Introduction to Nature Language Processing

Boston University

CS 640, AI

Slides by Margrit Betke, Yiwen Gu

Outline

- What is NLP?
- Key NLP tasks
- NLP Techniques and Approaches
  - RNNs
  - Attention Mechanism & Transformers (lecture on Thursday)

What is Natural Language Processing (NLP)?

- Natural language processing (NLP) is a field of artificial intelligence and linguistics concerned with the interactions between computers and human (natural) languages.
- NPL systems convert human language into formal representations that computer programs can then manipulate (e.g., translate into another language)
  - NLP systems may handle both text and speech but work on speech recognition has evolved into a separate field. → Speech recognition
- **Goal**: Enable computers to understand, interpret, and generate human languages.

Key NLP Tasks

- Text Classification
- Named Entity Recognition (NER)
- Machine Translation
- Sentiment Analysis
- Speech Recognition
- Question Answering
- Summarization

Text Classification

- **Definition**: Assign labels to a piece of text based on its content.
- **Examples**: Spam detection, Topic classification.
- **Key Techniques**: Bag of Words, Word Embeddings, RNNs, Transformers.

Named Entity Recognition (NER)

- **Definition**: Identifying entities like names, dates, locations within text.
- **Examples**: Extracting names in articles, Identifying product names in reviews.
- **Challenges**: Ambiguity, Contextuality.

Machine Translation

- **Definition**: Automatically translating text from one language to another.
- **Examples**: Google Translate
- **Key Techniques**: Rule-based methods, Statistical MT, Neural MT (Transformer, Seq2Seq).

Sentiment Analysis

- **Definition**: Analyzing the sentiment (positive, negative, neutral) of text.
- **Examples**: Product reviews, Social media posts.
- **Challenges**: Sarcasm, context-sensitivity.

Question Answering

- **Definition**: Systems that can answer questions posed in natural language.
- **Examples**: Google Search, Chatbots.
- **Challenges**: Understanding context, dealing with ambiguous questions.

Summarization

- **Definition**: Creating a shorter version of a text while preserving the main ideas.
- **Types**: Extractive Summarization, Abstractive Summarization.
- **Challenges**: Capturing key information, Coherence.

Some Challenges

- Text Segmentation

  Some written languages like Chinese, Japanese and Thai do not have single-word boundaries, so text parsing, which requires the identification of word boundaries, becomes a non-trivial task.

Green = correct segments
Red = in lexicon, but not correct here.

There are 80 (eighty!) distinct ways to segment this sentence.
Most of them are nonsense.

Some Challenges

- Disambiguation of Meaning of Words

  Many words have more than one meaning; NPL systems select the meaning that makes the most sense in the context.

Examples:

George Bush went to Washington, D.C.

Is “Washington” a person, a place, or an organization?

Is “Bush” a person or vegetation?

Apple released IOS 18. Have you upgraded?

Is “apple” a fruit or an organization?

NLP Techniques

- **N**-grams
  - A contiguous sequence of **n** items from a given sample of text
  - Unigrams (n=1), Bigrams (n = 2), Trigrams (n=3)
- TF-IDF (Term Frequency-Inverse Document Frequency)
  - A numerical statistic that reflects the importance of a word in a document relative to a collection of documents.
  - TF: Term Freq. → how often a word appears in a document
  - IDF: Inverse Document Freq. → How common or rare a word is across all documents in the corpus.
- Part-of-Speech (POS) Tagging
  - aka. grammatical tagging or word-category disambiguation
  - Identification of words as nouns, verbs, adjectives, adverbs, etc
  - syntax parsing, understanding sentence structure
  - HMM ← later in the semester

NLP Techniques

- Word Embeddings
  - Dense vector representations of words that capture semantic meaning

Example:

king - man + woman = queen

  - Earlier works (e.g Word2Vec, GloVe, FastText):
    - **fixed** embedding per word. ~ **Static Embeddings**
  - Later works (e.g. BERT, GPT, etc):
    - embeddings **change** based on the context of the word in a sentence. ~ **Contextual Embeddings**

Recurrent Neural Networks (RNNs)

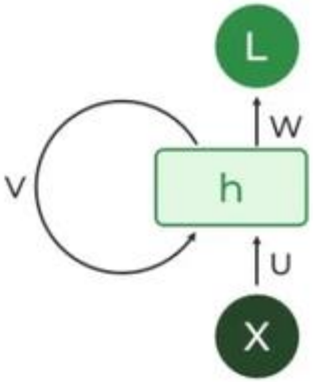
- A type of neural network designed to process sequences of data by maintaining **a hidden state** that captures information from previous steps in the sequence.
- In tasks like language modeling, text generation, and machine translation, **word order** and **context** are crucial. RNNs are effective because they process data sequentially, allowing information from earlier in the sequence to influence later outputs.
- **Sequential Data** (of various length)
- The output from the previous step is fed as input to the current step.

