Chapter ML:IX (continued)

IX. Deep Learning

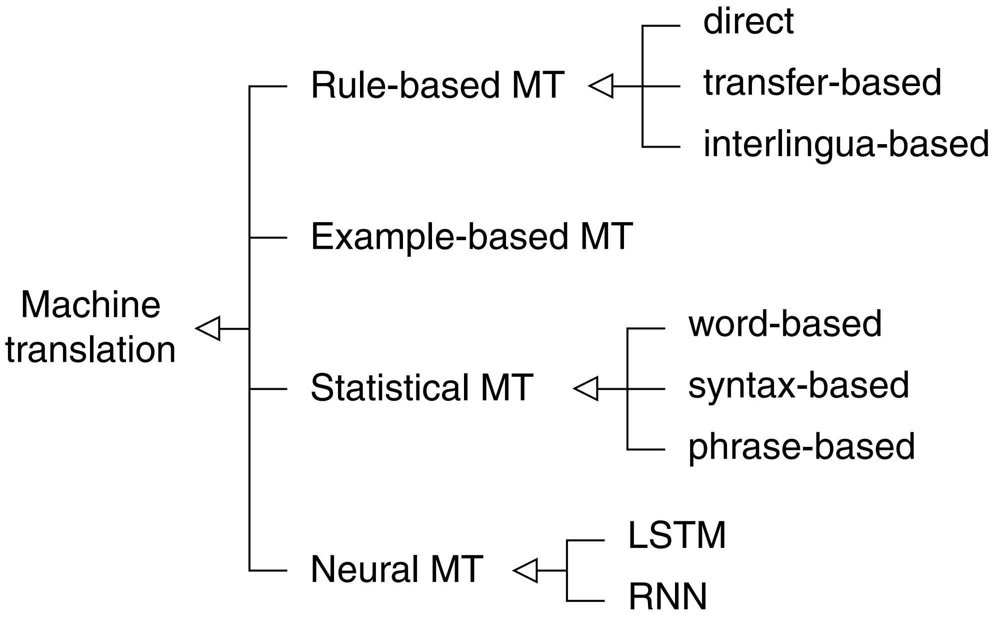
- Elements of Deep Learning
- Convolutional Neural Networks
- Autoencoder Networks

- Recurrent Neural Networks
- RNNs for Machine Translation
- Attention Mechanism
- Self Attention and Transformers

- Transformer Language Models

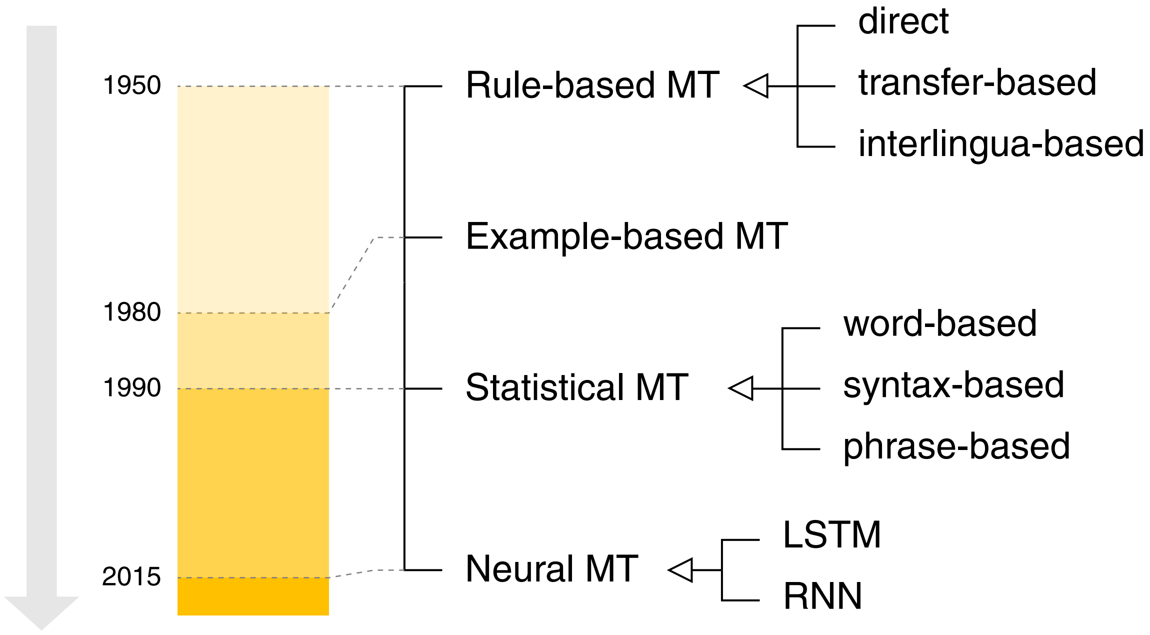
RNNs for Machine Translation

Statistical Machine Translation (SMT)



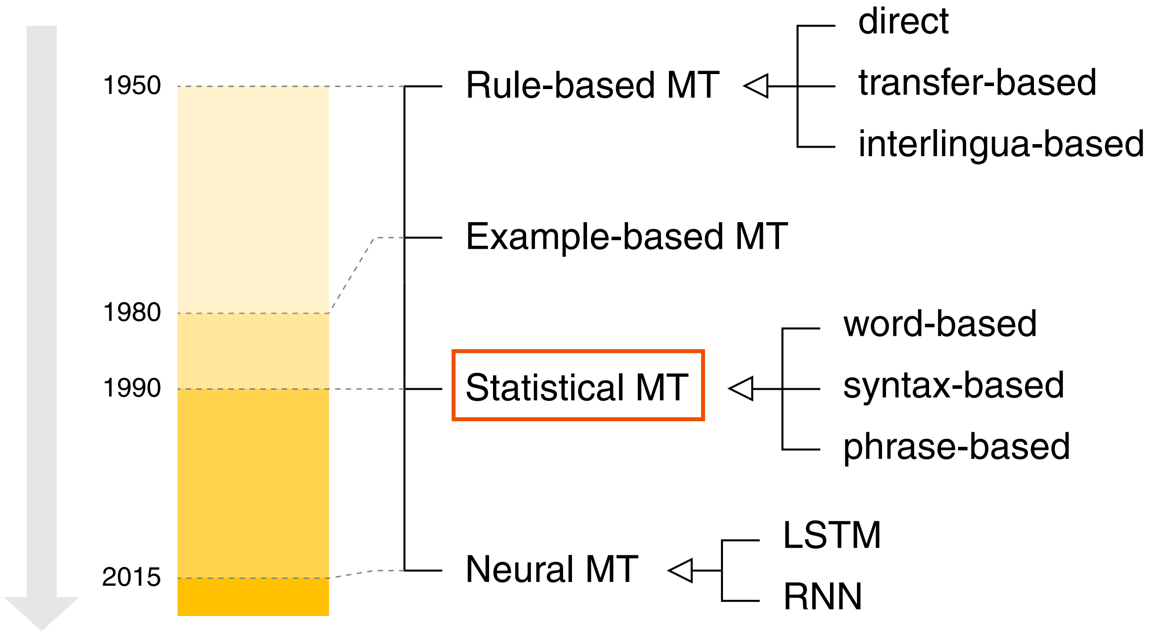
RNNs for Machine Translation

Statistical Machine Translation (SMT) (continued)



RNNs for Machine Translation

Statistical Machine Translation (SMT) (continued)

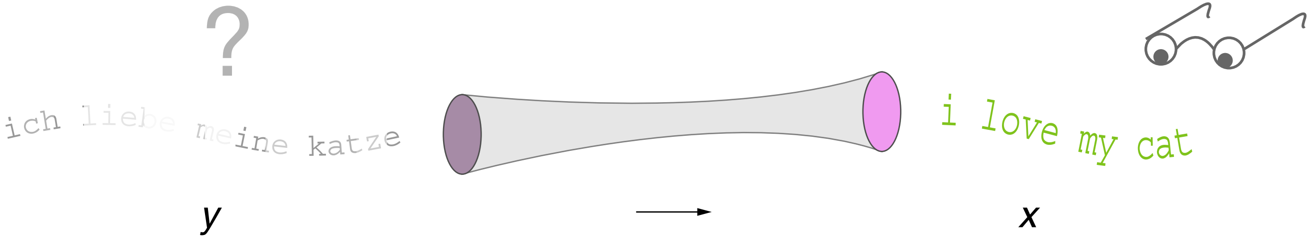


“Noisy channel” model applied to SMT:

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)).

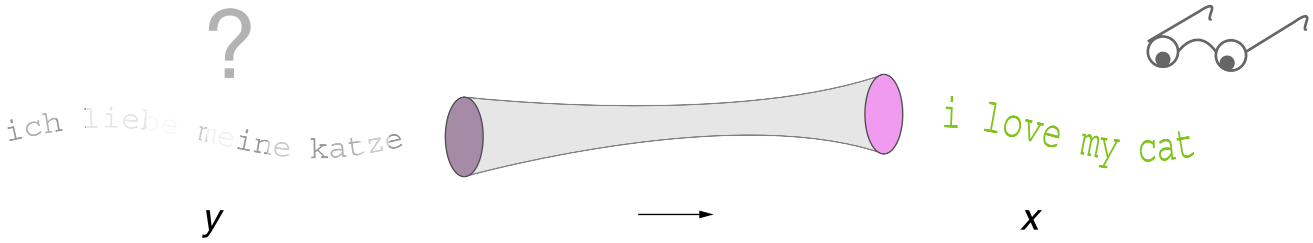
RNNs for Machine Translation

Statistical Machine Translation (SMT) (continued)



RNNs for Machine Translation

Statistical Machine Translation (SMT) (continued)



$$p\big(\text{“ich jage eine katze”} \;\big|\; \text{“i love my cat”}\big)$$
$$p\big(\text{“ich habe keine katze”} \;\big|\; \text{“i love my cat”}\big)$$
$$\vdots$$
$$p\big(\text{“ich liebe meine katze”} \;\big|\; \text{“i love my cat”}\big)$$

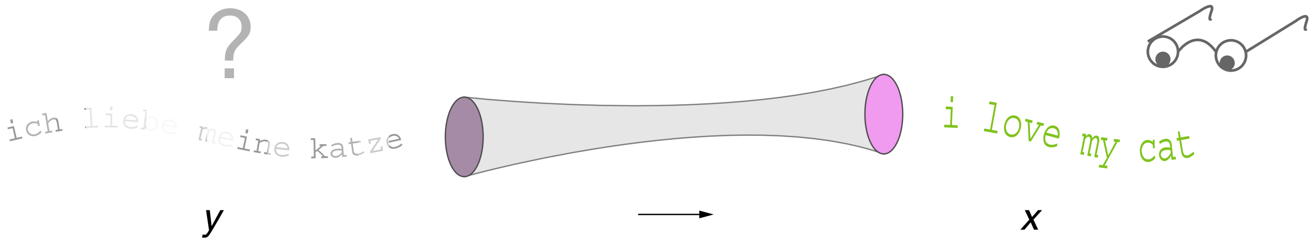
$$p\big(\text{German\_target\_sentence} \;\big|\; \text{English\_source\_sentence}\big)$$

$$p\big(\text{sentence\_in\_own\_language} \;\big|\; \text{sentence\_in\_foreign\_language}\big)$$

$$p\big(y \;\big|\; x\big)$$

RNNs for Machine Translation

Statistical Machine Translation (SMT) (continued)



$$
\begin{gathered}
p\big(\text{“}\texttt{ich jage eine katze}\text{”} \mid \text{“}\texttt{i love my cat}\text{”}\big) \\
p\big(\text{“}\texttt{ich habe keine katze}\text{”} \mid \text{“}\texttt{i love my cat}\text{”}\big) \\
\vdots \\
p\big(\text{“}\texttt{ich liebe meine katze}\text{”} \mid \text{“}\texttt{i love my cat}\text{”}\big) \\
p\big(\text{German\_target\_sentence} \mid \text{English\_source\_sentence}\big) \\
p\big(\text{sentence\_in\_own\_language} \mid \text{sentence\_in\_foreign\_language}\big) \\
p(y \mid x)
\end{gathered}
$$

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

$$p(y \mid x) \to \text{max}$$

Remarks:

- 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 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 $\mathbf{y}^{\text{e}}(T^{\text{e}})$ 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 $p(y \mid x)$:

$$\text{argmax}_y\, p(y \mid x) = \text{argmax}_y\; p(x \mid y) \cdot p(y) \qquad \Leftarrow$$

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

$X \mathrel{\widehat{=}}$ X$=x$, $\quad x \mathrel{\widehat{=}}$ English sentence

$Y \mathrel{\widehat{=}}$ Y$=y$, $\quad y \mathrel{\widehat{=}}$ 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 $p(y \mid x)$:

$$\text{argmax}_y\, p(y \mid x) = \text{argmax}_y\; p(x \mid y) \cdot \boxed{p(y)} \qquad \Leftarrow$$

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

$X \,\widehat{=}\,$ X$=x$, $\quad x \,\widehat{=}\,$ English sentence

$Y \,\widehat{=}\,$ Y$=y$, $\quad y \,\widehat{=}\,$ 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 $p(x, \mathbf{a} \mid y)$, where "$\mathbf{a}$" is a vector of alignment features. Training data are bilingual corpora.

