

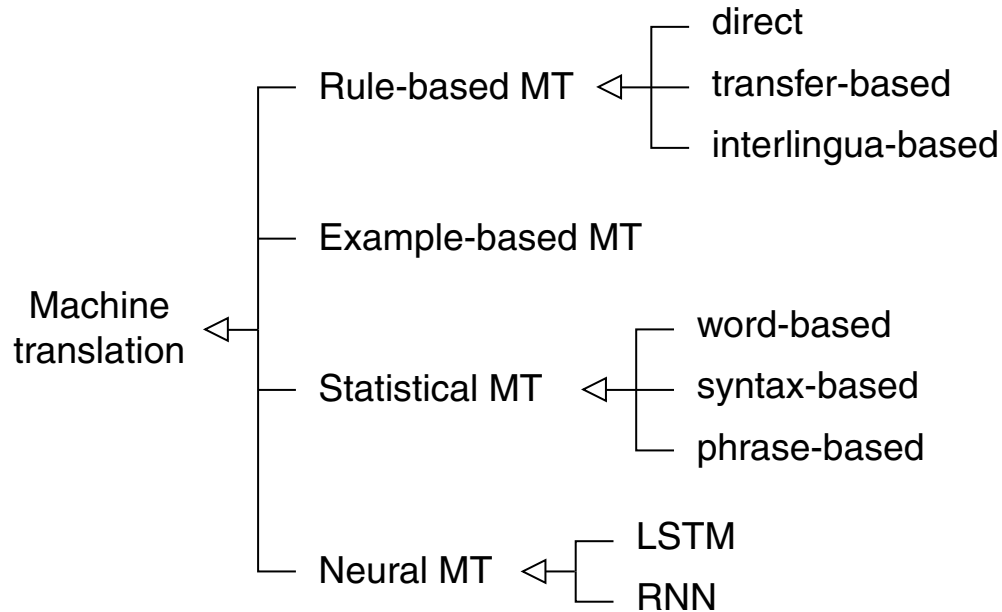
# Chapter ML:IX (continued)

## IX. Deep Learning

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

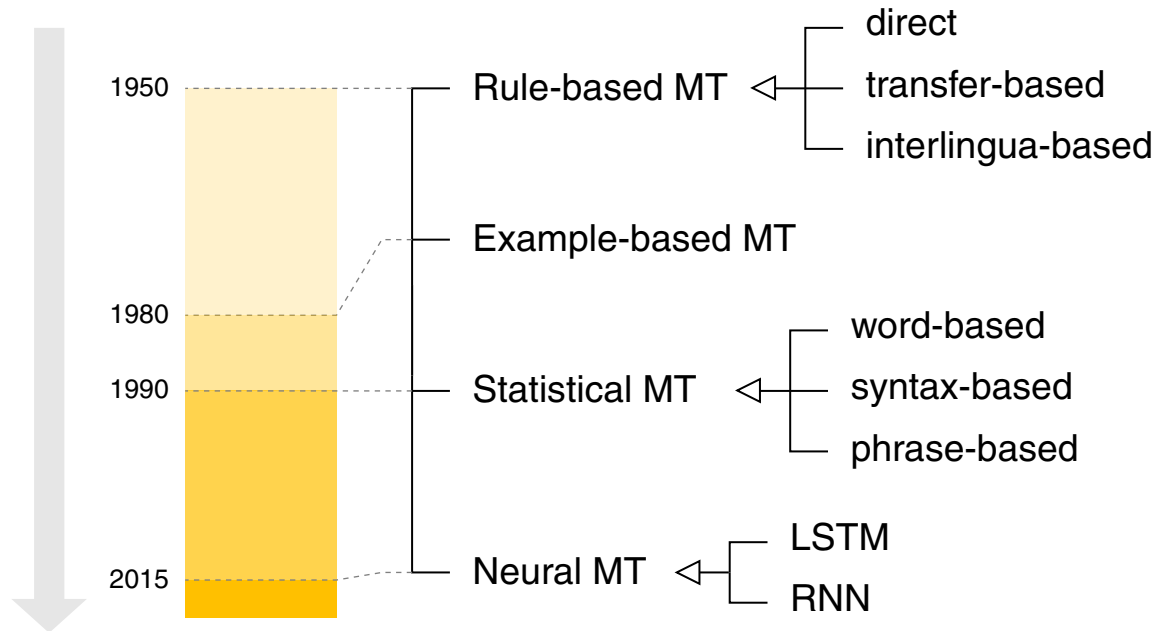
# RNNs for Machine Translation

## Statistical Machine Translation (SMT)



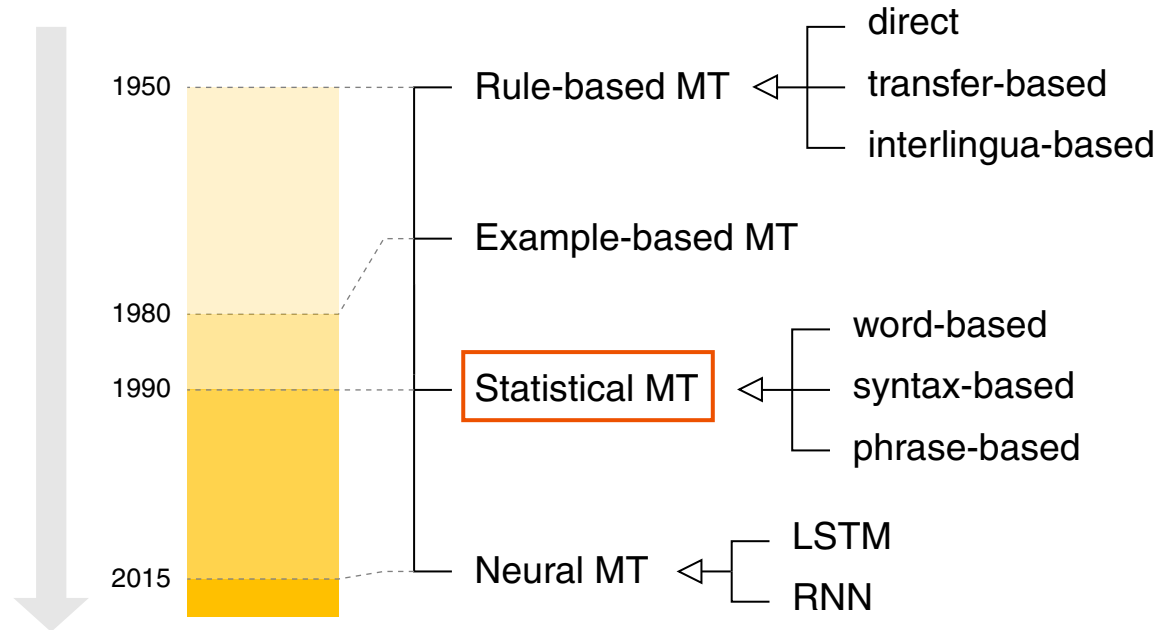
# RNNs for Machine Translation

## Statistical Machine Translation (SMT) (continued)



# RNNs for Machine Translation

## Statistical Machine Translation (SMT) (continued)

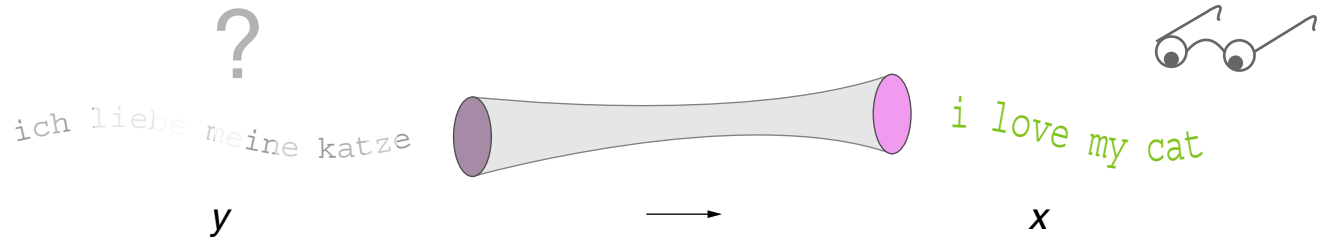


“Noisy channel” model applied to SMT:

Learn from a parallel corpus  $D$  a probabilistic model,  $P(Y | X)$ , 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)).