Generating Text with Recurrent Neural Networks

Sutskever et al., 2011

Ilya Sutskever ILYA@CS.UTORONTO.CA
James Martens JMARTENS@CS.TORONTO.EDU
Geoffrey Hinton HINTON@CS.TORONTO.EDU
University of Toronto, 6 King's College Rd., Toronto, ON M5S 3G4 CANADA

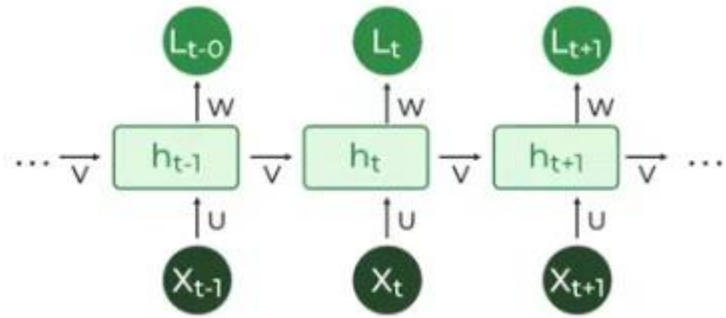
RNN Architecture



- **Input:** A sequence of tokens
- **Hidden State:** At each time step, the RNN has a hidden state, h_t, which is updated based on the current input x_t and the previous hidden state h_{t-1}
  - The hidden state acts as memory, capturing information about what has happened previously in the sequence.
  - The hidden state is computed as:

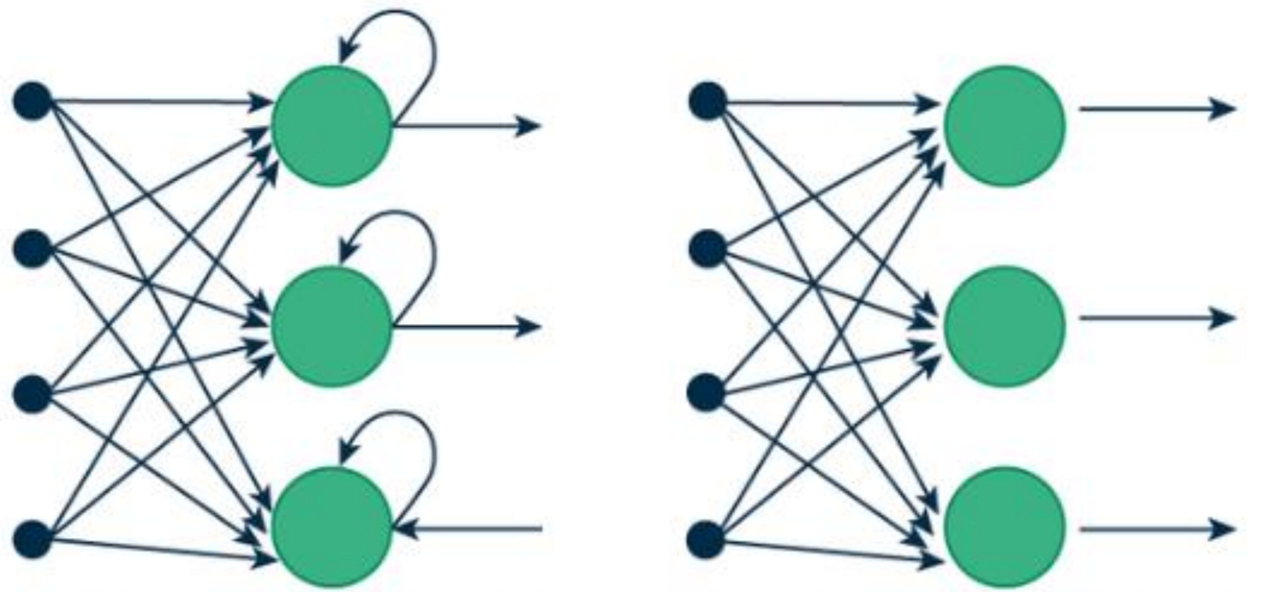
$$h_t = \sigma(W_x \cdot x_t + W_h \cdot h_{t-1})$$

- **Output:** The network can produce an output at each time step, which could be a word prediction, classification label, or some other task-specific output.



