Tuesday • October 19, 2021

Encoder – Decoder Networks

Yale

LING 380/780

Neural Network Models of Linguistic Structure

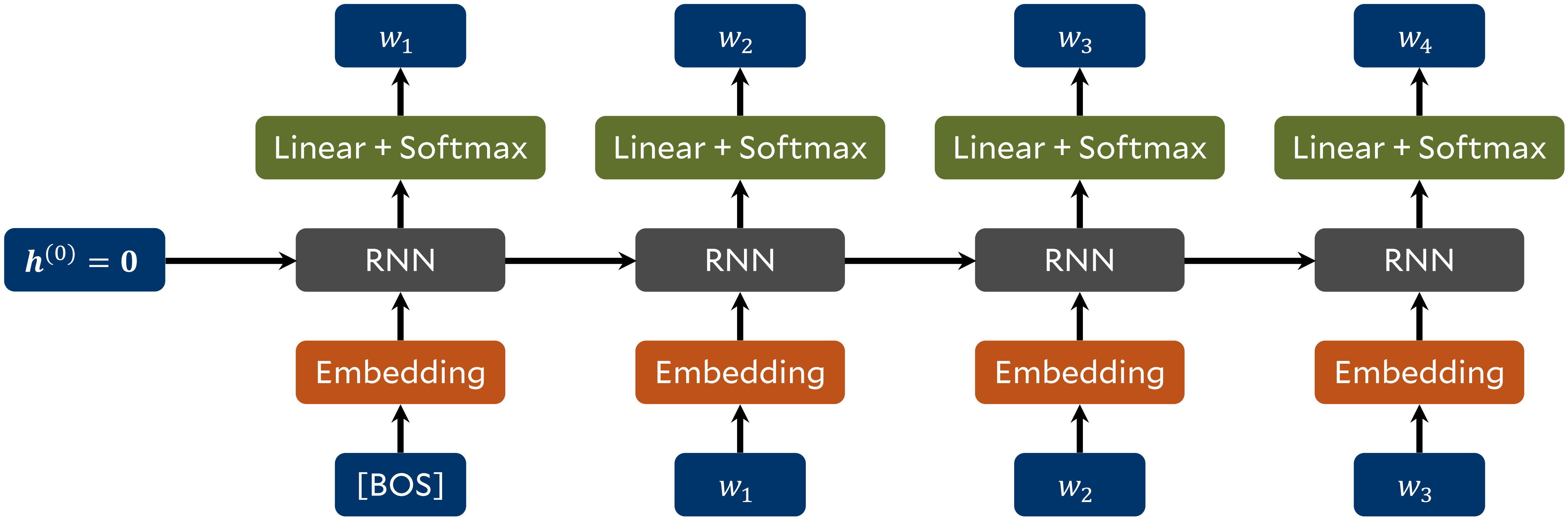
Recurrent Neural Network Applications

What kinds of tasks are RNN architectures suitable for?

- Word prediction (language modeling)
- Classification

Prediction and Generation

- Language models **generate text**.



Greedy Generation Algorithm

- Initialize $\boldsymbol{h} \leftarrow \mathbf{0}$, $w =$ [BOS].
- While $w \neq$ [EOS]:
  - Feed w and $\boldsymbol{h}$ into the RNN.
  - Let $\boldsymbol{h}$ be the new RNN state.
  - Let w be the word assigned the highest probability.