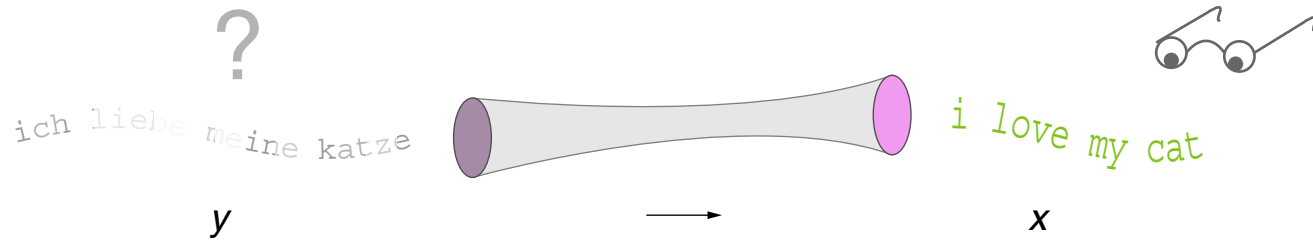
# RNNs for Machine Translation

## Statistical Machine Translation (SMT) (continued)



# RNNs for Machine Translation

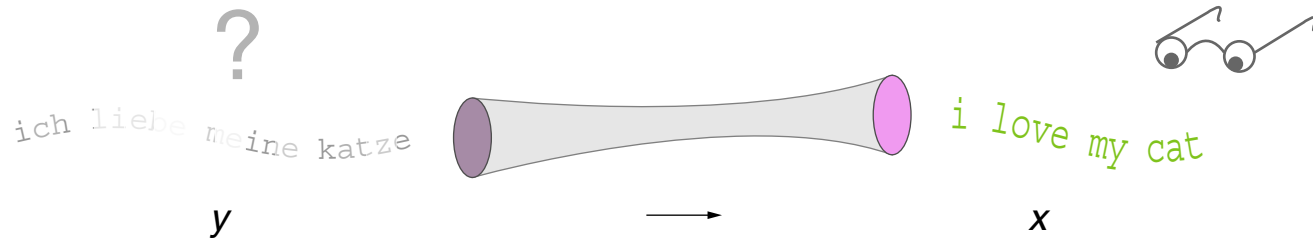
## Statistical Machine Translation (SMT) (continued)



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

# RNNs for Machine Translation

## Statistical Machine Translation (SMT) (continued)



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

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

$$p(y \mid x) \rightarrow \max$$

## Remarks:

- ❑ Here,  $x$  denotes a sentence  $x_1x_2 \dots x_{|x|}$ . Likewise,  $y$  denotes a sentence  $y_1y_2 \dots y_{|y|}$ . 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  $\mathbf{y}^e(T^e)$  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  $p(y \mid x)$ :

$$\operatorname{argmax}_y p(y \mid x) = \operatorname{argmax}_y p(x \mid y) \cdot p(y) \quad \Leftarrow$$

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

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

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

# 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  $p(y \mid x)$ :

$$\operatorname{argmax}_y p(y \mid x) = \operatorname{argmax}_y p(x \mid y) \cdot \boxed{p(y)} \quad \Leftarrow$$
$$P(Y \mid X) = \frac{P(X \mid Y) \cdot P(Y)}{P(X)}$$

$X \hat{=} X=x, \quad x \hat{=} \text{English sentence}$   
 $Y \hat{=} Y=y, \quad y \hat{=} \text{German sentence}$

1.  $p(y)$  is called “language model” and takes care of the *fluency* in the target language. It is modeled as  $p(y_1, \dots, y_m) = \prod_{i=1}^m p(y_i \mid y_{i-(n-1)}, \dots, y_{i-1})$ . 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.  $\operatorname{argmax}_y$  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  $p(y \mid x)$ :

$$\operatorname{argmax}_y p(y \mid x) = \operatorname{argmax}_y p(x \mid y) \cdot p(y) \quad \Leftarrow$$
$$P(Y \mid X) = \frac{P(X \mid Y) \cdot P(Y)}{P(X)}$$

$X \hat{=} X=x, \quad x \hat{=} \text{English sentence}$   
 $Y \hat{=} Y=y, \quad y \hat{=} \text{German sentence}$

1.  $p(y)$  is called “language model” and takes care of the *fluency* in the target language. It is modeled as  $p(y_1, \dots, y_m) = \prod_{i=1}^m p(y_i \mid y_{i-(n-1)}, \dots, y_{i-1})$ . 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.  $\operatorname{argmax}_y$  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  $p(y \mid x)$ :

$$\operatorname{argmax}_y p(y \mid x) = \operatorname{argmax}_y p(x \mid y) \cdot p(y) \quad \Leftarrow$$
$$P(Y \mid X) = \frac{P(X \mid Y) \cdot P(Y)}{P(X)}$$

$X \hat{=} X=x, \quad x \hat{=} \text{English sentence}$   
 $Y \hat{=} Y=y, \quad y \hat{=} \text{German sentence}$

1.  $p(y)$  is called “language model” and takes care of the *fluency* in the target language. It is modeled as  $p(y_1, \dots, y_m) = \prod_{i=1}^m p(y_i \mid y_{i-(n-1)}, \dots, y_{i-1})$ . 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.  $\operatorname{argmax}_y$  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,  $p(y) = p(y_1, y_2, \dots, y_m)$  denotes the probability of the event to observe the sentence  $y = y_1 y_2 \dots y_m$ , 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,  $p(y_1, y_2, \dots, y_m)$  is a short form of  $P(Y_1=y_1, Y_2=y_2, \dots, Y_m=y_m)$ , 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 are complex and given as latent variables only.

# RNNs for Machine Translation

## Neural Machine Translation (NMT)

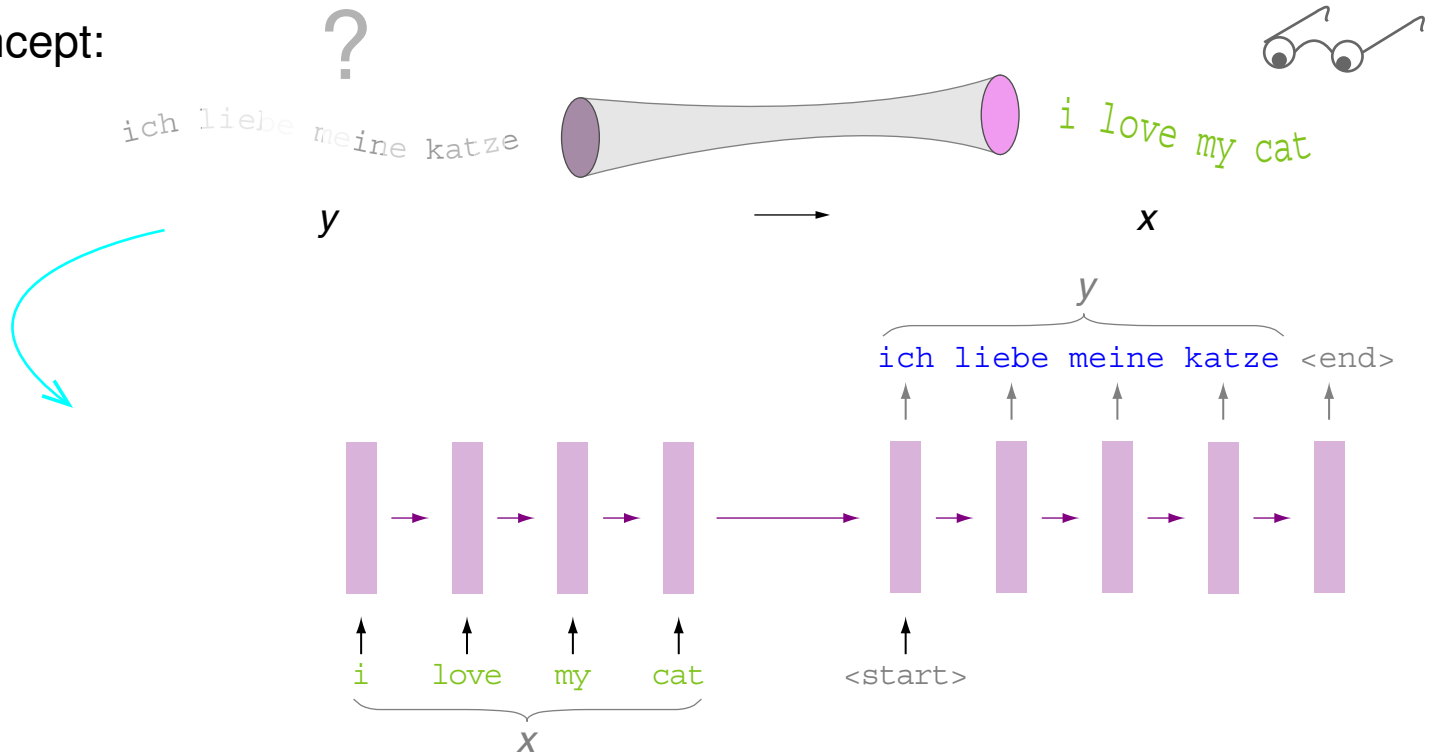
### Concept:

- ❑ 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)

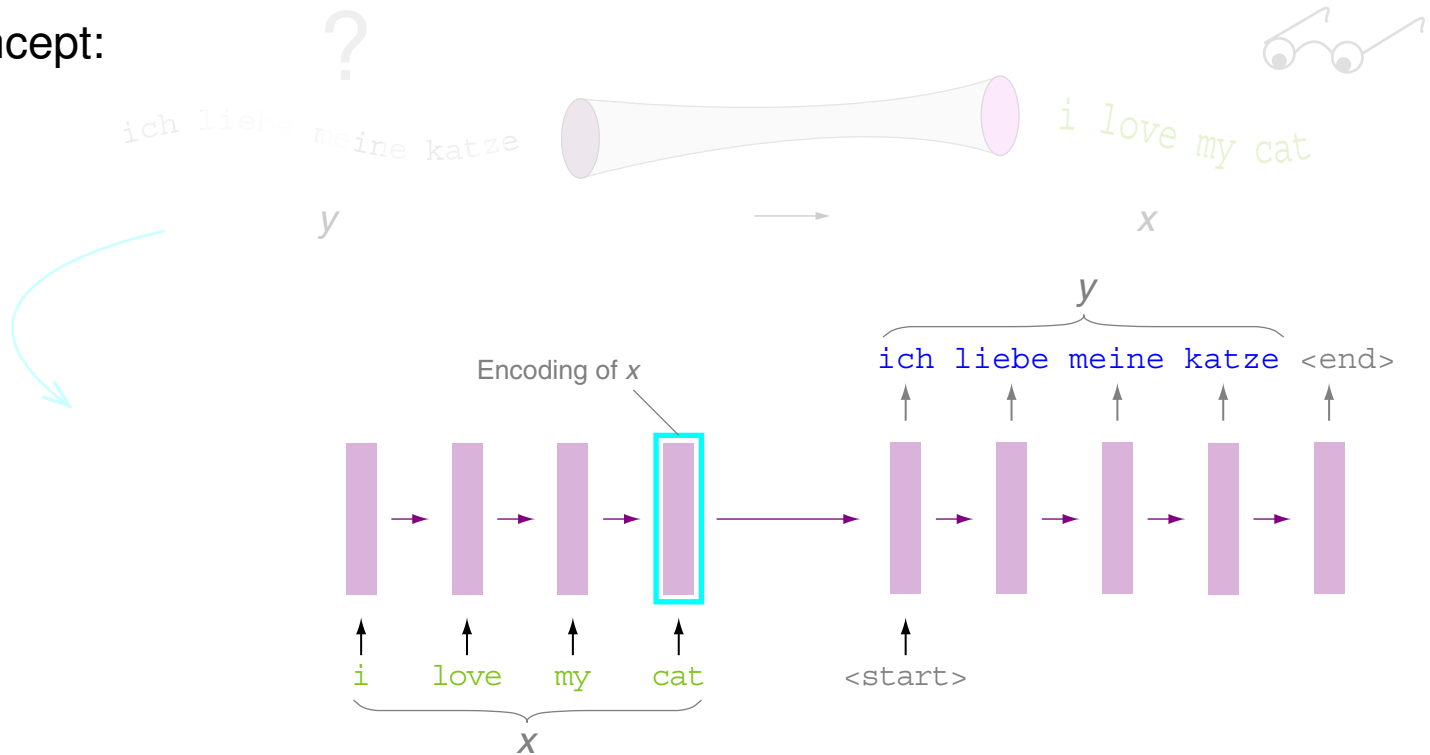
Concept:



