Recurrent Neural Networks I

Swati Mishra

Applications of Machine Learning (4AL3)

Fall 2024

McMaster University
ENGINEERING

Introduction

- In some types of dataset, the information is captured not just in the instances , but also the sequence in which they occur.
  - Example: Language, Time-series,

Introduction

- In some types of dataset, the information is captured not just in the instances , but also the sequence in which they occur.
  - Example: Language, Time-series,

- When applying machine learning to sequences:
  - Input variable is a sequence
  - Target variable is also a sequence (only in a different domain)

Introduction

- In some types of dataset, the information is captured not just in the instances , but also the sequence in which they occur.
  - Example: Language, Time-series,

- When applying machine learning to sequences:
  - Input variable is a sequence
  - Target variable is also a sequence (only in a different domain)

- Rarely do we have a fixed target sequences. In such cases, we predict the next item in sequence at next time step t+1, based on the input data until time step 1.

Introduction

- In some types of dataset, the information is captured not just in the instances , but also the sequence in which they occur.
  - Example: Language, Time-series,

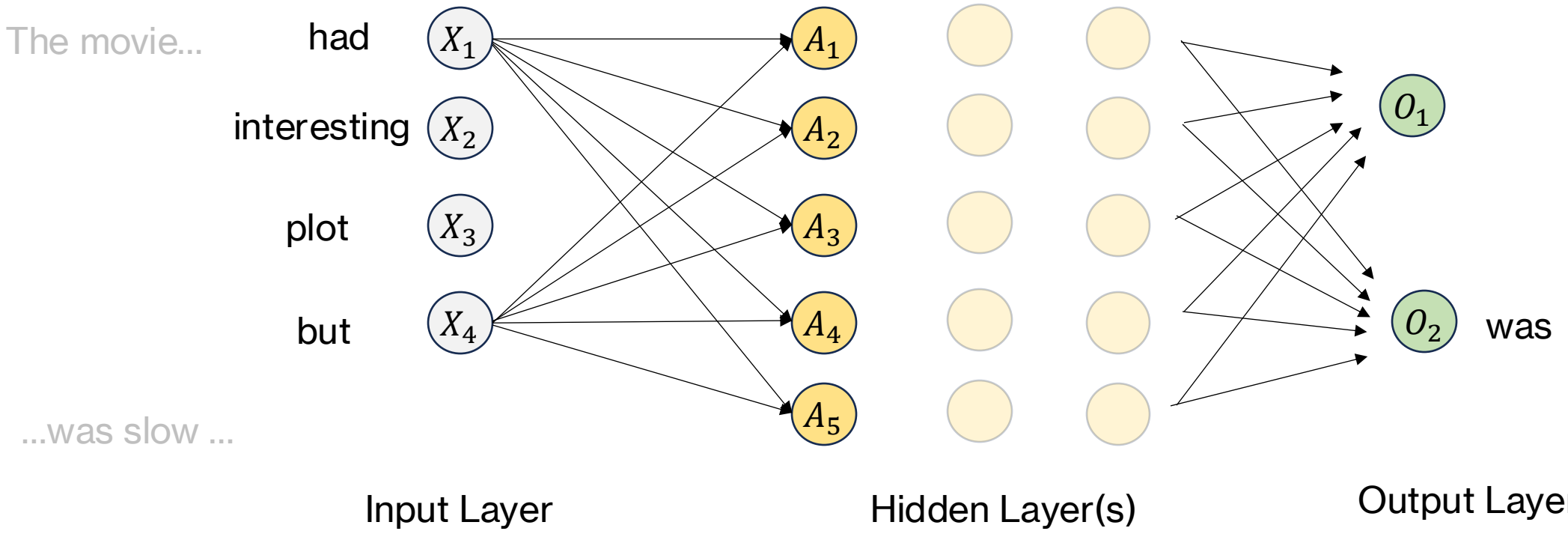
- When applying machine learning to sequences:
  - Input variable is a sequence
  - Target variable is also a sequence (only in a different domain)

- Rarely do we have a fixed target sequences. In such cases, we predict the next item in sequence at next time step t+1, based on the input data until time step 1.

Is RNN supervised or semi supervised?

Feedforward Neural Networks

- Feedforward Neural networks employ fixed-size input vectors with associated weights to capture all relevant aspects of an example at once .



Recurrent Neural Networks

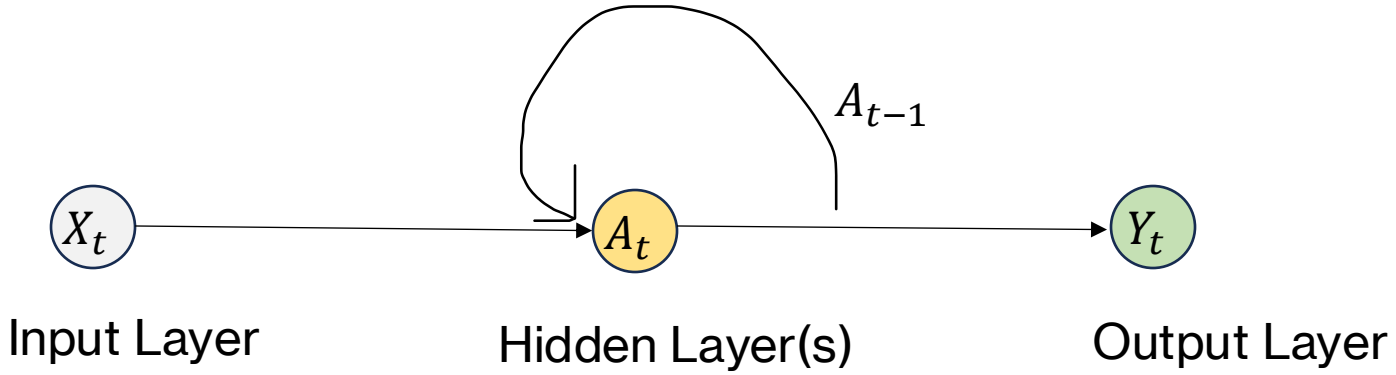
- A recurrent neural network (RNN) is any network that contains a feedback connection within its network connections .

