

2. Intro to Natural Language Processing

Natural Language Processing

- NLP Tasks
 - Classification
 - Hate speech detection
 - Spam filtering
 - Social Media drug adverse effect identification
 - Generation
 - Translation
 - Translating natural language into code
 - “Hey siri what is the weather like?”
 - Chatbots

Natural Language Processing

- Regression
 - Easy scoring
 - Predict Click Through Rate of news article
- Information Extraction
 - Who are the people mentioned in this text?
 - What date was the procedure performed?
- Document Retrieval
 - Google search
 - Automatic literature review

Natural Language Processing

- Some names we will be using throughout this lecture:
 - Token - A single “atom” of text, usually a word
 - Document - A complete datapoint of text
 - Vocabulary - A list of all possible tokens
 - Embedding Model or Tokenizer: convert a token / sentence / document into a fixed dimension vector
 - Embedding or Representation: the vector given by the embedding model

Spam Filtering

- Given a set of observations $(x_i, y_i)_{i=1:N}$ where x_i is one email, $y_i = 1$ if this email is spam otherwise 0.
- Goal is to create / learn a function f_θ such that $f_\theta(x_i) = p(y_i | x_i)$
- A two-classes classification task.

Spam Filtering

Heuristics

- If x_i contains “advertisement, price, discount”, we conclude $\hat{y}_i = 1$
- If x_i contains mostly capital letters, we conclude $\hat{y}_i = 1$
- Lack of Adaptability, Poor Generalization, etc,.

Bag of Words

- Example
 - First email: Dear Zhenghang, Good Morning!
 - Second email: Dear Zhenghang, Good Night!
 - Third email: Dear Zhenghang, Good Morning and Good Night!
 - Vocabulary: {Dear, Zhenghang, Good, Morning, Night, and}

Email	Dear	Zhenghang	Good	Morning	Night	and
1	1	1	1	1	0	0
2	1	1	1	0	1	0
3	1	1	2	1	1	1

Bag of Words

- Ignore the order information.
 - 1. “The cat chased the dog.”
 - 2. “The dog chased the cat.”
- In BoW representation with vocabulary “the, cat, chased, dog”, they have the same representation.

Word	Count
the	2
cat	1
chased	1
dog	1

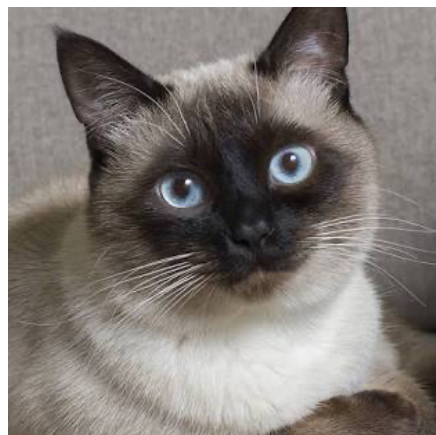
Bag of Words

- Ignore semantic information.
 - 1. “The movie was great.”
 - 2. “The film was excellent.”
 - 3. “The film was bad.”
- Vocabulary: ["the", "movie", "film", "was", "great", "excellent", “bad”]
- Compute dot product as metric for similarity:
 - $S_2 * S_1 = 2$
 - $S_2 * S_3 = 3$
 - which suggests $d(S_2, S_1) < d(S_2, S_3)$

Word	S1 (“great”)	S2 (“excellent”)	S3 (“bad”)
the	1	1	1
movie	1	0	0
film	0	1	1
was	1	1	1
great	1	0	0
excellent	0	1	0
bad	0	0	1

Better Representations

- We want an embedding model g
- We hope this model can give meaningful representations
 - $g(\text{movie}) \approx g(\text{film})$
 - $g(\text{image of Siamese}) \approx g(\text{image of Maine Coon})$



- $g(\text{the word cat}) \approx g(\text{image of a cat})$

Better Representations

- We want an embedding model g
- We hope this embedding model can be reused across different tasks.
 - $h_i = g(x_i)$, where x_i is the original data (one email for example)
 - For classification task, we want a classifier on top of h_i
 - $c_{\theta_1}(h_i) = p(x_i \text{ belongs to class 1} \mid x_i)$, where c_{θ} is the classification model.
 - For regression task (evaluate the quality of one email)
 - $r_{\theta_2}(h_i)$ = a scalar measures the quality of x_i