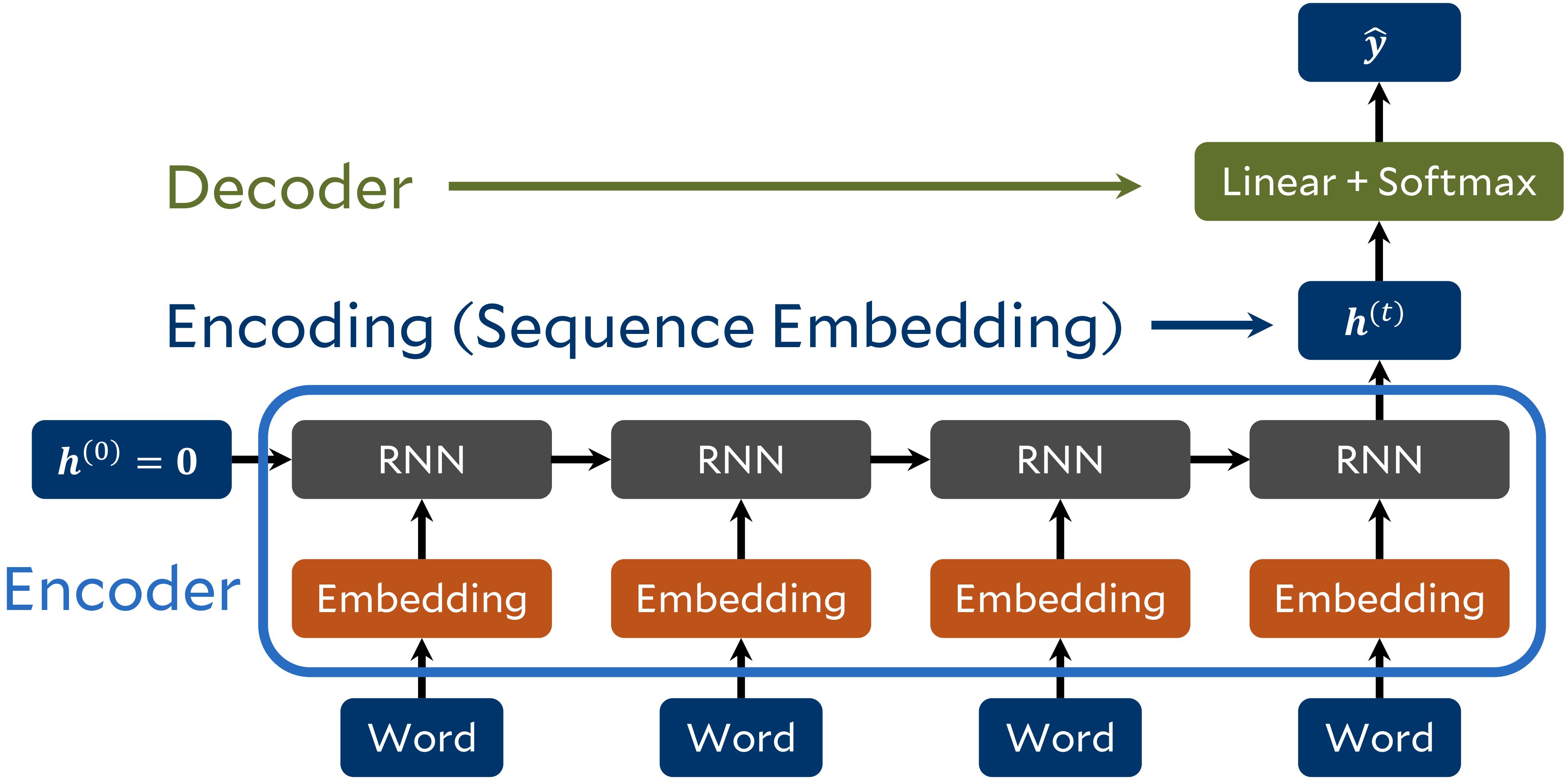
Greedy Generation Algorithm

- "Greedy" means that at every step, the locally best choice is made.
- The next word is always the most likely word.
- But this doesn't necessarily give the best sequence!

