

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

Attention Mechanism

Vanishing Gradient Problem

Attention Mechanism

RNN with Long Short-Term Memory (LSTM)

[SKIPPED]

Remarks:

- ❑ LSTM is a recurrent neural network architecture that is very efficient at remembering long term dependencies and that is less vulnerable to the vanishing gradient problem.

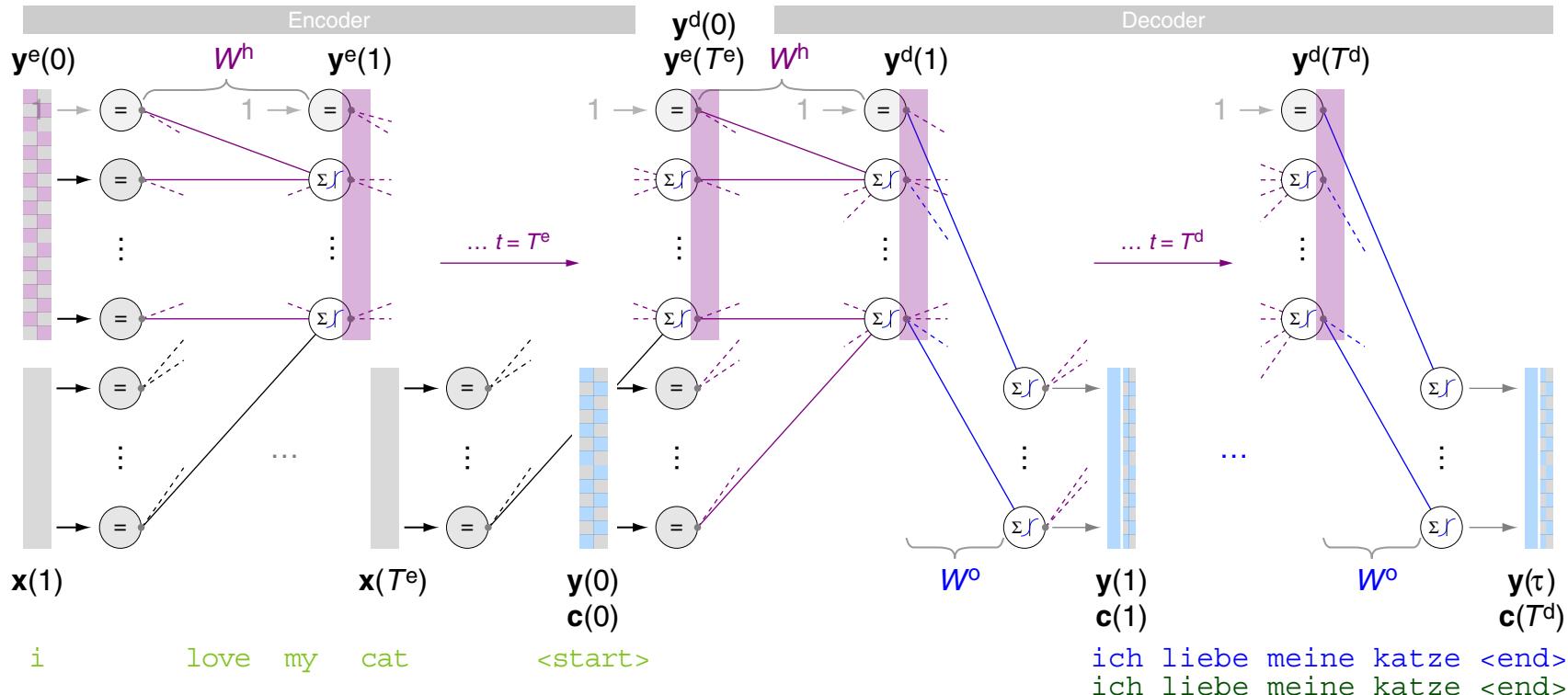
Attention Mechanism

RNN with Gated Recurrent Units (GRU)

[SKIPPED]

Attention Mechanism

Notation II (Omitting the Maths)



$x(t)$ input
in_word

$y^h(t)$ hidden
 $y^e(t)$ hidden encoder
 $y^d(t)$ hidden decoder

predefined
hidden

$y^a(t)$ attention

$y(t)$ output
out_word (test)