Word2Vec

Get meaningful representations

- Maps each word to a fixed dimension vector.
- Assumption: “Words that have similar meanings will occur in similar contexts”
- Given one document $x_i = (x_{i,1}, x_{i,2}, \dots, x_{i,M_i})$
- **Context** of size k of token $x_{i,j}$ is defined as
 - $\text{context}(x_{i,j}) = \{x_{i,j-k}, x_{i,j-(k-1)}, \dots, x_{i,j-1}, x_{i,j+1}, \dots, x_{i,j+k}\}$

Word2Vec

- Example
 - Problem Definition Much of the **focus** of queueing theory (QT) is on performance evaluation that supports comparative analytics, i.e., comparing performance measures under different intervention. However, analytical results are very sensitive to model assumptions.
- The context of size 3 of token “focus” is {Much, of, the, of, queueing, theory}

Word2Vec

- Given one document $x_i = (x_{i,1}, x_{i,2}, \dots, x_{i,M_i})$
- **Context** of size k of token $x_{i,j}$ is defined as
 - $\text{context}(x_{i,j}) = \{x_{i,j-k}, x_{i,j-(k-1)}, \dots, x_{i,j-1}, x_{i,j+1}, \dots, x_{i,j+k}\}$
- Loss function (details omitted) $-\log \sum_{i=1}^{M_i} \sum_{k \in \text{context}(x_{i,j})} p(h(x_{i,k}) \mid h(x_{i,j}))$

Word2Vec

- Properties of this Word2Vec model:
 - Words with similar semantic meaning has similar embedding vector.
 - Try it out! <https://projector.tensorflow.org/>
- Word2Vec translates words into numbers so that relationships can be captured.
 - For example, man - woman $\approx$ king - queen

Language Modeling

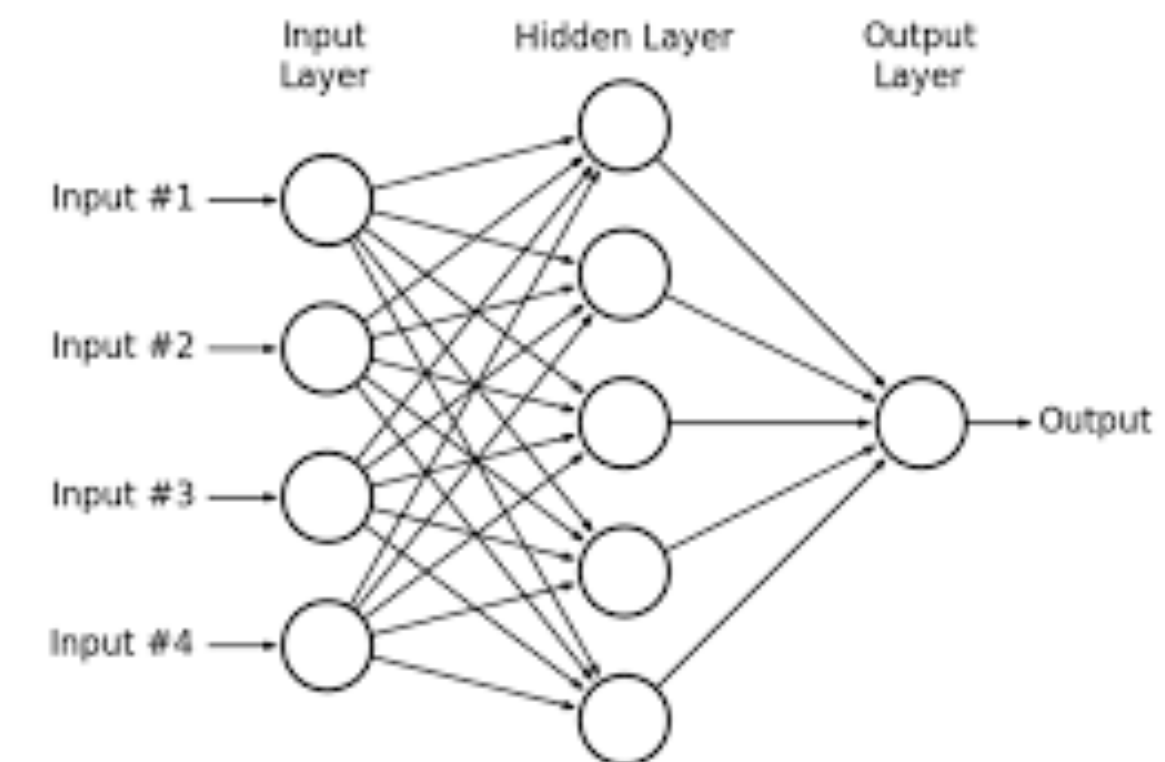
- Better option?
 - Given one sentence $x = (x_1, x_2, \dots, x_M)$
 - Assumption 1. $p(x_1, x_2, \dots, x_M) = p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_2)\cdots$
 - Next token depends only on previous token, but we are able to model the orders of tokens!

