Recurrent Neural Networks II

COSC 410: Applied Machine Learning
Fall 2025
Prof. Forrest Davis

October 28, 2025

Warm-up

1. It's spooky season. Discuss with your neighbor the best type of Halloween candy.
2. The partial derivative of the loss with respect to the weight that updates the hidden representation (e.g., $\frac{\partial \text{Loss}}{\partial h}$ based on our equation last class) contains the hidden representation from the time step before (A^{t-1} in the terms we used last class). Consider two activation functions used to create the new hidden representation:
   - Sigmoid
   - ReLU

   Which is more prone to exploding gradient? Which to vanishing gradient? Why?

Logistics

- Codelet 4 on Feed-Forward Neural Networks out, due Friday Oct 31
- Prof. Vijay is observing the course today

Learning Objectives

- Map the mathematical representation of RNNs to graphs
- Understand the properties captured by a RNN and why they are useful
- Describe key RNN architectures and relevant tasks
- Describe the key challenge with image data

Summary: We link the mathematical representation of a recurrent neural network to its graphical representation. We lay out some key architectures and tasks that uses these networks, with a deeper dive into one application and some remarks on training. We end by motivating the next architecture we will explore.

Refresher and Mathematical Expression for a RNN

LAST CLASS, WE DISTINGUISHED between a rolled and unrolled graphical representation of a recurrent neural network. The book gives the following expressions to characterize a one-layer RNN.

$$H_t = \phi(X_t W_{xh} + H_{t-1} W_{hh} + b_h) \tag{1}$$
$$O_t = (H_t W_{hq} + b_q) \tag{2}$$

Question

1. Give a rolled representation of the general expression for a one-layer RNN as outlined by the book.

2. How would you handle the hidden representation for the first time step (when t=1)?

Motivating Core Properties of RNNs

WE IDENTIFIED three key properties we wanted to encode in a network architecture:

1. Uniform treatment of the same features across time
2. Retention of past information
3. A notion that time is unfolding[1]

[1] This could include biases (in a technical sense) like a greater weight to more recent information, but that isn't necessarily desirable in all contexts. What is essential is capturing the temporal dynamics of sequential data.

Let's refresh these points and outline some other ways to motivate why we care about them/what we are trying to capture with them (on the slides).

Types of Architectures

THERE ARE MANY BROAD architectures that are used in applied settings for sequential data. I'll highlight the main ones. The key concepts remain relevant even with different sequential mechanisms than recurrence (like attention in transformers), so it is worth learning these ideas. I'll demonstrate them on the slides, but I'll add the names here:

- **Encoders** (many to one)
- **Decoders** (one to many)
- **One to One**
- **Encoder-Decoders** (many to many)
- **Deep/Stacked** recurrent neural networks
- **Bidirectional** recurrent neural networks

Deeper Application Dive: Character-Level Language Model

ONE QUITE DOMINANT USE of a recurrent neural network is in language modeling.[2] Let's lay out the application in the character domain at some depth, so you can get a sense for the design.

[2] Language models are the backbone of models like ChatGPT. Their aim is to estimate the probability of a word conditioned on the preceding textual context. Only minor tweaks have to be added to switch this discussion to the word domain.

Practice Problems

Give two concrete tasks for each architecture. Describe the input and the output.

1. Encoder
2. Decoder
3. One to One
4. Encoder-Decoder

Notes on Training

WE DETERMINED THAT IN ORDER for the gradient to be computationally feasible, we needed to truncate backpropagation through time and calculate the gradient for a recurrent neural network based on sub-intervals of time. How we do this impacts learning. See the slides for considerations.

Additionally, I want to point out an issue with training decoders. Namely, the model can be hard to train because it can make mistakes along the generation pipeline, which derail the network. We can make the training in these domains faster by using **teacher forcing**. See this slides for an example.

Motivating the Next Architecture

EVERYDAY DATA IS INHERENTLY HIGH-DIMENSIONAL. Consider the representation of images on the slide.

Question

For these questions consider the task of building a fully connected, feed-forward neural network architecture.

1. How many input features are there from one 1080p HD color image? (Note: 1080p is 1920 pixels by 1080 pixels)?

2. Consider a 4 min HD YouTube video at 30fps. How many features would it have?

Before Next Class

- Reading and pre-class quiz
- Work on Codelet 4