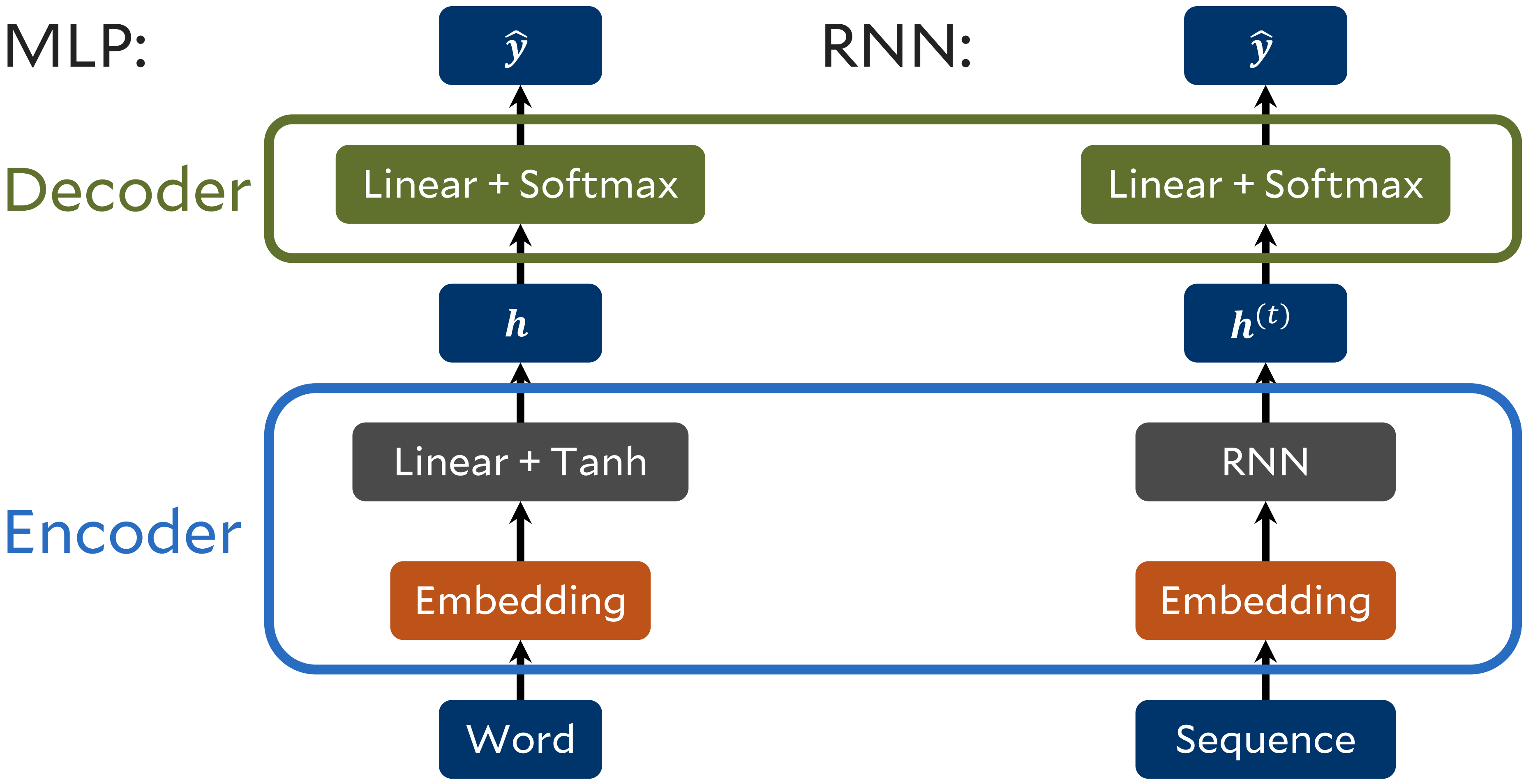
Beam Search Generation Algorithm

- Initialize $s^{(i)} = $ [BOS] for $i \in \{1, 2, \dots, k\}$.
- While none of the $s^{(i)}$s end with [EOS]:
  - For each i, let $s^{(i,j)}$ be the sequence obtained by adding the jth most likely next token to $s^{(i)}$.
  - Let $s^{(1)}, s^{(2)}, \dots, s^{(k)}$ be the k most likely sequence among the $s^{(i,j)}$s.

