Recurrent Neural Networks

Madhumitha Sakthi, Dr. Edison Thomaz - EE380L Data Mining- Spring 2020

Topics covered

- Sequence Modeling
- RNN – Introduction
- Traditional RNN Architecture
- Types of RNN and it’s usage
- RNN Variants
- BPTT
- RNN Limitations
- LSTM
- GRU
- Applications – LSTM, GRU
- Seq2Seq
- Attention Networks
- Introduction to ^^NLP^^

Sequence Modeling

- It is the task of predicting the next timestep based on the past
  - Text Autocomplete predicts the word based on letters
- $x_t \sim p(x_t \mid x_{t-1}, \ldots\ldots, x_1)$ i.e., predict the current value based on a distribution from the past
- What if we have (t-1) observations that makes it computationally expensive?
  - 1st Strategy: Assume that you need only 'm' observations from the past: makes 'm' constant and train autoregressive models
  - 2nd Strategy: Learn a summary about the past and keep updating this: $x_t = p(x_t | x_{t-1}, h_t)$ Latent Autoregressive models

RNN - Introduction

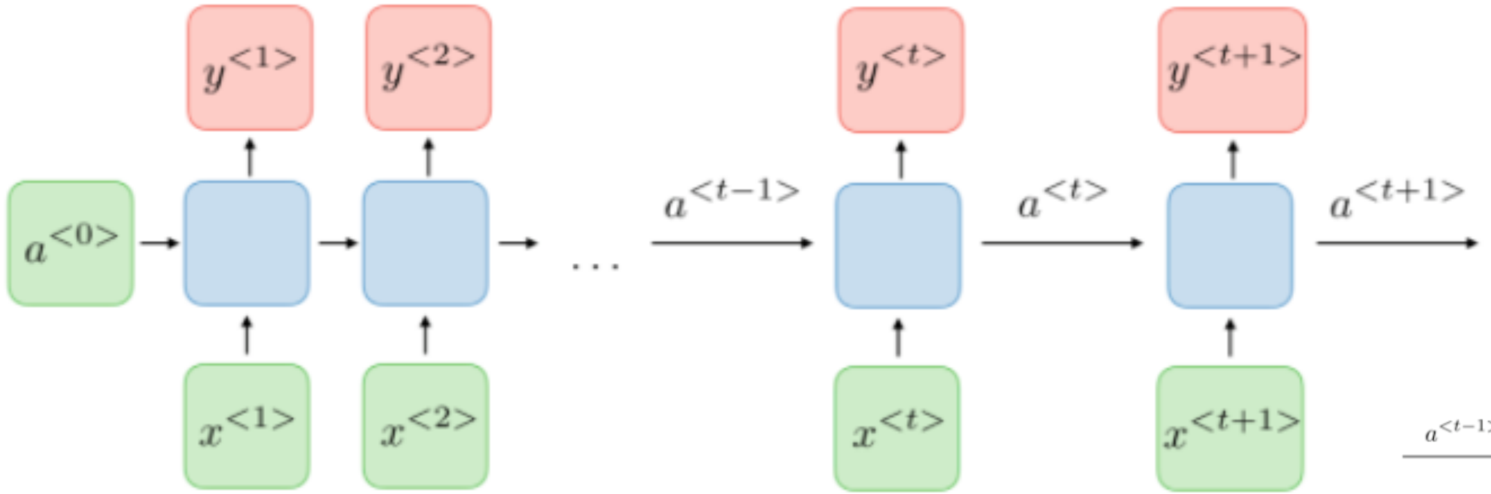
- A simple neural network for sequence modeling
- The hidden state of each step would depend on the previous step
- The hidden state h_t acts as a memory about the past
- There are various types of RNN widely used in various NLP applications
- Variants in the internal architecture comes with added computational costs

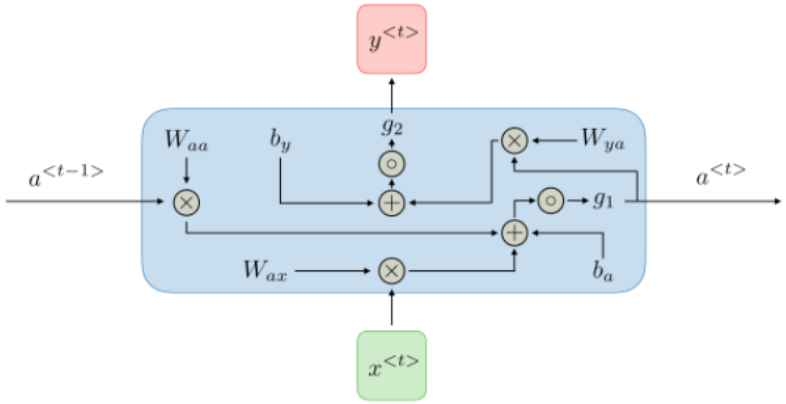
- Unlike the regular GD, we would use BPTT (BackPropagation Through Time) to update the RNN weights