Language Modeling

- Better option?
 - Given one sentence $x = (x_1, x_2, \dots, x_M)$
 - Assumption 2.
$$p(x_1, x_2, \dots, x_M) = p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_1, x_2)p(x_4 \mid x_1, x_2, x_3)\cdots$$
 - Next token depends on all previous tokens (how ChatGPT works)

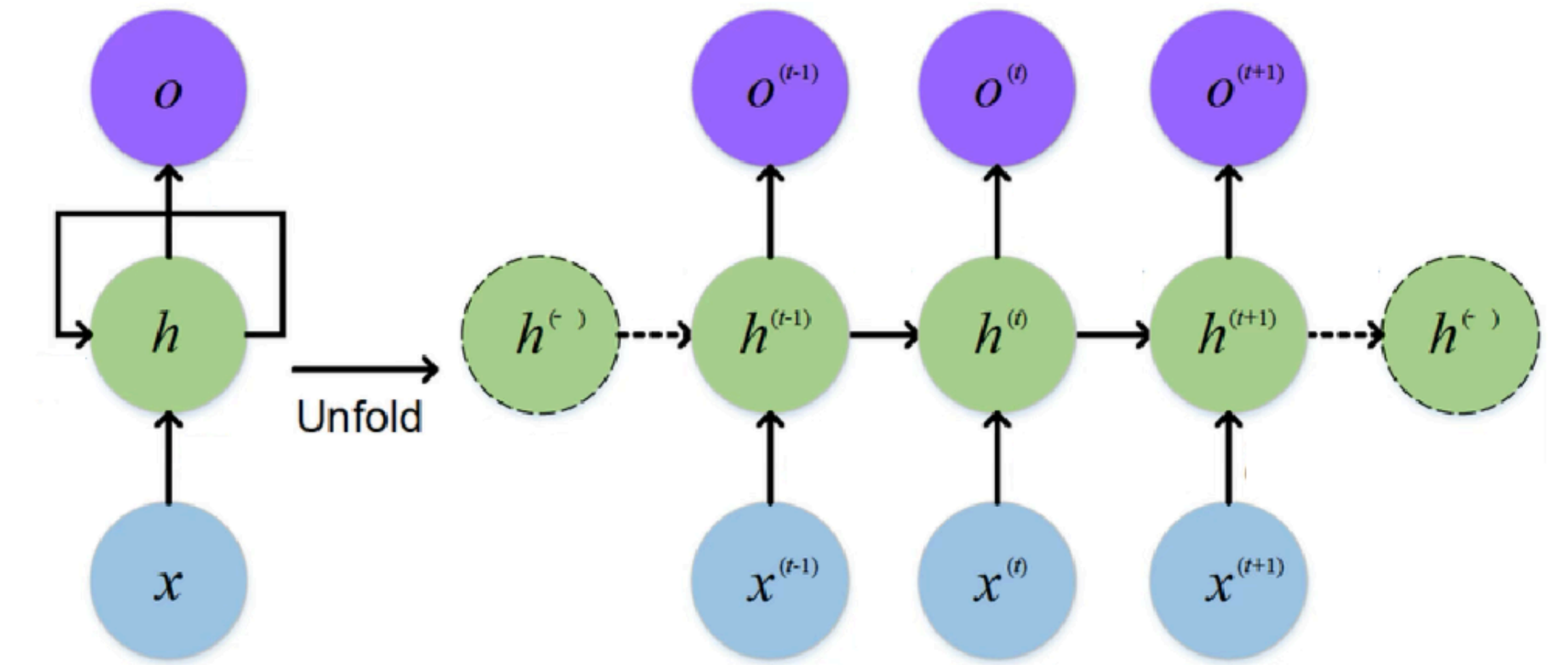
Recurrent Neural Network

- $p(x_1, x_2, \dots, x_M) = p(x_1)p(x_2 | x_1)p(x_3 | x_1, x_2)p(x_4 | x_1, x_2, x_3)\dots$
- RNN can handle inputs of different length.
 - Beyond language modeling: time series modeling / prediction / classification
- Recall Multi Layer Perceptron, can only accept fixed dimension input
 - pixel vector for one image
 - embedding for one token