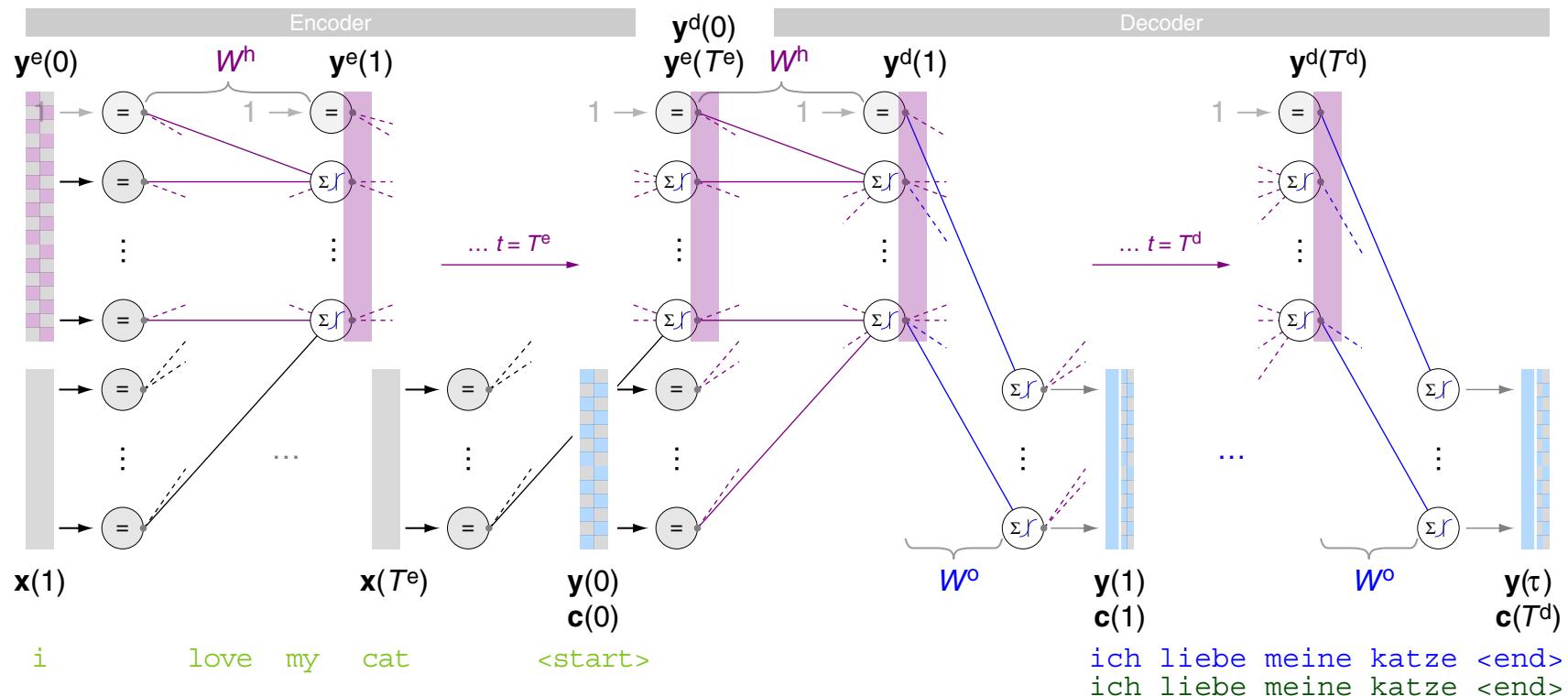
$c(t)$ target
out_word (training)

$y(t)$ output or $c(t)$ target
(depends on training or test phase)

[Notation I]