3. 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 $p(y \mid x)$:

$$\text{argmax}_y\, p(y \mid x) = \text{argmax}_y \boxed{p(x \mid y)} \cdot p(y) \qquad \Leftarrow$$

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

$X \mathrel{\widehat{=}}$ X$=x$, $x \mathrel{\widehat{=}}$ English sentence

$Y \mathrel{\widehat{=}}$ Y$=y$, $y \mathrel{\widehat{=}}$ 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 $p(x, \mathbf{a} \mid y)$, where “$\mathbf{a}$” is a vector of alignment features. Training data are bilingual corpora.

3. 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 $p(y \mid x)$:

$$\text{argmax}_y \, p(y \mid x) = \boxed{\text{argmax}_y} \; p(x \mid y) \cdot p(y) \qquad \Leftarrow$$

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

$X \mathrel{\widehat{=}}$ X$=x$, $x \mathrel{\widehat{=}}$ English sentence

$Y \mathrel{\widehat{=}}$ Y$=y$, $y \mathrel{\widehat{=}}$ 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 $p(x, \mathbf{a} \mid y)$, where "$\mathbf{a}$" is a vector of alignment features. Training data are bilingual corpora.

3. argmax_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 the language model syntax, $p(y) = p(y_1, y_2, \ldots, y_m)$ denotes the probability of the event to observe the sentence $y = y_1 y_2 \ldots y_m$, 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 $P(\mathsf{Y}_1 = y_1, \mathsf{Y}_2 = y_2, \ldots, \mathsf{Y}_m = y_m)$, 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 $p(x, \mathbf{a} \mid y)$ from a parallel corpus D is a highly sophisticated endeavor since the alignments features 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:

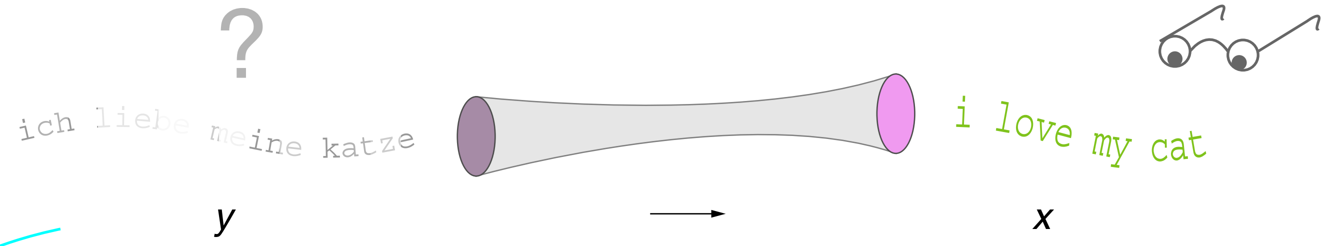
  1. Encoder RNN, calculates an encoding of the source sentence x.
  2. Decoder RNN, generates the target sentence y. The decoder RNN is a *conditional* language model—it is conditioned on the RNN encoding.

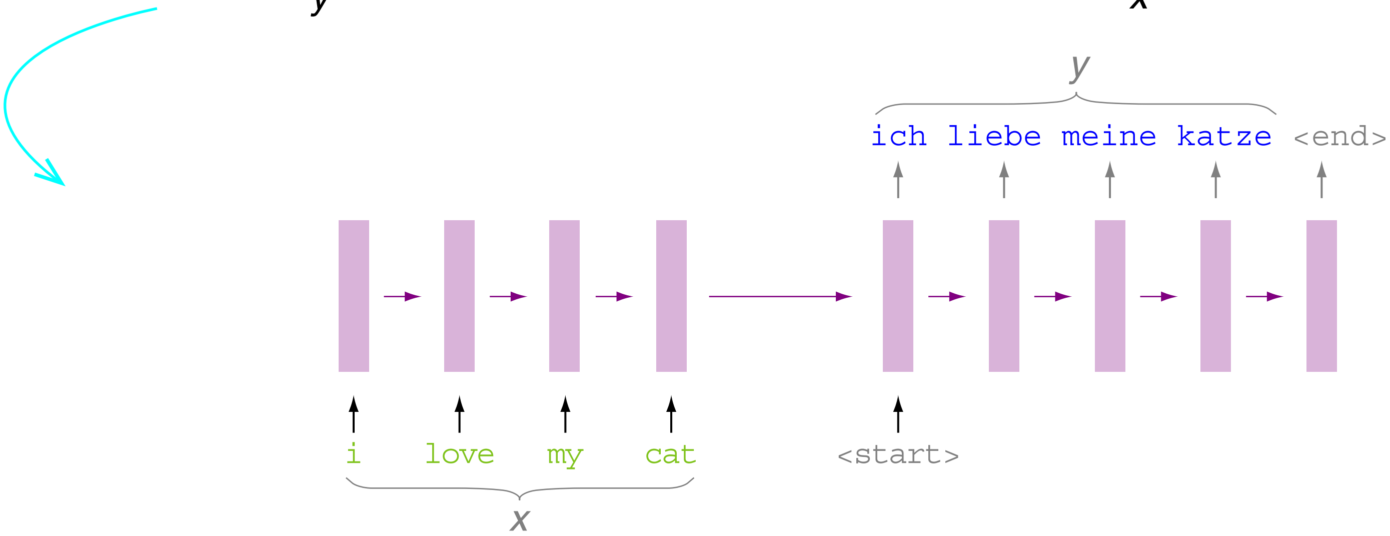
- Optimization (loss minimization) is done for the network as a whole, which means that backpropagation is performed “end-to-end”.

RNNs for Machine Translation

Neural Machine Translation (NMT) (continued)

Concept:

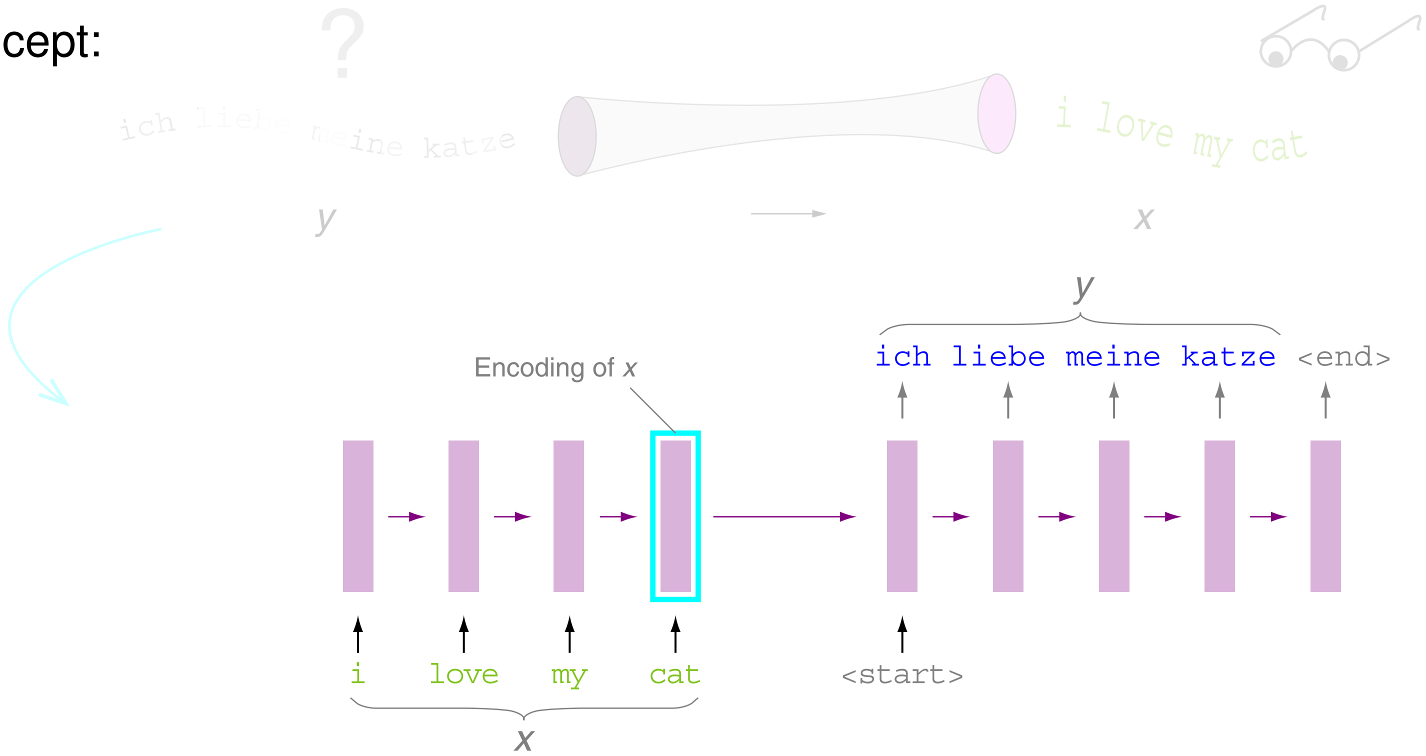




RNNs for Machine Translation

Neural Machine Translation (NMT) (continued)