Credit: Piyush Rai

Traditional RNN Architecture

Previous outputs to be used as inputs while having hidden states



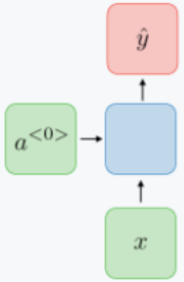
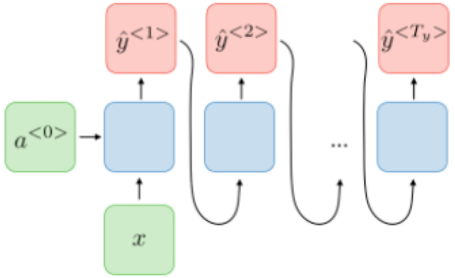
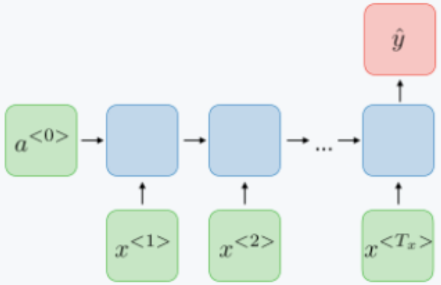
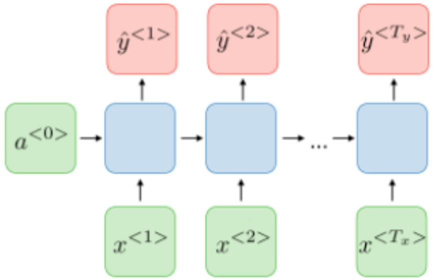
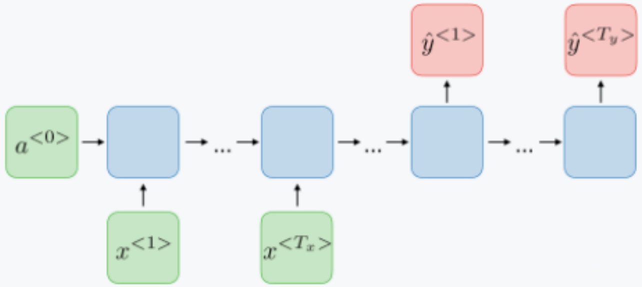


For each timestep t, the activation $a^{<t>}$ and the output $y^{<t>}$ are expressed as follows:

$$\boxed{a^{<t>} = g_1(W_{aa}a^{<t-1>} + W_{ax}x^{<t>} + b_a)} \quad \text{and} \quad \boxed{y^{<t>} = g_2(W_{ya}a^{<t>} + b_y)}$$

