IX. Deep Learning

- Elements of Deep Learning
- Convolutional Neural Networks
- Autoencoder Networks
- Recurrent Neural Networks
- Long-Term Dependencies
- RNNs for Machine Translation
- Attention Mechanism
- Transformer
- Transformer Language Models
- Pretraining and Finetuning
RNNs for Machine Translation

Statistical Machine Translation (SMT)

- Rule-based MT
  - direct
  - transfer-based
  - interlingua-based

- Example-based MT

- Statistical MT
  - word-based
  - syntax-based
  - phrase-based

- Neural MT
  - LSTM
  - RNN
RNNs for Machine Translation

Statistical Machine Translation (SMT) (continued)

- Rule-based MT
  - direct
  - transfer-based
  - interlingua-based

- Example-based MT

- Statistical MT
  - word-based
  - syntax-based
  - phrase-based

- Neural MT
  - LSTM
  - RNN

Timeline:
- 1950
- 1980
- 1990
- 2015
The “Noisy Channel” Model applied to SMT:

Learn from a parallel corpus $D$ a probabilistic model, $P(Y \mid X)$, which can be used to decode the channel input (the target sentence $y$, e.g. in German) from the channel output (the source sentence $x$ in a foreign language (e.g., English)).
RNNs for Machine Translation

Statistical Machine Translation (SMT) (continued)

ich liebe meine katze

i love my cat

Translation
RNNs for Machine Translation
Statistical Machine Translation (SMT) (continued)

\[ p(\text{"ich jage eine katze"} | \text{"i love my cat"}) \]
\[ p(\text{"ich habe keine katze"} | \text{"i love my cat"}) \]
\[ \vdots \]
\[ p(\text{"ich liebe meine katze"} | \text{"i love my cat"}) \]
\[ p(\text{German\_target\_sentence} | \text{English\_source\_sentence}) \]
\[ p(\text{sentence\_in\_own\_language} | \text{sentence\_in\_foreign\_language}) \]
\[ p(y | x) \]
RNNs for Machine Translation

Statistical Machine Translation (SMT) (continued)

Task: Given a sentence \( x \) in a foreign language (here: English), what is the most probable translation \( y \) in our own language (here: German)?

\[
p(y \mid x) \rightarrow \text{max}
\]
Remarks:

- Here, $x$ denotes a sentence $x_1x_2 \ldots x_{|x|}$. Likewise, $y$ denotes a sentence $y_1y_2 \ldots y_{|y|}$. The $x_t$ and $y_t$ denote the words at position (at time point) $t$ in the respective sentences.

- Noisy Channel Model I. When the (German) sentence $y$ was transmitted over a noisy channel, it got corrupted and came out as sentence $x$ in a foreign language (English). The task is to recover the original sentence, i.e., to decode (= translate) the English (source) into German (target).

- Noisy Channel Model II. We can observe only the sentence $x$, and we ask ourselves which sentence $y$ might have induced $x$? Among the candidates for $y$ we search the most probable sentence, which we then consider as translation of $x$.

  I.e., the Noisy Channel Model does not take sentence $y$ and looks for a translation $x$ (= varies $x$), but takes “the condition” $x$ as given and varies among the $y$.

Tackling this translation task with coupled RNNs (= **Neural Machine Translation**) reflects this view: Conditioned by the hidden vector encoding of $x$, denoted as $y^e(T^e)$ in the **figure**, the decoder has to generate the most probable sentence $y$. 
RNNs for Machine Translation
Statistical Machine Translation (SMT) (continued)

Based on a parallel corpus $D$, the best translation $y$ of a sentence $x$ given in the foreign language maximizes under $D$ the probability $p(y \mid x)$:

$$\arg\max_y p(y \mid x) = \arg\max_y p(x \mid y) \cdot p(y)$$

$$P(Y \mid X) = \frac{P(X \mid Y) \cdot P(Y)}{P(X)}$$

$X \equiv X=x$, $x \equiv$ English sentence

$Y \equiv Y=y$, $y \equiv$ German sentence
RNNs for Machine Translation
Statistical Machine Translation (SMT) (continued)