RNN Classification



MLP vs. RNN Classifiers



Types of RNNs

- **RNN Generators:** Autoregressively generate text from $\boldsymbol{h}^{(0)} = \mathbf{0}$ and $w_0 =$ [BOS].
- **RNN Classifiers:** “Encode” an embedding for the sequence, then “decode” it using a linear + softmax.
- **Sequence-to-Sequence RNNs:** A third type of RNN!

Sequence-to-Sequence (Seq2Seq) Tasks

Sequence-to-sequence tasks (seq2seq) are ones that take a **sequence** as input and **produce a sequence** as output.

Examples:

- Machine translation
- Text summarization
- Question answering

Sequence-to-Sequence (Seq2Seq) Tasks

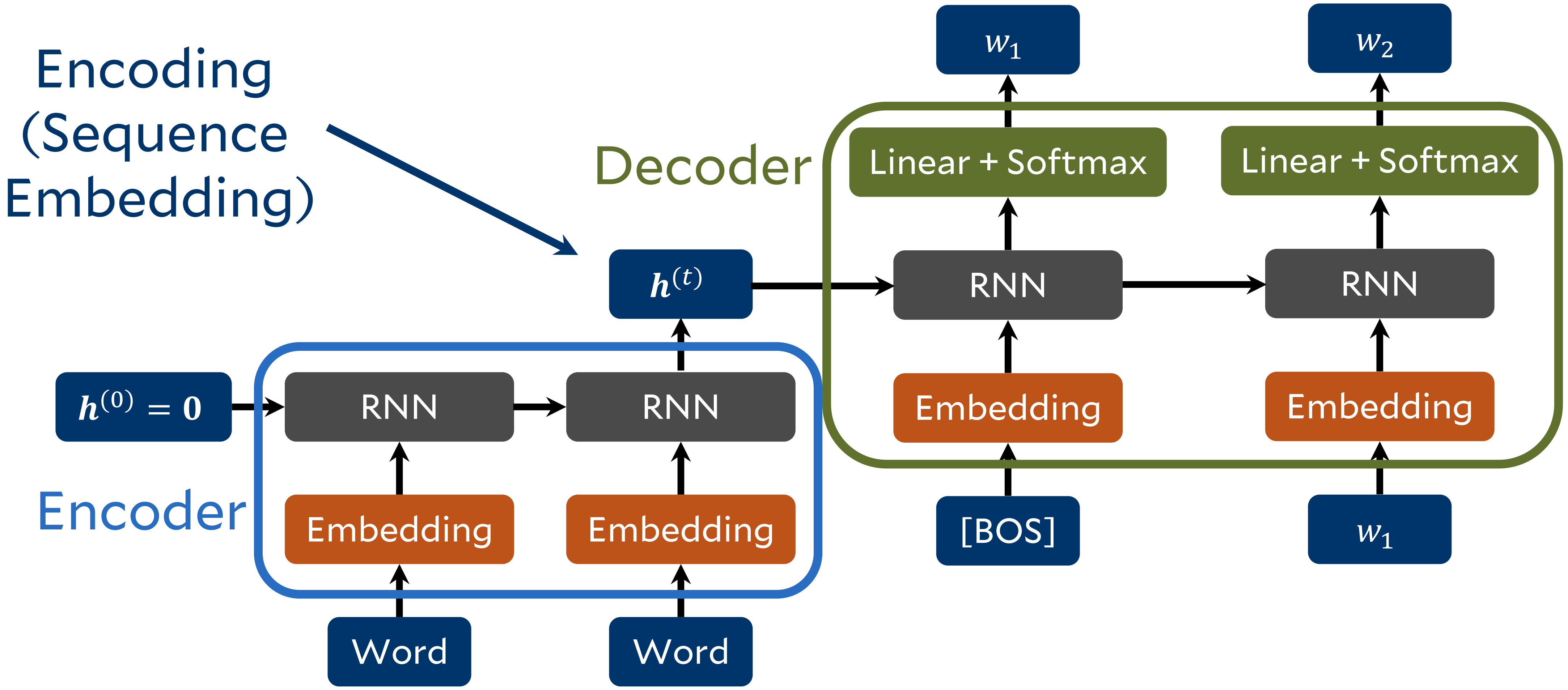
How do you apply RNNs to seq2seq tasks?

