

# Neural Models for Sequences

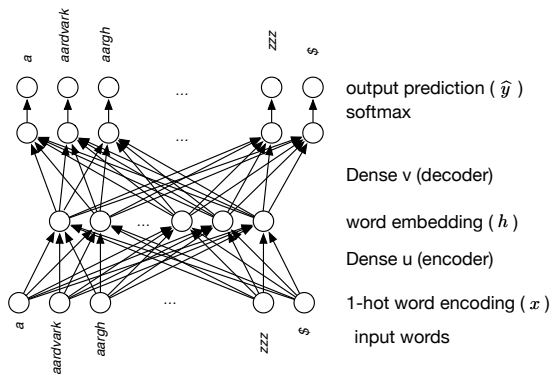
- Fully-connected networks, perhaps including convolutional layers, can handle fixed-size images and sequences.
- What about variable-length sequences?
- Sequences arise in natural language processing, biology, and any domain involving time.

# Neural Language Models

- A **corpus** is the text used for training. The corpus could be, for example, a novel, a set of news articles, all of Wikipedia, or a subset of the text on the web.
- A **token** is a sequence of characters that are grouped together, such as the characters between blanks or punctuation. The process of splitting a corpus into tokens is called **tokenization**.
- The **vocabulary**, the set of **words** that will be considered, typically including names, common phrases, slang, punctuation, and markers for the beginning and end:  $\langle start \rangle$  and  $\langle stop \rangle$ .
- Sometimes there is processing to group words into tokens, or to split words into tokens (such as the word “eating” becoming the tokens “eat” and “ing”).
- In a character-level model, the vocabulary could be the set of Unicode characters that appear in the corpus.

# Word Embeddings

Consider a model takes a single word and makes a prediction about what word appears near (e.g., following) it:



The vector of values in the hidden layer for the input word  $i$ , namely  $[u[i, 0], u[i, 1], u[i, 2], \dots]$ , is its **word embedding**.

## Simple Word Embedding Example

The text “The history of AI is a history of fantasies, possibilities, demonstrations, and promise. . .” (ignore punctuation, with  $\langle start \rangle$  as the start of a sentence) becomes the training data:

Input	Target
$\langle start \rangle$	the
the	history
history	of
of	ai
ai	is
is	a
a	history
history	of
of	fantasies

It usually works better to make predictions based on multiple surrounding words, rather than just one. The following methods use the  $k$  words before and after as a **context**:

- In the **continuous bag of words (CBOW)** model, each word in the context contributes  $n/(2 * k)$  in the one-hot encoding, where  $n$  is the number of times the word appears in the context.
- In the **Skip-gram model**, the neural network model is used for each  $(w_{i+j}, w_i)$ , for  $j \in \{-k, \dots, -1, 1, \dots, k\}$ , and the prediction of  $w_i$  is proportional to the product of each of the predictions. Thus, this assumes that each context word gives an independent prediction of word  $w_i$ .

- The embeddings resulting from these models can be added or subtracted point-wise, for example

$$\textit{Paris} - \textit{France} + \textit{Japan} \approx \textit{Tokyo}$$

where value after “ $\approx$ ” is the mode of the prediction.

- Mikolov et al [2013] trained them on a corpus of 1.6 billion words, with up to 600 hidden units.
- Some other relationships found:

$$\textit{scientist} - \textit{Einstein} + \textit{Messi} \approx \textit{midfielder}$$

$$\textit{scientist} - \textit{Einstein} + \textit{Mozart} \approx \textit{violinist}$$

$$\textit{scientist} - \textit{Einstein} + \textit{Picasso} \approx \textit{painter}$$

$$\textit{sushi} - \textit{Japan} + \textit{Germany} \approx \textit{bratwurst}$$

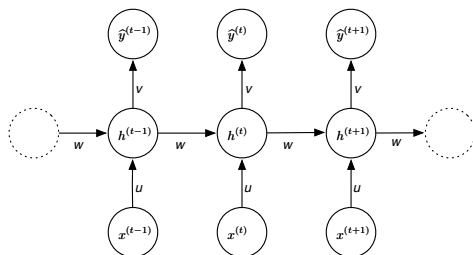
$$\textit{sushi} - \textit{Japan} + \textit{USA} \approx \textit{pizza}$$

$$\textit{sushi} - \textit{Japan} + \textit{France} \approx \textit{tapas}.$$

- There was about 60% accuracy picking the mode compared to what the authors considered to be the correct answer.