Based on a parallel corpus $D$, the best translation $y$ of a sentence $x$ given in the foreign language maximizes under $D$ the probability $p(y \mid x)$:

$$\arg\max_y p(y \mid x) = \arg\max_y p(x \mid y) \cdot p(y) \iff$$

$$P(Y \mid X) = \frac{P(X \mid Y) \cdot P(Y)}{P(X)}$$

$x \triangleq X=x$, $x \triangleq$ English sentence

$Y \triangleq Y=y$, $y \triangleq$ German sentence

1. $p(y)$ is called “language model” and takes care of the fluency in the target language. It is modeled as $p(y_1, \ldots, y_m) = \prod_{i=1}^{m} p(y_i \mid y_{i-(n-1)}, \ldots, y_{i-1})$. Training data are (monolingual) corpora in the target language.

2. $p(x \mid y)$ is called “translation model” and captures the translation fidelity between two languages. It is modeled as $p(x, a \mid y)$, where “$a$” is a vector of alignment features. Training data are bilingual corpora.

3. $\arg\max_y$ is called “decoder” and operationalizes the search for the maximization problem. Keyword: beam search
RNNs for Machine Translation
Statistical Machine Translation (SMT) (continued)

Based on a parallel corpus $D$, the best translation $y$ of a sentence $x$ given in the foreign language maximizes under $D$ the probability $p(y \mid x)$:

$$\text{argmax}_y p(y \mid x) = \text{argmax}_y \left( p(x \mid y) \cdot p(y) \right) \iff P(Y \mid X) = \frac{P(X \mid Y) \cdot P(Y)}{P(X)}$$

$X \equiv X=x, \quad x \equiv \text{English sentence}$

$Y \equiv Y=y, \quad y \equiv \text{German sentence}$

1. $p(y)$ is called “language model” and takes care of the fluency in the target language. It is modeled as $p(y_1, \ldots, y_m) = \prod_{i=1}^{m} p(y_i \mid y_{i-(n-1)}, \ldots, y_{i-1})$. Training data are (monolingual) corpora in the target language.

2. $p(x \mid y)$ is called “translation model” and captures the translation fidelity between two languages. It is modeled as $p(x, a \mid y)$, where “$a$” is a vector of alignment features. Training data are bilingual corpora.

3. $\text{argmax}_y$ is called “decoder” and operationalizes the search for the maximization problem. Keyword: beam search
RNNs for Machine Translation
Statistical Machine Translation (SMT) (continued)

Based on a parallel corpus $D$, the best translation $y$ of a sentence $x$ given in the foreign language maximizes under $D$ the probability $p(y \mid x)$:

$$\arg\max_y p(y \mid x) = \arg\max_y p(x \mid y) \cdot p(y) \iff P(Y \mid X) = \frac{P(X \mid Y) \cdot P(Y)}{P(X)}$$

1. $p(y)$ is called “language model” and takes care of the fluency in the target language. It is modeled as $p(y_1, \ldots, y_m) = \prod_{i=1}^{m} p(y_i \mid y_{i-(n-1)}, \ldots, y_{i-1})$. Training data are (monolingual) corpora in the target language.

2. $p(x \mid y)$ is called “translation model” and captures the translation fidelity between two languages. It is modeled as $p(x, a \mid y)$, where “a” is a vector of alignment features. Training data are bilingual corpora.

3. $\arg\max_y$ is called “decoder” and operationalizes the search for the maximization problem. Keyword: beam search
Remarks (statistical machine translation):

- Although $p(y \mid x)$ can be maximized directly, Bayes rule is applied since the decomposition of $p(y \mid x)$ into $p(x \mid y)$ and $p(y)$ comes along with a number of advantages.

- In $\prod_{i=1}^{m} p(y_i \mid y_{i-(n-1)}, \ldots, y_{i-1})$, $m$ denotes the length of the entire sequence or sentence, and $n$ denotes the order of the model (the “window” size).

- In the language model syntax, $p(y) \equiv p(y_1, y_2, \ldots, y_m)$ denotes the probability of the event to observe the sentence $y \equiv y_1 y_2 \ldots y_m$, where $y_1$ corresponds to the first word of the sentence, $y_2$ to the second, etc.