- The outputs of one unit is directly dependent on the previous unit.

Recurrent Neural Networks

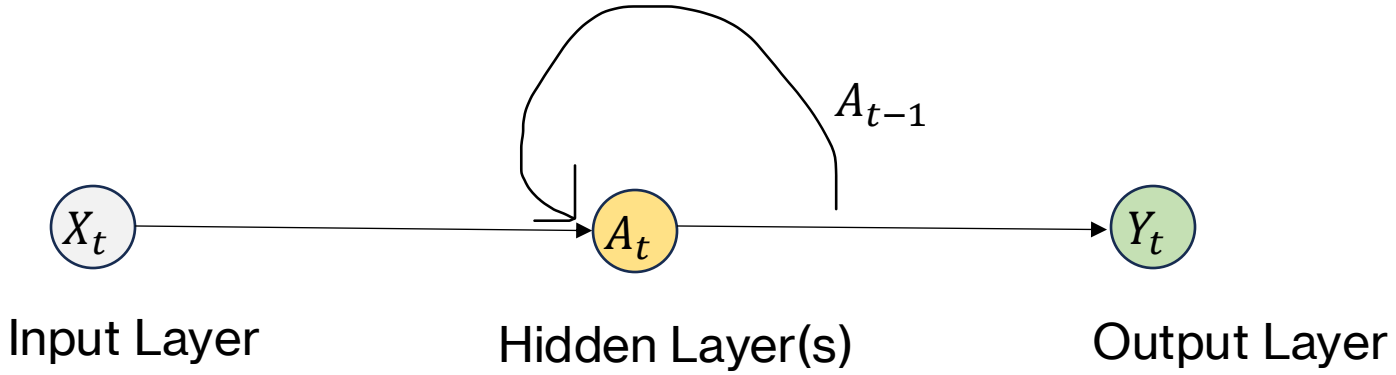
- A recurrent neural network (RNN) is any network that contains a feedback connection within its network connections .

- The outputs of one unit is directly dependent on the previous unit.



Recurrent Neural Networks

- A recurrent neural network (RNN) is any network that contains a feedback connection within its network connections .
- The outputs of one unit is directly dependent on the previous unit.



- The new set of weights determine how the network should make use of past context to calculate the output for the current input.

Recurrent Neural Networks: Inference

- The mapping between input and outputs is **almost** like traditional NNs.

$$h_t = g(Uh_{t-1} + Wx_t)$$

$$y_t = softmax(Vh_t)$$

Recurrent Neural Networks: Inference

- The mapping between input and outputs is **almost** like traditional NNs.

$$h_t = g(Uh_{t-1} + Wx_t)$$

$$y_t = softmax(Vh_t)$$

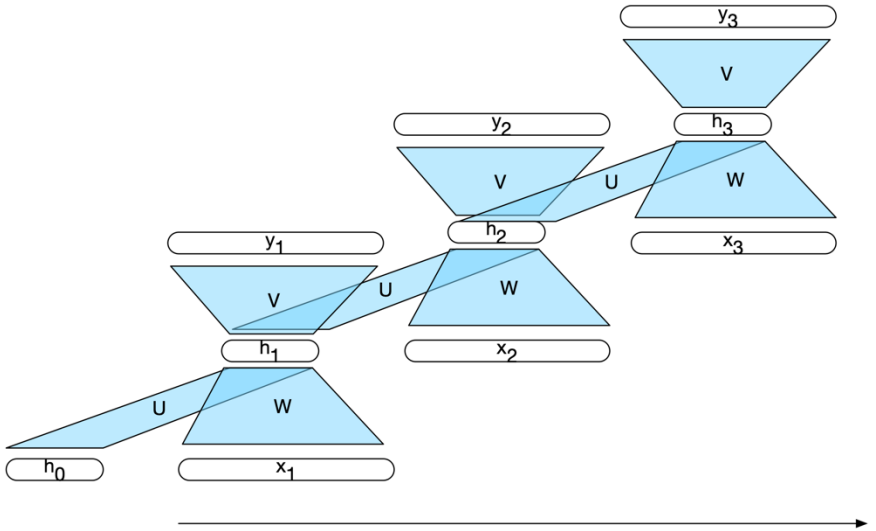
- Giving a hidden state to our model, allows for storage of information.
- Our goal is to get a good enough estimate of probability distribution over the space of hidden state vectors, to compute the next output.