# Matched Recurrent Neural Network

A recurrent neural network with matched input–output:



- takes sequence  $x^{(0)}, x^{(1)}, x^{(2)} \dots$  and outputs  $y^{(0)}, y^{(1)}, y^{(2)} \dots$ , where  $y^{(i)}$  only depends on  $x^{(j)}$  for  $j \leq i$ .
- $h^{(t)}$  represents a **memory** or **belief state**: the information remembered from the previous times.
- A recurrent neural network represents
  - ▶ **belief state transition function**:  $x^{(t)}, h^{(t-1)} \rightarrow h^{(t)}$
  - ▶ **command function**:  $h^{(t)} \rightarrow \hat{y}^{(t)}$

# Basic Matched Recurrent Neural Network

- **belief state transition function:**  $x^{(t)}, h^{(t-1)} \rightarrow h^{(t)}$ .  
The  $i$ th component of vector  $h^{(t)}$  is

$$h^{(t)}[i] = \phi \left( b[i] + \sum_j w[i, j] * h^{(t-1)}[j] + \sum_k u[i, k] * x^{(t)}[k] \right)$$

for nonlinear activation function  $\phi$ , bias weight vector  $b$ , weight matrices  $w$  and  $u$ .

Weight vector and matrices do not depend on time,  $t$ .

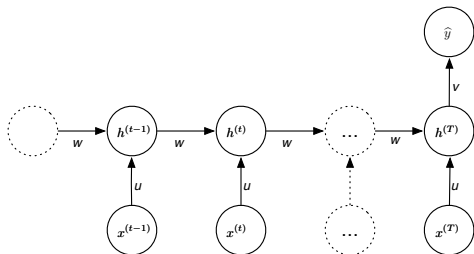
- **command function:**  $h^{(t)} \rightarrow \hat{y}^{(t)}$ .  
If the  $m$ th component of  $\hat{y}^{(t)}$  is Boolean:

$$\hat{y}^{(t)}[m] = \text{sigmoid}(b'[m] + \sum_i v[m, i] * h^{(t)}[i])$$



# Single output recurrent neural network

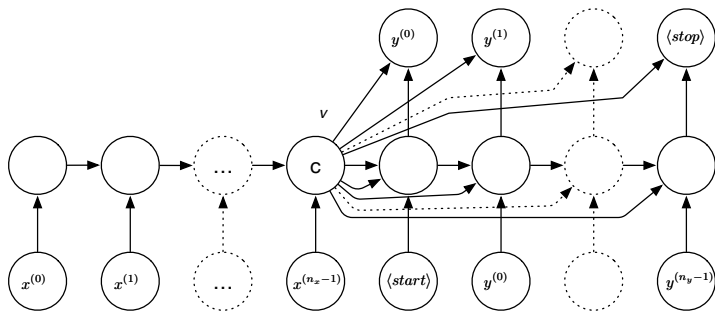
A recurrent neural network with single output after time  $T$ :



- takes sequence  $x^{(0)}, x^{(1)}, x^{(2)} \dots$  and outputs  $\hat{y}$ .
- A recurrent neural network represents
  - ▶ **belief state transition function**:  $x^{(t)}, h^{(t-1)} \rightarrow h^{(t)}$
  - ▶ **command function**:  $h^{(T)} \rightarrow \hat{y}$

# Encoder–decoder recurrent neural network

An **encoder–decoder recurrent neural network** does **sequence-to-sequence mapping**:



- $c$  is a vector representing the context for the decoder.
- The **decoder** is a **generative language model** that takes the context and emits an output sequence.
- The decoder is like the matched RNN, but with  $c$  as an input for each hidden value and each output value.

# Long short-term memory (LSTM)

- In a basic RNN, the gradients either exponentially vanish and explode.
- A **long short-term memory (LSTM)** network is a special kind of recurrent neural network designed so that the memory is maintained unless replaced by new information.
- Instead of learning a function from  $h^{(t-1)}$  to  $h^{(t)}$ , it learns the change in  $h$ , written  $\Delta h^{(t)}$ , so that  $h^{(t)} = h^{(t-1)} + \Delta h^{(t)}$ .
- The value of  $h^{(t)}$  is  $h^{(0)} + \sum_{i \leq t} \Delta h^{(i)}$ .
- The error in  $h^{(t)}$  is passed to all predecessors, and is not vanishing exponentially as it does in a traditional RNN.