  The $y_i$ are realizations of random variables, which can be written in any order as arguments of $p()$. I.e., to capture the word order, $y_i$ does not only denote the word but also its position: $y_i$ corresponds to the event “Word $y_i$ at position $i$.”

  In summary, $p(y_1, y_2, \ldots, y_m)$ is a short form of $P(Y_1=y_1, Y_2=y_2, \ldots, Y_m=y_m)$, where the $Y_i$ are random variables whose realizations are the possible words at position $i$. Note that these random variables are neither independent nor identically distributed.

- Learning $p(x, a \mid y)$ from a parallel corpus $D$ is a highly sophisticated endeavor since the alignments features, $a$, are complex and given as latent variables only.
RNNs for Machine Translation
Neural Machine Translation (NMT)

Concept:

- Machine translation with a multilayer perceptron (MLP).
- Network architecture is a sequence-to-sequence model:
  1. Encoder RNN, calculates an encoding of the source sentence $x$.
  2. Decoder RNN, generates the target sentence $y$. The decoder RNN is a conditional language model—it is conditioned on the RNN encoding.
- Optimization (loss minimization) is done for the network as a whole, which means that backpropagation is performed “end-to-end”.
RNNs for Machine Translation
Neural Machine Translation (NMT) (continued)

Concept:

\[ \text{i love my cat} \]

\[ \text{ich liebe meine katze} \]

Translation

\[ \text{ich liebe meine katze} \]

<end>

\[ \text{i love my cat} \]

<start>

ML:IX-138 Deep Learning © STEIN/VÖLSKE 2023
RNNs for Machine Translation
Neural Machine Translation (NMT) (continued)

Concept:
RNNs for Machine Translation
Neural Machine Translation (NMT) (continued)

Concept:

The sequence-to-sequence RNN directly calculates $p(y \mid x)$:

$$p(y \mid x) = p(y_1 \mid x) \cdot p(y_2 \mid y_1, x) \cdot p(y_3 \mid y_1, y_2, x) \cdots \cdot p(y_\tau \mid y_1, \ldots, y_{\tau-1}, x)$$
Remarks:

- “End-to-end” is not an architectural feature of a network (observe that every network is used in this way). It is a strategy for solving a task by not decomposing it, but by processing the original input-output examples in an indivisible manner.

- The sequence-to-sequence model is an example of a conditional language model: (1) It is a language model because the decoder is predicting the next word $y_t$ of the target sentence based on the preceding words $y_1, \ldots, y_{t-1}$. (2) It is conditional because its predictions are conditioned on the source sentence $x$. [Manning 2021, lecture CS224N]

- In the following slides, the hidden vector $y^e(T^e)$ represents the RNN encoding of the source sentence $x$. In particular,
  - the word $x_t$ from a source (input) sentence $x$ is denoted as $x(t)$,
  - the word $y_t$ from an output sentence is denoted as $y(t)$,
  - the word $y_t$ from a target sentence $y$ is denoted as $c(t)$.

  Note that we have until now not distinguished that $y_t$ can be output or target; $y_t$ has been considered as an (output) variable whose optimum value has to be determined.

- Don’t get confused: The input $y$ of the noisy channel becomes the output (or target) of the RNN. Similarly, the output $x$ of the noisy channel becomes the input of the RNN.
RNNs for Machine Translation

Types of Learning Tasks

[S1] sequence → class

sentence → \{⊕, ⊖\}
i love my cat → ⊕

[S2] class → sequence

{⊕, ⊖} → sentence
⊕ → i love my cat

[S3] sequence → sequence

English sentence → German sentence
i love my cat → ich liebe meine katze
RNNs for Machine Translation
(S3) Sequence-to-Sequence: Machine Translation

- I love my cat. → Ich liebe meine Katze.
- Cats and dogs lap water. → Katzen und Hunde lecken Wasser.
- It is raining cats and dogs. → Es regnet in Strömen.
- Cats and dogs are not allowed. → Katzen oder Hunde sind nicht erlaubt.