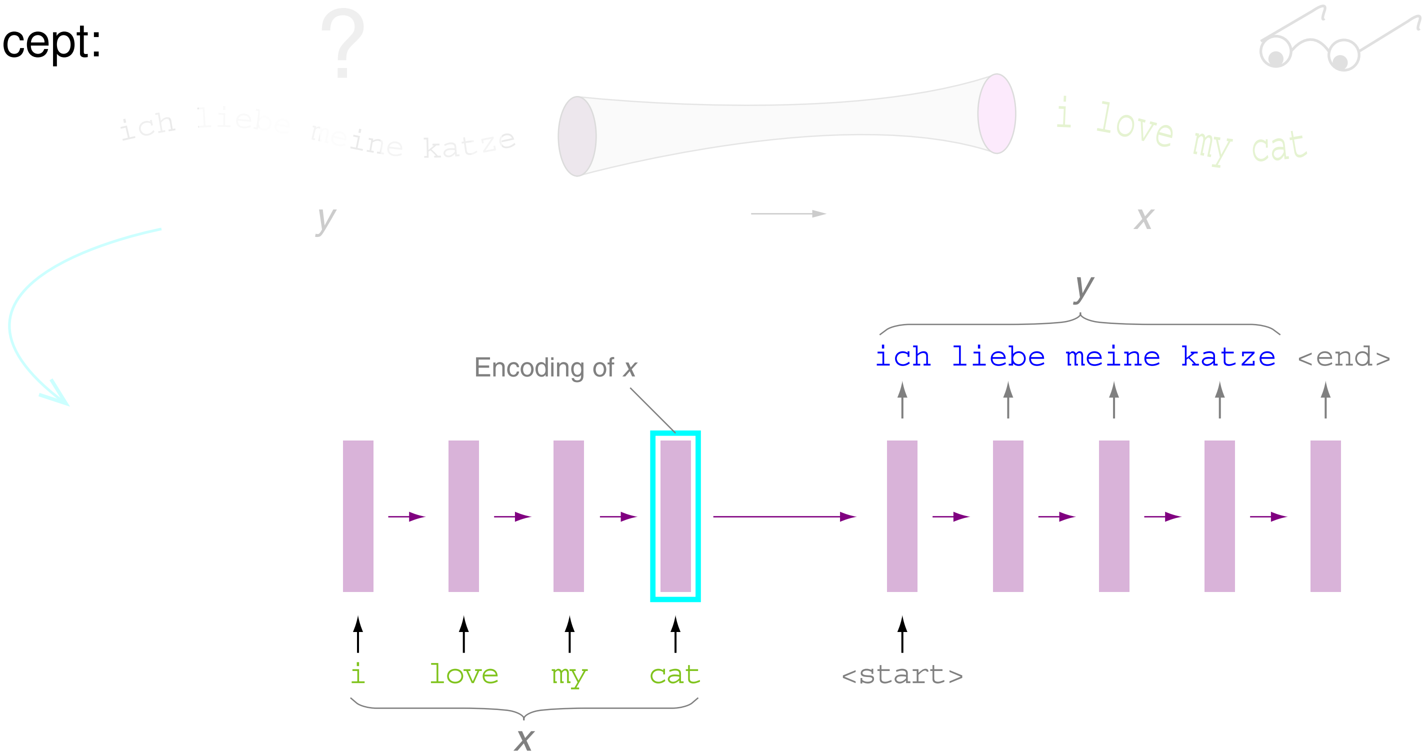
Concept:



RNNs for Machine Translation

Neural Machine Translation (NMT) (continued)

Concept:



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

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

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 also conditioned on the source sentence x. [Manning 2021, lecture CS224N]

- In the following slides, the hidden vector $\mathbf{y}^{\mathrm{e}}(T^{\mathrm{e}})$ represents the RNN encoding of the source sentence x. In particular,

  - the words x_t from a source (input) sentence x are denoted as $\mathbf{x}(t)$,
  - the words y_t from a output sentence are denoted as $\mathbf{y}(t)$,
  - the words y_t from a target sentence y are denoted as $\mathbf{c}(t)$.

  Note that we have not distinguished whether y_t is output or target.

- Don’t get confused: The input y of the noisy channel becomes the target (output) of the RNN. Similary, the output x of the noisy channel becomes the source (input) of the RNN.

RNNs for Machine Translation

Types of Learning Tasks [Recap]

(S1) sequence → class

sentence $\rightarrow \{\oplus, \ominus\}$

`i love my cat` $\rightarrow \oplus$

(S2) class → sequence

$\{\oplus, \ominus\} \rightarrow$ sentence

$\oplus \rightarrow$ `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$^{\mathsf{e}}$: (allowed and are cat cats dogs i is it lap love my not raining water)

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

Input: $[\,[\,[\,[\,[\mathbf{x},\ \mathbf{y}(0)\,],\ \mathbf{y}(1)],\ \mathbf{y}(2)],\ldots],\ \mathbf{y}(\tau-1)],\ \mathbf{x} = \left[\begin{pmatrix}0\\ \vdots\\ 1\\ 0\\ \vdots\end{pmatrix}, \begin{pmatrix}0\\ \vdots\\ \vdots\\ 1\\ 0\\ \vdots\end{pmatrix}, \begin{pmatrix}0\\ \vdots\\ \vdots\\ 1\\ 0\\ \vdots\end{pmatrix}, \begin{pmatrix}0\\ \vdots\\ 1\\ 0\\ \vdots\\ \vdots\end{pmatrix}\right] \mathrel{\widehat{=}}$ I love my cat

Output: $[\mathbf{y}(1),\mathbf{y}(2),\mathbf{y}(3),\ldots,\mathbf{y}(\tau^{\mathsf{d}})]$, $\mathbf{y}(0) \equiv \mathbf{c}(0) \mathrel{\widehat{=}}$ <start>, $\mathbf{y}(\tau) \mathrel{\widehat{=}} \mathbf{c}(5) \mathrel{\widehat{=}}$ <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$^{\text{e}}$: (`allowed and are cat cats dogs i is it lap love my not raining water`)

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

Input:

$$[\,[\,[\,[\,[\mathbf{x},\ \mathbf{y}(0)],\ \mathbf{y}(1)],\ \mathbf{y}(2)],\ldots],\ \mathbf{y}(\tau{-}1)],\quad \mathbf{x} = \left[\begin{pmatrix}0\\ \vdots\\ 1\\ 0\\ \vdots\end{pmatrix}, \begin{pmatrix}0\\ \vdots\\ 1\\ 0\\ \vdots\end{pmatrix}, \begin{pmatrix}0\\ \vdots\\ 1\\ 0\\ \vdots\end{pmatrix}, \begin{pmatrix}0\\ \vdots\\ 1\\ 0\\ \vdots\end{pmatrix}\right] \mathrel{\widehat{=}} \text{I love my cat}$$

Output:

$$[\mathbf{y}(1), \mathbf{y}(2), \mathbf{y}(3), \ldots, \mathbf{y}(\tau^{\text{d}})],\quad \mathbf{y}(0) \equiv \mathbf{c}(0) \mathrel{\widehat{=}} \texttt{<start>},\quad \mathbf{y}(\tau) \mathrel{\widehat{=}} \mathbf{c}(5) \mathrel{\widehat{=}} \texttt{<end>}$$

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: $[\,[\,[\,[\,[\mathbf{x},\ \mathbf{y}(0)],\ \mathbf{y}(1)],\ \mathbf{y}(2)],\ldots],\ \mathbf{y}(\tau{-}1)],\ \ \mathbf{x} = \left[\begin{pmatrix}0\\ \vdots\\ 1\\ 0\\ \vdots\end{pmatrix}, \begin{pmatrix}0\\ \vdots\\ \vdots\\ 1\\ 0\\ \vdots\end{pmatrix}, \begin{pmatrix}0\\ \vdots\\ \vdots\\ 1\\ 0\\ \vdots\end{pmatrix}, \begin{pmatrix}0\\ \vdots\\ 1\\ 0\\ \vdots\\ \vdots\end{pmatrix}\right]$ $\widehat{=}$ I love my cat

Output: $[\mathbf{y}(1), \mathbf{y}(2), \mathbf{y}(3), \ldots, \mathbf{y}(\tau^{\mathsf{d}})]$, $\quad \mathbf{y}(0) \equiv \mathbf{c}(0) \ \widehat{=}$ <start>, $\quad \mathbf{y}(\tau) \ \widehat{=}\ \mathbf{c}(5) \ \widehat{=}$ <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: $[\,[\,[\,[\,[\mathbf{x},\ \mathbf{y}(0)],\ \mathbf{y}(1)],\ \mathbf{y}(2)], \ldots],\ \mathbf{y}(\tau{-}1)],\quad \mathbf{x} = \left[\begin{pmatrix}0\\ \vdots\\ 1\\ 0\\ \vdots\end{pmatrix}, \begin{pmatrix}0\\ \vdots\\ \vdots\\ 1\\ 0\\ \vdots\end{pmatrix}, \begin{pmatrix}0\\ \vdots\\ \vdots\\ \vdots\\ 1\\ 0\\ \vdots\end{pmatrix}, \begin{pmatrix}0\\ \vdots\\ 1\\ 0\\ \vdots\\ \vdots\end{pmatrix}\right] \mathrel{\widehat{=}}$ I love my cat

Output: $[\mathbf{y}(1), \mathbf{y}(2), \mathbf{y}(3), \ldots, \mathbf{y}(\tau^{d})],\quad \mathbf{y}(0) \equiv \mathbf{c}(0) \mathrel{\widehat{=}} \texttt{<start>},\quad \mathbf{y}(\tau) \mathrel{\widehat{=}} \mathbf{c}(5) \mathrel{\widehat{=}} \texttt{<end>}$

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: $[\,[\,[\,[\,[\mathbf{x}, \mathbf{y}(0)], \mathbf{y}(1)], \mathbf{y}(2)], \ldots], \mathbf{y}(\tau{-}1)], \quad \mathbf{x} = \left[\begin{pmatrix}0\\ \vdots\\ 1\\ 0\\ \vdots\end{pmatrix}, \begin{pmatrix}0\\ \vdots\\ 1\\ 0\\ \vdots\end{pmatrix}, \begin{pmatrix}0\\ \vdots\\ 1\\ 0\\ \vdots\end{pmatrix}, \begin{pmatrix}0\\ \vdots\\ 1\\ 0\\ \vdots\end{pmatrix}\right] \;\widehat{=}$ I love my cat

Output: $[\mathbf{y}(1), \mathbf{y}(2), \mathbf{y}(3), \ldots, \mathbf{y}(\tau^{d})], \quad \mathbf{y}(0) \equiv \mathbf{c}(0) \;\widehat{=}$ <start>, $\quad \mathbf{y}(\tau) \;\widehat{=}\; \mathbf{c}(5) \;\widehat{=}$ <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>`)

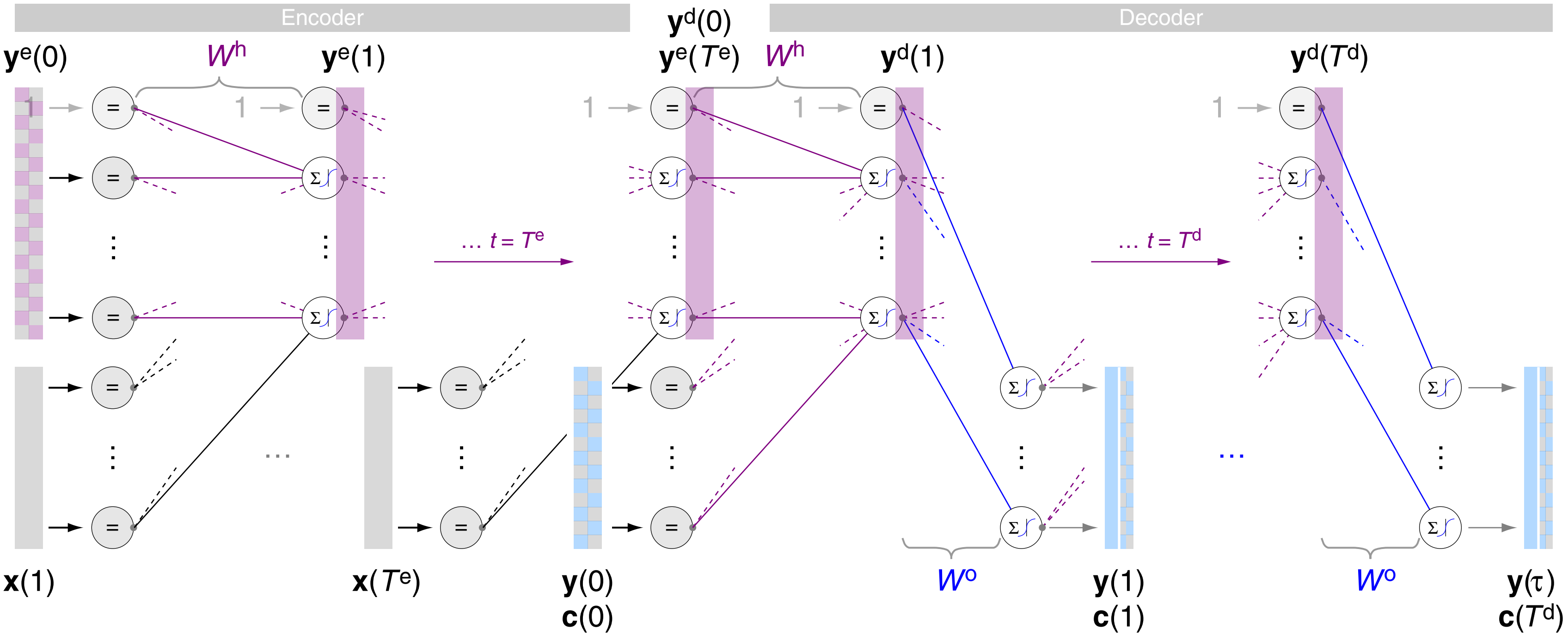
Input: $$[\,[\,[\,[\,[\mathbf{x},\ \mathbf{y}(0)],\ \mathbf{y}(1)],\ \mathbf{y}(2)], \ldots],\ \mathbf{y}(\tau{-}1)],\quad \mathbf{x} = \left[\begin{pmatrix}0\\ \vdots\\ 1\\ 0\\ \vdots\end{pmatrix}, \begin{pmatrix}0\\ \vdots\\ 1\\ 0\\ \vdots\end{pmatrix}, \begin{pmatrix}0\\ \vdots\\ 1\\ 0\\ \vdots\end{pmatrix}, \begin{pmatrix}0\\ 1\\ 0\\ \vdots\end{pmatrix}\right] \mathrel{\widehat{=}} \text{I love my cat}$$