Credit: Amidi

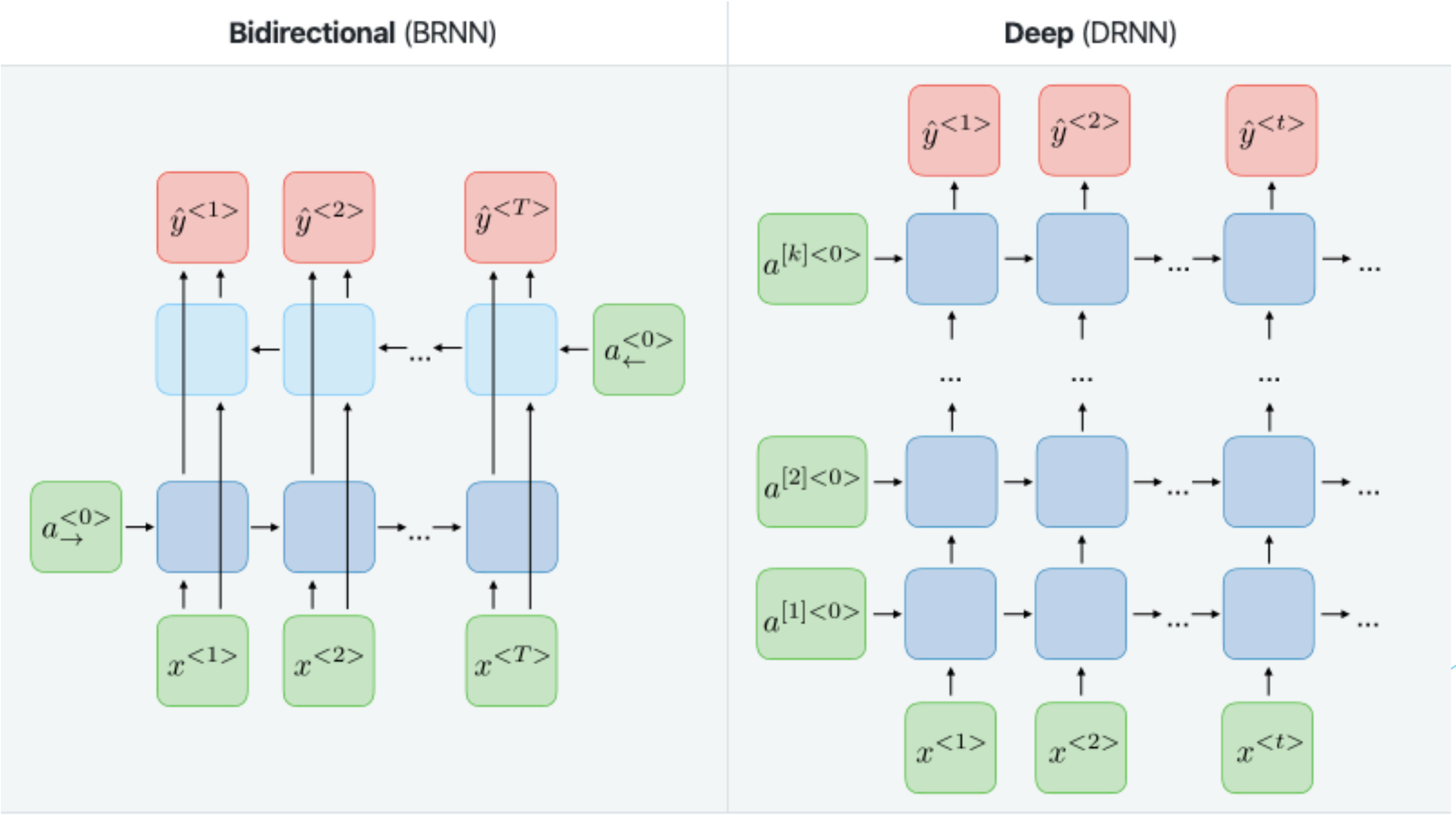
Types of RNN and it’s usage

Type of RNN	Illustration	Example
One-to-one $T_x = T_y = 1$		Traditional neural network
One-to-many $T_x = 1, T_y > 1$		Music generation
Many-to-one $T_x > 1, T_y = 1$		Sentiment classification
Many-to-many $T_x = T_y$		Name entity recognition
Many-to-many $T_x \neq T_y$		Machine translation

Credit: Amidi

RNN Variants

Two commonly used BRNN variants



Credit: Amidi

BPTT

- The hidden state depends on previous hidden states
- Cannot apply regular backpropagation on this

$$\boxed{a^{<t>} = g_1(W_{aa}a^{<t-1>} + W_{ax}x^{<t>} + b_a)} \quad \text{and} \quad \boxed{y^{<t>} = g_2(W_{ya}a^{<t>} + b_y)}$$

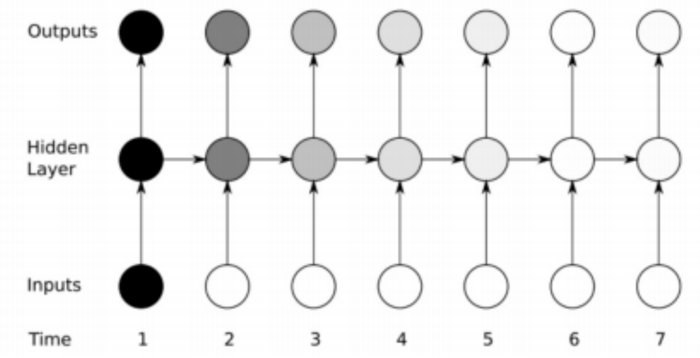
$$\boxed{\mathcal{L}(\hat{y}, y) = \sum_{t=1}^{T_y} \mathcal{L}(\hat{y}^{<t>}, y^{<t>})} \qquad \boxed{\frac{\partial \mathcal{L}^{(T)}}{\partial W} = \sum_{t=1}^{T} \left. \frac{\partial \mathcal{L}^{(T)}}{\partial W} \right|_{(t)}}$$

