RNN

Dr. Hasan Mahmud

Different types of data in machine learning

- **Spatial data:** Spatial data is actually tabular data, but its observation has spatial attributes. Spatial data is directly or indirectly references a specific geographical area or location.
- **Temporal data:** Temporal data is simply data that represents a state over time. Temporal data is collected to analyze weather patterns and other environmental variables, monitor traffic conditions, study demographic trends, and so on.
- **Time-series data:** is a sequence of data points collected over time intervals, allowing us to *track changes over time*. Time-series data can track changes over milliseconds, days, or even years. Time-series data is also sequential data
- **Sequential Data:** sequential Data refers to any data that contain elements that are ordered into sequences. Examples include time series, DNA sequences (see biomedical informatics) and sequences of user actions.

Real-life Sequence Learning Applications

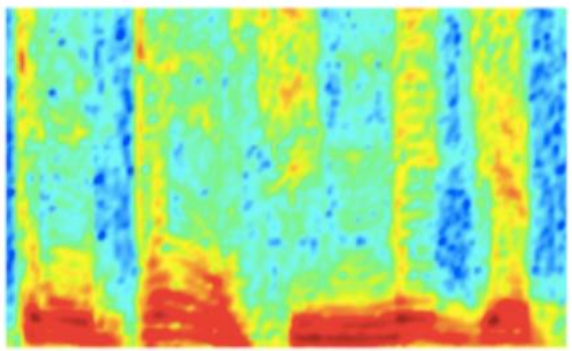
- RNNs can be applied to various type of sequential data to learn the temporal patterns.
  - Time-series data (e.g., stock price) → Prediction, regression
  - Raw sensor data (e.g., signal, voice, handwriting) → Labels or text sequences
  - Text → Label (e.g., sentiment) or text sequence (e.g., translation, summary, answer)
  - Image and video → Text description (e.g., captions, scene interpretation)

Task	**Input**	**Output**
Activity Recognition (Zhu et al. 2018)	Sensor Signals	Activity Labels
Machine translation (Sutskever et al. 2014)	English text	French text
Question answering (Bordes et al. 2014)	Question	Answer
Speech recognition (Graves et al. 2013)	Voice	Text
Handwriting prediction (Graves 2013)	Handwriting	Text
Opinion mining (Irsoy et al. 2014)	Text	Opinion expression

Sequence Sources

* Elements of a sequence occur in a certain order
* Elements depend on each other

AUDIO



Audio Spectrogram

IMAGES

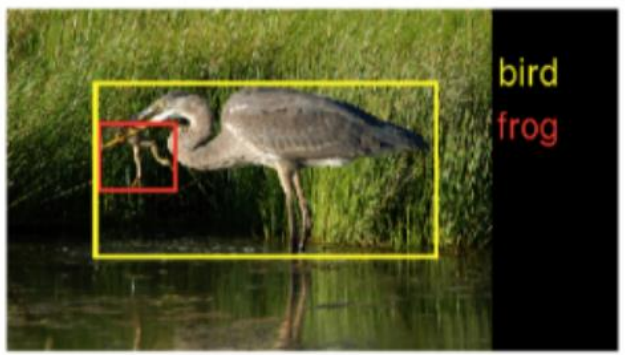
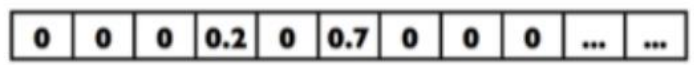