# RNNs for Machine Translation

## Neural Machine Translation (NMT) (continued)

Concept:

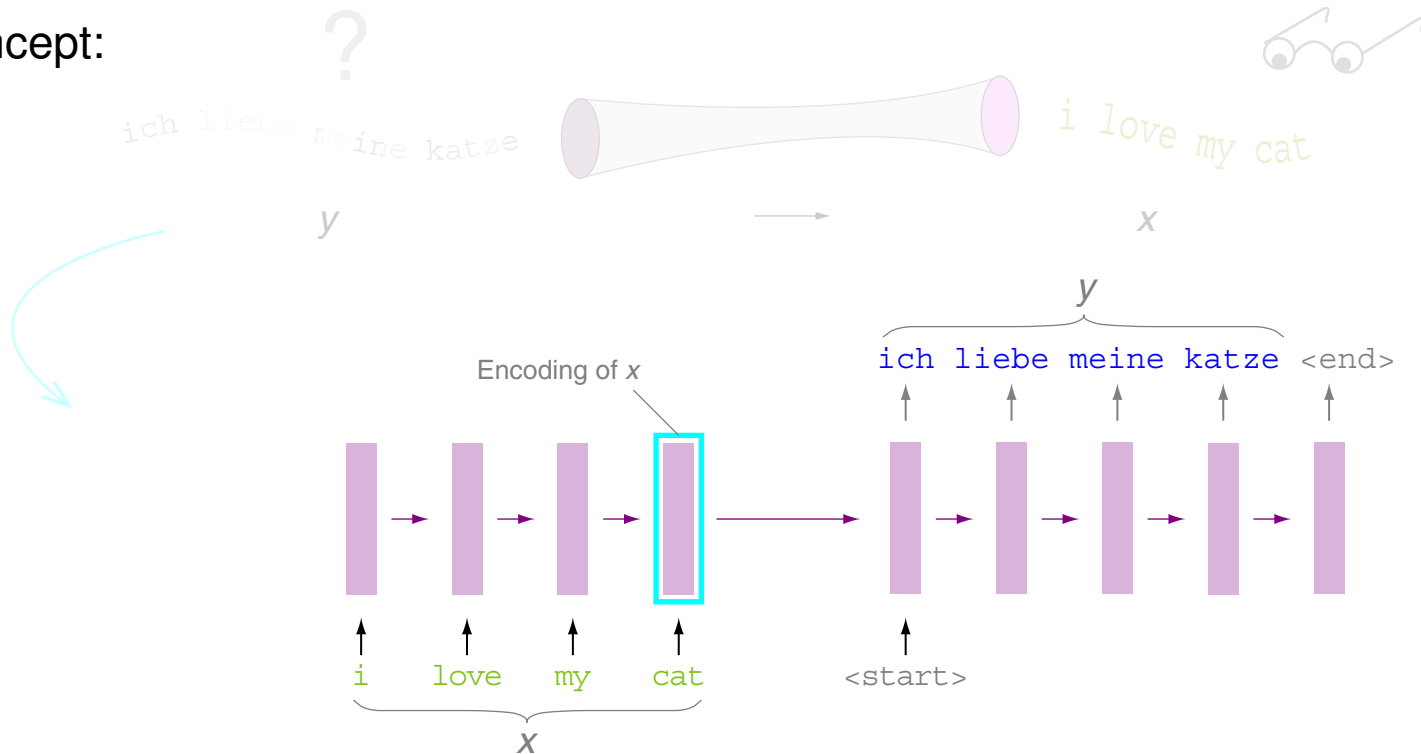




# RNNs for Machine Translation

## Neural Machine Translation (NMT) (continued)

Concept:



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

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

## 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  $y_1, \dots, y_{t-1}$ . (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  $\mathbf{y}^e(T^e)$  represents the RNN encoding of the source sentence  $x$ . In particular,
  - the word  $x_t$  from a source (input) sentence  $x$  is denoted as  $\mathbf{x}(t)$ ,
  - the word  $y_t$  from a output sentence is denoted as  $\mathbf{y}(t)$ ,
  - the word  $y_t$  from a target sentence  $y$  is 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. Similarly, 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  $\rightarrow$  class

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

i love my cat  $\rightarrow \oplus$

(S2) class  $\rightarrow$  sequence

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

$\oplus \rightarrow$  i love my cat

(S3) **sequence  $\rightarrow$  sequence**

**English sentence  $\rightarrow$  German sentence**

i love my cat  $\rightarrow$  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<sup>e</sup>: ( allowed and are cat cats dogs i is it lap love my not  
raining water )

Vocabulary<sup>d</sup>: ( 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<sup>e</sup>: ( allowed and are cat cats dogs i is it lap love my not  
raining water )

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

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

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

# 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<sup>e</sup>: ( allowed and are cat cats dogs i is it lap love my not raining water )

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

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

Output:  $[y(1), y(2), y(3), \dots, y(\tau^d)], y(0) \equiv c(0) \hat{=} \text{<start>}, y(\tau) \hat{=} c(5) \hat{=} \text{<end>}$

# 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<sup>e</sup>: ( allowed and are cat cats dogs i is it lap love my not raining water )

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

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

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

# 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<sup>e</sup>: ( allowed and are cat cats dogs i is it lap love my not raining water )

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

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

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



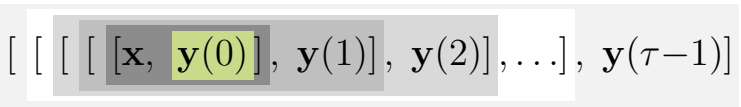
# RNNs for Machine Translation

## (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<sup>e</sup>: ( allowed and are cat cats dogs i is it lap love my not raining water )

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

Input:   $\mathbf{x} = \begin{bmatrix} \begin{pmatrix} 0 \\ \vdots \\ 1 \\ 0 \\ \vdots \end{pmatrix}, \begin{pmatrix} 0 \\ \vdots \\ 1 \\ 0 \\ \vdots \end{pmatrix}, \begin{pmatrix} 0 \\ \vdots \\ 1 \\ 0 \\ \vdots \end{pmatrix}, \begin{pmatrix} 0 \\ \vdots \\ 1 \\ 0 \\ \vdots \end{pmatrix} \end{bmatrix} \hat{=} \text{I love my cat}$

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

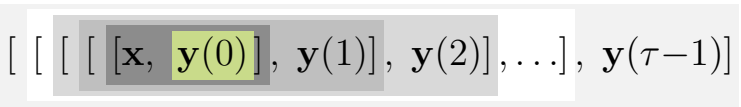
# RNNs for Machine Translation

## (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<sup>e</sup>: ( allowed and are cat cats dogs i is it lap love my not raining water )