Vocabulary\(e\): (allowed and are cat cats dogs i is it lap love my not raining water)

Vocabulary\(d\): (erlaubt es hunde ich in katze lecken liebe meine nicht regnet sind strömen und wasser <start> <end>)
RNNs for Machine Translation
(S3) Sequence-to-Sequence: Machine Translation (continued)

- I love my cat. → Ich liebe meine Katze.
- Cats and dogs lap water. → Katzen und Hunde lecken Wasser.
- It is raining cats and dogs. → Es regnet in Strömen.
- Cats and dogs are not allowed. → Katzen oder Hunde sind nicht erlaubt.

Vocabulary\textsuperscript{e}: (allowed and are cat cats dogs i is it lap love my not raining water)

Vocabulary\textsuperscript{d}: (erlaubt es hunde ich in katze lecken liebe meine nicht regnet sind strömen und wasser <start> <end>)

Input: 
\[
\begin{bmatrix}
\begin{bmatrix}
[x, y(0)], y(1), y(2)], \ldots], y(\tau-1)]
\end{bmatrix}
\end{bmatrix}
\]
\[x = \begin{bmatrix}
0 \\
0 \\
0 \\
\vdots \\
0 \\
1 \\
0 \\
\vdots \\
1 \\
1 \\
\vdots \\
1 \\
0 \\
\vdots \\
1 \\
\vdots \\
1 \\
\vdots \\
1 \\
\end{bmatrix} \equiv I love my cat
\]

Output: 
\[y(1), y(2), y(3), \ldots, y(\tau^d)], \quad y(0) \equiv c(0) \equiv <start>, \quad y(\tau) \equiv c(5) \equiv <end>\]
RNNs for Machine Translation

(S3) Sequence-to-Sequence: Machine Translation (continued)

- I love my cat. → Ich liebe meine Katze.
- Cats and dogs lap water. → Katzen und Hunde lecken Wasser.
- It is raining cats and dogs. → Es regnet in Strömen.
- Cats and dogs are not allowed. → Katzen oder Hunde sind nicht erlaubt.

Vocabulary\(^e\): (allowed and are cat cats dogs i is it lap love my not raining water)

Vocabulary\(^d\): (erlaubt es hunde ich in katze lecken liebe meine nicht regnet sind strömen und wasser <start> <end>)

Input:

\[
\begin{bmatrix}
\begin{bmatrix} x, y(0) \end{bmatrix}, y(1), y(2), \ldots, y(\tau-1) \end{bmatrix}, \quad x = \begin{bmatrix}
0 \\ 1 \\ 0 \\ 0 \\
\vdots \\
\end{bmatrix}
\end{bmatrix}
\]

≡ I love my cat

Output:

\[
[y(1), y(2), y(3), \ldots, y(\tau^d)], \quad y(0) \equiv c(0) \equiv <start>, \quad y(\tau) \equiv c(5) \equiv <end>
\]
RNNs for Machine Translation
(S3) Sequence-to-Sequence: Machine Translation (continued)

- I love my cat. → Ich liebe meine Katze.
- Cats and dogs lap water. → Katzen und Hunde lecken Wasser.
- It is raining cats and dogs. → Es regnet in Strömen.
- Cats and dogs are not allowed. → Katzen oder Hunde sind nicht erlaubt.

Vocabulary\textsuperscript{e}: (allowed and are cat cats dogs i is it lap love my not raining water)

Vocabulary\textsuperscript{d}: (erlaubt es hunde ich in katze lecken liebe meine nicht regnet sind strömen und wasser \texttt{<start> \texttt{<end>}})

Input: \[
\left[ \left[ \left[ [x, y(0)], y(1) \right], y(2) \right], \ldots, y(\tau-1) \right], x = \begin{bmatrix}
0 \\ \\
\vdots \\ \\
1 \\
0 \\ \\
\vdots \\ \\
1 \\
0 \\
\vdots \\ \\
\end{bmatrix} \right] \triangleq \text{I love my cat}
\]

Output: \[
[y(1), y(2), y(3), \ldots, y(\tau^d)], \quad y(0) \equiv c(0) \triangleq \texttt{<start>}, \quad y(\tau) \equiv c(5) \triangleq \texttt{<end>}
\]
RNNs for Machine Translation
(S3) Sequence-to-Sequence: Machine Translation (continued)

- I love my cat. → Ich liebe meine Katze.
- Cats and dogs lap water. → Katzen und Hunde lecken Wasser.
- It is raining cats and dogs. → Es regnet in Strömen.
- Cats and dogs are not allowed. → Katzen oder Hunde sind nicht erlaubt.