- **Classification:** RNN encoder, 1-layer MLP decoder
- **Seq2Seq:** RNN encoder, autoregressive RNN decoder

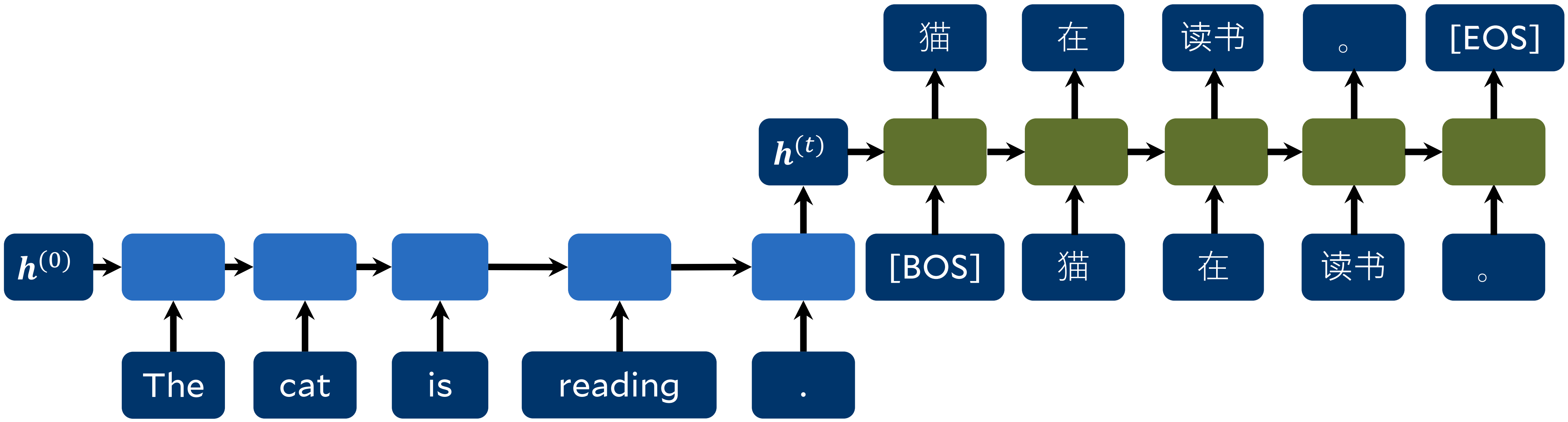
Types of Neural Networks

	Single Input	Sequence Input
Single Output	MLP	RNN Classifier
Sequence Output	RNN Generator	RNN Encoder–Decoder

Encoder–Decoder Architecture



Example: Machine Translation



Problem

- In the encoder–decoder architecture, all information about the sequence must be stored in the encoding vector.
- If the text is long, the vector might be too small to store the entire sequence.
- **Solution:** allow the decoder to peek at the input sequence!