Image pixels

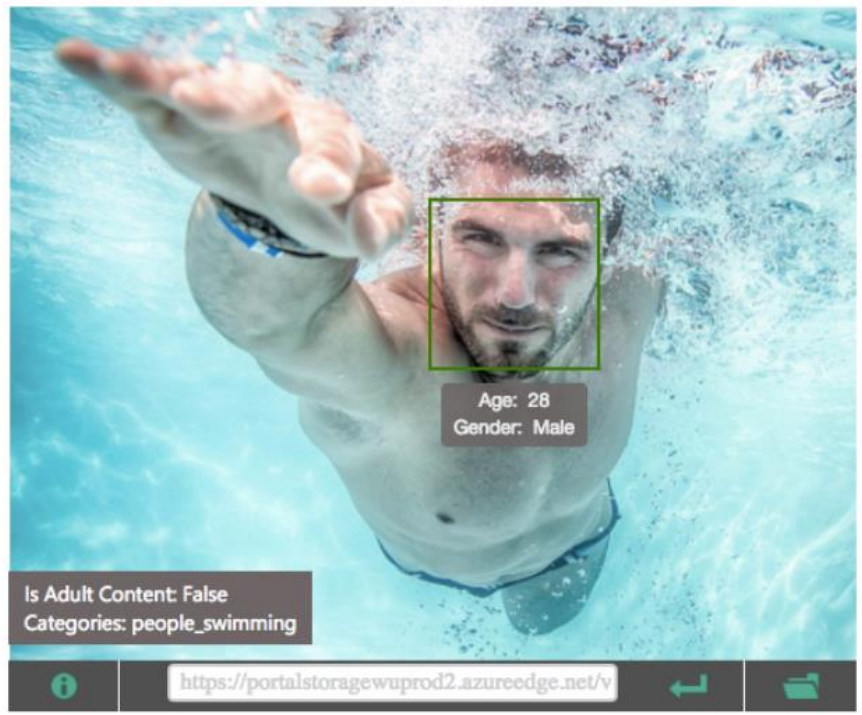
TEXT



Word, context, or document vectors

Sequence Applications: One-to-Many

- **Input**: fixed-size
- **Output**: sequence
- e.g., image captioning



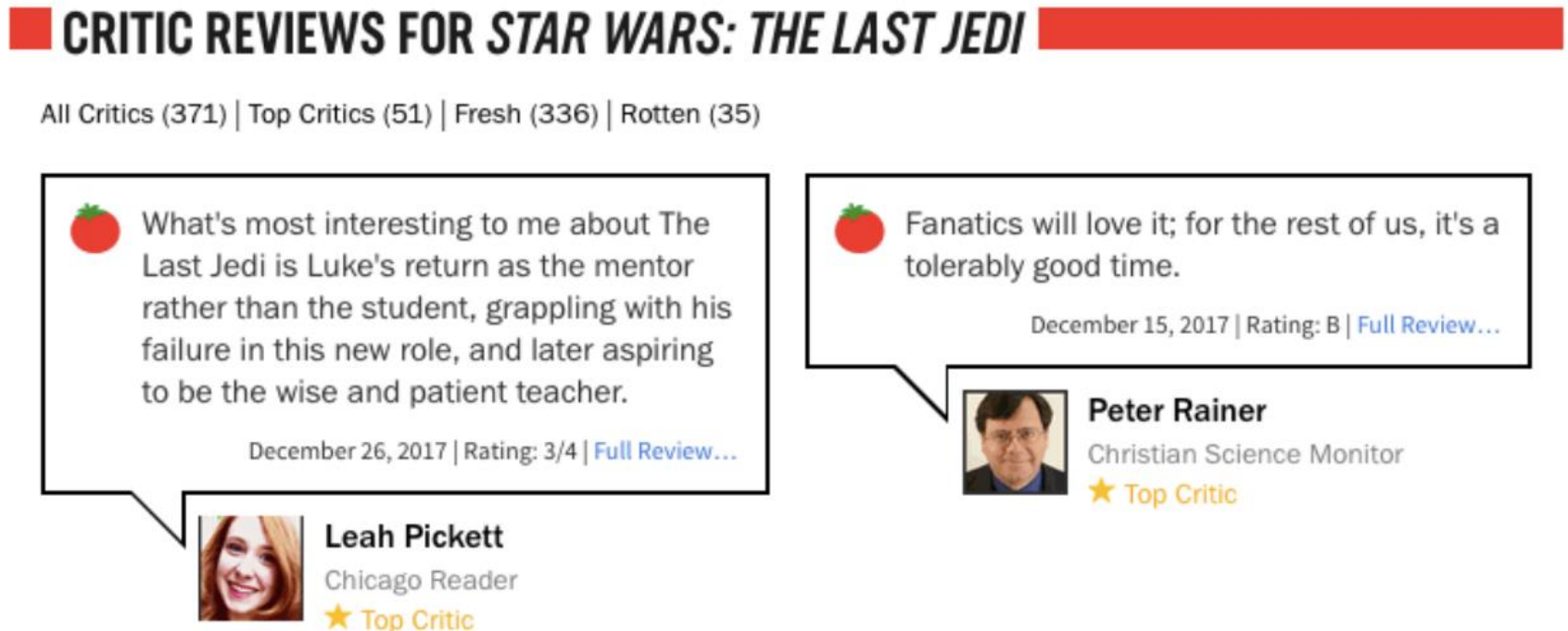
Features:

Feature Name	Value
Description	{ "type": 0, "captions": [{ "text": "a man swimming in a pool of water", "confidence": 0.7850108693093019 }] }
Tags	[{ "name": "water", "confidence": 0.9996442794799805 }, { "name": "sport", "confidence": 0.9504992365837097 }, { "name": "swimming", "confidence": 0.9062818288803101, "hint": "sport" }, { "name": "pool", "confidence": 0.8787588477134705 }, { "name": "water sport", "confidence": 0.631849467754364, "hint": "sport" }]
Image Format	jpeg
Image Dimensions	1500 x 1155
Clip Art Type	0 Non-clipart
Line Drawing Type	0 Non-LineDrawing
Black & White Image	False

Captions: https://www.microsoft.com/cognitive-services/en-us/computer-vision-api

Sequence Applications: Many-to-One

- **Input**: sequence
- **Output**: fixed-size
- e.g., sentiment analysis (hate? love?, etc)



https://www.rottentomatoes.com/m/star_wars_the_last_jedi

Sequence Applications: Many-to-Many

- **Input**: sequence
- **Output**: sequence
- e.g., language translation

1-of-N encoding

How to represent each word as a vector?

1-of-N Encoding lexicon = {apple, bag, cat, dog, elephant}

The vector is lexicon size.

Each dimension corresponds to a word in the lexicon

The dimension for the word is 1, and others are 0

apple = [1 0 0 0 0]

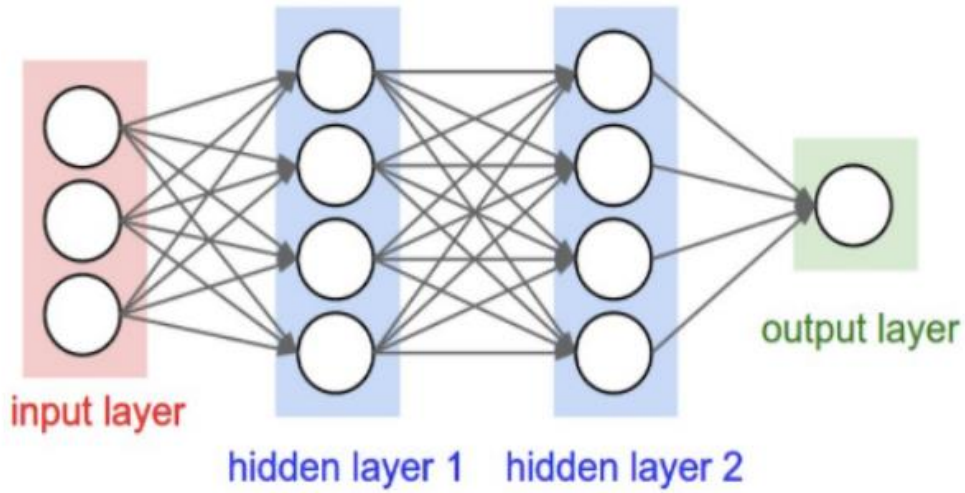
bag = [0 1 0 0 0]