Vocabulary
\( e \): (allowed and are cat cats dogs i is it lap love my not raining water)

Vocabulary
\( d \): (erlaubt es hunde ich in katze lecken liebe meine nicht regnet sind strömen und wasser <start> <end>)

Input:
\[
[ [ [ \mathbf{x}, \mathbf{y}(0) ], \mathbf{y}(1) ], \mathbf{y}(2) ], \ldots, \mathbf{y}(\tau-1) ], \quad \mathbf{x} = \begin{bmatrix} \begin{bmatrix} 0 \\ 1 \\ \vdots \end{bmatrix}, & \begin{bmatrix} 0 \\ 0 \\ \vdots \end{bmatrix}, & \begin{bmatrix} 0 \\ 1 \\ \vdots \end{bmatrix}, & \begin{bmatrix} 0 \\ 1 \\ \vdots \end{bmatrix} \end{bmatrix} \] \]
≡ I love my cat

Output:
\[
[ \mathbf{y}(1), \mathbf{y}(2), \mathbf{y}(3), \ldots, \mathbf{y}(\tau^d) ], \quad \mathbf{y}(0) \equiv \mathbf{c}(0) \equiv \text{<start>}, \quad \mathbf{y}(\tau) \equiv \mathbf{c}(5) \equiv \text{<end>}
\]
RNNs for Machine Translation
(S3) Sequence-to-Sequence: Machine Translation (continued)

- I love my cat. → Ich liebe meine Katze.
- Cats and dogs lap water. → Katzen und Hunde lecken Wasser.
- It is raining cats and dogs. → Es regnet in Strömen.
- Cats and dogs are not allowed. → Katzen oder Hunde sind nicht erlaubt.

**Vocabulary**

\[ e : ( \text{allowed and are cat cats dogs i is it lap love my not raining water} ) \]

\[ d : ( \text{erlaubt es hunde ich in katze lecken liebe meine nicht regnet sind strömen und wasser } <\text{start}> <\text{end}> ) \]

**Input:**

\[
[ [ [ [ x, y(0) ], y(1) ], y(2) ], \ldots ], y(\tau-1) ] , \quad x = \begin{bmatrix}
0 \\
1 \\
\vdots \\
0
\end{bmatrix},
\begin{bmatrix}
0 \\
0 \\
1 \\
0
\end{bmatrix},
\begin{bmatrix}
0 \\
0 \\
1 \\
0
\end{bmatrix} \quad \equiv \text I \text love my cat
\]

**Output:**

\[
y(1), y(2), y(3), \ldots, y(\tau^d) , \quad y(0) \equiv c(0) \equiv <\text{start}>, \quad y(\tau) \equiv c(5) \equiv <\text{end}> \]
RNNs for Machine Translation

(S3) Sequence-to-Sequence: Machine Translation (continued)

- I love my cat. \( \rightarrow \) Ich liebe meine Katze.
- Cats and dogs lap water. \( \rightarrow \) Katzen und Hunde lecken Wasser.
- It is raining cats and dogs. \( \rightarrow \) Es regnet in Strömen.
- Cats and dogs are not allowed. \( \rightarrow \) Katzen oder Hunde sind nicht erlaubt.

Vocabulary\(^e\): (allowed and are cat cats dogs i is it lap love my not raining water)

Vocabulary\(^d\): (erlaubt es hunde ich in katze lecken liebe meine nicht regnet sind strömen und wasser \(<\text{start}>\ <\text{end}>\)

Input:

\[
\hat{x} = \begin{bmatrix}
\begin{bmatrix} 0 \\ \vdots \\ 1 \cdot 10 \end{bmatrix}, \\
\begin{bmatrix} 0 \\ \vdots \\ 1 \cdot 10 \end{bmatrix}, \\
\begin{bmatrix} 0 \\ \vdots \\ 1 \cdot 10 \end{bmatrix}, \\
\begin{bmatrix} 0 \\ \vdots \\ 1 \cdot 10 \end{bmatrix}
\end{bmatrix} \triangleq \text{I love my cat}
\]