Recurrent Neural Networks: Inference

- The mapping between input and outputs is **almost** like traditional NNs.

$$h_t = g(Uh_{t-1} + Wx_t)$$

$$y_t = softmax(Vh_t)$$



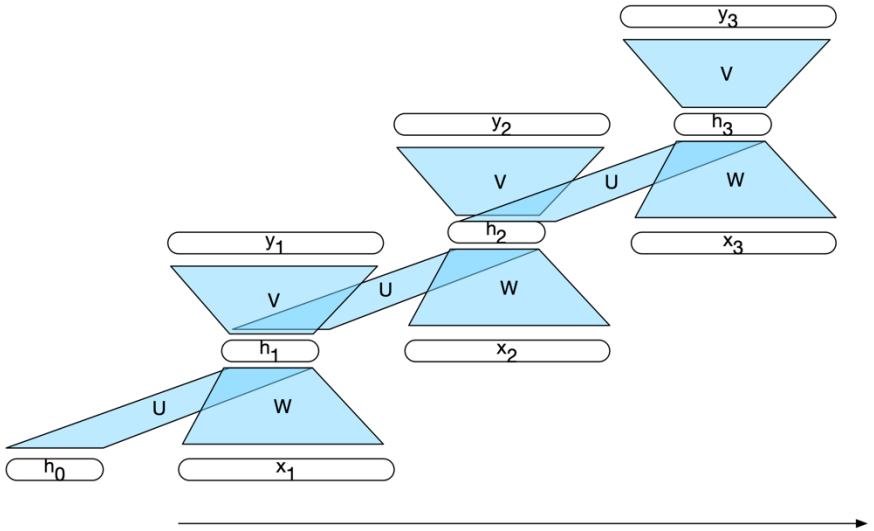
- Giving a hidden state to our model, allows for storage of information.
- Our goal is to get a good enough estimate of probability distribution over the space of hidden state vectors, to compute the next output.

Picture Credit: Speech and Language Processing - Jurafsky and Martin

Recurrent Neural Networks: Inference

- The mapping between input and outputs is **almost** like traditional NNs.

  - $h_0 \leftarrow 0$
    - for $i \leftarrow 1$ to length (x) do
      - $h_i \leftarrow g(Uh_{i-1} + Wx_i)$
      - $y_i \leftarrow f(Vh_i)$
    - return y

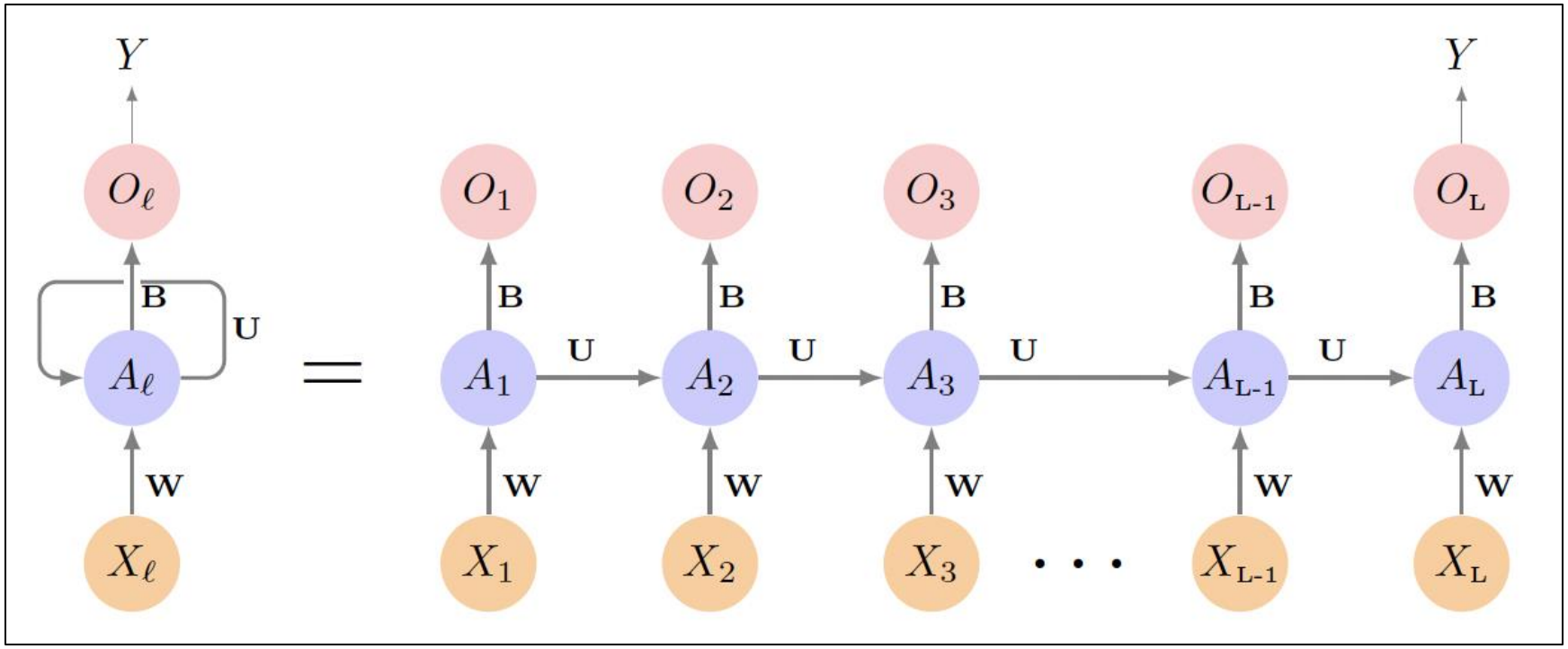


- Giving a hidden state to our model, allows for storage of information.

- Our goal is to get a good enough estimate of probability distribution over the space of hidden state vectors, to compute the next output.