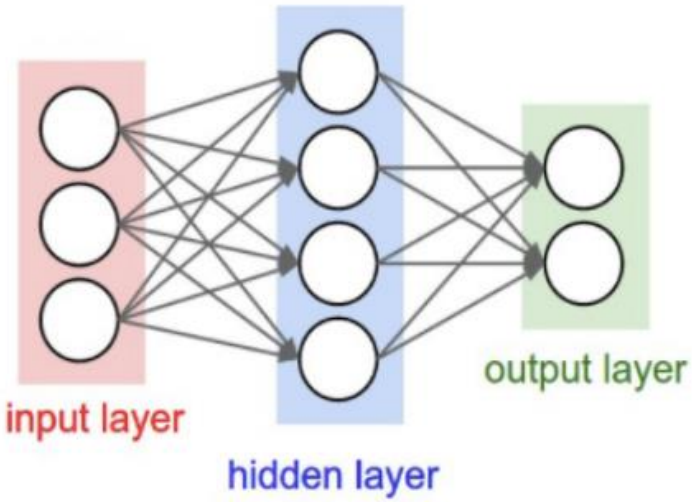
cat = [0 0 1 0 0]

dog = [0 0 0 1 0]

elephant = [0 0 0 0 1]

Recall: Feedforward Neural Networks





Problem: many model parameters!

Problem: no memory of past since weights learned independently

Each layer serves as input to the next layer with no loops

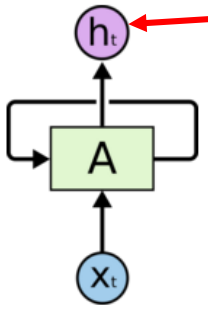
Figure Source: http://cs231n.github.io/neural-networks-1/

Recurrent Neural Networks

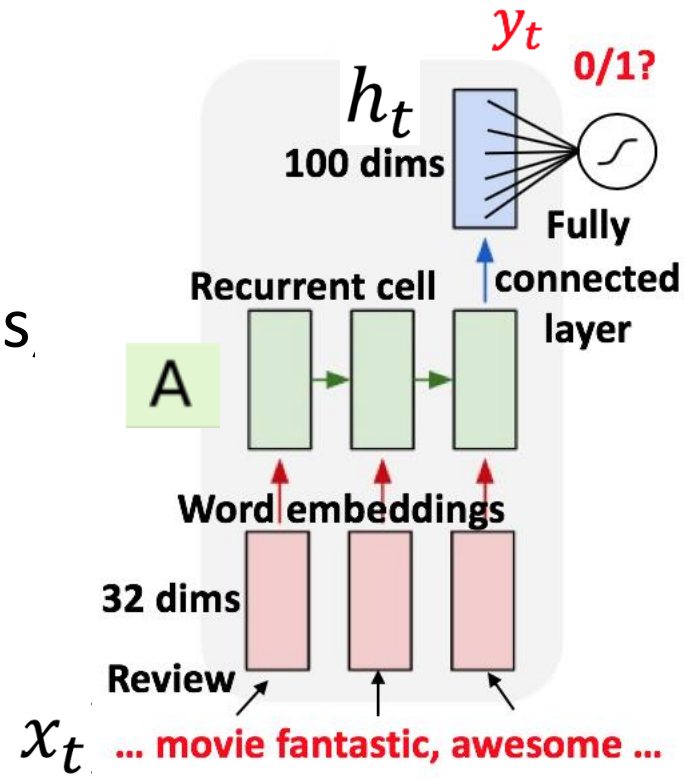
- Human brain deals with information streams. Most data is obtained, processed, and generated sequentially.
  - E.g., listening: soundwaves → vocabularies/sentences
  - E.g., action: brain signals/instructions → sequential muscle movements
- Human thoughts have persistence; humans don't start their thinking from scratch every second.
  - As you read this sentence, you understand each word based on your prior knowledge.
- The applications of standard Artificial Neural Networks (and also Convolutional Networks) are limited due to:
  - They only accepted a fixed-size vector as input (e.g., an image) and produce a fixed-size vector as output (e.g., probabilities of different classes).
  - These models use a fixed amount of computational steps (e.g. the number of layers in the model).
- Recurrent Neural Networks (RNNs) are a family of neural networks introduced to **learn sequential data**.
  - Inspired by the temporal-dependent and persistent human thoughts

Recurrent Neural Networks

- Recurrent Neural Networks are networks with loops allowing information to persist.