Output:

\[
y(1), y(2), y(3), \ldots, y(\tau^d), y(0) = c(0) \triangleq <\text{start}>, y(\tau) = c(5) \triangleq <\text{end}>
\]

Target:

\[
[c(1), \ldots, c(5)] = \begin{bmatrix}
\begin{bmatrix} 0 \\ \vdots \\ 1 \cdot 10 \end{bmatrix}, \\
\begin{bmatrix} 0 \\ \vdots \\ 1 \cdot 10 \end{bmatrix}, \\
\begin{bmatrix} 0 \\ \vdots \\ 1 \cdot 10 \end{bmatrix}, \\
\begin{bmatrix} 0 \\ \vdots \\ 1 \cdot 10 \end{bmatrix}
\end{bmatrix} \triangleq \text{Ich liebe meine Katze}
\]
RNNs for Machine Translation

(S3) Sequence-to-Sequence Mapping with RNNs

Input:
\[ x, [y(1), \ldots, y(\tau-1)] \]

Output:
\[ y(t) = \sigma \left( W^o y^d(t) \right), t = 1, \ldots, \tau \]

Hidden:
\[ y^e(t) = \sigma \left( W^h y^e(t-1) \right), t = 1, \ldots, T^e \]

Target:
\[ c(1), \ldots, c(T) \]
\[ c(T) \equiv <\text{end}> \]
RNNs for Machine Translation
(S3) Sequence-to-Sequence Mapping with RNNs (continued)

Input:
\[ x, [y(1), \ldots, y(\tau-1)] \]

Output:
\[ y(t) = \sigma_1 (W^o y^d(t)), t = 1, \ldots, \tau \]

Hidden:
\[ y^e(t) = \sigma \left( W^h \left( y^e(t-1) \right) \right), t = 1, \ldots, T^e \]
\[ y^d(t) = \sigma \left( W^h \left( y^d(t-1) \right) \right), t = 1, \ldots, \tau \]

Target:
\[ [c(1), \ldots, c(T)] \]
\[ c(T) \equiv \langle \text{end} \rangle \]
RNNs for Machine Translation

(S3) Sequence-to-Sequence Mapping with RNNs (continued)

Input:
\[ x, [y(1), \ldots, y(\tau-1)] \]

Output:
\[ y(t) = \sigma_1 \left( W^o y^d(t) \right) , t = 1, \ldots, \tau \]

Hidden:
\[ y^e(t) = \sigma \left( W^h \left( y^e(t-1) \right) \right) , t = 1, \ldots, T^e \]
\[ y^d(t) = \sigma \left( W^h \left( y^d(t-1) \right) \right) , t = 1, \ldots, \tau \]

Target:
\[ [c(1), \ldots, c(T)] \]
\[ c(T) \equiv <\text{end}> \]
RNNs for Machine Translation
(S3) Sequence-to-Sequence Mapping with RNNs (continued)

Input:
\( x, \{ y(1), \ldots, y(\tau-1) \} \)

Output:
\( y(t) = \sigma_1 \left( W^o y^d(t) \right), t = 1, \ldots, \tau \)

Hidden:
\( y^e(t) = \sigma \left( W^h \left( y^e(t-1) \right) x(t) \right), t = 1, \ldots, T^e \)

Target:
\( [c(1), \ldots, c(T)] \)
\( c(T) \equiv <\text{end}> \)
RNNs for Machine Translation

(S3) Sequence-to-Sequence Mapping with RNNs (continued)

Input:
\[ x, [y(1), \ldots, y(\tau-1)] \]

Output:
\[ y(t) = \sigma_1\left(W^o y^d(t)\right), t = 1, \ldots, \tau \]

Hidden:
\[ y^e(t) = \sigma\left(W^h\left(y^e(t-1)\right)\right), t = 1, \ldots, T^e \]
\[ y^d(t) = \sigma\left(W^h\left(y^d(t-1)\right)\right), t = 1, \ldots, \tau \]