Visualization Courtesy: www.geeksforgeeks.org/

RNN vs FFN



(a) Recurrent Neural Network (b) Feed-Forward Neural Network

Backpropagation of RNN:

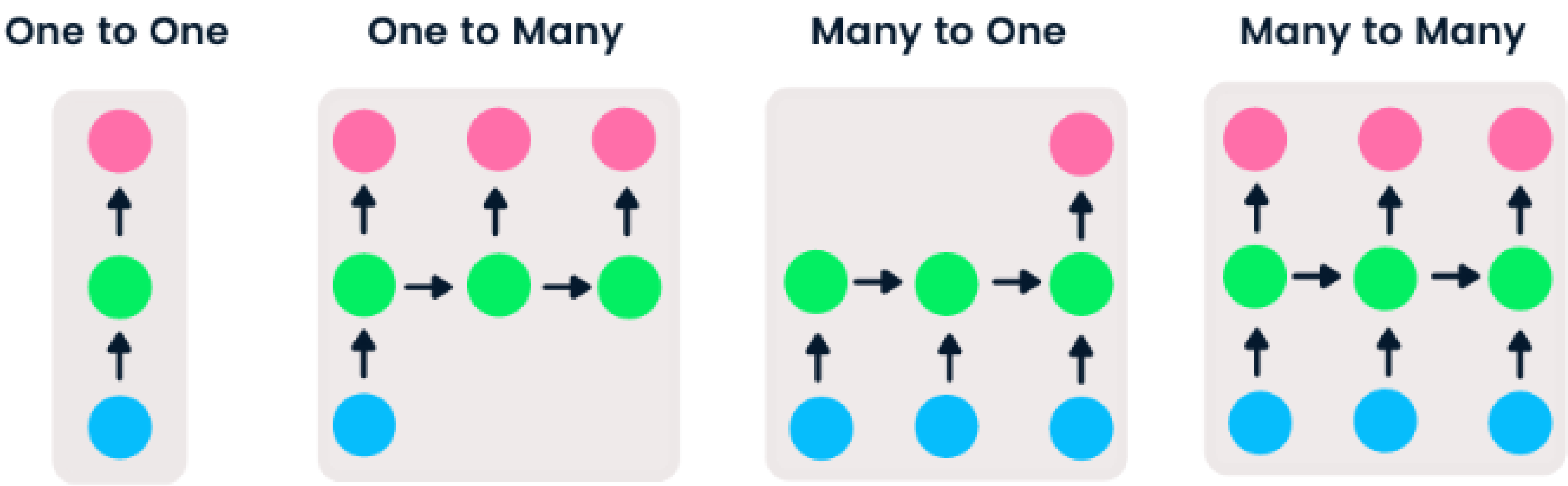
- **Loss function:** In the case of a recurrent neural network, the loss function L of all time steps is defined based on the loss at every time step as follows:

$$\boxed{\mathcal{L}(\widehat{y},y)=\sum_{t=1}^{T_y}\mathcal{L}(\widehat{y}^{<t>},y^{<t>})}$$

- **Backpropagation through time:** Backpropagation is done at each point in time. At timestep T, the derivative of the loss L with respect to weight matrix W is expressed as follows:

$$\boxed{\frac{\partial\mathcal{L}^{(T)}}{\partial W}=\sum_{t=1}^{T}\left.\frac{\partial\mathcal{L}^{(T)}}{\partial W}\right|_{(t)}}$$

RNN Types



RNN Pros & Cons

Pros

- Possibility of processing input of any length
- Model size not increasing with size of input
- Computation takes into account historical information
- **Weights are shared across time**

Cons

- Computation being slow
- Difficulty of accessing information from **a long time ago**
  - Exploding Gradient
  - Vanishing Gradient
- Cannot consider any future input for the current state

Advanced RNN:
Long Short-Term Memory (LSTM)

- LSTMs maintain a **cell state** Ct that runs through the entire sequence with only minor linear interactions, reducing the vanishing gradient problem.
- **Components:**
  - Forget Gate: Decides which information from the cell state should be discarded.
  - Input Gate: Decides which new information should be added to the cell state.
  - Output Gate: Determines the output based on the cell state

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad \text{(Forget gate)}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad \text{(Input gate)}$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \quad \text{(Candidate cell state)}$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \quad \text{(New cell state)}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad \text{(Output gate)}$$

$$h_t = o_t * \tanh(C_t) \quad \text{(Hidden state)}$$

Advanced RNN:
Gated Recurrent Unit (GRU)

- A simplified version of LSTMs that combine the forget and input gates into a single "update gate"
- **Components:**
  - Update Gate: Controls how much of the previous memory to retain.
  - Reset Gate: Controls how much of the past information to forget.