Output is to predict a vector h_t, where $output\ y_t = \varphi(h_t)$ at some time steps (t)



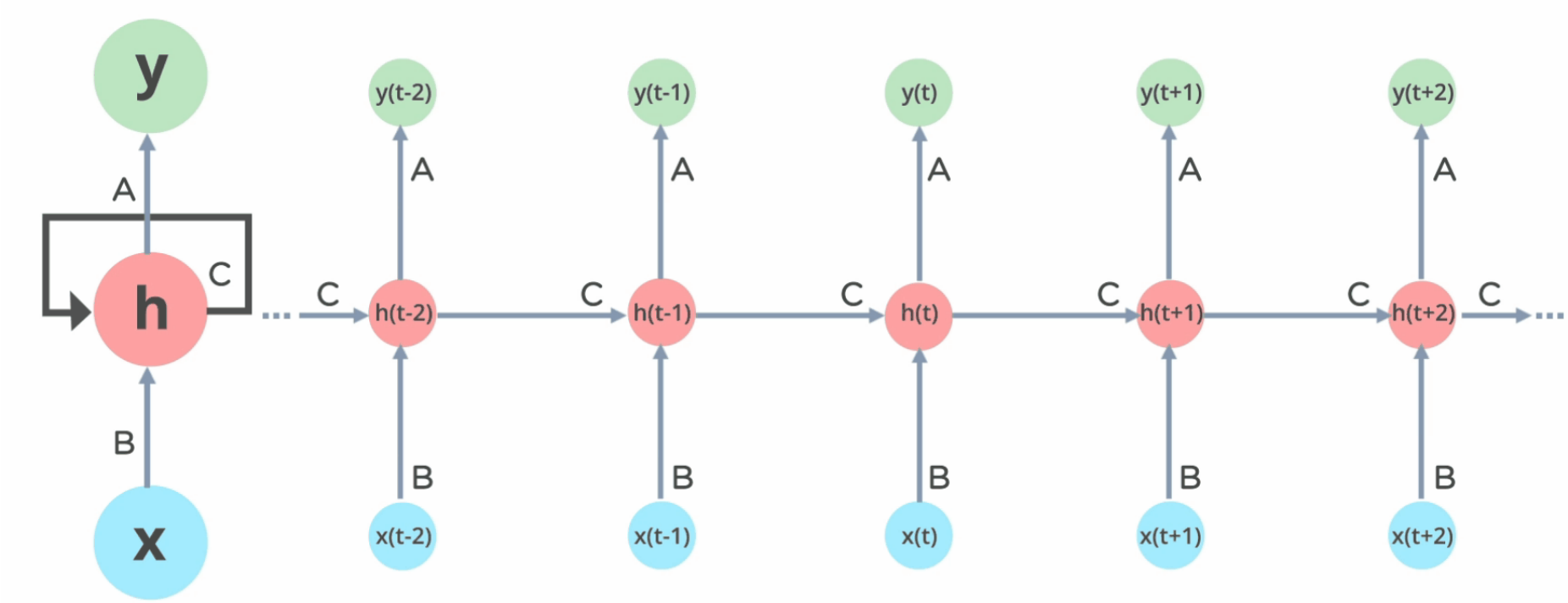
Recurrent Neural Networks have loops.

In the above diagram, a chunk of neural network, $A = f_W$, looks at some input x_t and outputs a value h_t. A loop allows information to be passed from one step of the network to the next.

new state — old state

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

function with parameter W — Input vector at some time step



A recurrent neural network can be thought of as multiple copies of the same network, each passing a message to a successor. The diagram above shows what happens if we **unroll the loop**.

Two problems with RNN

- Vanishing gradient
- Exploding gradient