窥视

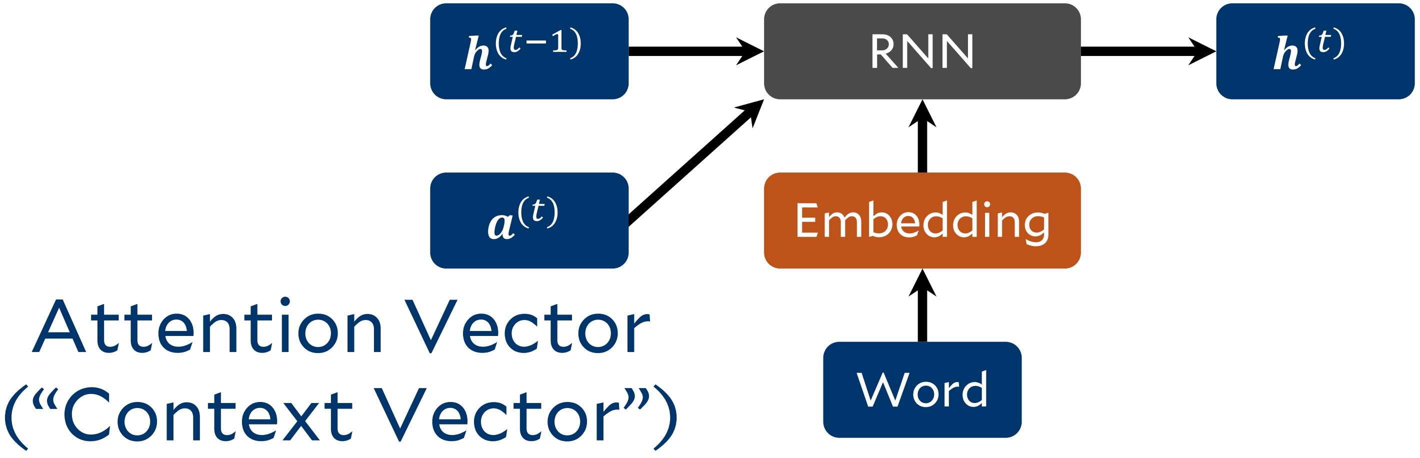
Attention

Two methods for incorporating the input sequence in the decoder:

- Bahdanau attention
- Luong attention

Bahdanau Attention

Add a third input to the RNN decoder unit:



Bahdanau Attention

- At each decoding step, the decoder looks at (“attends to”) parts of the encoder output and retrieves an attention vector $\boldsymbol{a}^{(t)}$.

$$\mathrm{Linear}\left(\boldsymbol{x}^{(t)}, \boldsymbol{h}^{(t-1)}, \boldsymbol{a}^{(t)}\right) = \boldsymbol{W}\begin{bmatrix} \boldsymbol{x}^{(t)} \\ \boldsymbol{h}^{(t-1)} \\ \boldsymbol{a}^{(t)} \end{bmatrix} + \boldsymbol{b}$$

$$= \boldsymbol{W}^{(x)}\boldsymbol{x}^{(t)} + \boldsymbol{W}^{(h)}\boldsymbol{h}^{(t-1)} + \boldsymbol{W}^{(a)}\boldsymbol{a}^{(t)} + \boldsymbol{b}$$

SRN with Attention

$$\boldsymbol{h}^{(t)} = \tanh(\boldsymbol{W}^{(x)}\boldsymbol{x}^{(t)} + \boldsymbol{W}^{(h)}\boldsymbol{h}^{(t-1)} + \boldsymbol{W}^{(a)}\boldsymbol{a}^{(t)} + \boldsymbol{b})$$

LSTM with Attention

$$\boldsymbol{f}^{(t)} = \sigma(\boldsymbol{W}^{(f,x)}\boldsymbol{x}^{(t)} + \boldsymbol{W}^{(f,h)}\boldsymbol{h}^{(t-1)} + \boldsymbol{W}^{(f,a)}\boldsymbol{a}^{(t)} + \boldsymbol{b}^{(f)})$$

$$\boldsymbol{i}^{(t)} = \sigma(\boldsymbol{W}^{(i,x)}\boldsymbol{x}^{(t)} + \boldsymbol{W}^{(i,h)}\boldsymbol{h}^{(t-1)} + \boldsymbol{W}^{(i,a)}\boldsymbol{a}^{(t)} + \boldsymbol{b}^{(i)})$$

$$\boldsymbol{o}^{(t)} = \sigma(\boldsymbol{W}^{(o,x)}\boldsymbol{x}^{(t)} + \boldsymbol{W}^{(o,h)}\boldsymbol{h}^{(t-1)} + \boldsymbol{W}^{(o,a)}\boldsymbol{a}^{(t)} + \boldsymbol{b}^{(o)})$$