Output: $$[\mathbf{y}(1), \mathbf{y}(2), \mathbf{y}(3), \ldots, \mathbf{y}(\tau^{\mathrm{d}})],\qquad \mathbf{y}(0) \equiv \mathbf{c}(0) \mathrel{\widehat{=}} \texttt{<start>},\quad \mathbf{y}(\tau) \mathrel{\widehat{=}} \mathbf{c}(5) \mathrel{\widehat{=}} \texttt{<end>}$$

Target: $$[\mathbf{c}(1), \ldots, \mathbf{c}(5)] = \left[\begin{pmatrix}0\\ \vdots\\ 1\\ 0\\ \vdots\end{pmatrix}, \begin{pmatrix}0\\ \vdots\\ 1\\ 0\\ \vdots\end{pmatrix}, \begin{pmatrix}0\\ \vdots\\ 1\\ 0\\ \vdots\end{pmatrix}, \begin{pmatrix}0\\ \vdots\\ 1\\ 0\\ \vdots\end{pmatrix}, \begin{pmatrix}0\\ \vdots\\ \vdots\\ 1\end{pmatrix}\right] \mathrel{\widehat{=}} \text{Ich liebe meine Katze}$$

RNNs for Machine Translation

(S3) Sequence-to-Sequence Mapping with RNNs



Encoder
Decoder
$\mathbf{y}^{\mathrm{e}}(0)$
W^{h}
$\mathbf{y}^{\mathrm{e}}(1)$
$\mathbf{y}^{\mathrm{d}}(0)$
$\mathbf{y}^{\mathrm{e}}(T^{\mathrm{e}})$
W^{h}
$\mathbf{y}^{\mathrm{d}}(1)$
$\mathbf{y}^{\mathrm{d}}(T^{\mathrm{d}})$
1
1
1
1
1
=
Σ
… $t = T^{\mathrm{e}}$
… $t = T^{\mathrm{d}}$
…
$\mathbf{x}(1)$
$\mathbf{x}(T^{\mathrm{e}})$
$\mathbf{y}(0)$
$\mathbf{c}(0)$
W^{o}
$\mathbf{y}(1)$
$\mathbf{c}(1)$
W^{o}
$\mathbf{y}(\tau)$
$\mathbf{c}(T^{\mathrm{d}})$

Input:

$\mathbf{x}, [\mathbf{y}(1), \ldots, \mathbf{y}(\tau-1)]$

Hidden:

$$\mathbf{y}^{\mathrm{e}}(t) = \boldsymbol{\sigma}\left(W^{\mathrm{h}}\begin{pmatrix}\mathbf{y}^{\mathrm{e}}(t-1)\\ \mathbf{x}(t)\end{pmatrix}\right), t = 1, \ldots, T^{\mathrm{e}}$$

$$\mathbf{y}^{\mathrm{d}}(t) = \boldsymbol{\sigma}\left(W^{\mathrm{h}}\begin{pmatrix}\mathbf{y}^{\mathrm{d}}(t-1)\\ \mathbf{y}(t-1)\end{pmatrix}\right), t = 1, \ldots, \tau$$

Target:

$[\mathbf{c}(1), \ldots, \mathbf{c}(T)]$

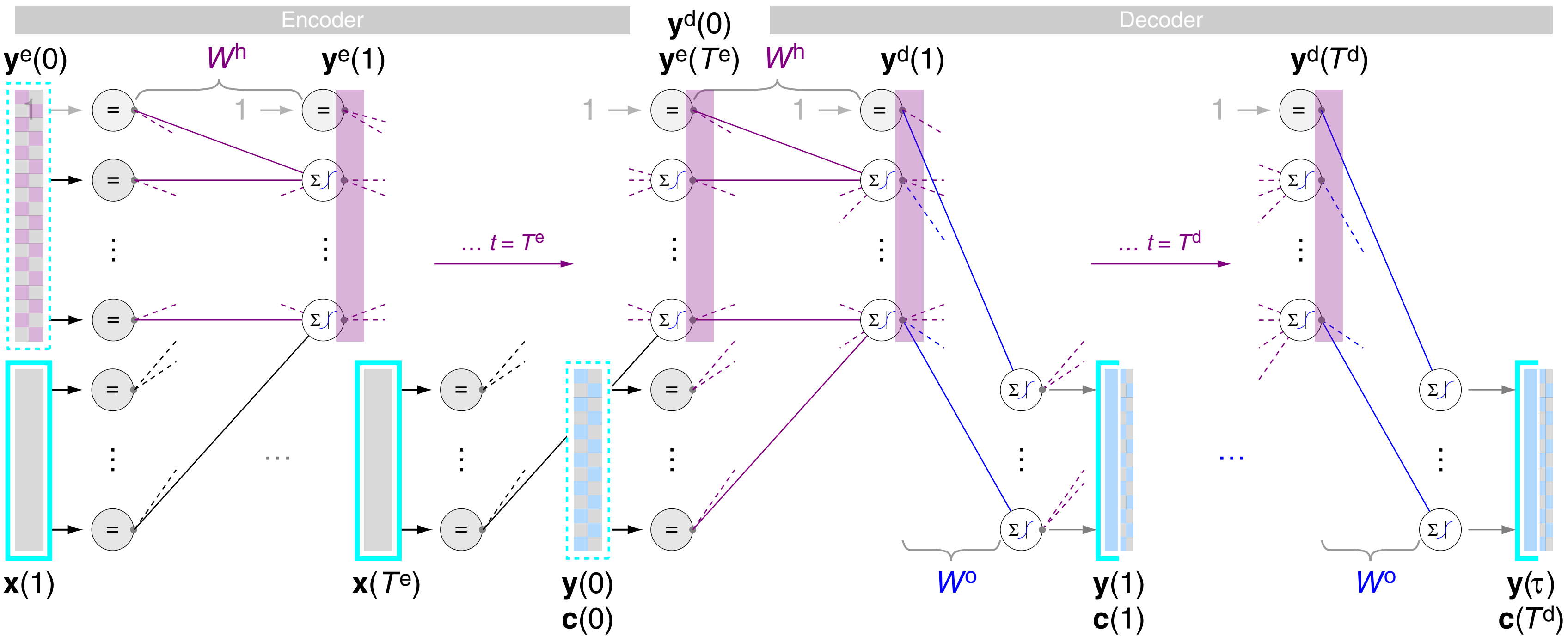
$\mathbf{c}(T) \,\widehat{=}\, \texttt{<end>}$

Output:

$\mathbf{y}(t) = \boldsymbol{\sigma}_1\left(W^{\mathrm{o}}\,\mathbf{y}^{\mathrm{d}}(t)\right), t = 1, \ldots, \tau$

RNNs for Machine Translation

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



Input:

$\mathbf{x}$, $[\mathbf{y}(1), \ldots, \mathbf{y}(\tau - 1)]$

Output:

$\mathbf{y}(t) = \sigma_1 \left(W^o \, \mathbf{y}^d(t) \right), t = 1, \ldots, \tau$

Hidden:

$$\mathbf{y}^e(t) = \boldsymbol{\sigma} \left(W^h \begin{pmatrix} \mathbf{y}^e(t-1) \\ \mathbf{x}(t) \end{pmatrix} \right), t = 1, \ldots, T^e$$

$$\mathbf{y}^d(t) = \boldsymbol{\sigma} \left(W^h \begin{pmatrix} \mathbf{y}^d(t-1) \\ \mathbf{y}(t-1) \end{pmatrix} \right), t = 1, \ldots, \tau$$

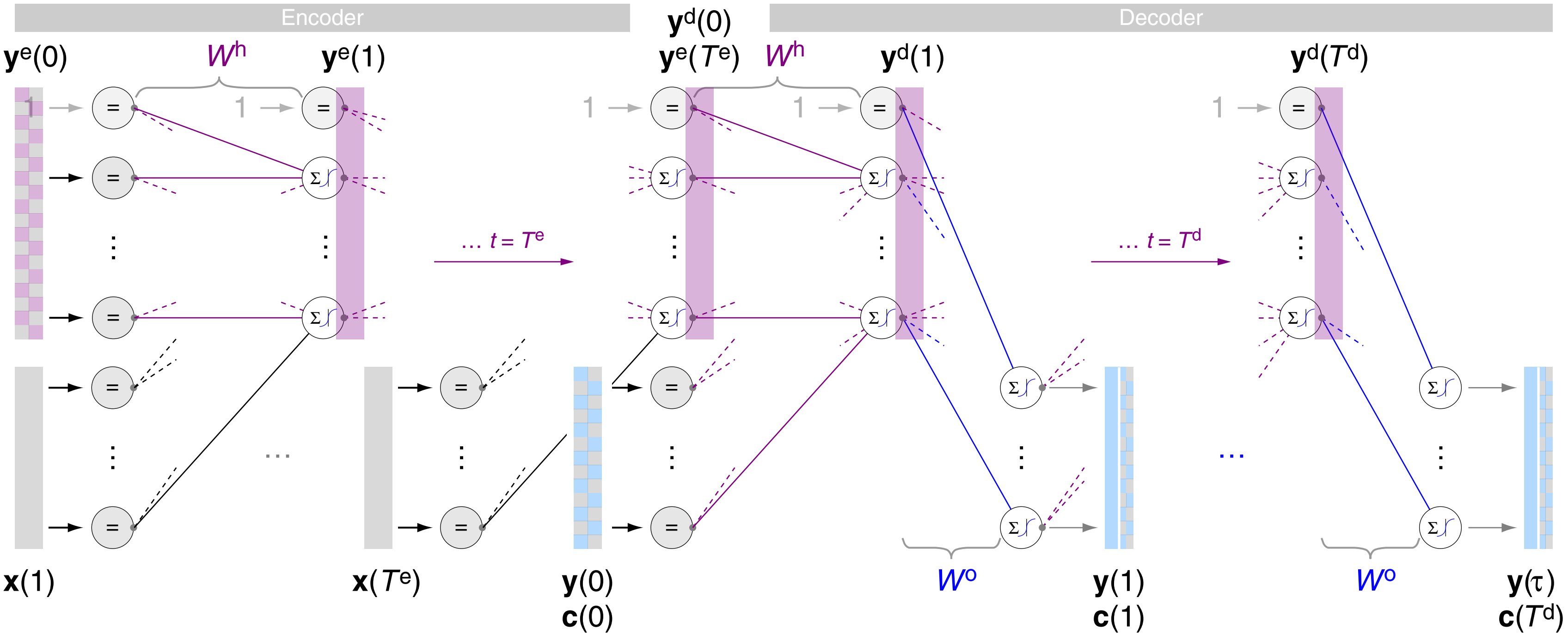
Target:

$[\mathbf{c}(1), \ldots, \mathbf{c}(T)]$

$\mathbf{c}(T) \mathrel{\widehat{=}} \text{<end>}$

RNNs for Machine Translation

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



Input:

$\mathbf{x}$, $[\mathbf{y}(1), \ldots, \mathbf{y}(\tau{-}1)]$

Output:

$$\mathbf{y}(t) = \boldsymbol{\sigma}_1\left(W^o\,\mathbf{y}^d(t)\right), t = 1, \ldots, \tau$$

Hidden:

$$\mathbf{y}^e(t) = \boldsymbol{\sigma}\left(W^h \begin{pmatrix} \mathbf{y}^e(t{-}1) \\ \mathbf{x}(t) \end{pmatrix}\right), t = 1, \ldots, T^e$$

$$\mathbf{y}^d(t) = \boldsymbol{\sigma}\left(W^h \begin{pmatrix} \mathbf{y}^d(t{-}1) \\ \mathbf{y}(t{-}1) \end{pmatrix}\right), t = 1, \ldots, \tau$$

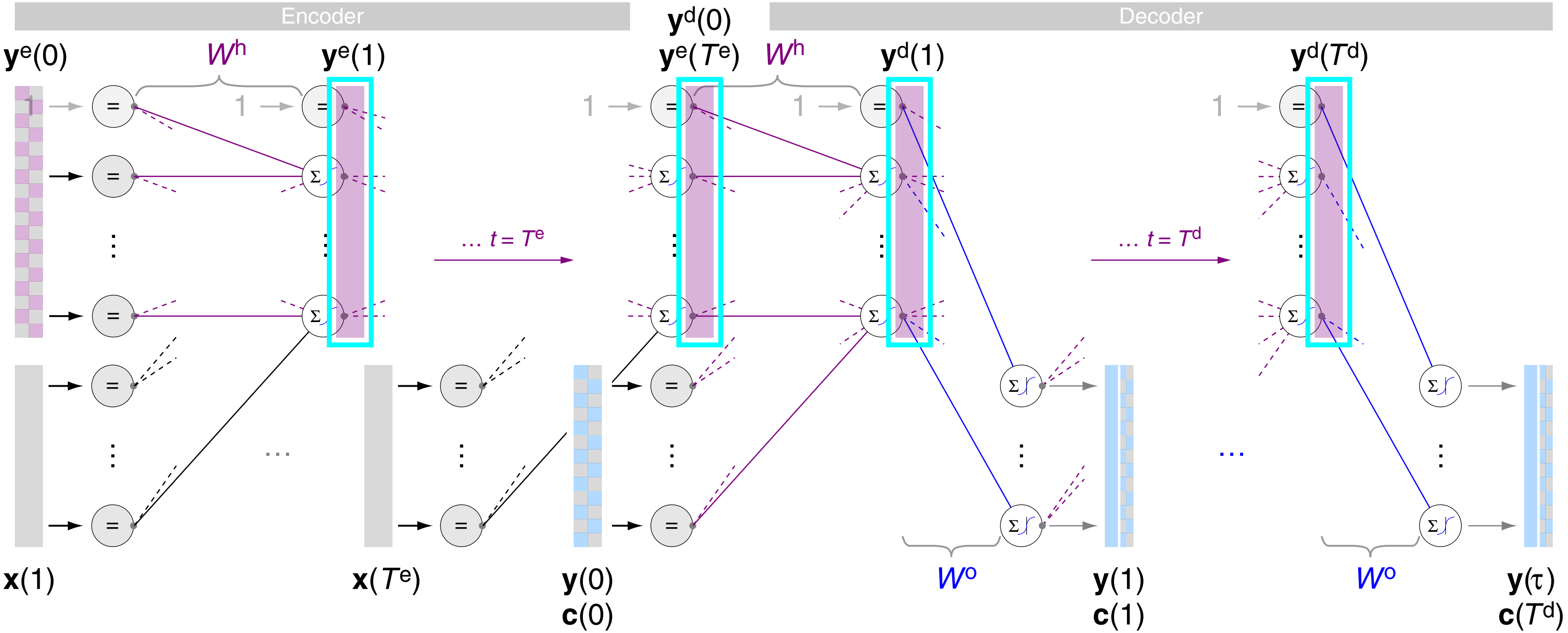
Target:

$[\mathbf{c}(1), \ldots, \mathbf{c}(T)]$

$\mathbf{c}(T) \mathrel{\widehat{=}} \text{<end>}$

RNNs for Machine Translation

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



Input:

$\mathbf{x}$, $[\mathbf{y}(1), \ldots, \mathbf{y}(\tau-1)]$

Output:

$\mathbf{y}(t) = \boldsymbol{\sigma}_1\left(W^o\,\mathbf{y}^d(t)\right), t = 1, \ldots, \tau$

Hidden:

$$\mathbf{y}^e(t) = \boldsymbol{\sigma}\left(W^h\begin{pmatrix}\mathbf{y}^e(t-1)\\ \mathbf{x}(t)\end{pmatrix}\right), t = 1, \ldots, T^e$$

$$\mathbf{y}^d(t) = \boldsymbol{\sigma}\left(W^h\begin{pmatrix}\mathbf{y}^d(t-1)\\ \mathbf{c}(t-1)\end{pmatrix}\right), t = 1, \ldots, T^d$$

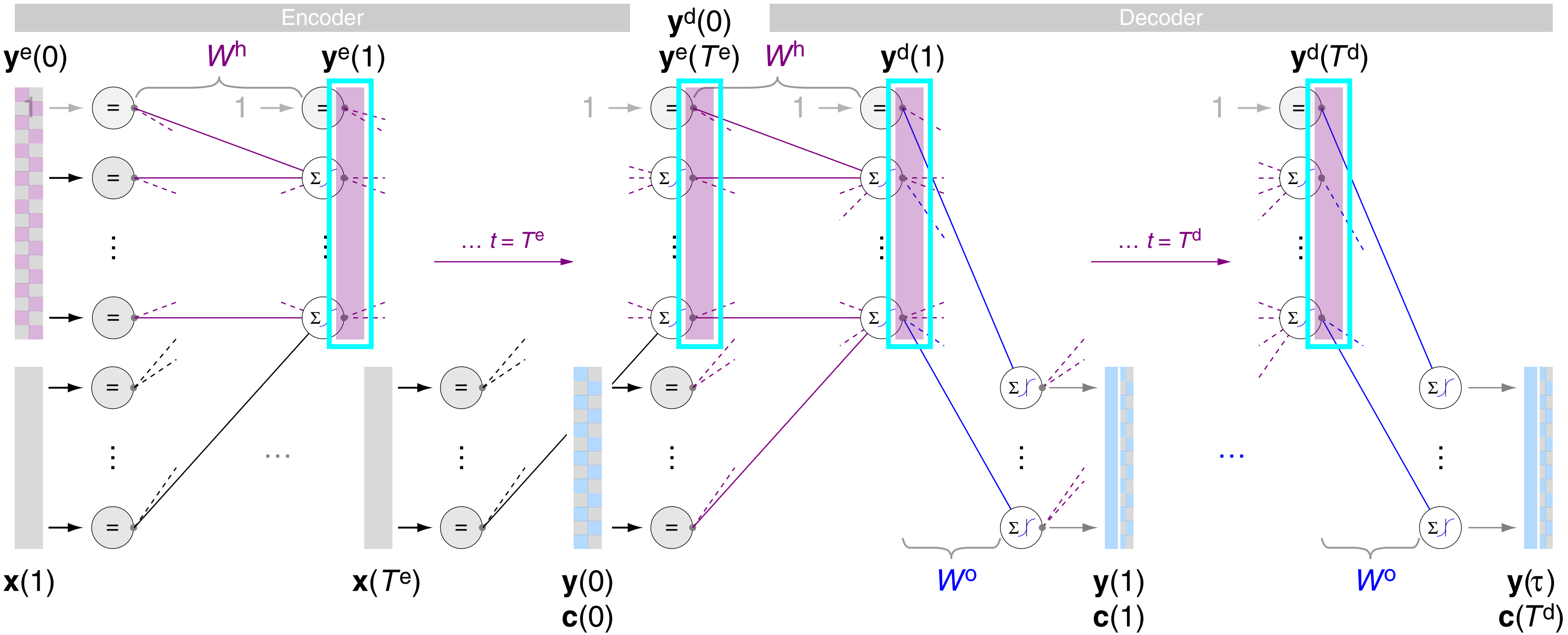
Target:

$[\mathbf{c}(1), \ldots, \mathbf{c}(T)]$

$\mathbf{c}(T) \mathrel{\widehat{=}}$ <end>

RNNs for Machine Translation

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



Input:

$\mathbf{x},\ [\mathbf{y}(1), \ldots, \mathbf{y}(\tau{-}1)]$

Output:

$\mathbf{y}(t) = \boldsymbol{\sigma}_1\left(W^o\,\mathbf{y}^d(t)\right), t = 1, \ldots, \tau$

Hidden:

$$\mathbf{y}^e(t) = \boldsymbol{\sigma}\left(W^h\begin{pmatrix}\mathbf{y}^e(t{-}1)\\ \mathbf{x}(t)\end{pmatrix}\right), t = 1, \ldots, T^e$$

$$\mathbf{y}^d(t) = \boldsymbol{\sigma}\left(W^h\begin{pmatrix}\mathbf{y}^d(t{-}1)\\ \mathbf{y}(t{-}1)\end{pmatrix}\right), t = 1, \ldots, \tau$$

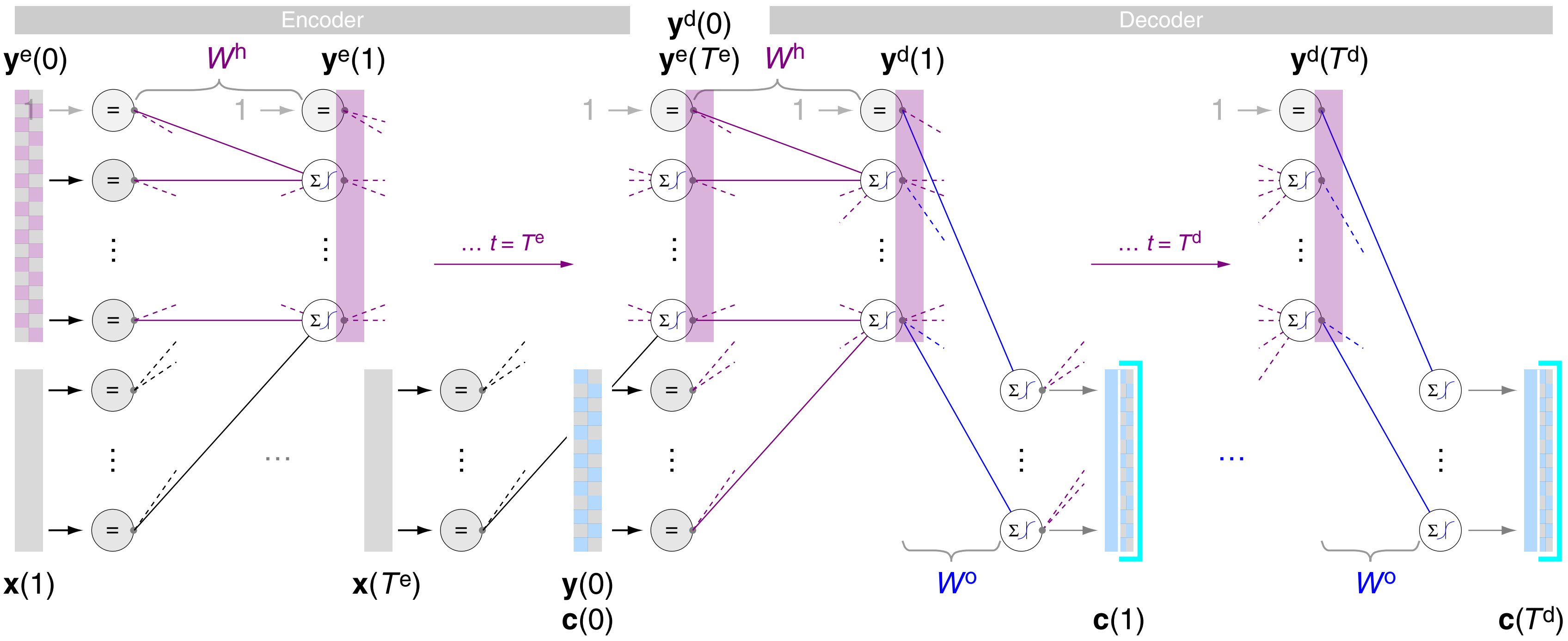
Target:

$[\mathbf{c}(1), \ldots, \mathbf{c}(T)]$

$\mathbf{c}(T) \mathrel{\widehat{=}} \text{<end>}$

RNNs for Machine Translation

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



Encoder
Decoder
$\mathbf{y}^{\text{e}}(0)$
W^{h}
$\mathbf{y}^{\text{e}}(1)$
$\mathbf{y}^{\text{d}}(0)$
$\mathbf{y}^{\text{e}}(T^{\text{e}})$
W^{h}
$\mathbf{y}^{\text{d}}(1)$
$\mathbf{y}^{\text{d}}(T^{\text{d}})$
1
… $t = T^{\text{e}}$
… $t = T^{\text{d}}$
…
$\mathbf{x}(1)$
$\mathbf{x}(T^{\text{e}})$
$\mathbf{y}(0)$
$\mathbf{c}(0)$
W^{o}
$\mathbf{c}(1)$
W^{o}
$\mathbf{c}(T^{\text{d}})$

Input:

$\mathbf{x}$, $[\mathbf{y}(1), \ldots, \mathbf{y}(\tau{-}1)]$

Output:

$$\mathbf{y}(t) = \boldsymbol{\sigma}_1\left(W^{\text{o}}\,\mathbf{y}^{\text{d}}(t)\right), t = 1, \ldots, \tau$$

Hidden:

$$\mathbf{y}^{\text{e}}(t) = \boldsymbol{\sigma}\left(W^{\text{h}}\begin{pmatrix}\mathbf{y}^{\text{e}}(t{-}1)\\ \mathbf{x}(t)\end{pmatrix}\right), t = 1, \ldots, T^{\text{e}}$$

$$\mathbf{y}^{\text{d}}(t) = \boldsymbol{\sigma}\left(W^{\text{h}}\begin{pmatrix}\mathbf{y}^{\text{d}}(t{-}1)\\ \mathbf{y}(t{-}1)\end{pmatrix}\right), t = 1, \ldots, \tau$$

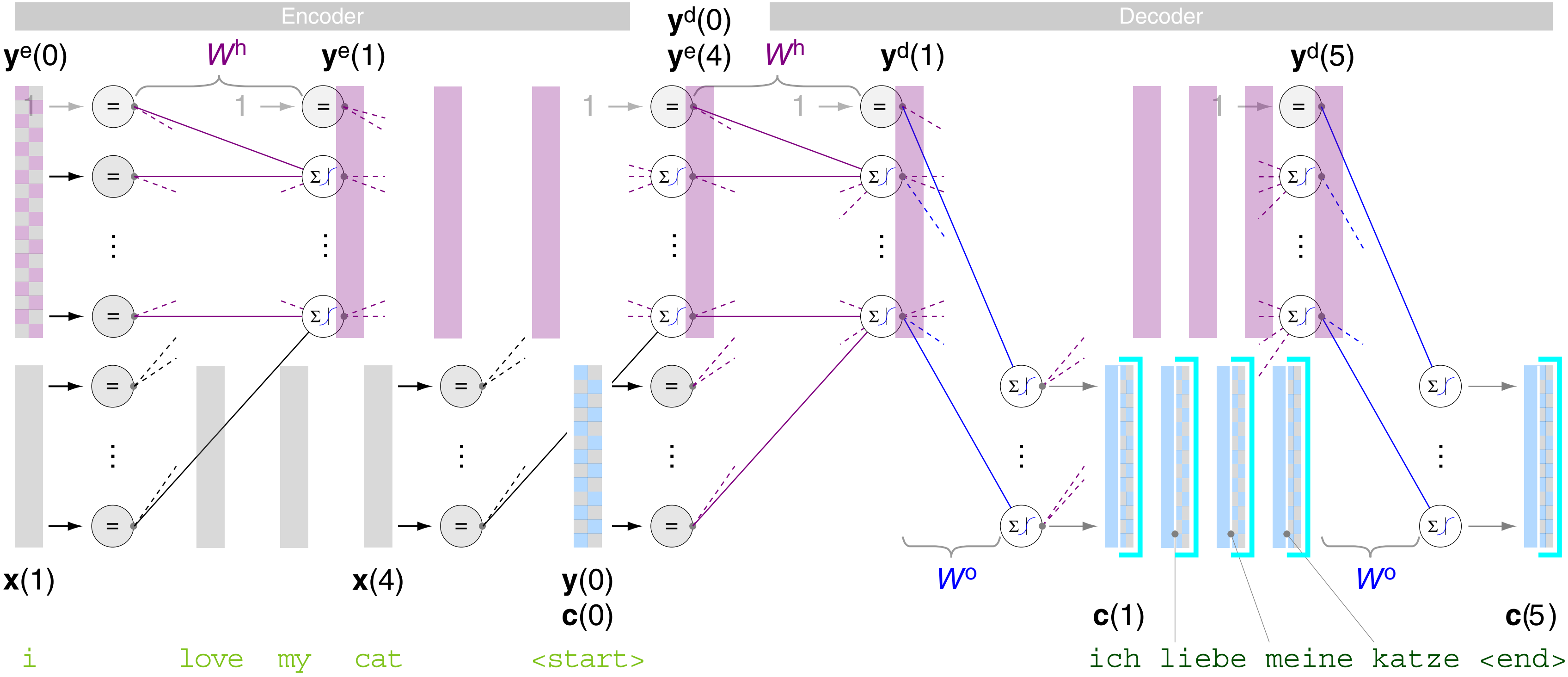
Target:

$[\mathbf{c}(1), \ldots, \mathbf{c}(T)]$

$\mathbf{c}(T) \mathrel{\widehat{=}}$ <end>

RNNs for Machine Translation

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



Input:

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

Output:

$$\mathbf{y}(t) = \boldsymbol{\sigma}_1 \left(W^o \, \mathbf{y}^d(t) \right), t = 1, \ldots, 5$$

Hidden:

$$\mathbf{y}^e(t) = \boldsymbol{\sigma} \left(W^h \begin{pmatrix} \mathbf{y}^e(t-1) \\ \mathbf{x}(t) \end{pmatrix} \right), t = 1, \ldots, 4$$

$$\mathbf{y}^d(t) = \boldsymbol{\sigma} \left(W^h \begin{pmatrix} \mathbf{y}^d(t-1) \\ \mathbf{c}(t-1) \end{pmatrix} \right), t = 1, \ldots, 5$$

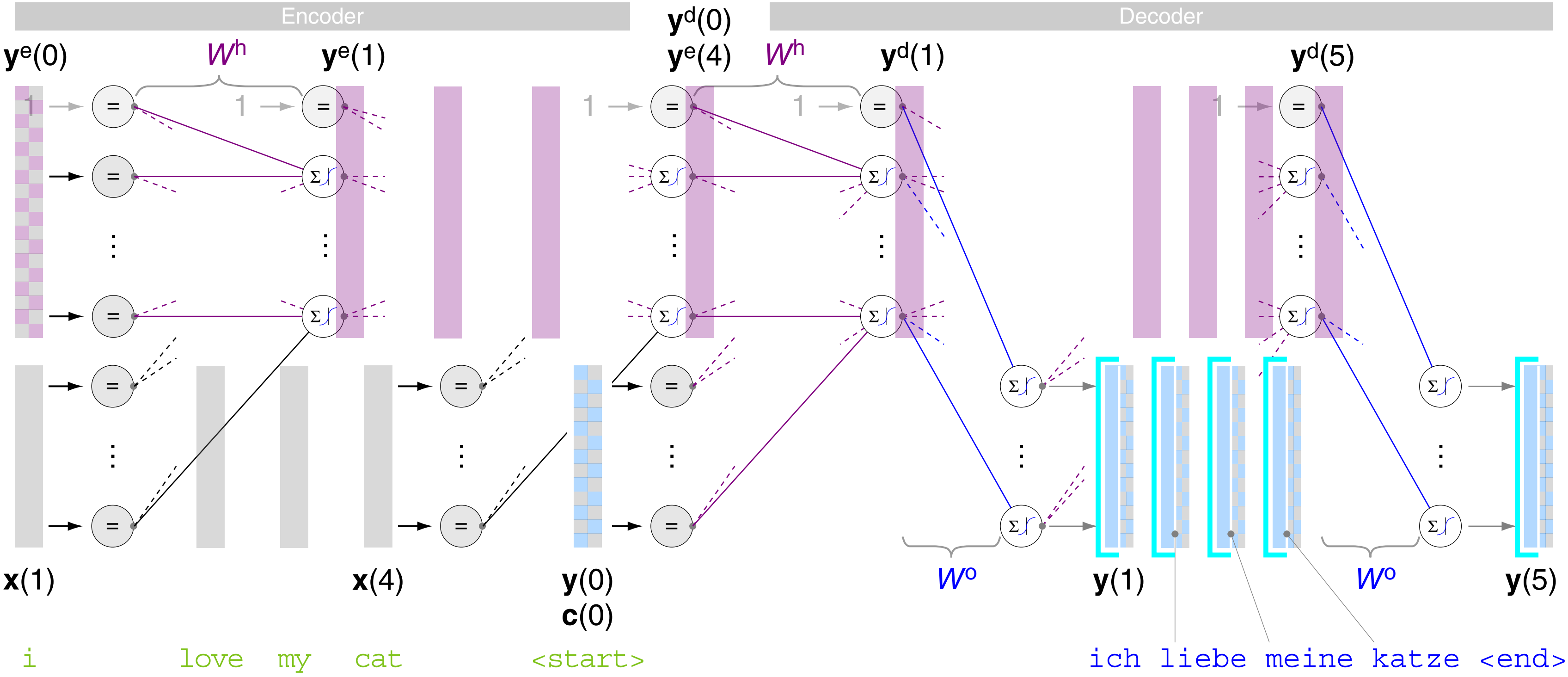
Target:

$[\mathbf{c}(1), \ldots, \mathbf{c}(5)]$

$\mathbf{c}(5) \mathrel{\widehat{=}}$ <end>

RNNs for Machine Translation

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



Input:

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

Output:

$$\mathbf{y}(t) = \boldsymbol{\sigma}_1\left(W^o\,\mathbf{y}^d(t)\right), t = 1, \ldots, 5$$

Hidden:

$$\mathbf{y}^e(t) = \boldsymbol{\sigma}\left(W^h\begin{pmatrix}\mathbf{y}^e(t-1)\\ \mathbf{x}(t)\end{pmatrix}\right), t = 1, \ldots, 4$$

$$\mathbf{y}^d(t) = \boldsymbol{\sigma}\left(W^h\begin{pmatrix}\mathbf{y}^d(t-1)\\ \mathbf{y}(t-1)\end{pmatrix}\right), t = 1, \ldots, 5$$

Target:

$[\mathbf{c}(1), \ldots, \mathbf{c}(5)]$

$\mathbf{c}(5) \mathrel{\widehat{=}}$ <end>

Remarks:

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

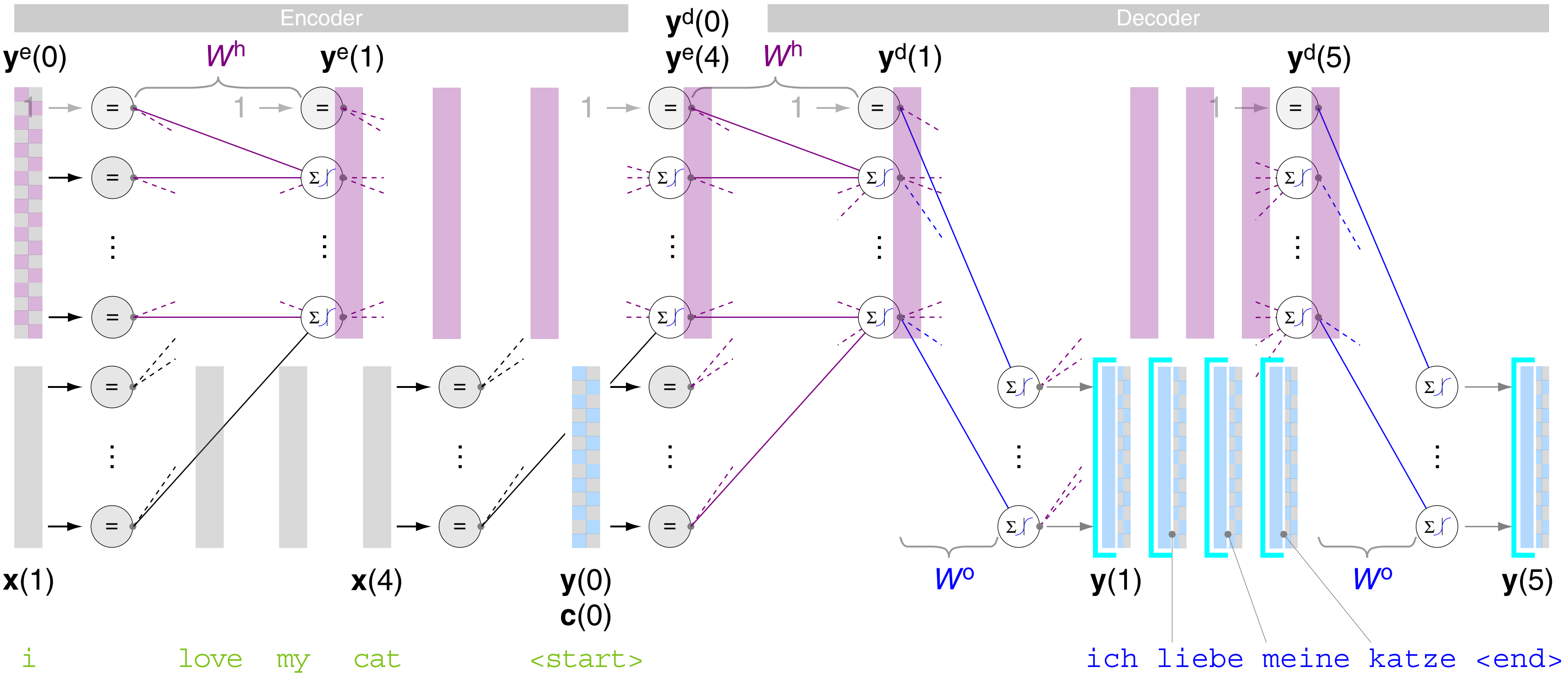
- The encoder hidden state $\mathbf{y}^{\text{e}}(t)$ represents the input sequence *up* to time step t, $[\mathbf{x}(1), \ldots, \mathbf{x}(t)]$.

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

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

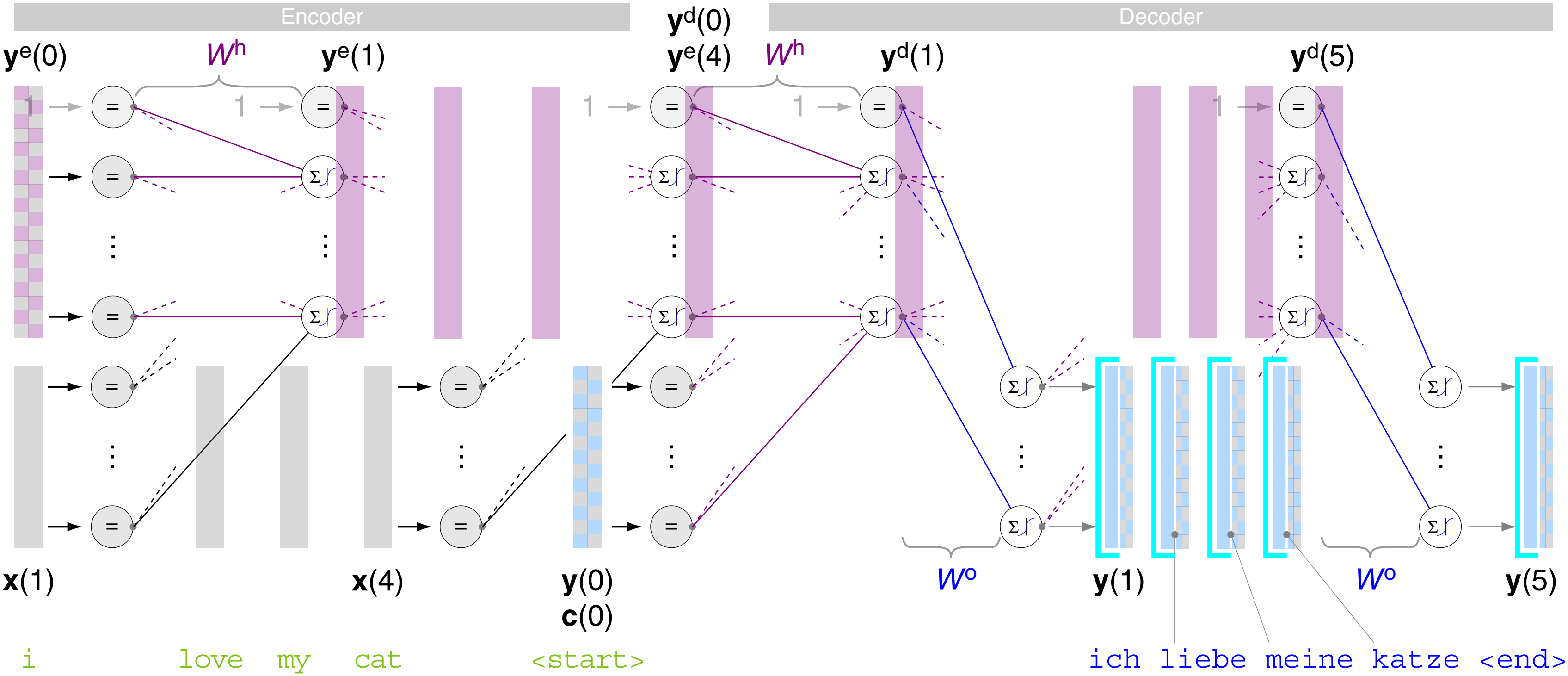
RNNs for Machine Translation

Sequence-to-Sequence RNNs are Conditional Language Models



RNNs for Machine Translation

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

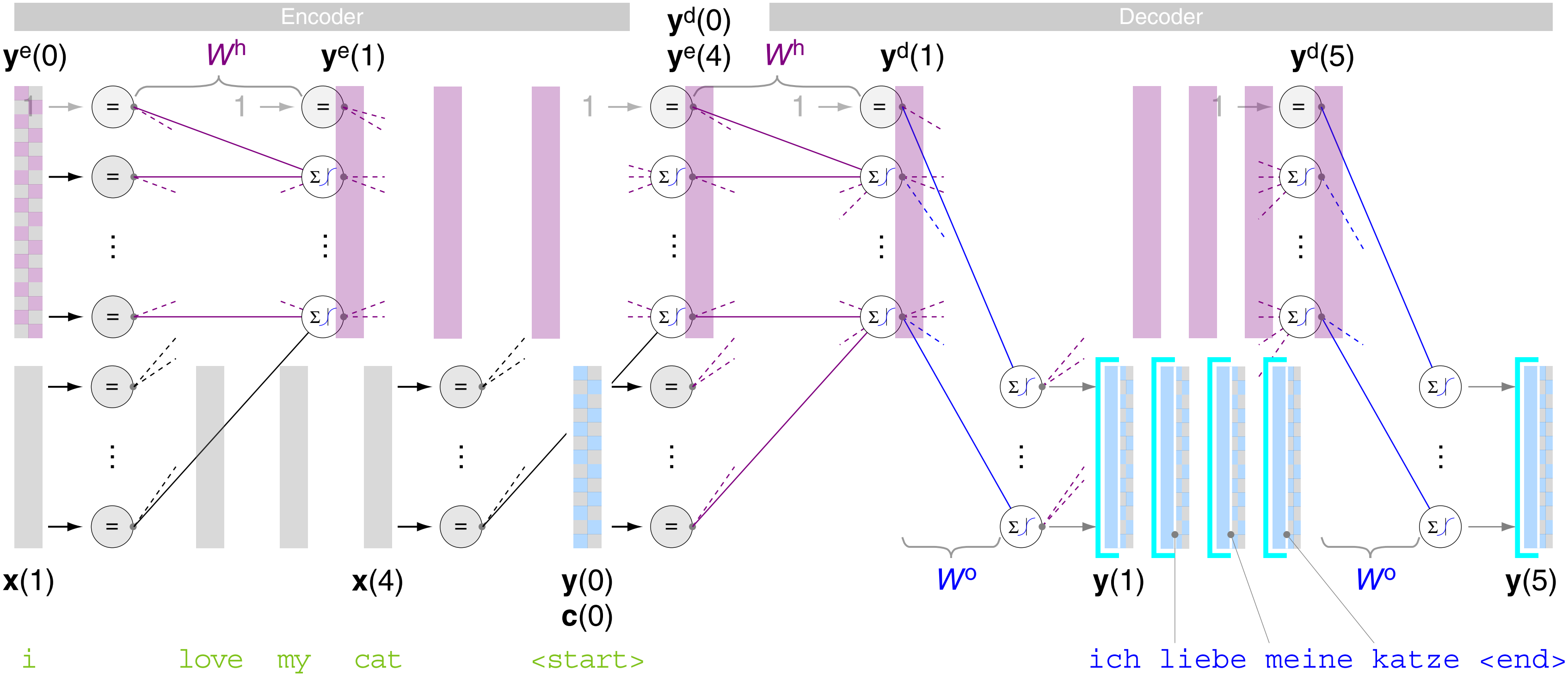


The sequence-to-sequence RNN directly calculates $p(y \mid x)$:

$$p(y \mid x) \;=\; p(y_1 \mid x) \cdot p(y_2 \mid y_1, x) \cdot p(y_3 \mid y_1, y_2, x) \cdot p(y_4 \mid y_1, y_2, y_3, x)$$