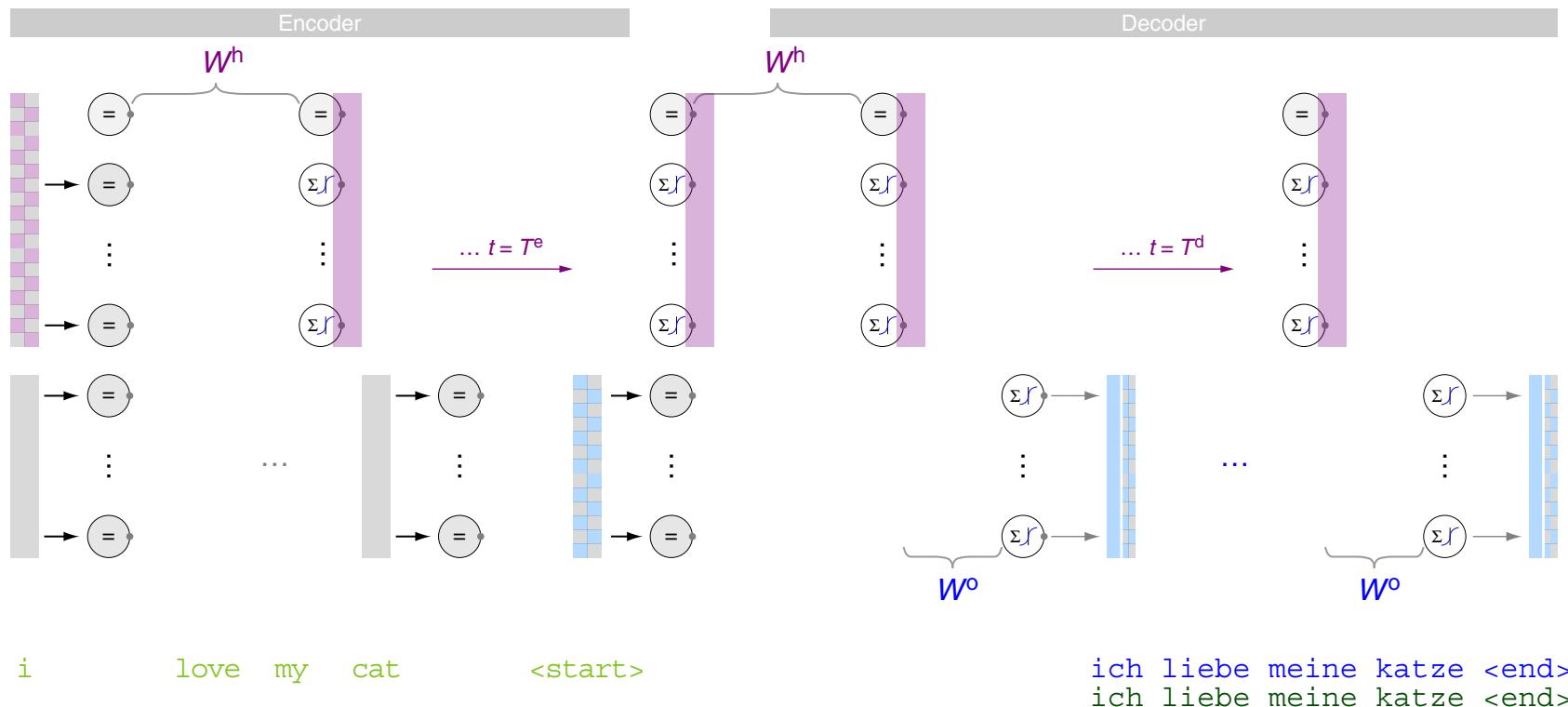
Attention Mechanism

Notation II (Omitting the Maths) (continued)