$$\tilde{\boldsymbol{c}}^{(t)} = \tanh(\boldsymbol{W}^{(c,x)}\boldsymbol{x}^{(t)} + \boldsymbol{W}^{(c,h)}\boldsymbol{h}^{(t-1)} + \boldsymbol{W}^{(c,a)}\boldsymbol{a}^{(t)} + \boldsymbol{b}^{(c)})$$

$$\boldsymbol{c}^{(t)} = \boldsymbol{f}^{(t)} \odot \boldsymbol{c}^{(t-1)} + \boldsymbol{i}^{(t)} \odot \tilde{\boldsymbol{c}}^{(t)}$$

$$\boldsymbol{h}^{(t)} = \boldsymbol{o}^{(t)} \odot \boldsymbol{c}^{(t)}$$

GRU with Attention

$$\boldsymbol{r}^{(t)} = \sigma(\boldsymbol{W}^{(r,x)}\boldsymbol{x}^{(t)} + \boldsymbol{W}^{(r,h)}\boldsymbol{h}^{(t-1)} + \boldsymbol{W}^{(r,a)}\boldsymbol{a}^{(t)} + \boldsymbol{b}^{(r)})$$

$$\boldsymbol{z}^{(t)} = \sigma(\boldsymbol{W}^{(z,x)}\boldsymbol{x}^{(t)} + \boldsymbol{W}^{(z,h)}\boldsymbol{h}^{(t-1)} + \boldsymbol{W}^{(z,a)}\boldsymbol{a}^{(t)} + \boldsymbol{b}^{(z)}))$$

$$\widetilde{\boldsymbol{h}}^{(t)} = \tanh(\boldsymbol{W}^{(h,x)}\boldsymbol{x}^{(t)} + \boldsymbol{W}^{(h,a)}\boldsymbol{a}^{(t)} + \boldsymbol{b}^{(h)}$$
$$+ \boldsymbol{r}^{(t)} \odot (\boldsymbol{W}^{(r,h)}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}^{(h,r)}))$$

$$\boldsymbol{h}^{(t)} = \boldsymbol{z}^{(t)} \odot \boldsymbol{h}^{(t-1)} + (1 - \boldsymbol{z}^{(t)}) \odot \widetilde{\boldsymbol{h}}^{(t)}$$

Computing the Attention Vector

First, use an MLP (2 layers, tanh activation, no softmax) to calculate an **attention score**

$$s_i^{(t)} = \mathrm{MLP}\left(\begin{bmatrix} \boldsymbol{h}^{(e,i)} \\ \boldsymbol{h}^{(d,t-1)} \end{bmatrix}\right)$$

where

- $\boldsymbol{h}^{(e,i)}$ is the ith encoder hidden state
- $\boldsymbol{h}^{(d,t)}$ is the $(t-1)$st decoder hidden state

Computing the Attention Vector

Then, convert the attention scores into **attention weights**:

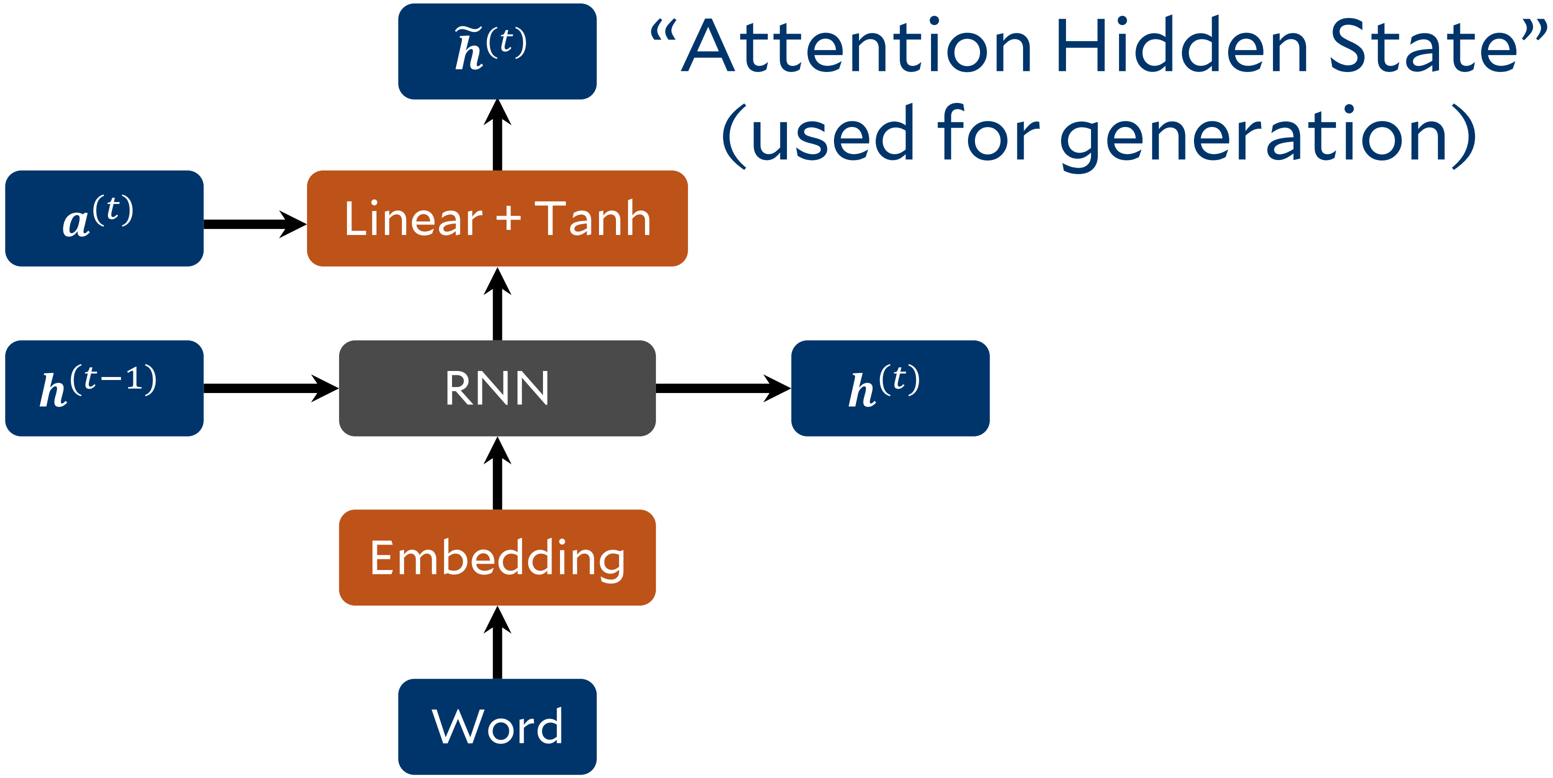
$$\boldsymbol{\alpha}^{(t)} = \mathrm{softmax}(\boldsymbol{s}^{(t)})$$

Use the attention weights to take a weighted average of encoder hidden states:

$$\boldsymbol{a}^{(t)} = \sum_i \alpha_i^{(t)} \boldsymbol{h}^{(e,i)}$$

Luong Attention

Add the attention vector to the RNN decoder hidden state:



Luong Attention

Computing attention scores

- "Dot": $s_i^{(t)} = \left(\boldsymbol{h}^{(d,t)}\right)^\top \boldsymbol{h}^{(e,i)}$
- "General": $s_i^{(t)} = \left(\boldsymbol{h}^{(d,t)}\right)^\top \boldsymbol{W}\boldsymbol{h}^{(e,i)}$
- "Concat" (similar to Bahdanau): $s_i^{(t)} = \mathrm{MLP}\left(\begin{bmatrix} \boldsymbol{h}^{(e,i)} \\ \boldsymbol{h}^{(d,t)} \end{bmatrix}\right)$

Visualizing Attention Weights

Bahdanau et al. (2015):
English to French

The
agreement
on
the
zone
economic
european
has
been
signed
in
August
1992