- Unroll in time and compute the derivative at each timestep and sum them

Credit: Amidi

RNN Limitations

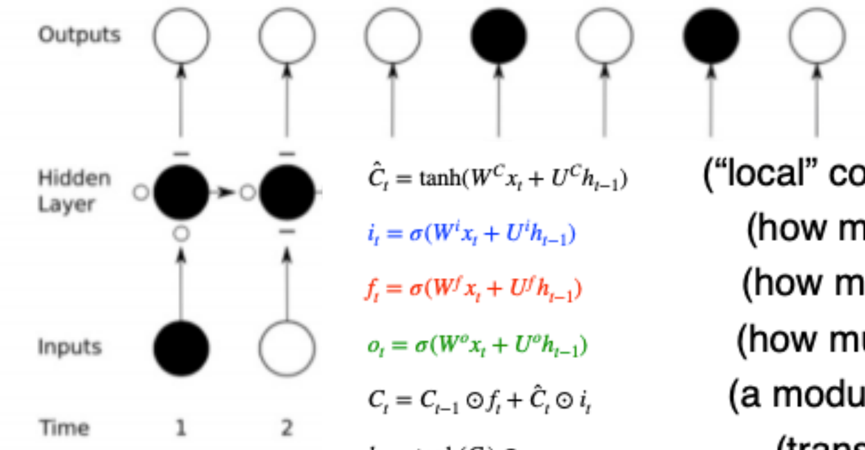
- Sensitivity of hidden states and outputs on a given input becomes weaker as we move away from it along the sequence (weak memory)
- New inputs “overwrite” the activations of previous hidden states
- Repeated multiplications can cause the gradients to vanish or explode



Credit: Piyush Rai

LSTM

- Captures long-range dependencies
- Open gate by ‘o’ and closed gate by ‘-’



$\hat{C}_t = \tanh(W^C x_t + U^C h_{t-1})$ (“local” context, only up to immediately preceding state)

$i_t = \sigma(W^i x_t + U^i h_{t-1})$ (how much to take in the local context)

$f_t = \sigma(W^f x_t + U^f h_{t-1})$ (how much to forget the previous context)

$o_t = \sigma(W^o x_t + U^o h_{t-1})$ (how much to output)

$C_t = C_{t-1} \odot f_t + \hat{C}_t \odot i_t$ (a modulated additive update for context)

$h_t = \tanh(C_t) \odot o_t$ (transform context into state and selectively output)

- State updated are now additive and not multiplicative
- State updated were multiplicative in RNNs

Credit: Piyush Rai

GRU

- The update gate
- Then the reset gate
- There is no longer a separate cell state

$$r_t = \sigma(W_r s_{t-1} + U_r x_t + b_r)$$
$$z_t = \sigma(W_z s_{t-1} + U_z x_t + b_z)$$

$$\tilde{s_t} = \phi(W(r_t \odot s_{t-1}) + U x_t + b)$$
$$s_t = z_t \odot s_{t-1} + (1 - z_t) \odot \tilde{s}_t$$