Target:
\[ [c(1), \ldots, c(T)] \]
\[ c(T) \equiv \text{<end>} \]
RNNs for Machine Translation
(S3) Sequence-to-Sequence Mapping with RNNs (continued)

Input:
\[ x, [y(1), \ldots, y(\tau - 1)] \]

Output:
\[ y(t) = \sigma_1(W^o y^d(t)) , t = 1, \ldots, \tau \]

Hidden:
\[ y^e(t) = \sigma(W^h y^e(t-1)) , t = 1, \ldots, T^e \]

Target:
\[ [c(1), \ldots, c(T)] \]
\[ c(T) \equiv \text{<end>} \]
RNNs for Machine Translation
(S3) Sequence-to-Sequence Mapping with RNNs (continued)

Input: \[ x, [y(1), \ldots, y(4)] \]

Output: \[ y(t) = \sigma_1 \left( W^o y^d(t) \right), t = 1, \ldots, 5 \]

Hidden: \[
\begin{align*}
y^e(t) &= \sigma \left( W^h \left( y^e(t-1) \right) \right), t = 1, \ldots, 4 \\
y^d(t) &= \sigma \left( W^h \left( y^d(t-1) \right) \right), t = 1, \ldots, 5
\end{align*}
\]

Target: \[
\begin{align*}
[c(1), \ldots, c(5)] \\
[c(5)] &\equiv <\text{end}> 
\end{align*}
\]
RNNs for Machine Translation
(S3) Sequence-to-Sequence Mapping with RNNs (continued)

Encoder

Decoder

\[
\begin{align*}
\mathbf{y}_e(0) &= W^h, \\
\mathbf{x}(1) &\rightarrow \mathbf{y}_e(1) \\
\mathbf{x}(4) &\rightarrow \mathbf{y}_e(4) \\
\mathbf{y}(0) &\rightarrow \mathbf{c}(0) \\
\end{align*}
\]

\[
\begin{align*}
\mathbf{y}_d(0) &= \mathbf{y}_e(0), \\
\mathbf{y}_d(1) &= \mathbf{y}_e(1), \\
\mathbf{y}_d(4) &= \mathbf{y}_e(4), \\
\mathbf{y}_d(5) &= \mathbf{y}_e(5) \\
\end{align*}
\]

\[
\begin{align*}
\mathbf{c}(1) &\rightarrow \mathbf{y}(1) \\
\mathbf{c}(5) &\rightarrow \mathbf{y}(5) \\
\end{align*}
\]

Input:
\[\mathbf{x}, [\mathbf{y}(1), \ldots, \mathbf{y}(4)]\]

Output:
\[\mathbf{y}(t) = \sigma_1 (W^o \mathbf{y}_d(t)) , t = 1, \ldots, 5\]

Hidden:
\[\mathbf{y}_e(t) = \sigma (W^h (\mathbf{y}_e(t-1))) , t = 1, \ldots, 4\]

Target:
\[\mathbf{c}(1), \ldots, \mathbf{c}(5)\]
\[\mathbf{c}(5) \equiv \langle \text{end} \rangle\]

ich liebe meine katze <end>
i love my cat <start>
Remarks:

- The final encoder hidden state, $y^e(T^e)$, represents the encoding of the source sentence. $y^e(T^e)$ is unified with the first decoder hidden state, $y^d(0)$.

- The encoder hidden state $y^e(t)$ represents the input sequence up to time step $t$, $[x(1), \ldots, x(t)]$.

- The decoder hidden state $y^d(t)$ represents the entire input sequence $[x(1), \ldots, x(T^e)]$, as well as the output sequence up to time step $t-1$, $[y(1), \ldots, y(t-1)]$.