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + b_z) \quad \text{(Update gate)}$$

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + b_r) \quad \text{(Reset gate)}$$

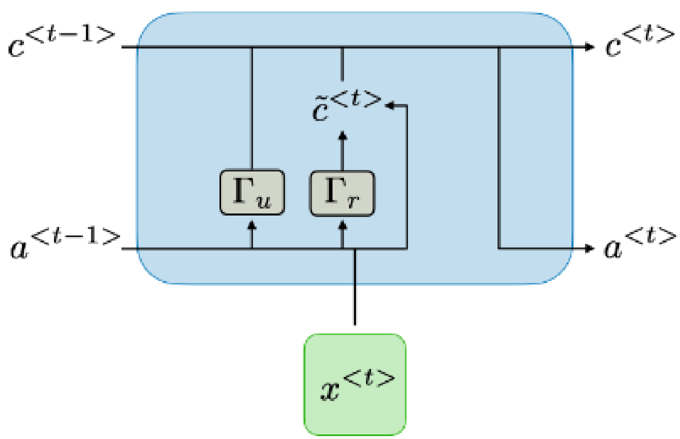
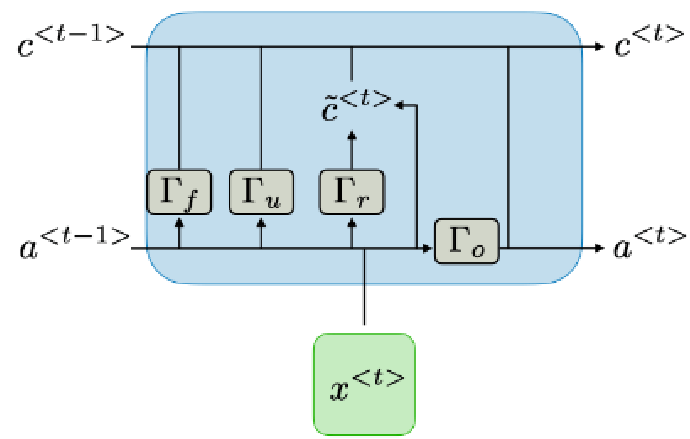
$$\tilde{h}_t = \tanh(W \cdot [r_t * h_{t-1}, x_t] + b) \quad \text{(Candidate hidden state)}$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t \quad \text{(New hidden state)}$$

GRU & LSTM

$$\boxed{\Gamma = \sigma(Wx^{<t>} + Ua^{<t-1>} + b)}$$

Type of gate	Role
Update gate Γ_u	How much past should matter now?
Relevance gate Γ_r	Drop previous information?
Forget gate Γ_f	Erase a cell or not?
Output gate Γ_o	How much to reveal of a cell?

Characterization	**Gated Recurrent Unit** (GRU)	**Long Short-Term Memory** (LSTM)
$\tilde{c}^{<t>}$	$\tanh(W_c[\Gamma_r \star a^{<t-1>}, x^{<t>}] + b_c)$	$\tanh(W_c[\Gamma_r \star a^{<t-1>}, x^{<t>}] + b_c)$
$c^{<t>}$	$\Gamma_u \star \tilde{c}^{<t>} + (1 - \Gamma_u) \star c^{<t-1>}$	$\Gamma_u \star \tilde{c}^{<t>} + \Gamma_f \star c^{<t-1>}$
$a^{<t>}$	$c^{<t>}$	$\Gamma_o \star c^{<t>}$
Dependencies		

BOSTON UNIVERSITY

Slide Courtesy: Standford CS230 -- **Different Notation Here**

Learning Outcomes

- Understand what is NLP
- Know the key tasks in NLP
- Get familiar with NLP techniques
  - N-grams, TF-IDF, POS
  - **Word Embeddings**
- Understand how **RNN** works
  - The architecture, the “recurrent” part
  - The 4 types and corresponding examples
  - Backpropagation through time (and the potential issues of exploding and vanishing gradients)
  - Pros & Cons (of the vanilla RNN)
  - LSTM and GRU

High level knowledge, e.g. know the “gates” and their differences.