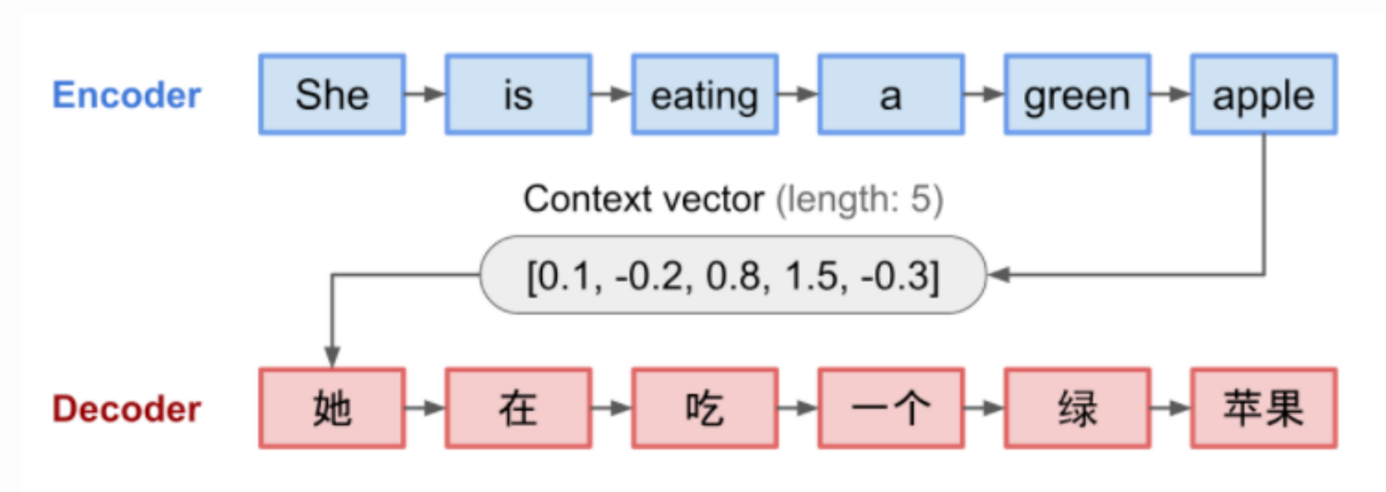
- GRU vs LSTM: GRUs are easier to train and less complex than LSTM

Applications – LSTM, GRU

- LSTM, GRU are widely used in various NLP Application
- Bi-LSTMs are used for Speech Recognition
- Machine translation – Translate text or speech from one language to another
- They are used for stock price prediction
- RNNs are used in BCI applications – Attention detection from EEG

Seq2Seq Models

- Encode-Decoder Model to predict a sequence from a sequence – Machine Translation



- This fixed-length context vector cannot remember long sequences

Credit: Lilian Wang

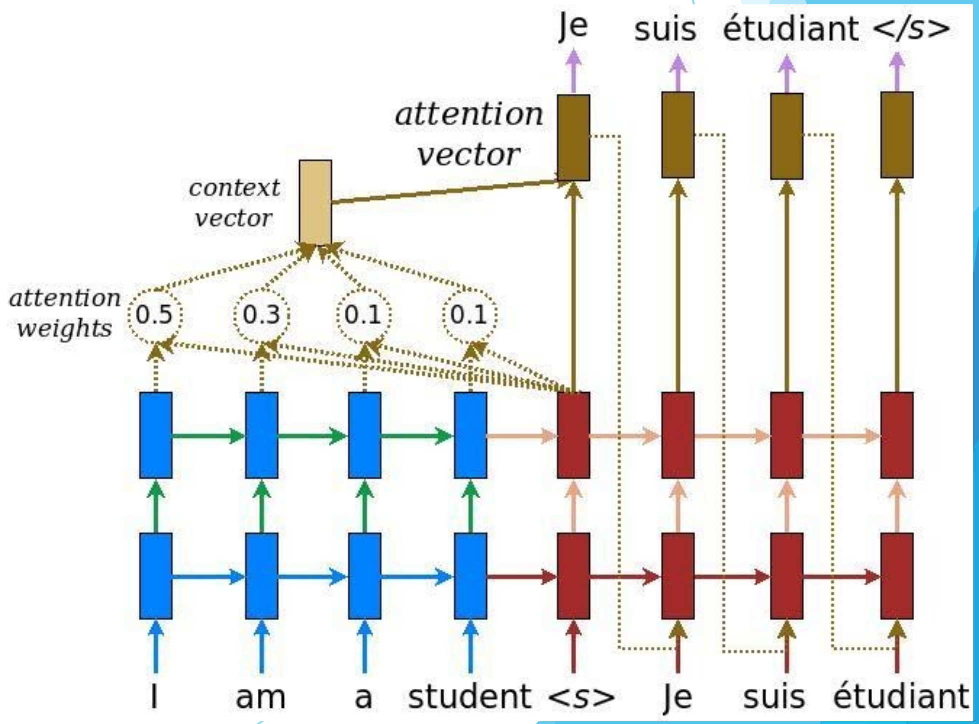
Attention Networks

- The alignment score is parametrized by a feed-forward network
- This network would be jointly trained with the other parameters

$$\mathbf{c}_t = \sum_{i=1}^{n} \alpha_{t,i} \boldsymbol{h}_i \quad \text{; Context vector for output } y_t$$

$$\alpha_{t,i} = \text{align}(y_t, x_i) \quad \text{; How well two words } y_t \text{ and } x_i \text{ are aligned.}$$

$$= \frac{\exp(\text{score}(\boldsymbol{s}_{t-1}, \boldsymbol{h}_i))}{\sum_{i'=1}^{n} \exp(\text{score}(\boldsymbol{s}_{t-1}, \boldsymbol{h}_{i'}))} \quad \text{; Softmax of some predefined alignment score..}$$