Picture Credit: Speech and Language Processing - Jurafsky and Martin

Recurrent Neural Networks: Inference



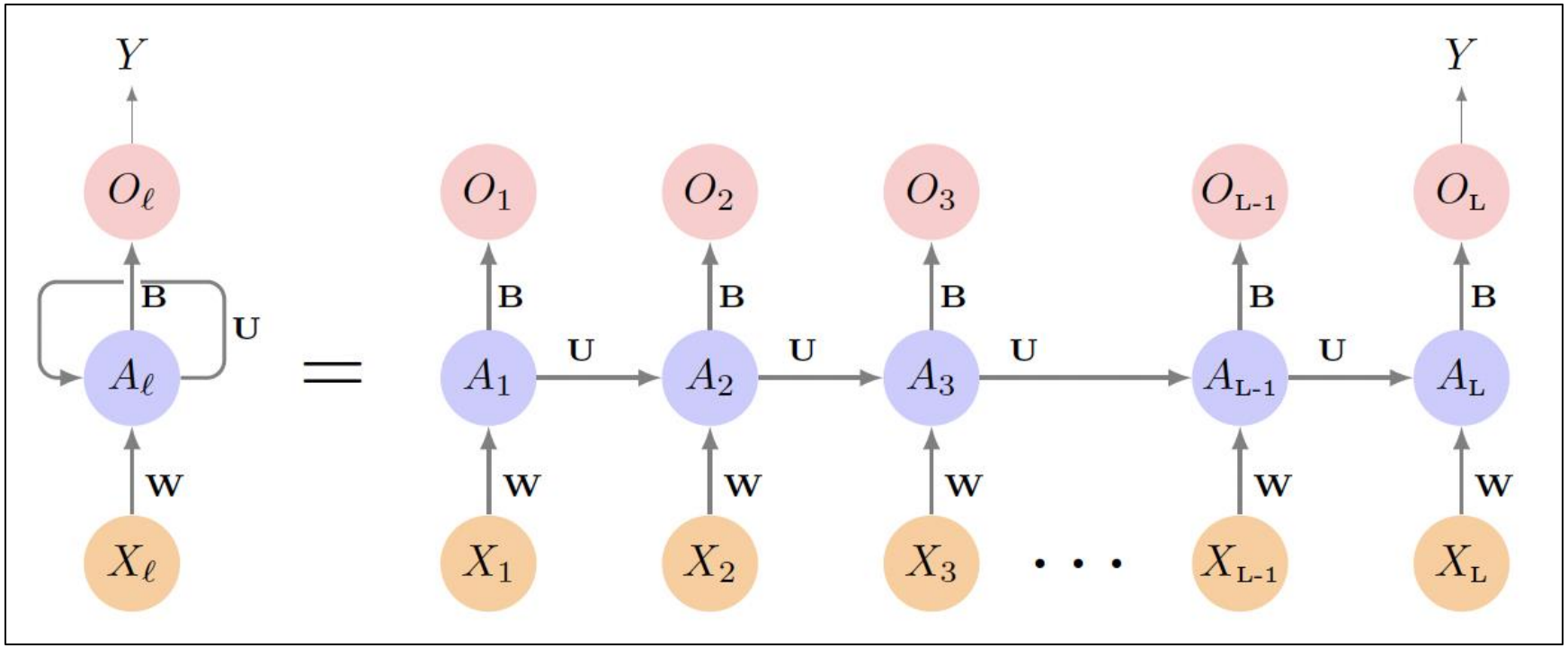
Picture Credit: ISLP

Recurrent Neural Networks: Inference

Output units

Hidden units

Input sequence



$$A_l^T = (A_{l1}, A_{l2}, \ldots, A_{lk}) \qquad A_{lk} = g\left(w_{k0} + \sum_{j=1}^{p} w_{kj} X_{lj} + \sum_{s=1}^{K} u_{ks} A_{l-1,s}\right) \qquad O_l = \beta_0 + \sum_{k=1}^{K} \beta_k A_{lk}$$

