UNC CHARLOTTE

The WILLIAM STATES LEE COLLEGE of ENGINEERING

Introduction to ML
Lecture 7: Recurrent Neural Network

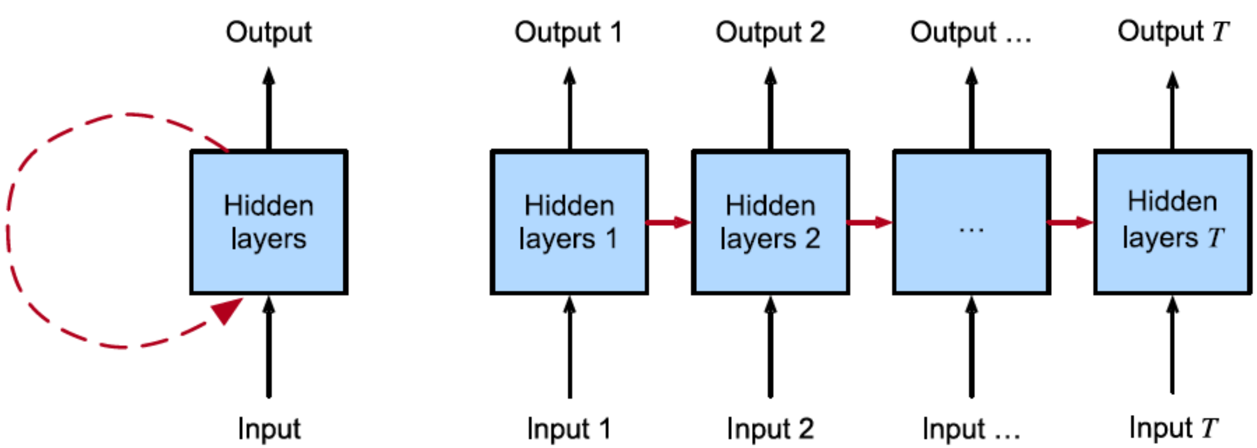
Hamed Tabkhi
Department of Electrical and Computer Engineering,
University of North Carolina Charlotte (UNCC)
htabkhiv@uncc.edu

UNC CHARLOTTE

Intro to RNNs

- Countless learning tasks require dealing with sequential data.
- Video captioning, speech synthesis, and music generation all require that models produce outputs consisting of sequences.
- In other domains, such as time series prediction, video analysis, and musical information retrieval, a model must learn from inputs that are sequences.

- Recurrent neural networks (RNNs) are deep learning models that capture the dynamics of sequences via *recurrent* connections, which can be thought of as cycles in the network of nodes.
- RNNs can be thought of as feedforward neural networks where each layer's parameters (both conventional and recurrent) are shared across



- On the left recurrent connections are depicted via cyclic edges. On the right, we unfold the RNN over time steps.

Intro to RNNs

State at any time step *t (ht)* could be computed based on both the current input *Xt* and the previous state *ht-1*