- Note that, as before, we are given a model function $y()$, which maps some input (actually, a sequence of feature vectors, $[x(1), \ldots, x(T^e)]$) to some output (a sequence of output vectors, $[y(1), \ldots, y(T^d)]$).
RNNs for Machine Translation

Sequence-to-Sequence RNNs are Conditional Language Models

<table>
<thead>
<tr>
<th>Encoder</th>
<th>Decoder</th>
</tr>
</thead>
<tbody>
<tr>
<td>$y^e(0)$</td>
<td>$W^h$</td>
</tr>
<tr>
<td>$y^e(1)$</td>
<td>$W^h$</td>
</tr>
<tr>
<td>$y^e(2)$</td>
<td>$W^h$</td>
</tr>
<tr>
<td>$y^e(3)$</td>
<td>$W^h$</td>
</tr>
<tr>
<td>$y^e(4)$</td>
<td>$y^d(5)$</td>
</tr>
</tbody>
</table>

$x(1)$  $x(4)$  $y(0)$  $c(0)$

i  love  my  cat  <start>

ich liebe meine katze  <end>
The sequence-to-sequence RNN directly calculates $p(y \mid x)$:

$$p(y \mid x) = p(y_1 \mid x) \cdot p(y_2 \mid y_1, x) \cdot p(y_3 \mid y_1, y_2, x) \cdot p(y_4 \mid y_1, y_2, y_3, x)$$
The sequence-to-sequence RNN directly calculates $p(y \mid x)$:

$$p(y \mid x) \equiv p(y^{(1)}, \ldots, y^{(5)} \mid x, y^{(0)}), \quad x := x^{(1)}, x^{(2)}, x^{(3)}, x^{(4)}$$

$$= p(y^{(1)} \mid x, y^{(0)}) \cdot p(y^{(2)} \mid x, y^{(0)}, y^{(1)}) \cdot \ldots \cdot p(y^{(5)} \mid x, y^{(0)}, \ldots, y^{(4)})$$
The sequence-to-sequence RNN directly calculates $p(y \mid x)$:

$$p(y \mid x) \equiv p(y(1), \ldots, y(5) \mid x, y(0)), \quad x := x(1), x(2), x(3), x(4)$$

$$= p(y(1) \mid x, y(0)) \cdot p(y(2) \mid x, y(0), y(1)) \cdot \ldots \cdot p(y(5) \mid x, y(0), \ldots, y(4))$$
RNNs for Machine Translation

Sequence-to-Sequence RNNs are Conditional Language Models (continued)

The sequence-to-sequence RNN directly calculates $p(y \mid x)$:

$$p(y \mid x) \equiv p(y(1), \ldots, y(5) \mid x, y(0)), \quad x := x(1), x(2), x(3), x(4)$$

$$= p(y(1) \mid x, y(0)) \cdot p(y(2) \mid x, y(0), y(1)) \cdot \ldots \cdot p(y(5) \mid x, y(0), \ldots, y(4))$$
RNNs for Machine Translation

Sequence-to-Sequence RNNs are Conditional Language Models (continued)

The sequence-to-sequence RNN directly calculates $p(y \mid x)$:

$$p(y \mid x) \equiv p(y(1), \ldots, y(5) \mid x, y(0)), \quad x := x(1), x(2), x(3), x(4)$$

$$= p(y(1) \mid x, y(0)) \cdot p(y(2) \mid x, y(0), y(1)) \cdot \ldots \cdot p(y(5) \mid x, y(0), \ldots, y(4))$$
Remarks:

- Each output vector $y(t)$ corresponds to a probability distribution over the Vocabulary\(^d\) (recall the $\sigma_1$-function). Here, the illustration of generation (aka decoding) steps shows an argmax-operation on each $y(t)$, called “greedy decoding”: the word with the highest probability is chosen.

- To maximize $\prod_{t=1}^{T} p(y(t) \mid x, y(0), \ldots, y(t-1))$, a complete search in the space of all sequences (target sentences) that can be generated is necessary, which is computationally intractable. Instead, heuristic search such as beam search is applied, where a beam size around 5 to 10 has shown good results in practice.

  The beam size is the number of generated successors in each decoding step; they are added to the OPEN list of the heuristic search algorithm. [Course on Search Algorithms]

- Sequence-to-sequence RNNs can be “stacked”, this way forming a multi-layered RNN, which is able to compute more complex representations. The idea is that the lower (higher) RNNs compute the lower-level (higher-level) features.

  Practice has shown that 2-4 layers are useful for neural machine translation, while transformer-based networks are typically deeper and comprise 12-24 layers. [Manning 2021, lecture CS224N]
RNNs for Machine Translation

Representation: Embeddings Instead of One-Hot Encoding

[TODO]