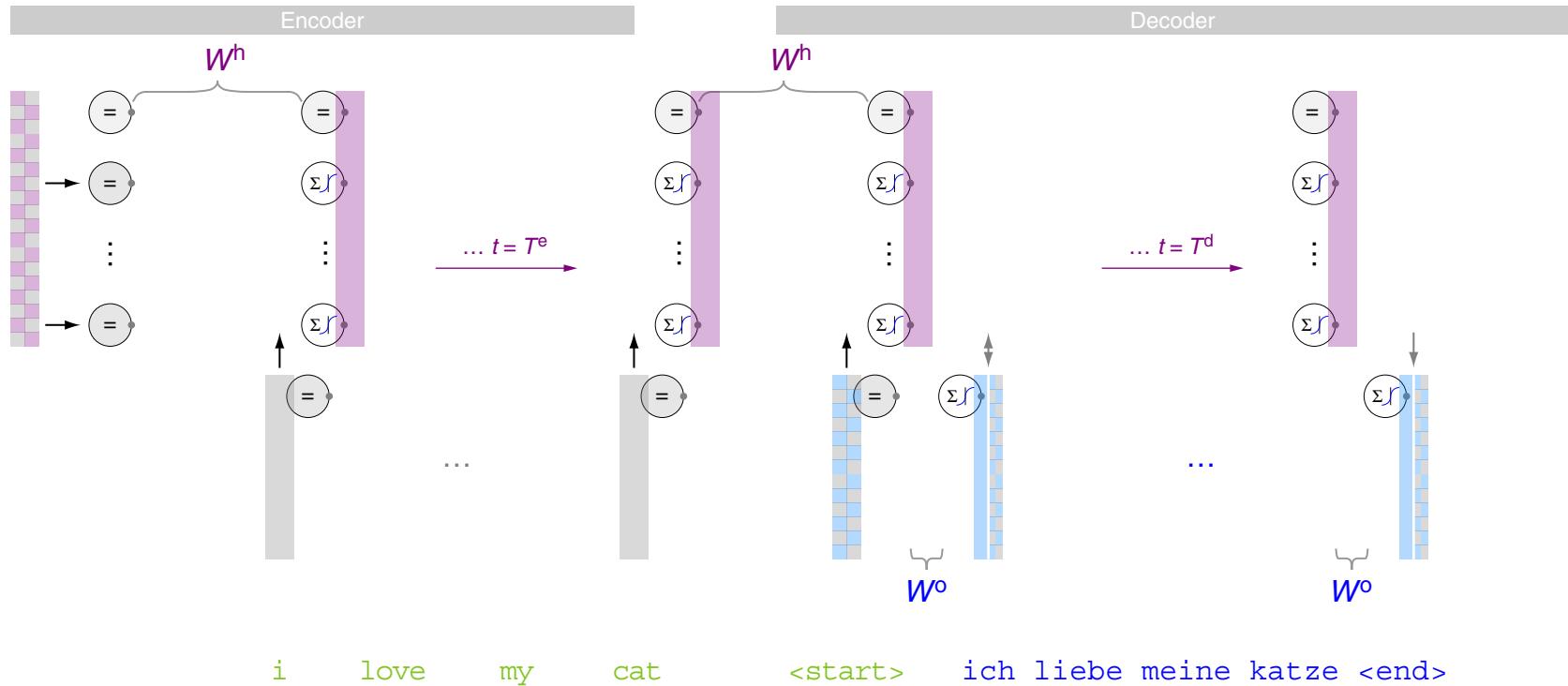
Attention Mechanism

Notation II (Omitting the Maths) (continued)