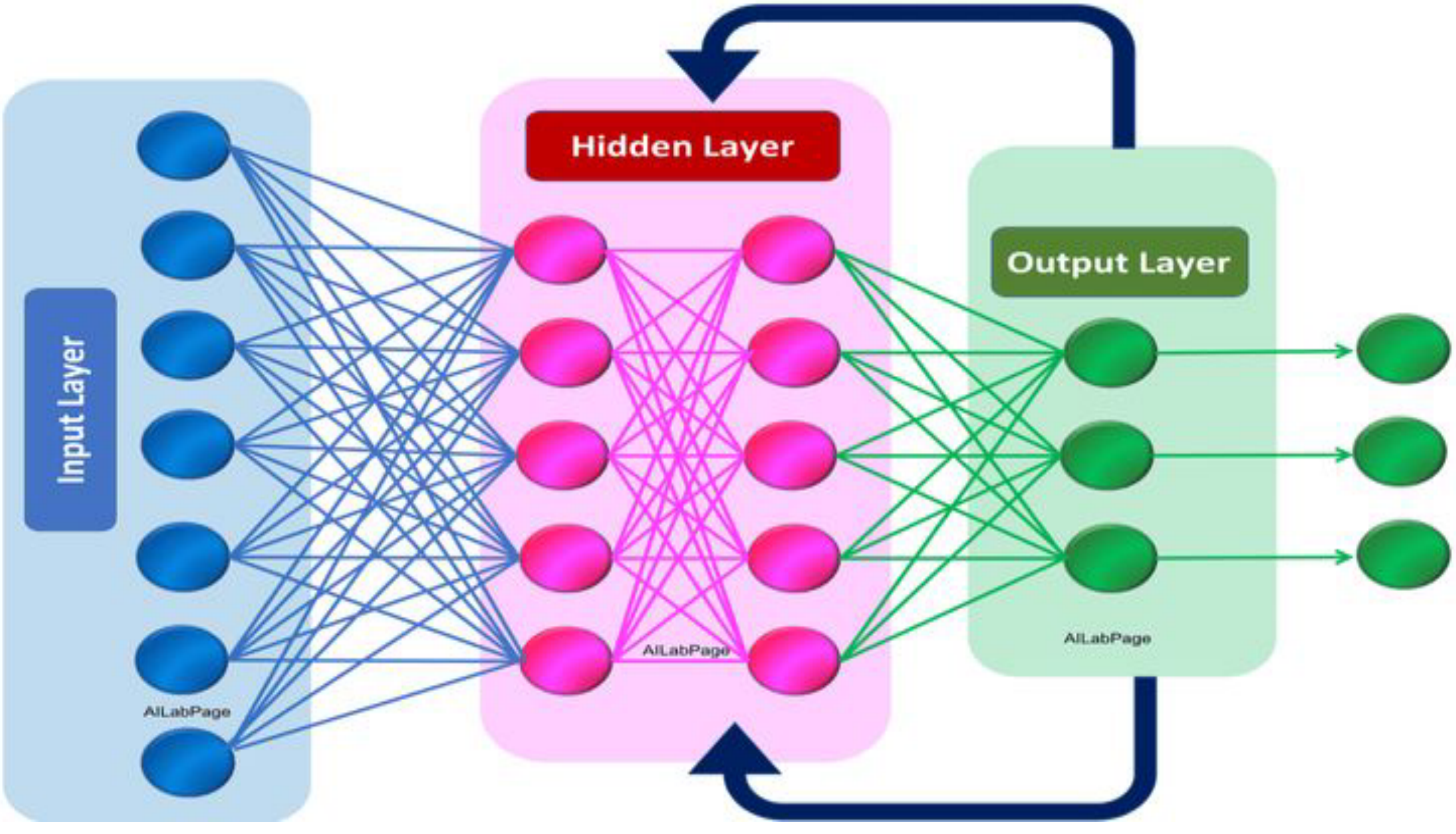
$$h_t = f(x_t, h_{t-1}).$$

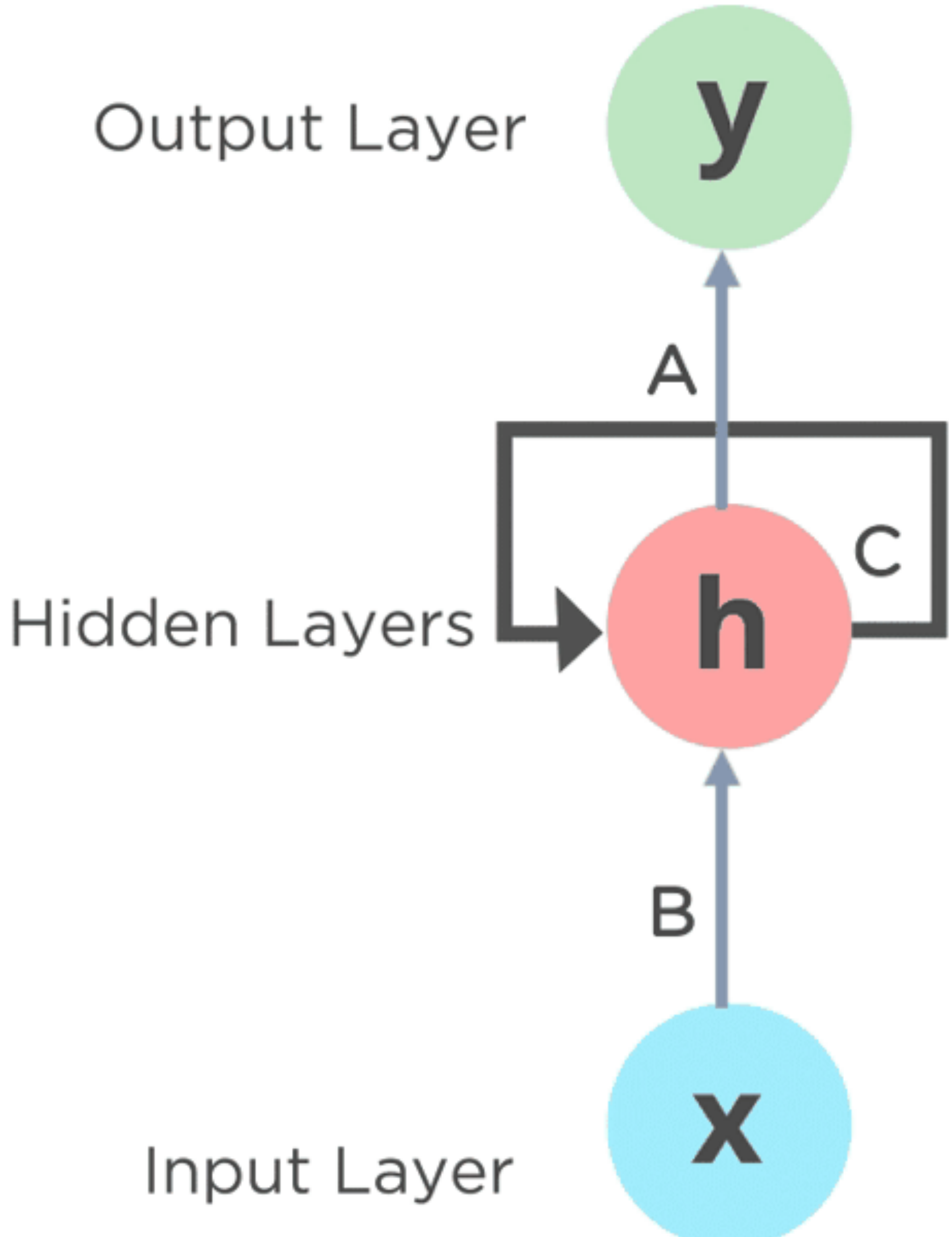
RNNs rose to prominence as the default models for handling complex sequential structure in deep learning and remain staple models for sequential modeling to this day.