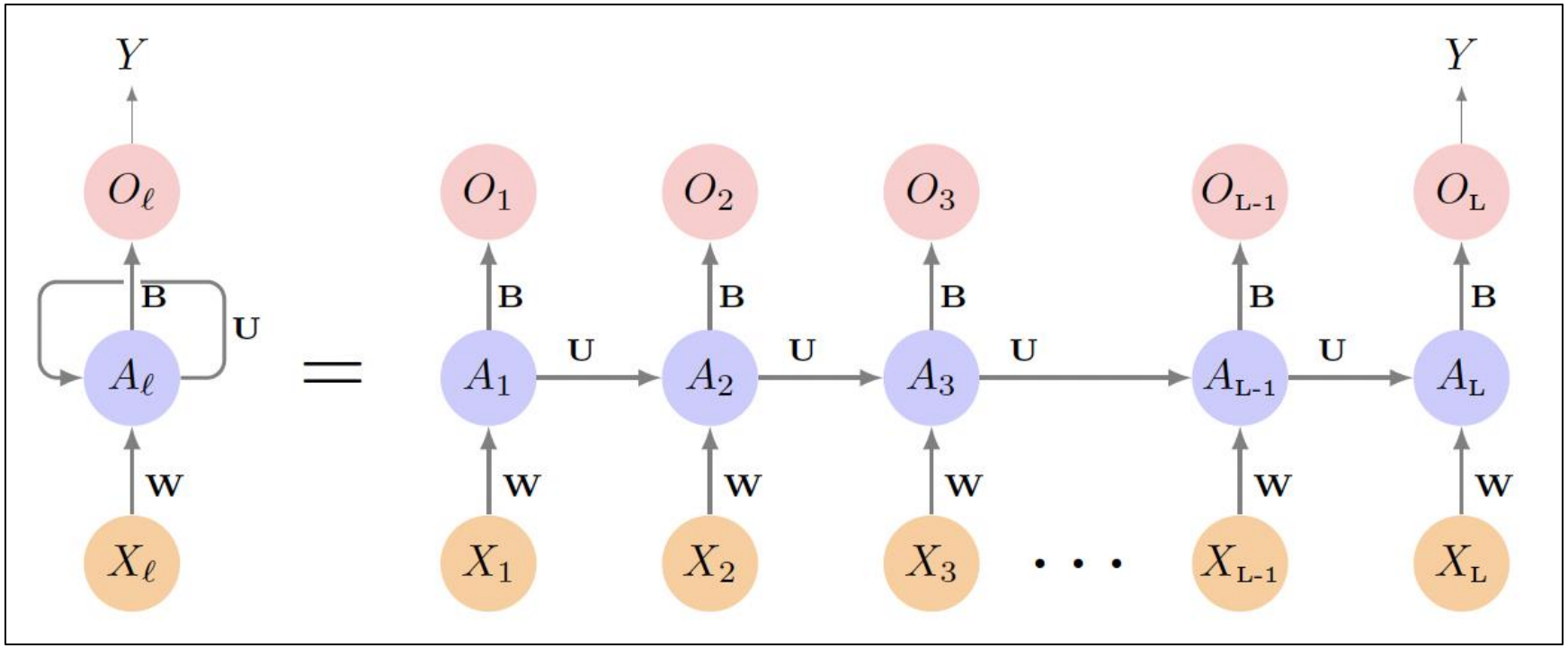
Picture Credit: ISLP

Recurrent Neural Networks: Inference

Output units

Hidden units

Input sequence

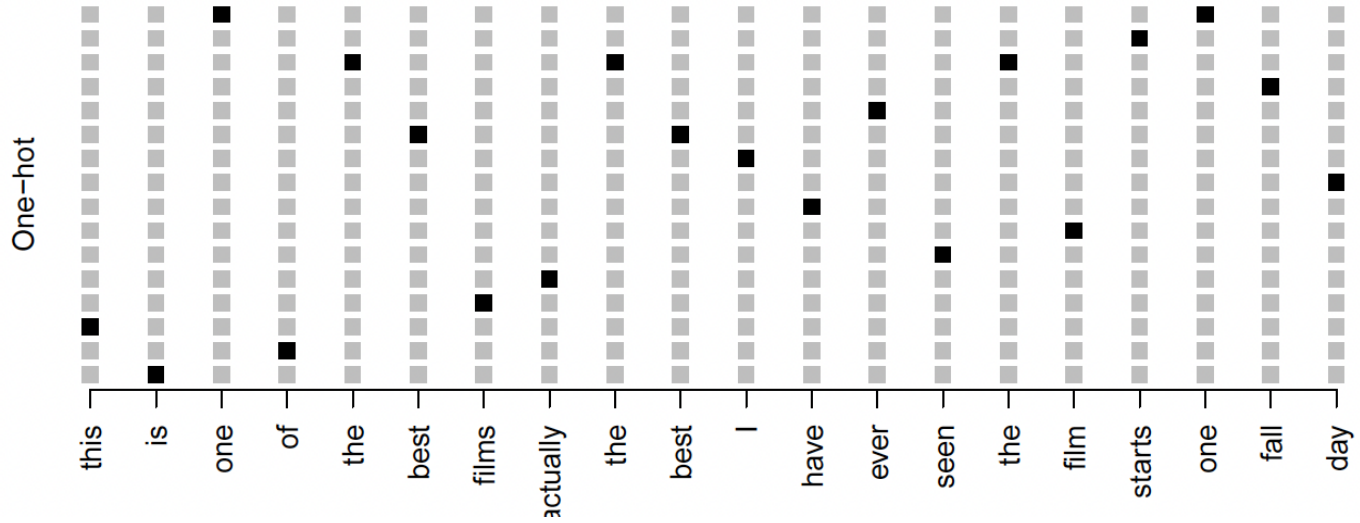


How do we evaluate if we are using RNN for regression task?