RNNs for Machine Translation

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



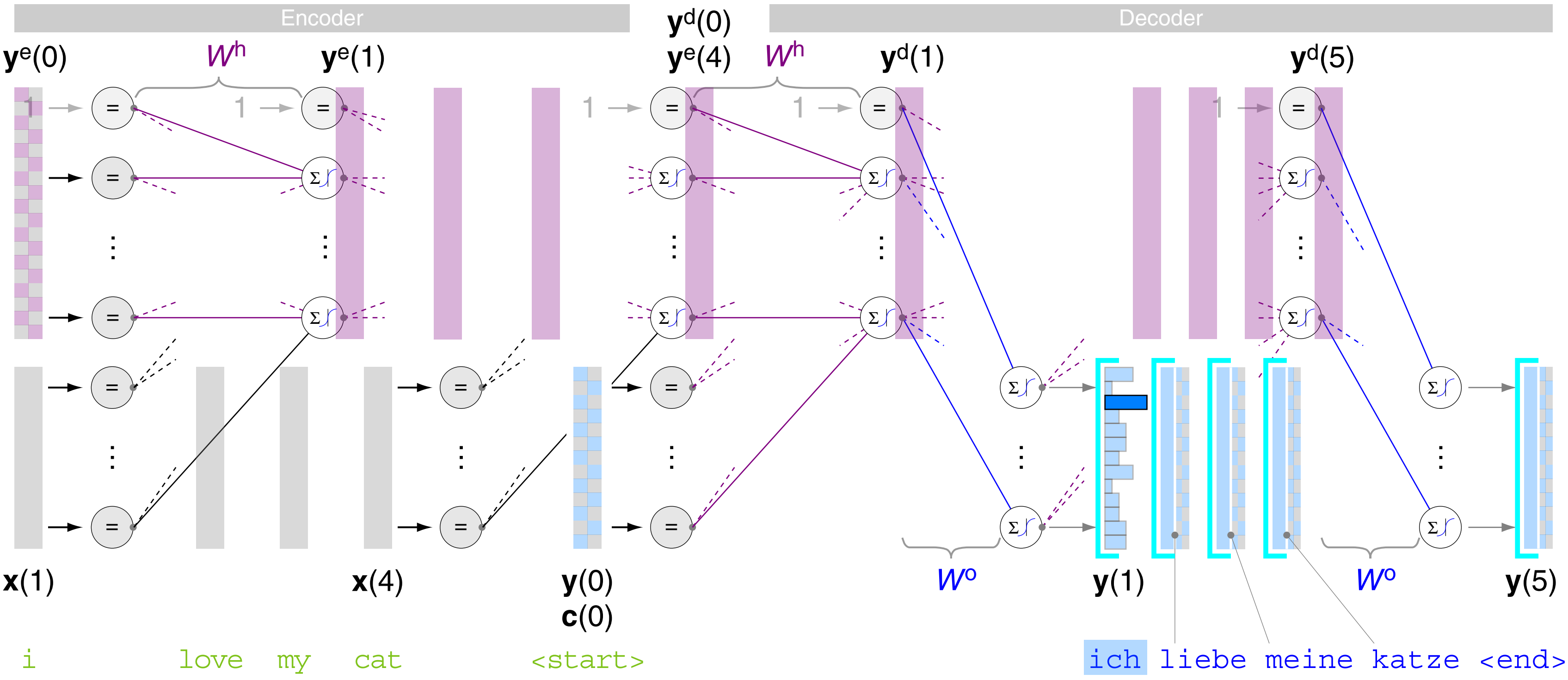
The sequence-to-sequence RNN directly calculates $p(y \mid x)$:

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

$$= p(\mathbf{y}(1) \mid \mathbf{x}, \mathbf{y}(0)) \cdot p(\mathbf{y}(2) \mid \mathbf{x}, \mathbf{y}(0), \mathbf{y}(1)) \cdot \ldots \cdot p(\mathbf{y}(5) \mid \mathbf{x}, \mathbf{y}(0), \ldots, \mathbf{y}(4))$$

RNNs for Machine Translation

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



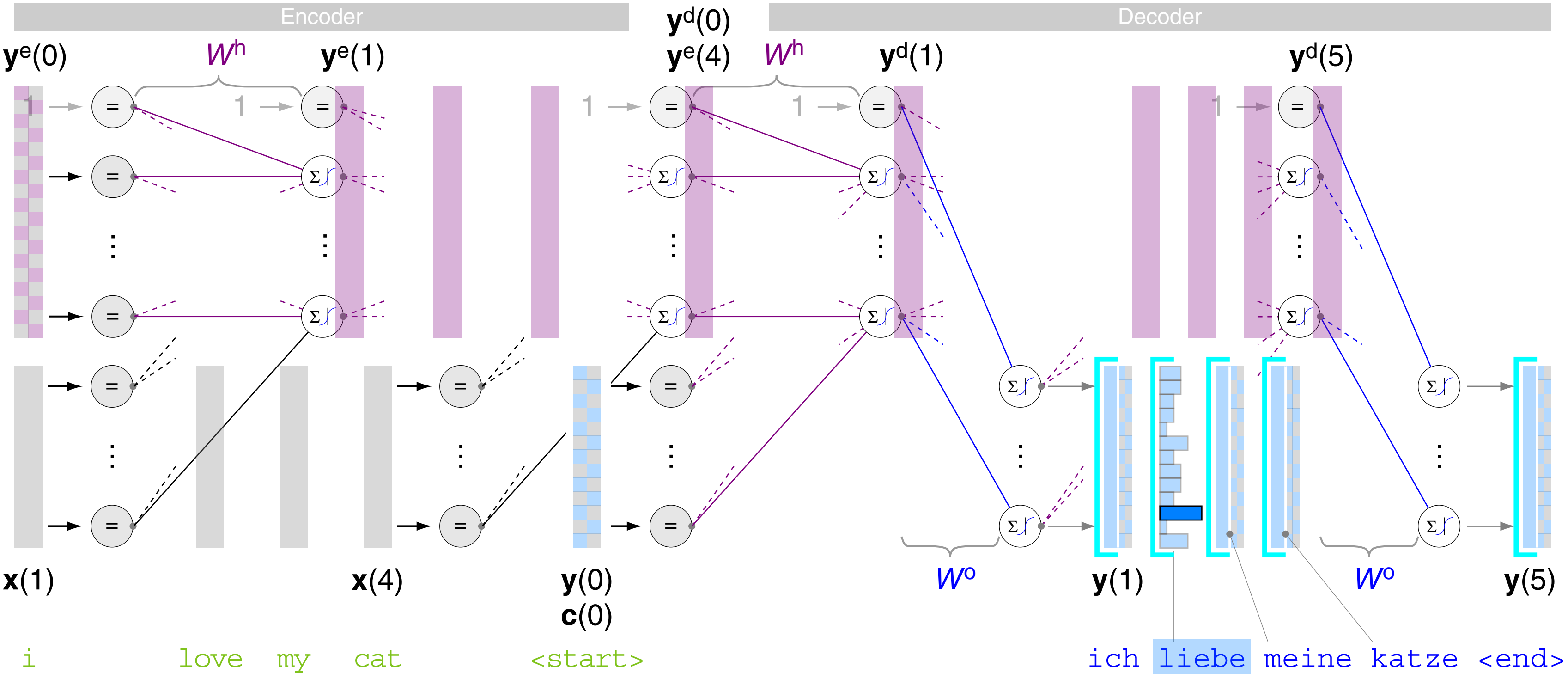
The sequence-to-sequence RNN directly calculates $p(y \mid x)$:

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

$$= \boxed{p(\mathbf{y}(1) \mid \mathbf{x}, \mathbf{y}(0))} \cdot p(\mathbf{y}(2) \mid \mathbf{x}, \mathbf{y}(0), \mathbf{y}(1)) \cdot \ldots \cdot p(\mathbf{y}(5) \mid \mathbf{x}, \mathbf{y}(0), \ldots, \mathbf{y}(4))$$

RNNs for Machine Translation

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



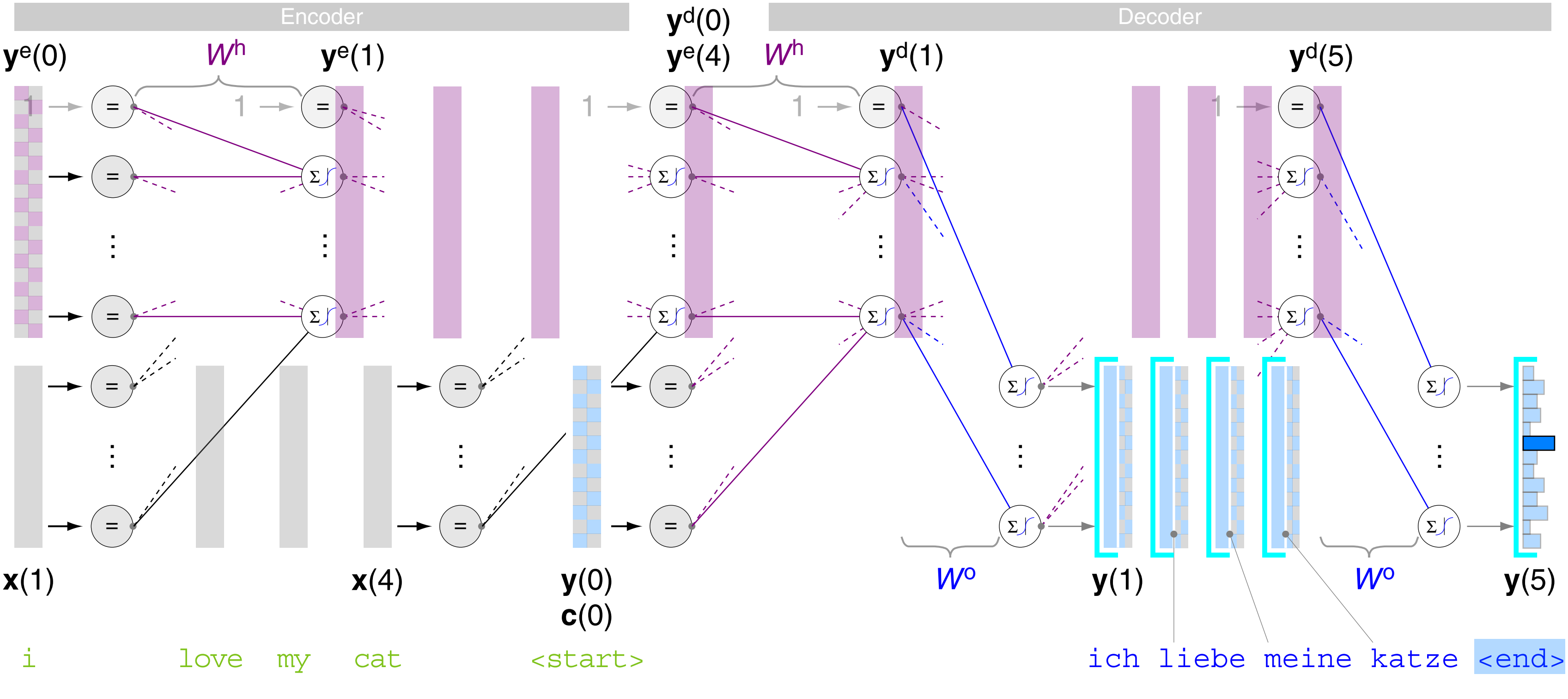
The sequence-to-sequence RNN directly calculates $p(y \mid x)$:

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

$$= p(\mathbf{y}(1) \mid \mathbf{x}, \mathbf{y}(0)) \cdot \boxed{p(\mathbf{y}(2) \mid \mathbf{x}, \mathbf{y}(0), \mathbf{y}(1))} \cdot \ldots \cdot p(\mathbf{y}(5) \mid \mathbf{x}, \mathbf{y}(0), \ldots, \mathbf{y}(4))$$

RNNs for Machine Translation

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



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

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

$$= \; p(\mathbf{y}(1) \mid \mathbf{x}, \mathbf{y}(0)) \cdot p(\mathbf{y}(2) \mid \mathbf{x}, \mathbf{y}(0), \mathbf{y}(1)) \cdot \ldots \cdot \boxed{p(\mathbf{y}(5) \mid \mathbf{x}, \mathbf{y}(0), \ldots, \mathbf{y}(4))}$$

Remarks:

- Each output vector $\mathbf{y}(t)$ corresponds to a probability distribution over Vocabulary^d (recall the $\boldsymbol{\sigma}_1$-function). Here, the illustration of generation (aka decoding) steps shows an argmax-operation on each $\mathbf{y}(t)$, called "greedy decoding" : the word with the highest probability is chosen.

- To maximize $\prod_{t=1}^{\tau} p\left(\mathbf{y}(t) \mid \mathbf{x}, \mathbf{y}(0), \ldots, \mathbf{y}(t-1)\right)$, 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 multilayer RNN, which is able to compute more complex representations. The idea is that the lower (higher) RNNs should compute 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]