{(x)} + b^{(y)}$$

$$a_t^{(y)} = g\left(z_t^{(y)}\right), g\left(\boxed{\cdot}\right) = \text{softmax}\left(\boxed{\cdot}\right)$$

# Cost Functions

## Definition

- Cross entropy loss is used with softmax activation as usual.

$$C_t = H\left(y_t, a_t^{(y)}\right)$$

$$C = \sum_t C_t$$

# BackPropogation Through Time

## Definition

- The gradient descent algorithm for recurrent neural networks is called BackPropogation Through Time (BPTT). The update procedure is the same as standard neural networks using the chain rule.

$$w = w - \alpha \frac{\partial C}{\partial w}$$

$$b = b - \alpha \frac{\partial C}{\partial b}$$

# Unfolded Network Diagram

## Definition

# Vanishing and Exploding Gradient

## Discussion

- If the weights are small, the gradient through many layers will shrink exponentially. This is called the vanishing gradient problem.
- If the weights are large, the gradient through many layers will grow exponentially. This is called the exploding gradient problem.
- Fully connected and convolutional neural networks only have a few hidden layers, so vanishing and exploding gradient is not a problem in training those networks.
- In a recurrent neural network, if the sequences are long, the gradients can easily vanish or explode.

# RNN Variants

## Discussion

- Long Short Term Memory (LSTM): gated units to keep track of long term dependencies.
- Gated Recurrent Unit (GRU): different gated units.
- Transformers (BERT, GPT): no recurrent units, positional encoding, attention mechansim.