Picture Credit: ISLP

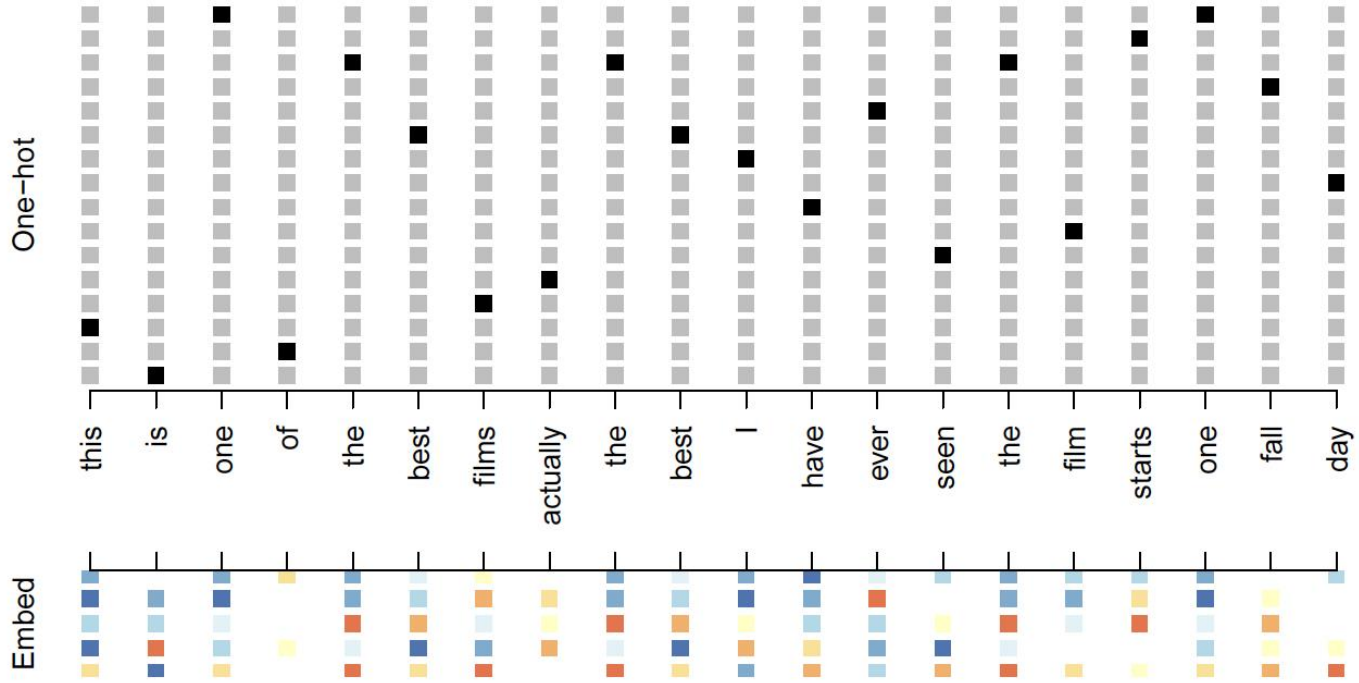
Recurrent Neural Network

- RNNs in text classification task.



Recurrent Neural Network

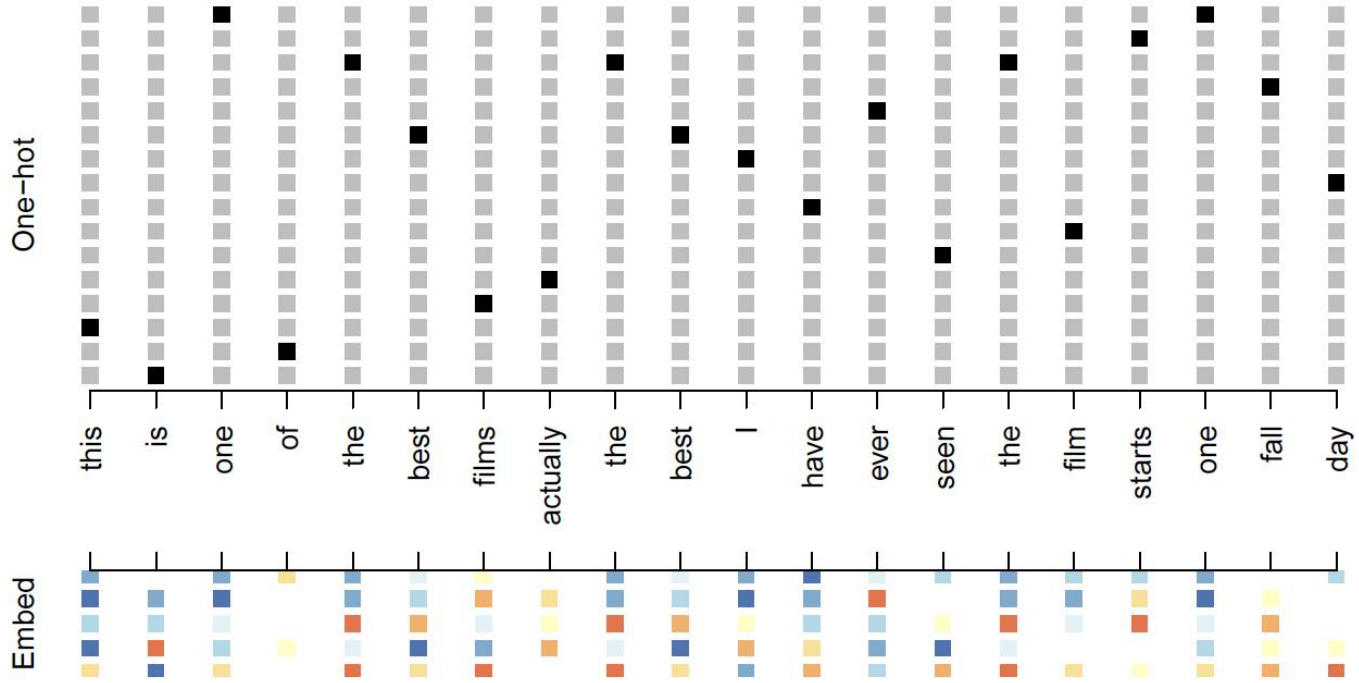
- RNNs in text classification task.



Using embedding spaces !

Recurrent Neural Network

- RNNs in text classification task.



Embedding space can be learned or can weights can be frozen.