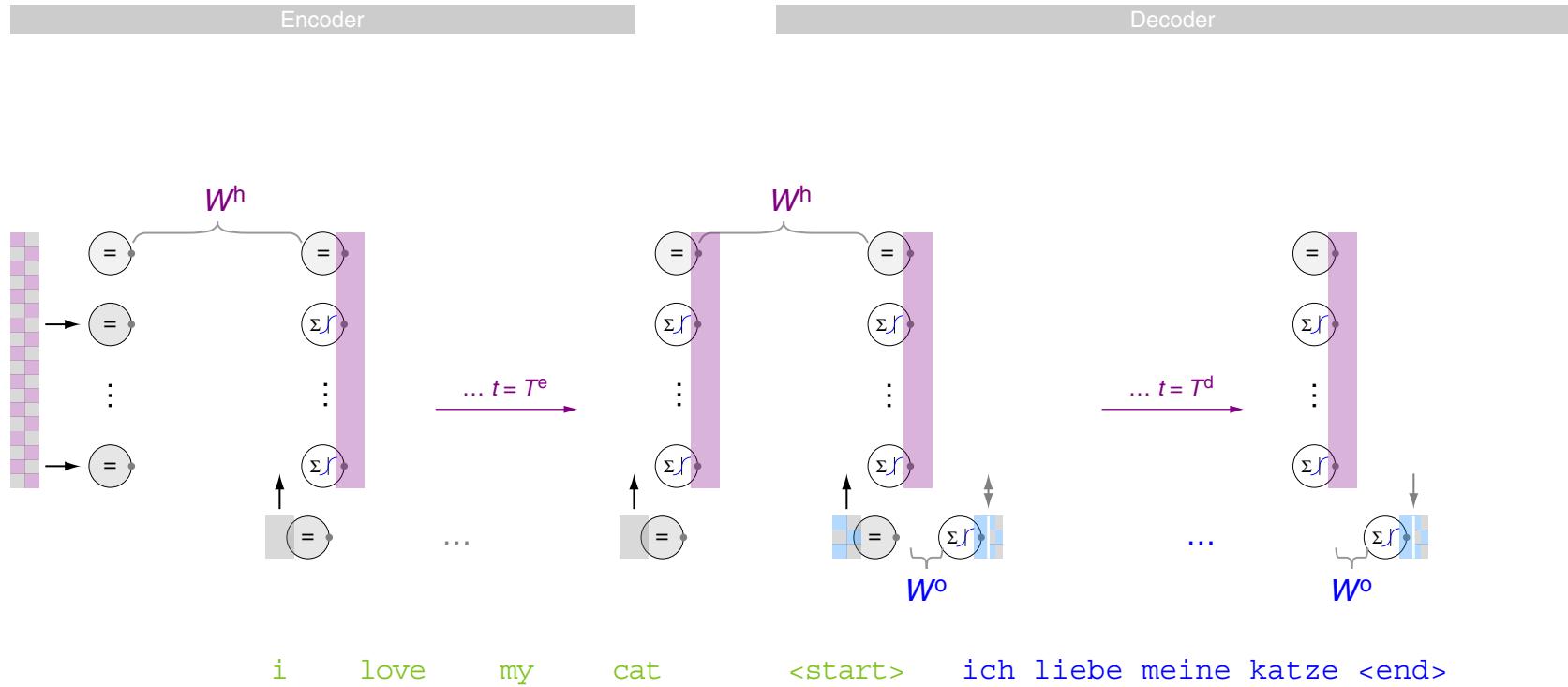
Attention Mechanism

Notation II (Omitting the Maths) (continued)