RNN Key insights:
While the inputs and targets for many fundamental tasks in machine learning cannot easily be represented as fixed length vectors, they can often nevertheless be represented as varying-length sequences of fixed length vectors.

For example, videos can be represented as varying-length sequences of still images.

RNN presented





A, B and C are the parameters

RNN Formulation with Hidden States (Layers)

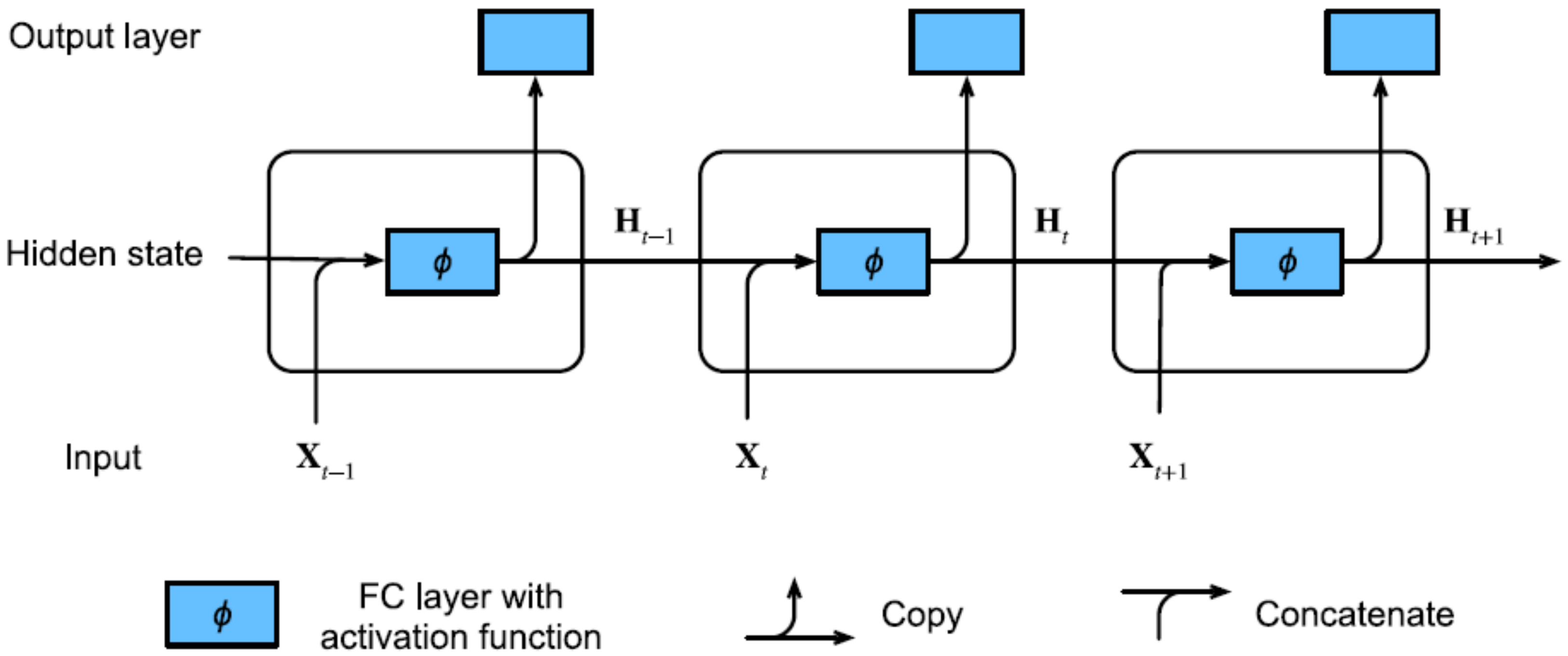


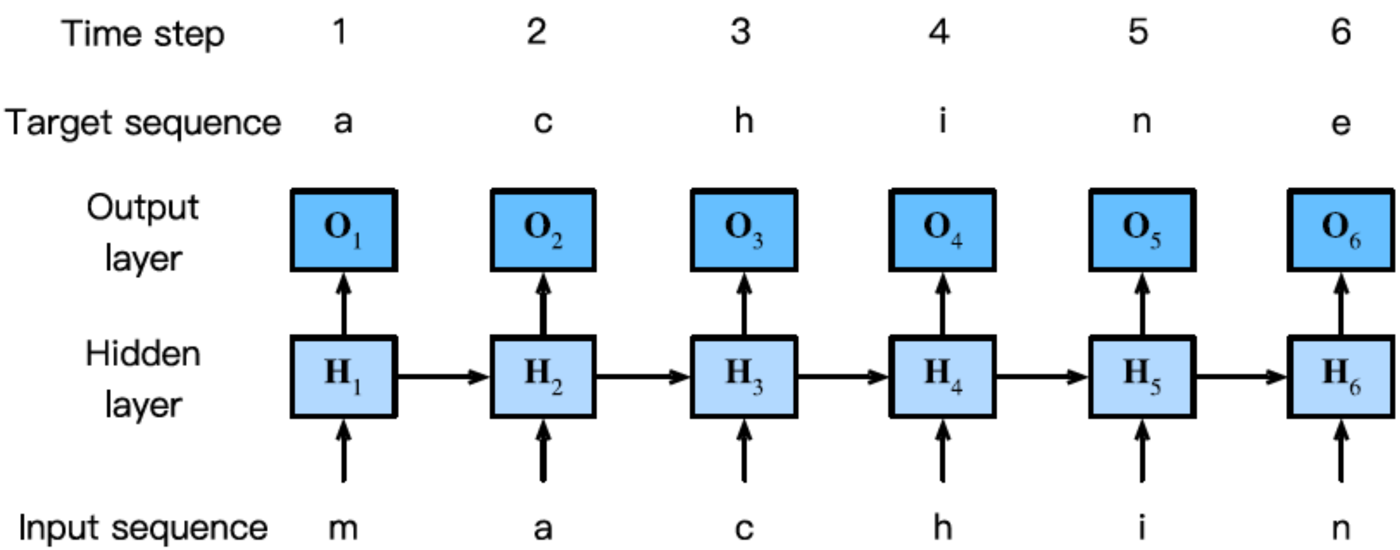
Fig illustrates the computational logic of an RNN at three adjacent time steps.
At any time step *t*, the computation of the hidden state can be treated as:

(i) concatenating the input **Xt** at the current time step *t* and the hidden state **Ht-1** at the previous time step t-1
(ii) feeding the concatenation result into a fully connected layer with the activation function ϕ.

The output of such a fully connected layer is the hidden state **Ht** of the current time step *t*.
In this case, the model parameters are the concatenation of **WXh** and **Whh**, and a bias of **Bh**.
What is more, **Ht** will also be fed into the fully connected output layer to compute the output **Ot** of the current time step *t*.

The hidden state of the current time step *t*, **Ht** will participate in computing the hidden state **Ht+1** of the next time step *t*+1.

Example: RNN for character-level language model



Predicting the next character based on the current and previous characters via an RNN for character-level language modeling

- Due to the recurrent computation of the hidden state in the
- hidden layer, the output of time step 3, is determined by the text sequence “m”, “a”, and “c”.
- Since the next character of the sequence in the training data is “h”, the correct prediction of time step 3 will depend on the probability distribution of the next character generated based on the feature sequence “m”, “a”, “c” and the target “h” of this time step.

UNC CHARLOTTE