Popular frozen weights:

Word2Vec
Glove

Recurrent Neural Network

- RNNs in text classification task.

```
word_to_ix = {"how": 0, "are": 1, "you":2}
embeds = nn.Embedding(3, 5)  # 3 words in vocab, 5 dimensional embeddings
lookup_tensor = torch.tensor([word_to_ix["how"]], dtype=torch.long)
how_embed = embeds(lookup_tensor)
print(how_embed)
```

[2] ✓ 0.0s

```
tensor([[ 0.6614,  0.2669,  0.0617,  0.6213, -0.4519]],
       grad_fn=<EmbeddingBackward0>)
```

Using embedding spaces !

https://pytorch.org/tutorials/beginner/nlp/word_embeddings_tutorial.html

Recurrent Neural Network

- RNNs in text classification task.
- GloVe is an unsupervised learning algorithm for obtaining vector representations for words.
- Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space.

GloVe: Global Vectors for Word Representation

Jeffrey Pennington, Richard Socher, Christopher D. Manning

Introduction

GloVe is an unsupervised learning algorithm for obtaining vector representations for words. Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space.

Getting started (Code download)

- Download the latest latest code (licensed under the Apache License, Version 2.0).
  Look for "Clone or download"
- Unpack the files: unzip master.zip
- Compile the source: cd GloVe-master && make
- Run the demo script: ./demo.sh
- Consult the included README for further usage details, or ask a question

Download pre-trained word vectors

- Pre-trained word vectors. This data is made available under the Public Domain Dedication and License v1.0 whose full text can be found at: http://www.opendatacommons.org/licenses/pddl/1.0/.
  - Wikipedia 2014 + Gigaword 5 (6B tokens, 400K vocab, uncased, 50d, 100d, 200d, & 300d vectors, 822 MB download): glove.6B.zip
  - Common Crawl (42B tokens, 1.9M vocab, uncased, 300d vectors, 1.75 GB download): glove.42B.300d.zip
  - Common Crawl (840B tokens, 2.2M vocab, cased, 300d vectors, 2.03 GB download): glove.840B.300d.zip
  - Twitter (2B tweets, 27B tokens, 1.2M vocab, uncased, 25d, 50d, 100d, & 200d vectors, 1.42 GB download): glove.twitter.27B.zip
- Ruby script for preprocessing Twitter data

Citing GloVe

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. [pdf] [bib]

Highlights

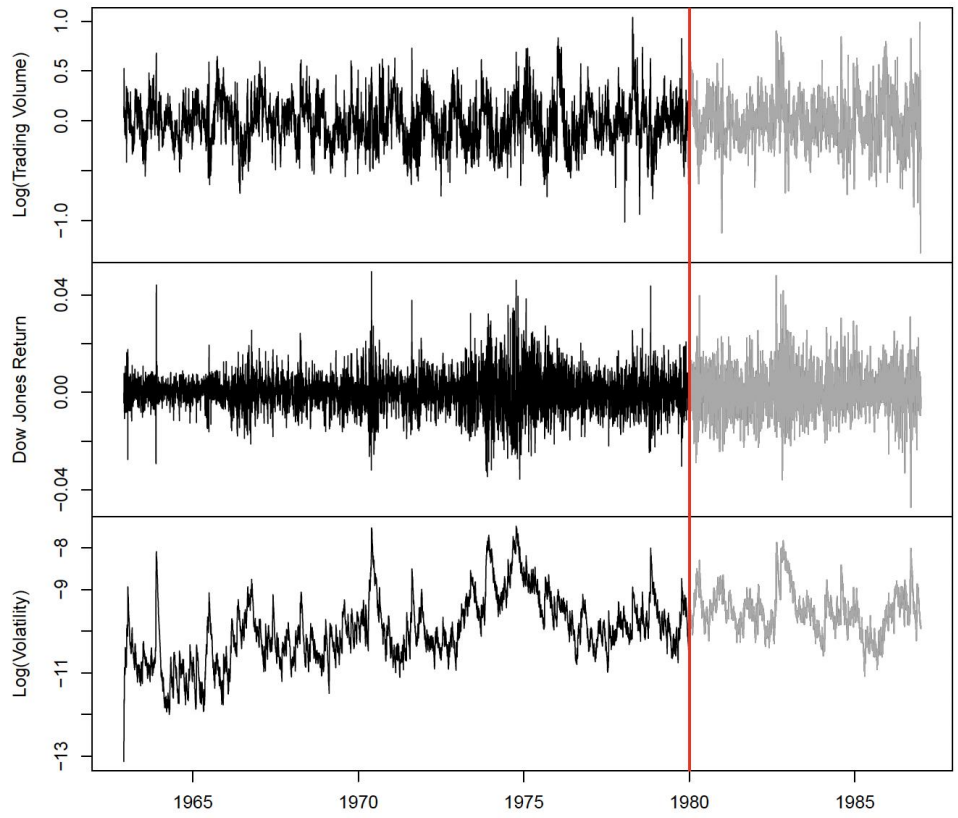
1. **Nearest neighbors**

The Euclidean distance (or cosine similarity) between two word vectors provides an effective method for measuring the linguistic or semantic similarity of the corresponding words. Sometimes, the nearest neighbors according to this metric reveal rare but relevant words that lie outside an average human's vocabulary. For example, here are the closest words to the target word *frog*:

0. *frog*
1. frogs
2. toad
3. litoria
4. leptodactylidae
5. rana
6. lizard
7. eleutherodactylus