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

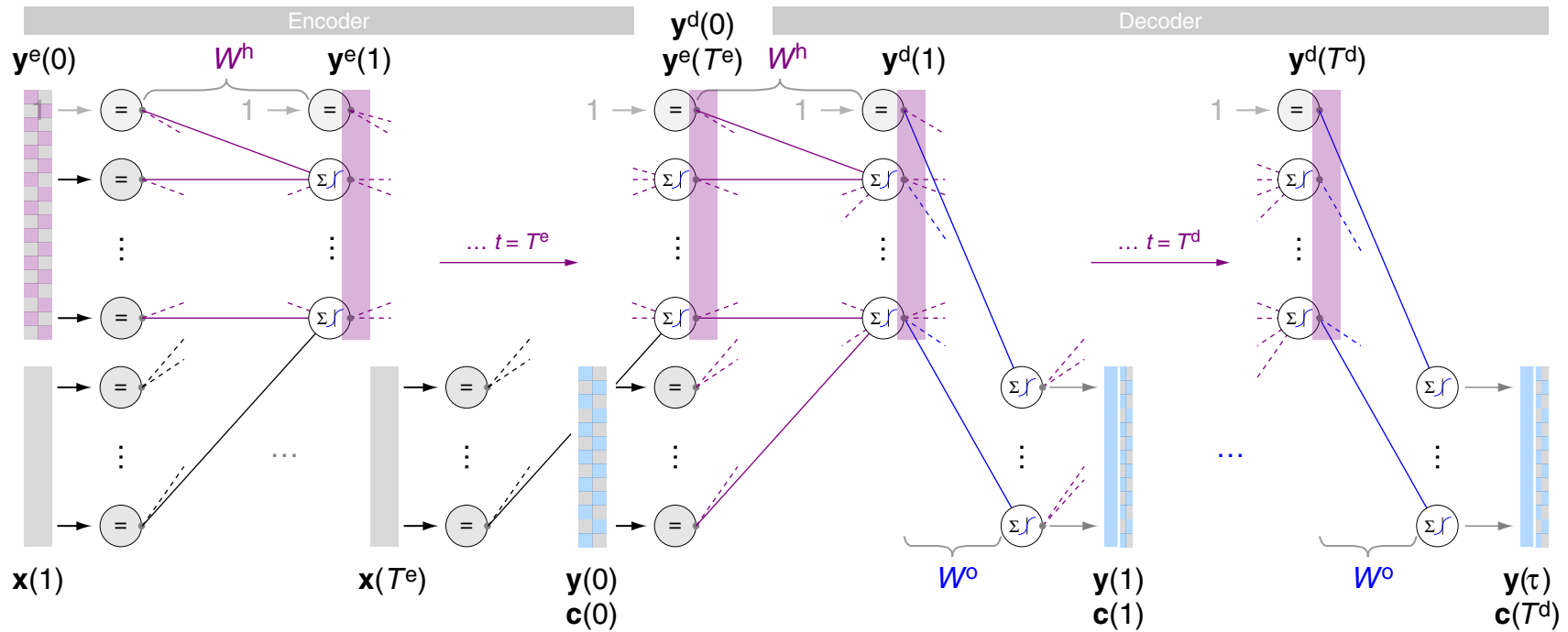
Input:   $\mathbf{x} = \begin{bmatrix} \begin{pmatrix} 0 \\ \vdots \\ 1 \\ 0 \\ \vdots \end{pmatrix}, \begin{pmatrix} 0 \\ \vdots \\ 1 \\ 0 \\ \vdots \end{pmatrix}, \begin{pmatrix} 0 \\ \vdots \\ 1 \\ 0 \\ \vdots \end{pmatrix}, \begin{pmatrix} 0 \\ \vdots \\ 1 \\ 0 \\ \vdots \end{pmatrix} \end{bmatrix} \hat{=} \text{I love my cat}$

Output:  $[\mathbf{y}(1), \mathbf{y}(2), \mathbf{y}(3), \dots, \mathbf{y}(\tau^d)], \quad \mathbf{y}(0) \hat{=} \mathbf{c}(0) \hat{=} \text{<start>}, \quad \mathbf{y}(\tau) \hat{=} \mathbf{c}(5) \hat{=} \text{<end>}$

Target:  $[\mathbf{c}(1), \dots, \mathbf{c}(5)] = \begin{bmatrix} \begin{pmatrix} 0 \\ \vdots \\ 1 \\ 0 \\ \vdots \end{pmatrix}, \begin{pmatrix} 0 \\ \vdots \\ 1 \\ 0 \\ \vdots \end{pmatrix}, \begin{pmatrix} 0 \\ \vdots \\ 1 \\ 0 \\ \vdots \end{pmatrix}, \begin{pmatrix} 0 \\ \vdots \\ 1 \\ 0 \\ \vdots \end{pmatrix}, \begin{pmatrix} 0 \\ \vdots \\ 1 \\ 0 \\ \vdots \end{pmatrix} \end{bmatrix} \hat{=} \text{Ich liebe meine Katze}$

# RNNs for Machine Translation

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



Input:

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

Output:

$$y(t) = \sigma_1(W^o y^d(t)), t = 1, \dots, \tau$$

Hidden:

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

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

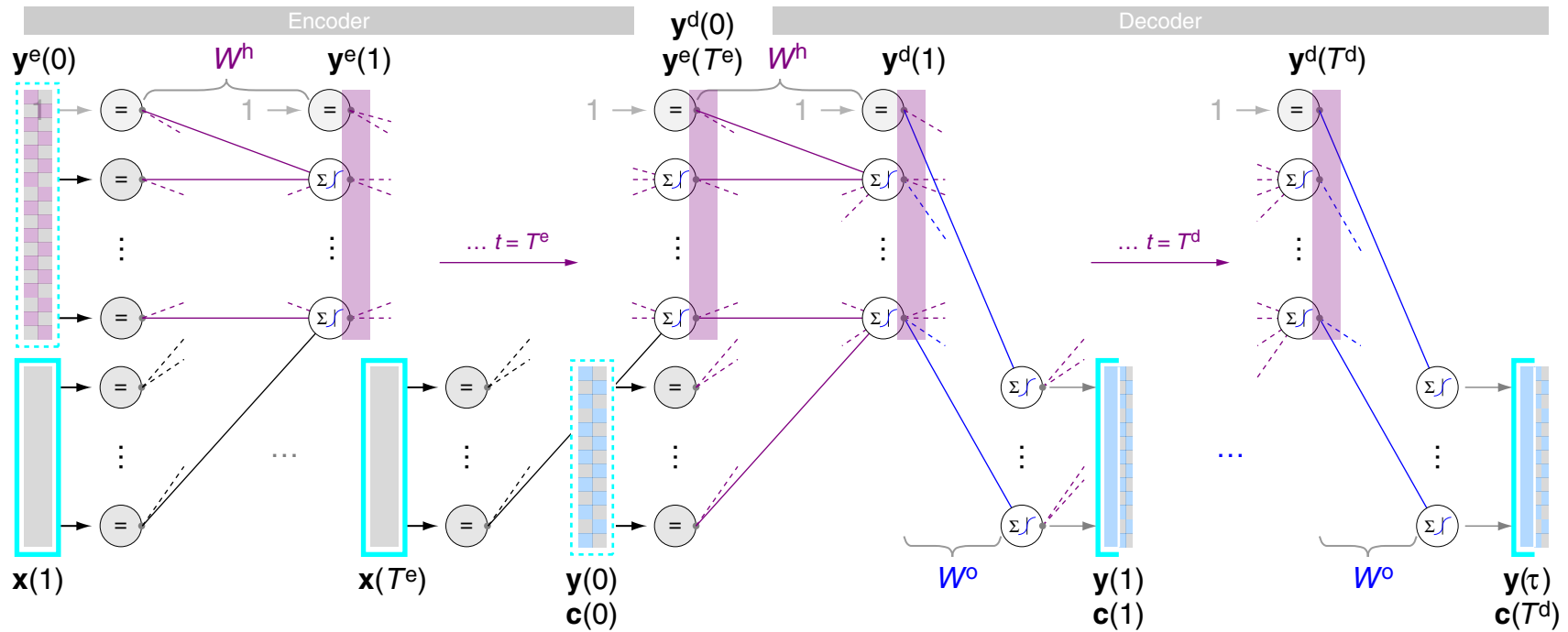
Target:

$$[c(1), \dots, c(T)]$$

$$c(T) \hat{=} \langle \text{end} \rangle$$

# RNNs for Machine Translation

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



Input:

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

Output:

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

Hidden:

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

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

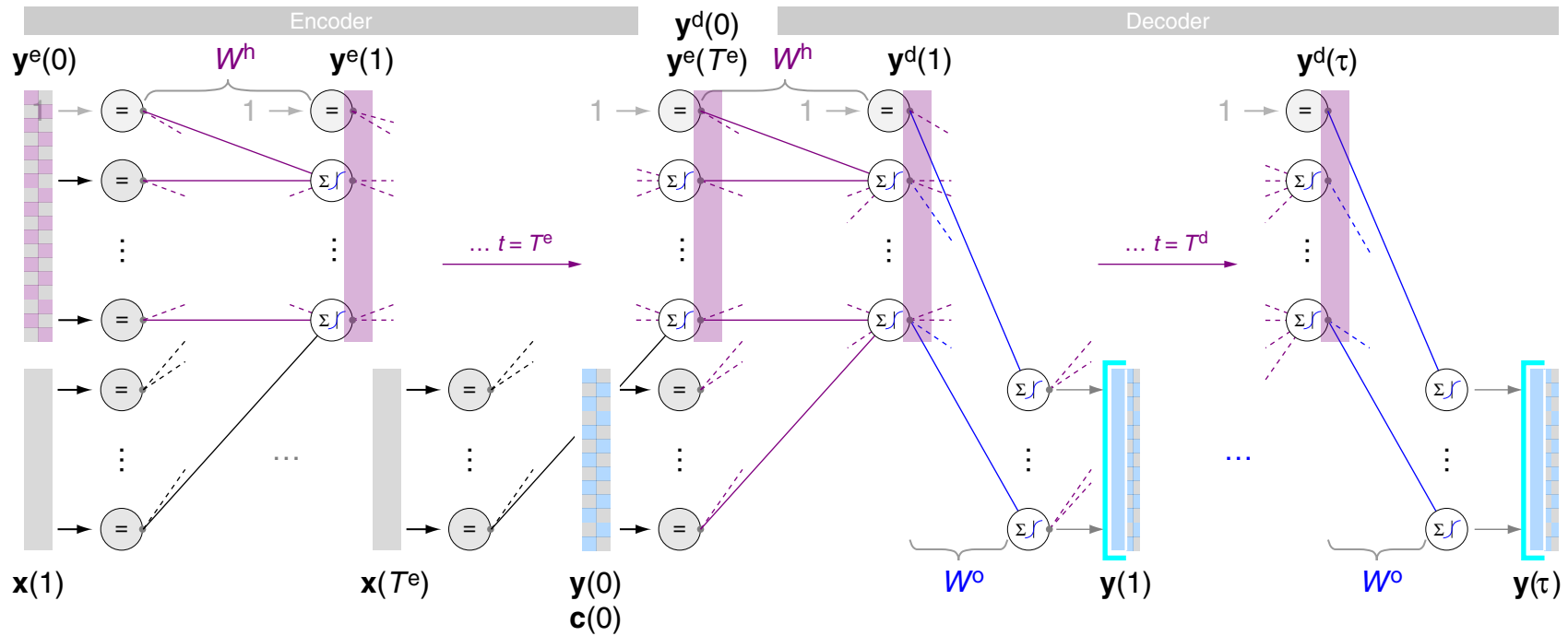
Target:

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

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

# RNNs for Machine Translation

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



Input:

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

Output:

$$y(t) = \sigma_1(W^o y^d(t)), t = 1, \dots, \tau$$

Hidden:

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

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

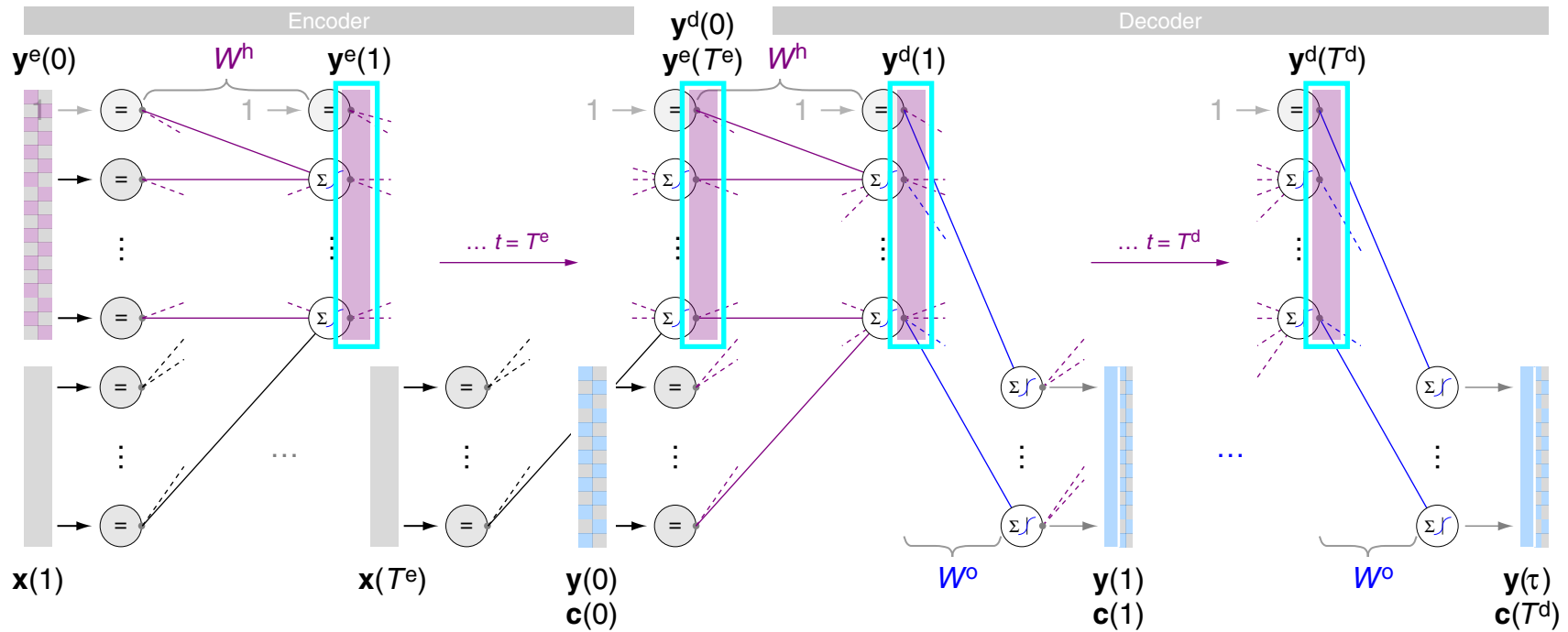
Target:

$$[c(1), \dots, c(T)]$$

$$c(T) \hat{=} \langle \text{end} \rangle$$

# RNNs for Machine Translation

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



Input:

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

Output:

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

Hidden:

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

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

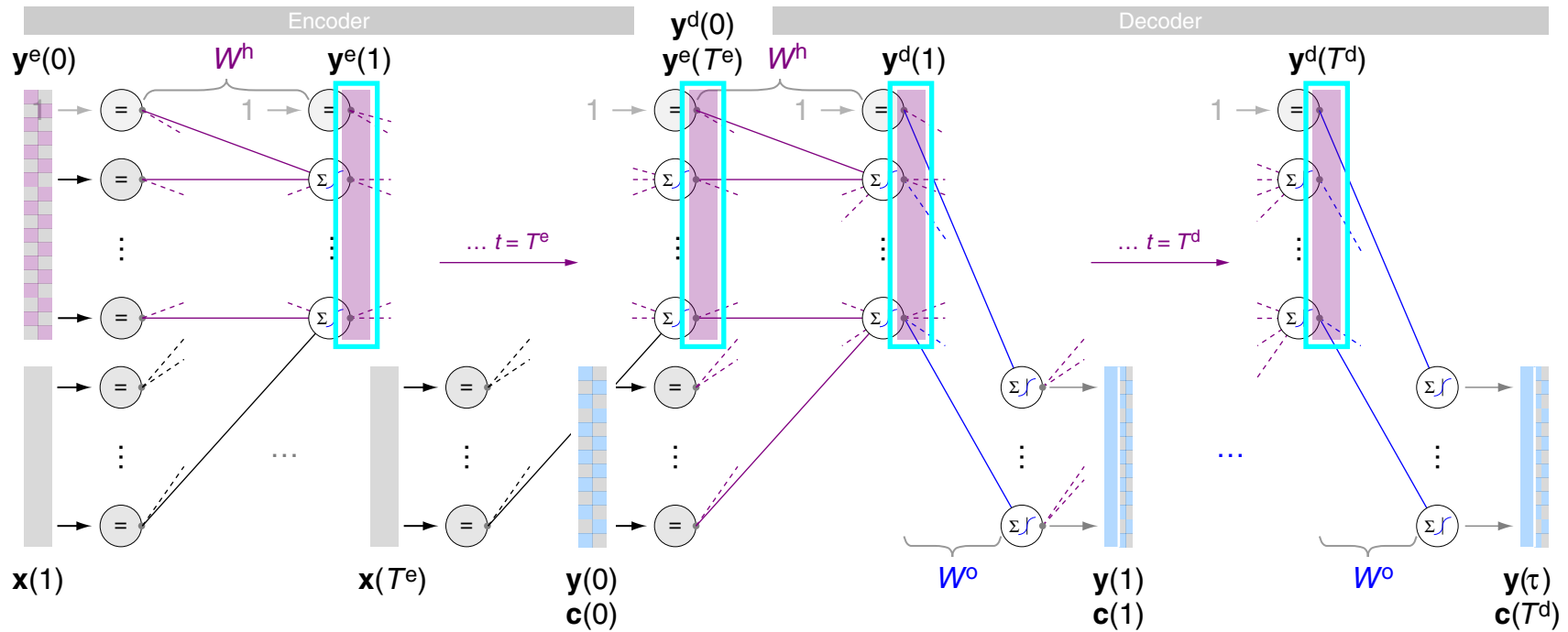
Target:

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

$$\mathbf{c}(T) \hat{=} \langle \text{end} \rangle$$

# RNNs for Machine Translation

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



Input:

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

Output:

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

Hidden:

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

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

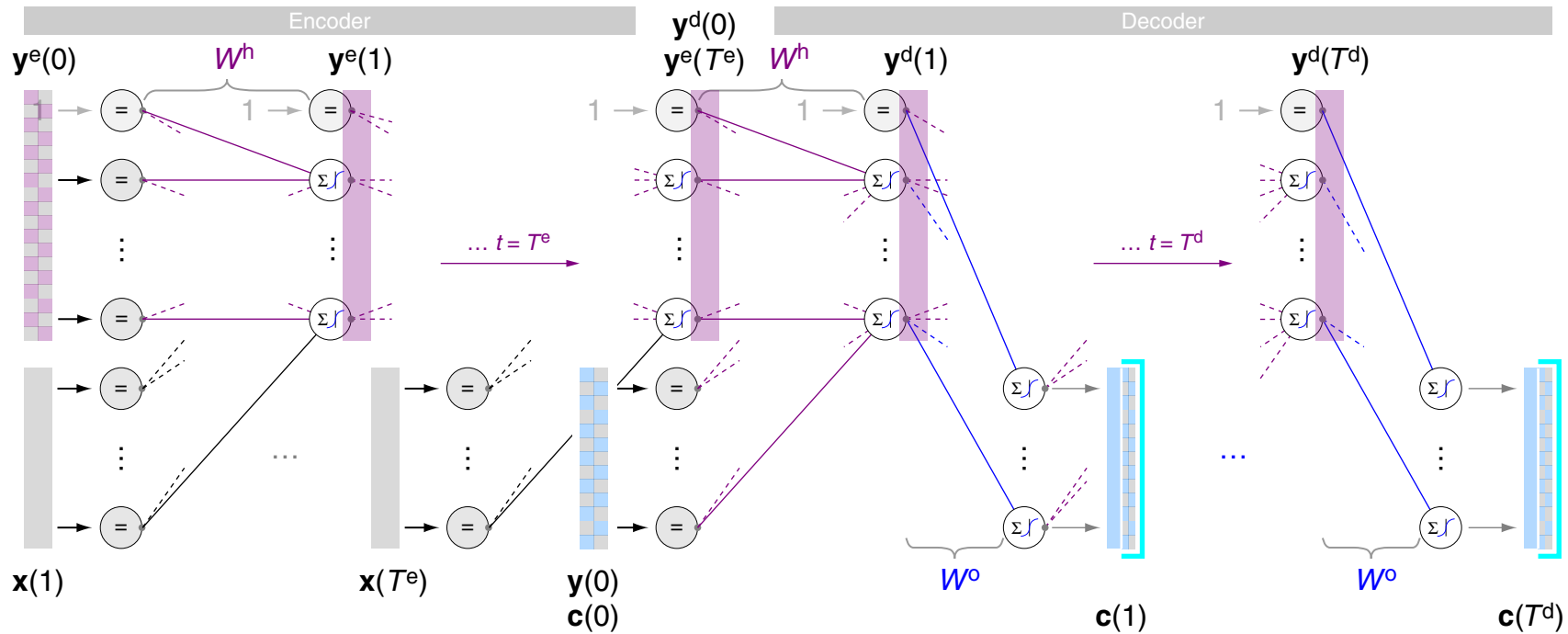
Target:

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

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

# RNNs for Machine Translation

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



Input:

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

Output:

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

Hidden:

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

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

Target:

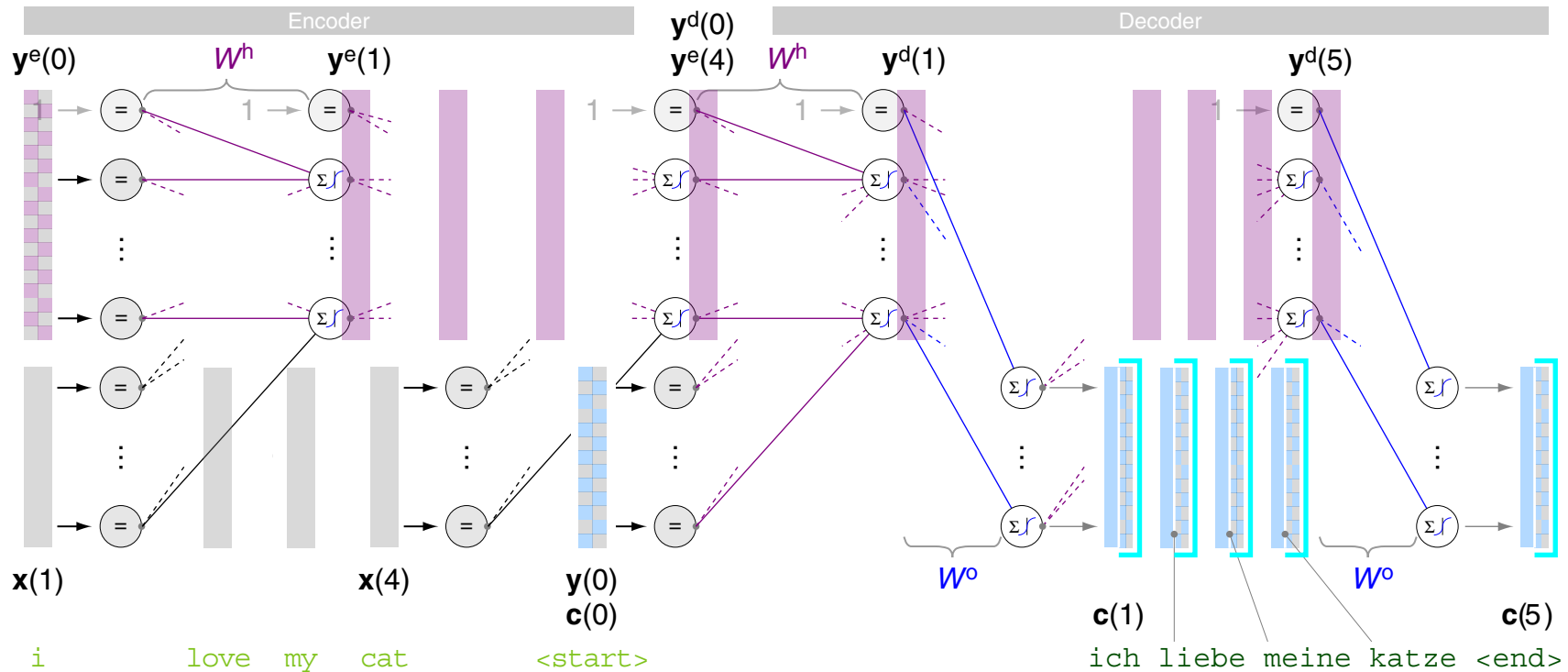
$$[\mathbf{c}(1), \dots, \mathbf{c}(T)]$$

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



# RNNs for Machine Translation

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



Input:

$x, [y(1), \dots, y(4)]$

Output:

$y(t) = \sigma_1(W^o y^d(t)), t = 1, \dots, 5$

Hidden:

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

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

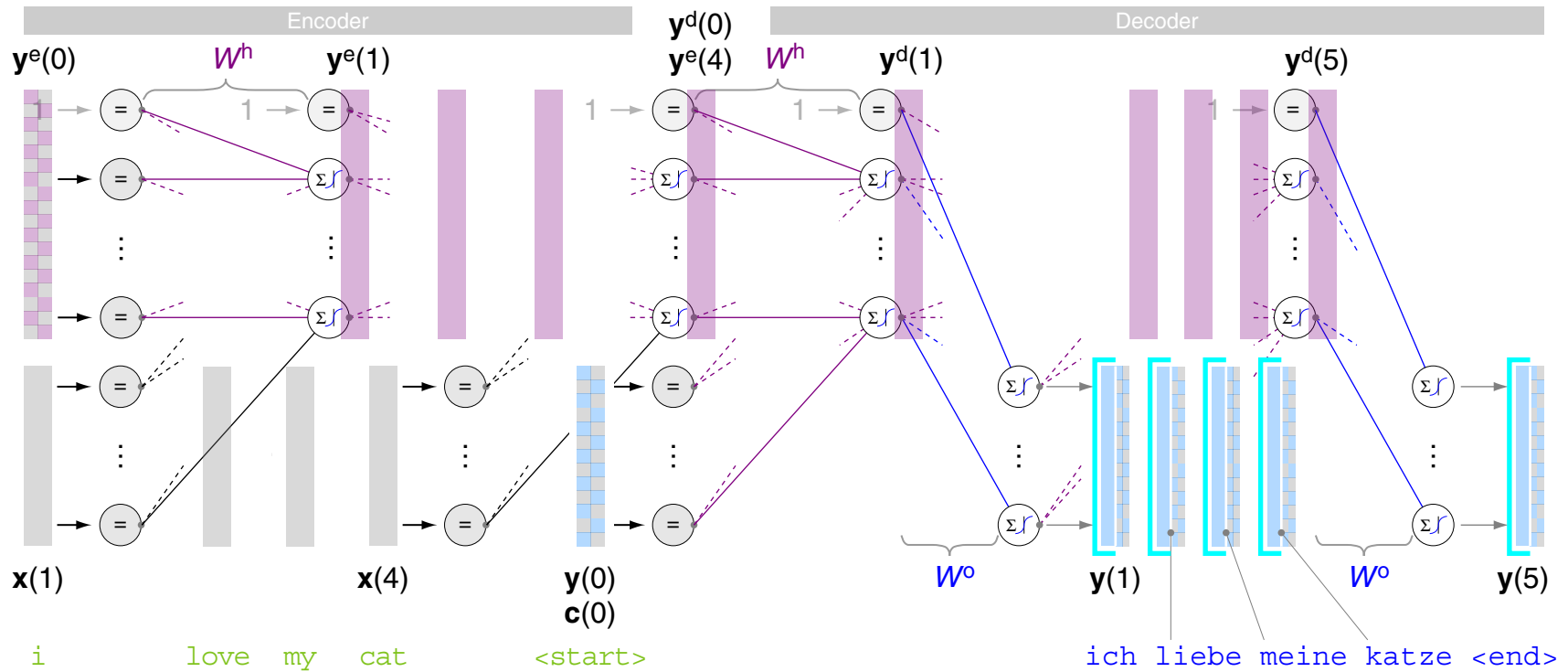
Target:

$[c(1), \dots, c(5)]$

$c(5) \hat{=} \text{<end>}$

# RNNs for Machine Translation

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



Input:

$x, [y(1), \dots, y(4)]$

Output:

$y(t) = \sigma_1(W^o y^d(t)), t = 1, \dots, 5$

Hidden:

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

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

Target:

$[c(1), \dots, c(5)]$

$c(5) \hat{=} \text{<end>}$

## Remarks:

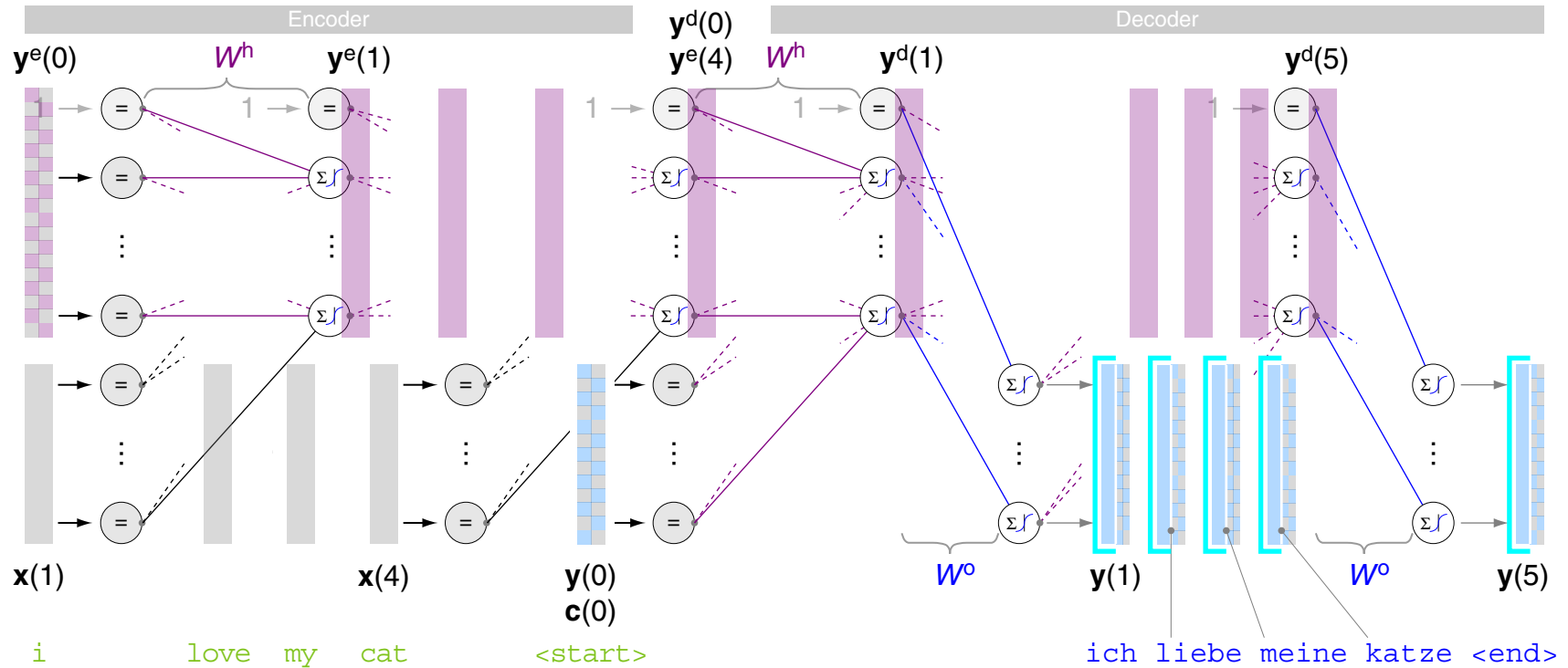
- ❑ The final encoder hidden state,  $\mathbf{y}^e(T^e)$ , represents the encoding of the source sentence.  $\mathbf{y}^e(T^e)$  is unified with the first decoder hidden state,  $\mathbf{y}^d(0)$ .
- ❑ The encoder hidden state  $\mathbf{y}^e(t)$  represents the input sequence *up* to time step  $t$ ,  $[\mathbf{x}(1), \dots, \mathbf{x}(t)]$ .
- ❑ The decoder hidden state  $\mathbf{y}^d(t)$  represents the entire input sequence  $[\mathbf{x}(1), \dots, \mathbf{x}(T^e)]$ , as well as the output sequence *up* to time step  $t-1$ ,  $[\mathbf{y}(1), \dots, \mathbf{y}(t-1)]$ .
- ❑ Note that, as before, we are given a model function  $\mathbf{y}()$ , which maps some input (actually, a *sequence* of feature vectors,  $[\mathbf{x}(1), \dots, \mathbf{x}(T^e)]$ ) to some output (a sequence of output vectors,  $[\mathbf{y}(1), \dots, \mathbf{y}(T^d)]$ ).

## Sequence-to-Sequence RNNs are Conditional Language Models



# RNNs for Machine Translation

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

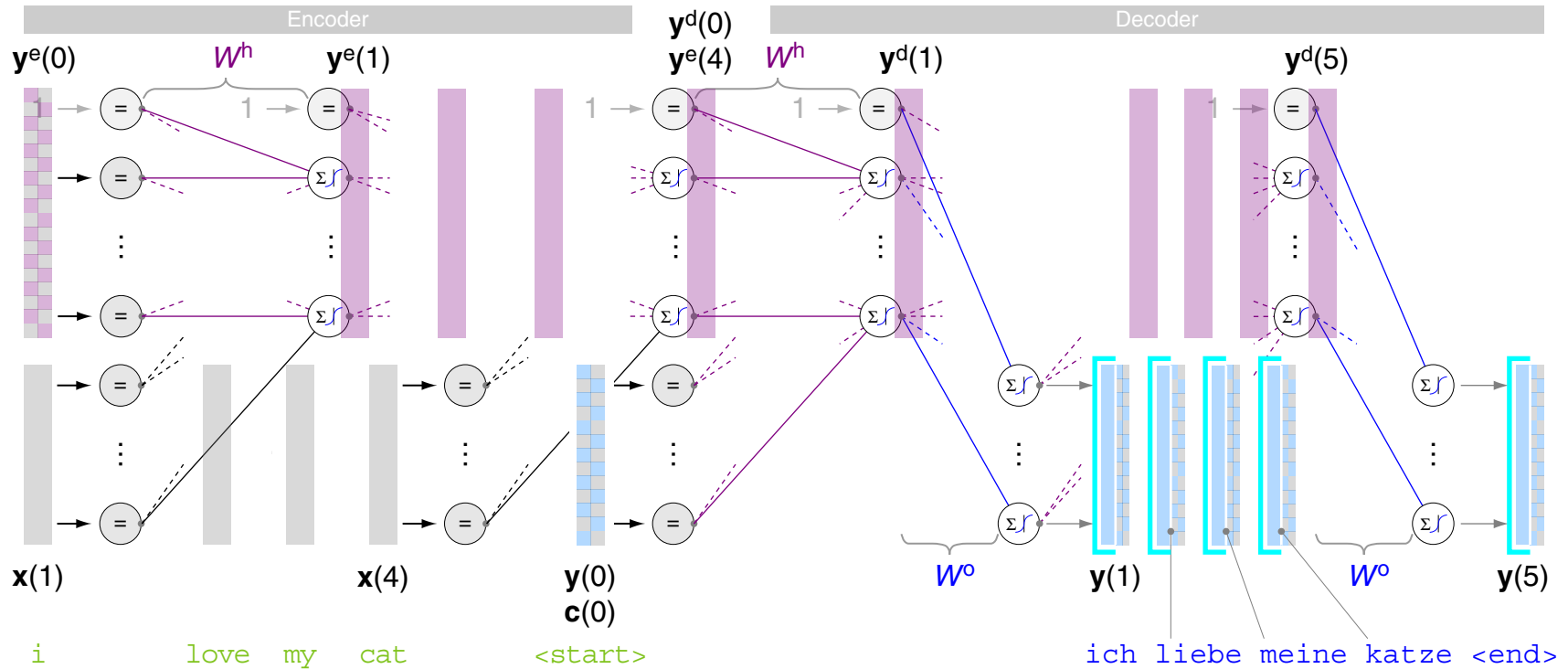


The sequence-to-sequence RNN directly calculates  $p(y \mid x)$ :

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

# RNNs for Machine Translation

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

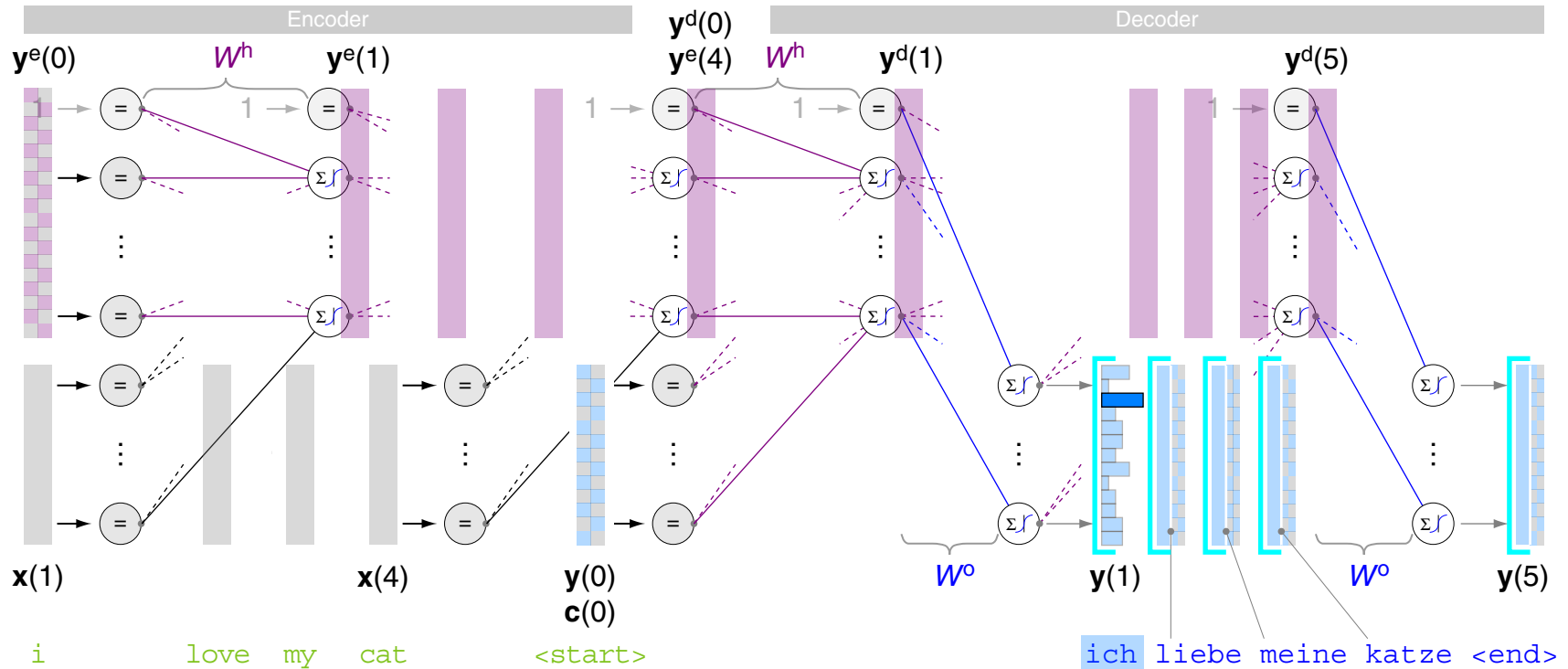


The sequence-to-sequence RNN directly calculates  $p(y \mid x)$ :

$$\begin{aligned} p(y \mid x) &\equiv p(y(1), \dots, y(5) \mid \mathbf{x}, y(0)), \quad \mathbf{x} := x(1), x(2), x(3), x(4) \\ &= p(y(1) \mid \mathbf{x}, y(0)) \cdot p(y(2) \mid \mathbf{x}, y(0), y(1)) \cdot \dots \cdot p(y(5) \mid \mathbf{x}, y(0), \dots, y(4)) \end{aligned}$$