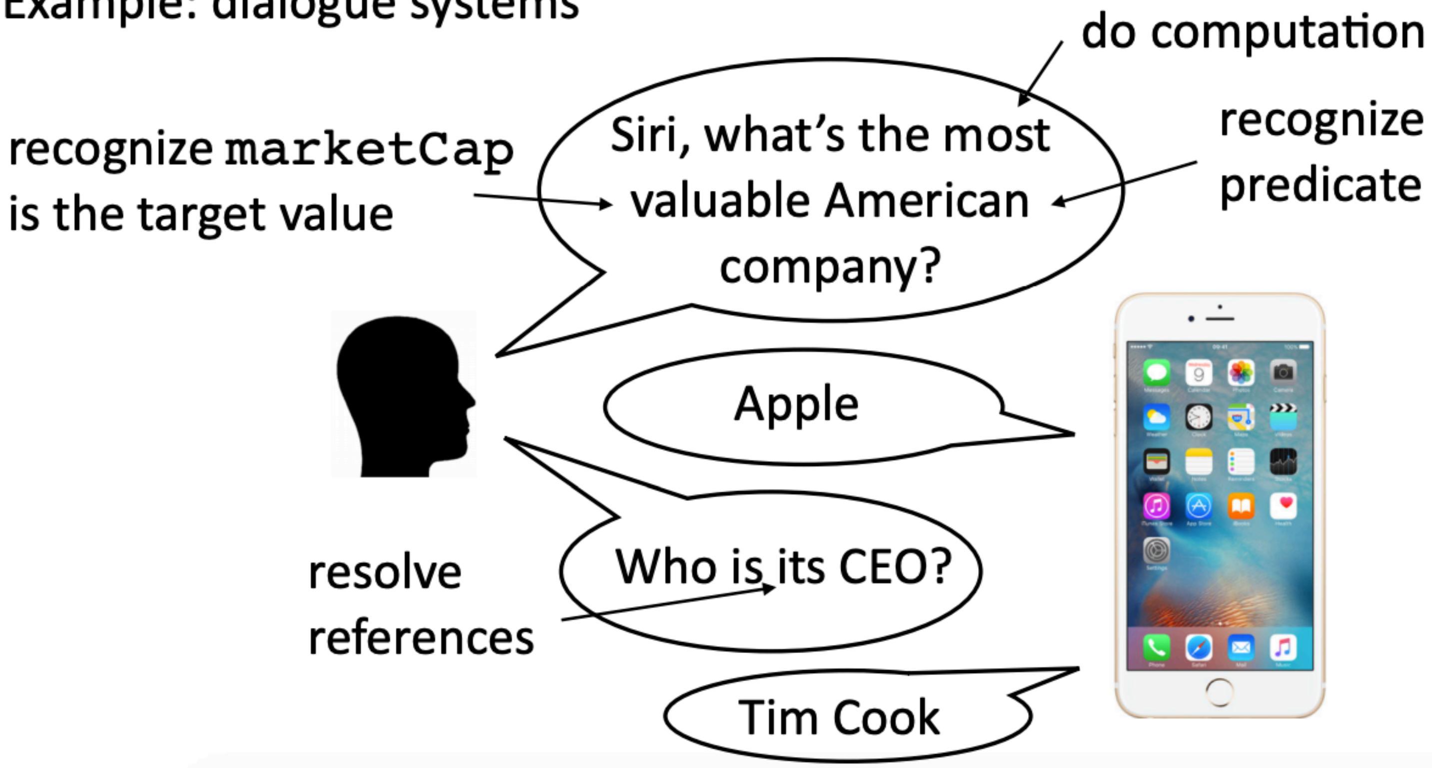


Credit: Lilian Wang

Introduction to ^^NLP^^

Be able to solve problems that require deep understanding of text

Example: dialogue systems



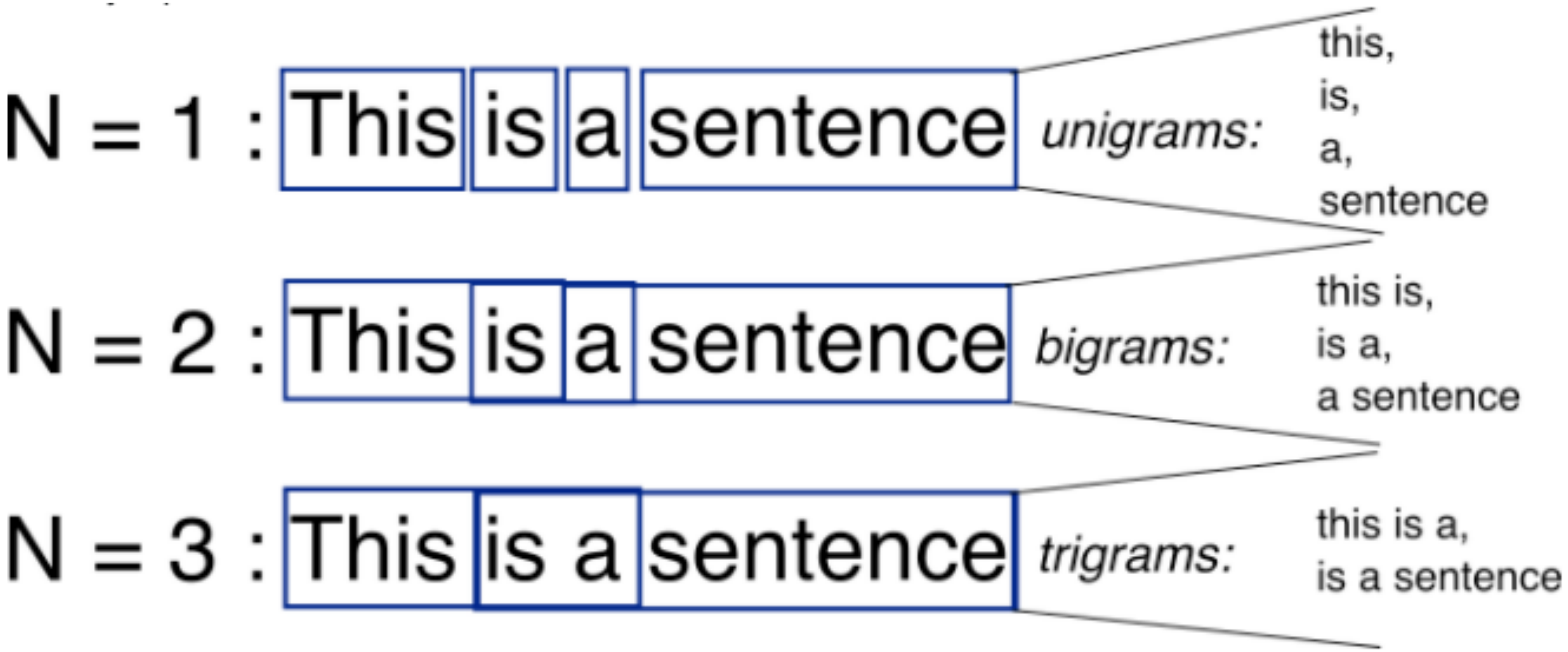
Credit: Greg Durrett

Sentiment Analysis on Text - Example

- Predict if a given sentence is positive, negative or neutral
- Text Preprocessing steps:
  - Text normalization: lower case all letters, punctuation removal, abbreviations, remove stop words, sparse words like a, an, he, is
  - Tokenization involves splitting words in a sentence as individual words
  - POS Tags: Parts of Speech tags as nouns, verbs, adjective and others
  - Stemming: Convert sentences to their root word
    - There are several types of stemming algorithms
    - There are sever type of stem algorithm

Sentiment Analysis on Text - Example

- N-gram features:



http://recognize-speech.com/language-model/n-gram-model/comparison

Sentiment Analysis on Text - Example

- Word2Vec
  - Statistical method to learn individual vectors for each word
  - The Continuous Bag-of-Words model learns the vector by predicting the current word based on its context
- Embedding layer:
  - This layer is learned jointly with the other layers
  - The size of the vector space is set to 50 or 100 or 300 dimensions
  - The input would be given as one-hot vector encoded words
  - Requires lot of training data but, can be trained to the specific NLP task data