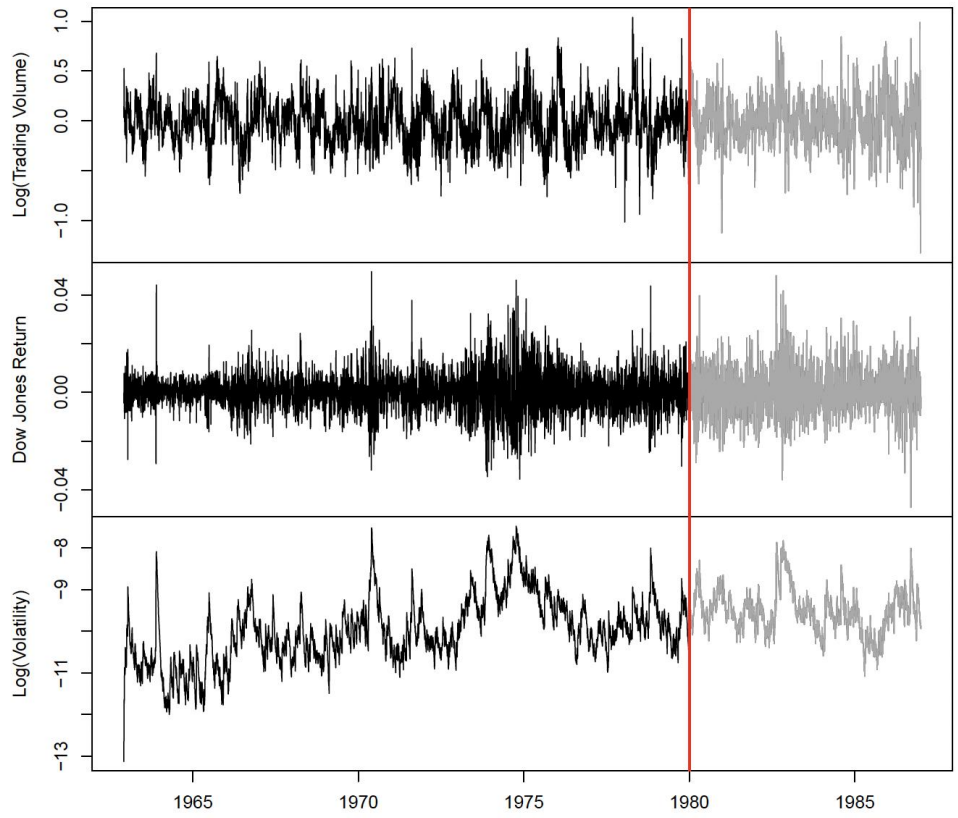
Recurrent Neural Network

- RNNs for time-series forecasting



Recurrent Neural Network

- RNNs for time-series forecasting
- Predicting trading trend (or stocks to buy and their prices) is extremely hard.
- Predicting trading volume is still something you can work with.

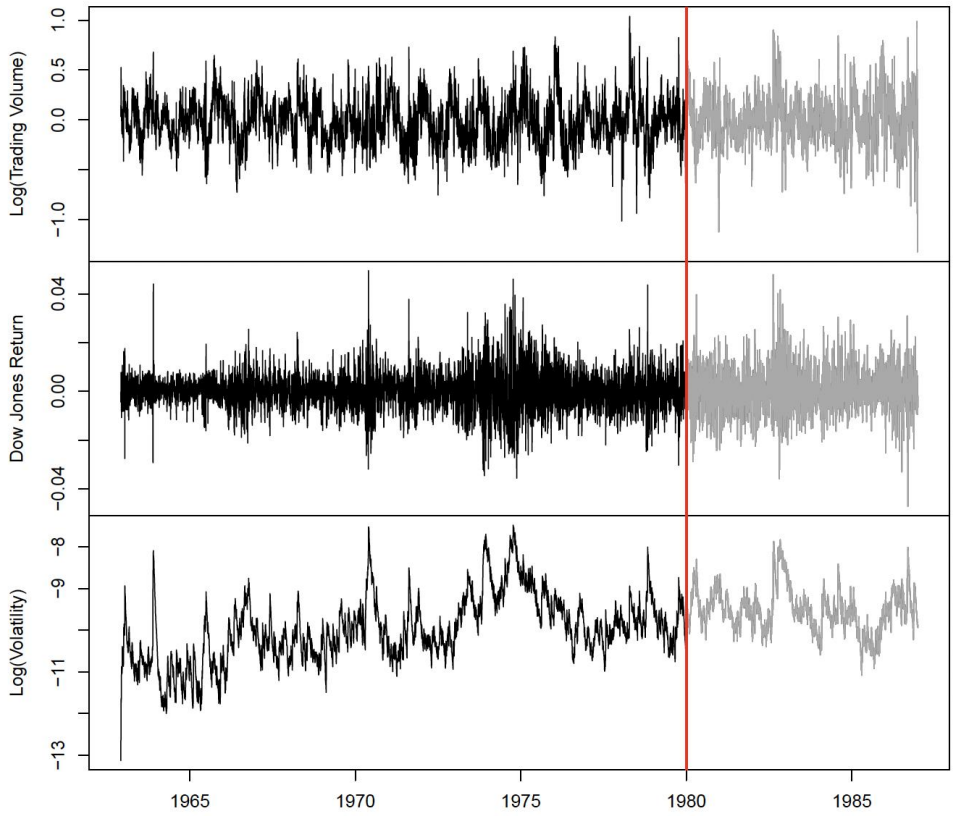


McMaster University

Recurrent Neural Network

- RNNs for time-series forecasting
- Predicting trading trend (or stocks to buy and their prices) is extremely hard.
- Predicting trading volume is still something you can work with.

Input variables: (Volume, Return, Volatility)



McMaster University

Readings

Supplemental Readings :

- ISLP: Chapter 10

Thank You

McMaster University