# RNNs for Machine Translation

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

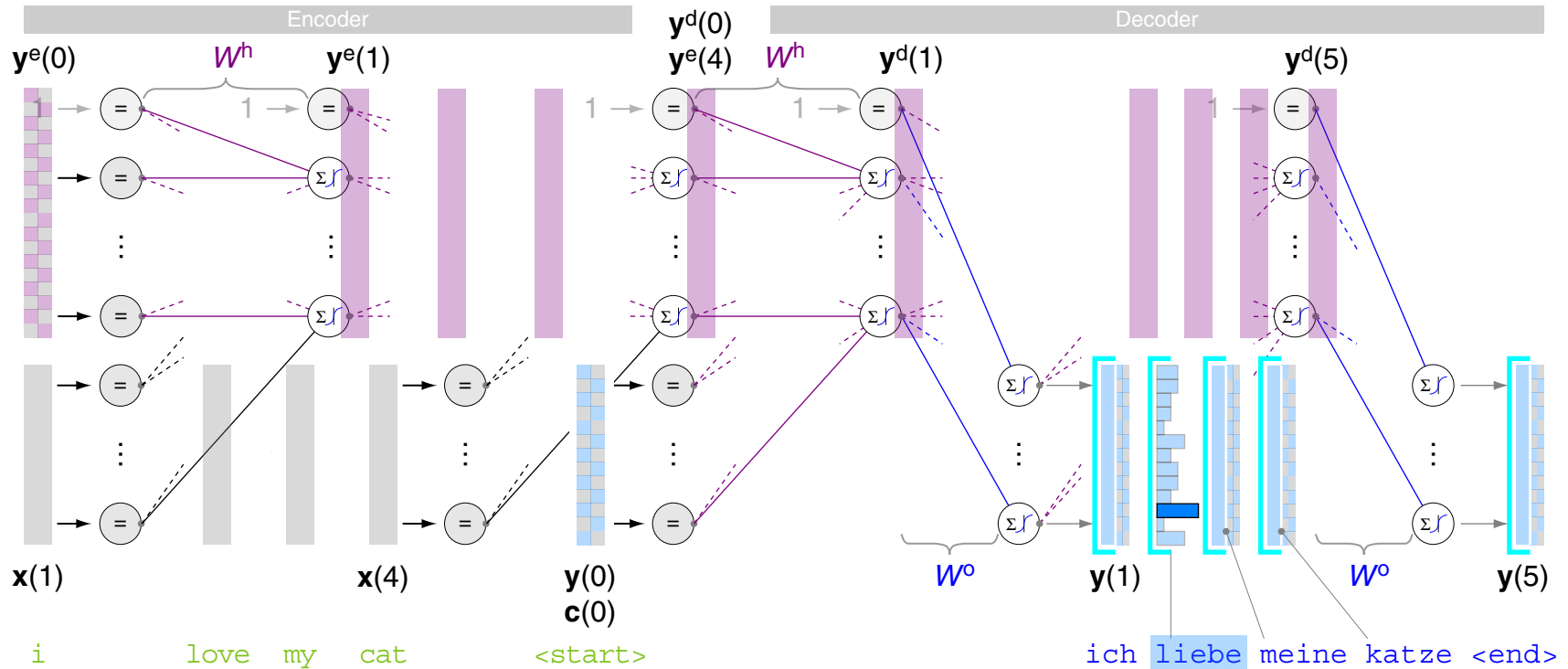


The sequence-to-sequence RNN directly calculates  $p(y \mid x)$ :

$$\begin{aligned}
 p(y \mid x) &\equiv p(y(1), \dots, y(5) \mid \mathbf{x}, y(0)), & \mathbf{x} &:= x(1), x(2), x(3), x(4) \\
 &= p(y(1) \mid \mathbf{x}, y(0)) \cdot p(y(2) \mid \mathbf{x}, y(0), y(1)) \cdot \dots \cdot p(y(5) \mid \mathbf{x}, y(0), \dots, y(4))
 \end{aligned}$$

# RNNs for Machine Translation

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



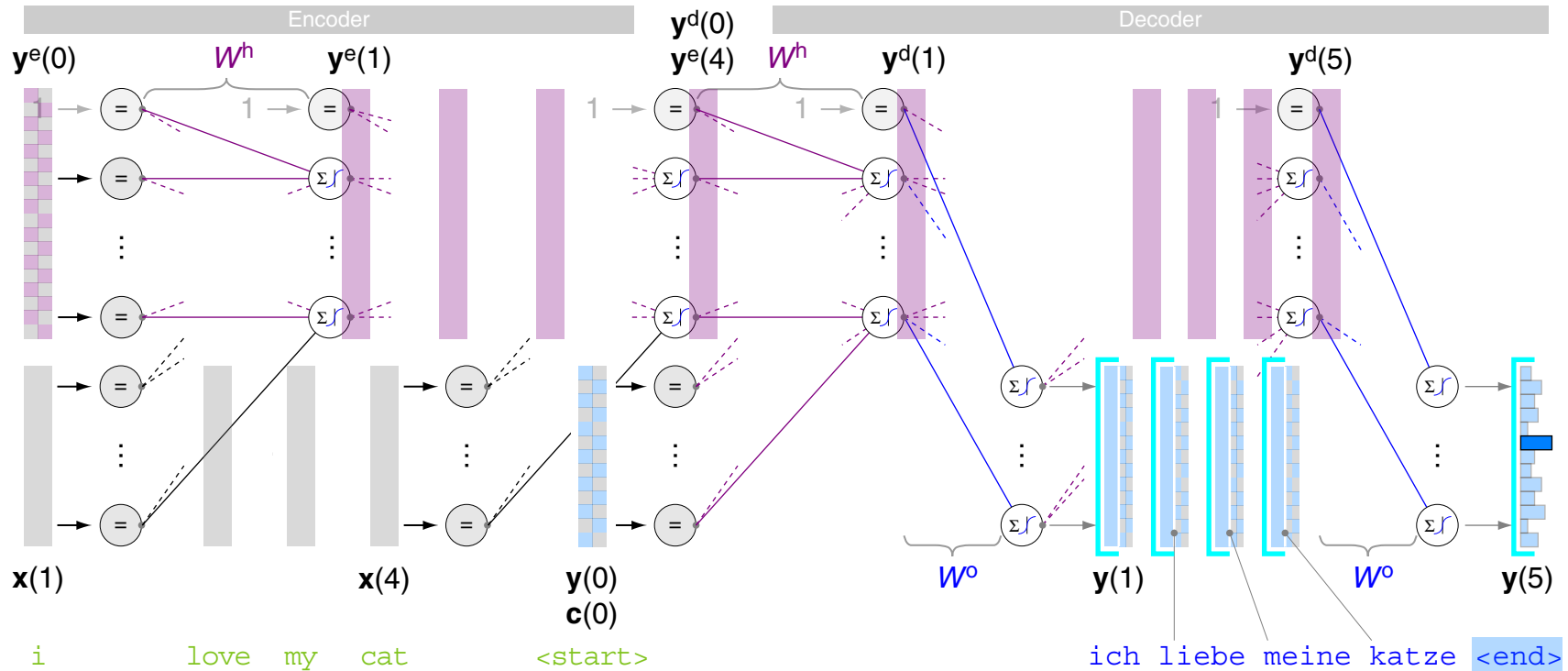
The sequence-to-sequence RNN directly calculates  $p(y \mid x)$ :

$$\begin{aligned}
 p(y \mid x) &\equiv p(y(1), \dots, y(5) \mid \mathbf{x}, y(0)), & \mathbf{x} &:= x(1), x(2), x(3), x(4) \\
 &= p(y(1) \mid \mathbf{x}, y(0)) \cdot p(y(2) \mid \mathbf{x}, y(0), y(1)) \cdot \dots \cdot p(y(5) \mid \mathbf{x}, y(0), \dots, y(4))
 \end{aligned}$$



# RNNs for Machine Translation

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



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

$$\begin{aligned} p(y \mid x) &\equiv p(y(1), \dots, y(5) \mid \mathbf{x}, y(0)), \quad \mathbf{x} := x(1), x(2), x(3), x(4) \\ &= p(y(1) \mid \mathbf{x}, y(0)) \cdot p(y(2) \mid \mathbf{x}, y(0), y(1)) \cdot \dots \cdot p(y(5) \mid \mathbf{x}, y(0), \dots, y(4)) \end{aligned}$$

## Remarks:

- ❑ Each output vector  $\mathbf{y}(t)$  corresponds to a probability distribution over [Vocabulary<sup>d</sup>](#) (recall the  $\sigma_1$ -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  $\prod_{t=1}^{\tau} p(\mathbf{y}(t) \mid \mathbf{x}, \mathbf{y}(0), \dots, \mathbf{y}(t-1))$ , 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 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]