Recurrent Neural Network

- Given a input sequence $x_1, x_2, \dots, x_M$
- Initialize h_0
- $h_t = f_{\theta_1}(x_t, h_{t-1})$
- $o_t = g_{\theta_2}(h_t)$
- Then we get $o_1, o_2, \dots, o_t$
- h_t is called “hidden state” and can store information from $x_1, x_2, \dots, x_t$.

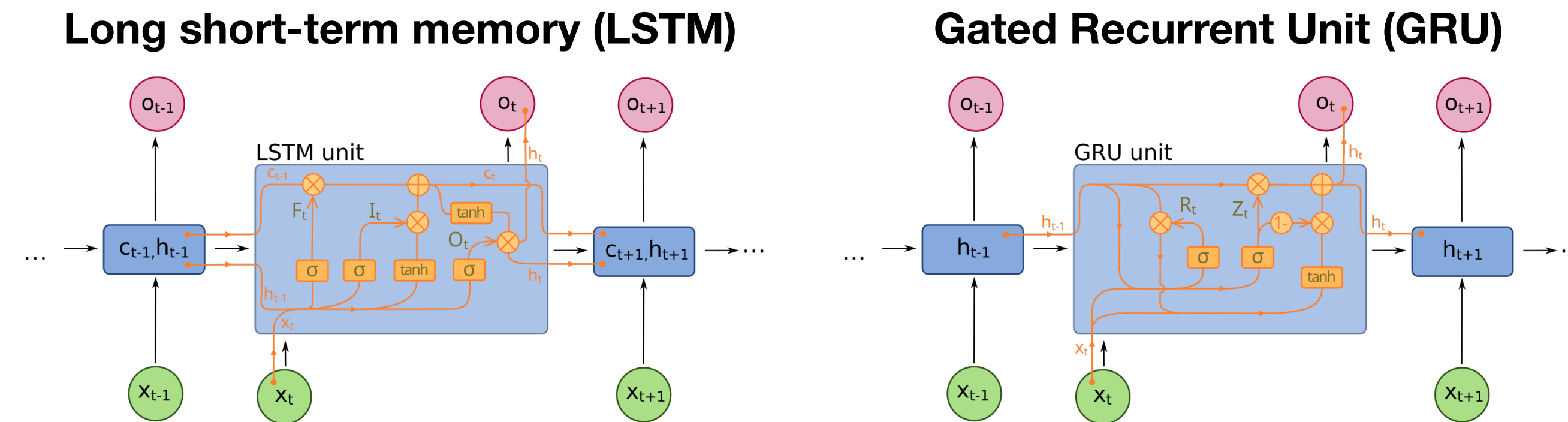


Recurrent Neural Network

- $h_t = f_{\theta_1}(x_t, h_{t-1})$
- $o_t = g_{\theta_2}(h_t)$
- Naive RNN
 - $h_t = \tanh(x_t W_{ih}^T + b_{ih} + h_{t-1} W_{hh}^T + b_{hh})$
 - $o_t = h_t$
 - Implement it!

Recurrent Neural Network

- Several types of RNN, different in the design of f_{θ_1} and g_{θ_2} . For example,



Recurrent Neural Network

For text completion

	embedding
apple	[1, 0, 0, 0]
banana	[0, 1, 0, 0]
red	[0, 0, 1, 0]
yellow	[0, 0, 0, 1]
red apple	[0, 0, 1, 0], [1, 0, 0, 0]
yellow banana	[0, 0, 0, 1], [0, 1, 0, 0]

- Convert token into vector:
 - word2vec
- One hot encoding:
 - For example, vocabulary = {apple, banana, red, yellow}
- Recall $p(x_1, x_2, \dots, x_M) = p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_1, x_2)p(x_4 \mid x_1, x_2, x_3)\dots$

- Loss function $\frac{1}{n} \sum_{t=1}^n -\log P(x_t \mid x_{t-1}, \dots, x_1)$

Reading List

- RNN https://d2l.ai/chapter_recurrent-neural-networks/index.html
- Word2Vec <https://arxiv.org/abs/1301.3781>