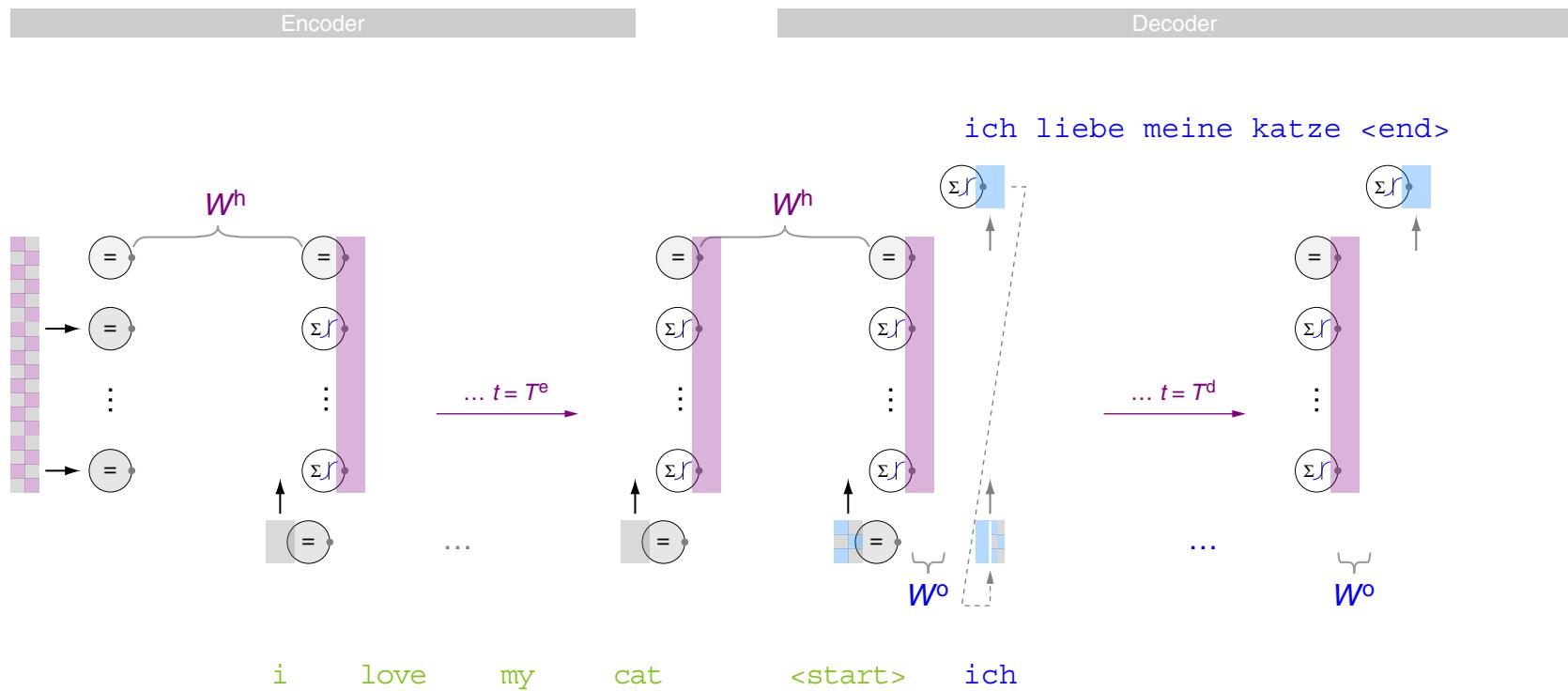
Attention Mechanism

Notation II (Omitting the Maths) (continued)



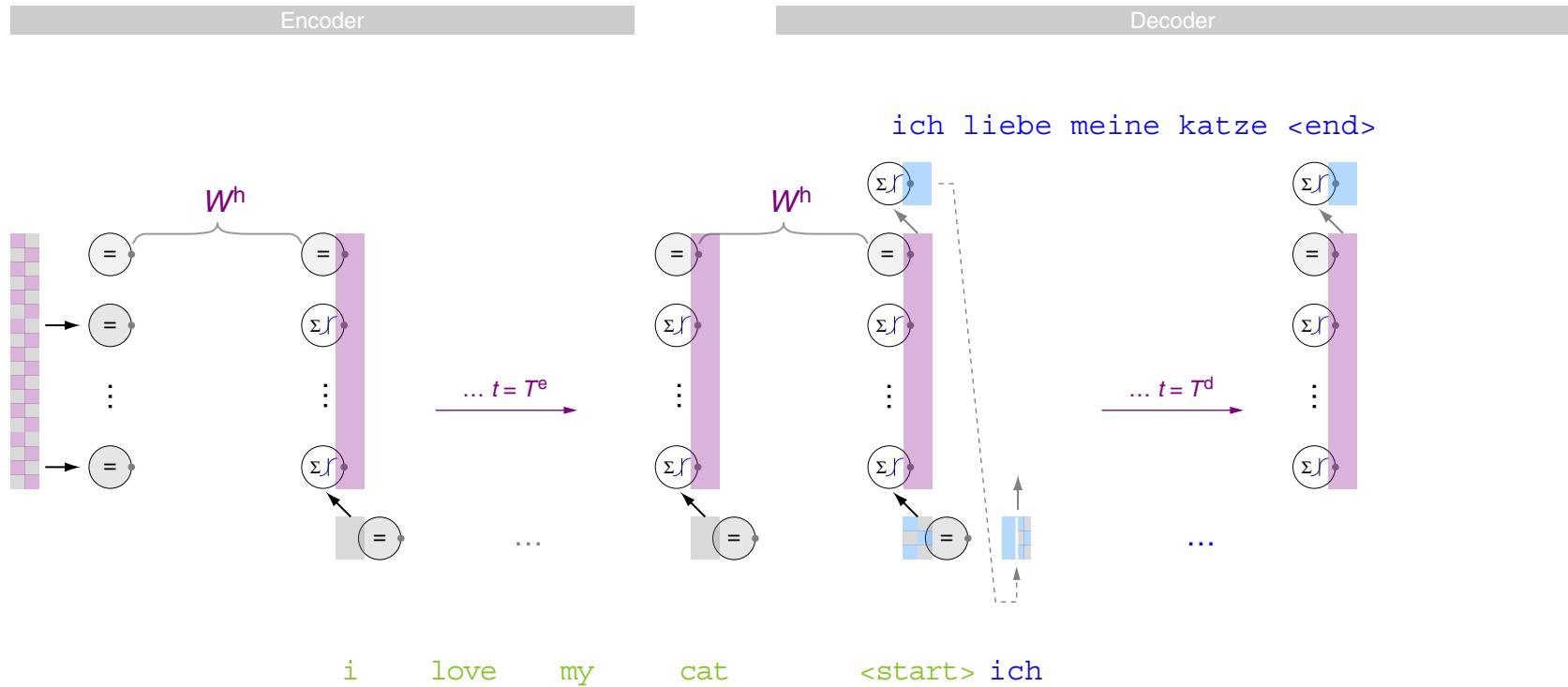
Attention Mechanism

Notation II (Omitting the Maths) (continued)



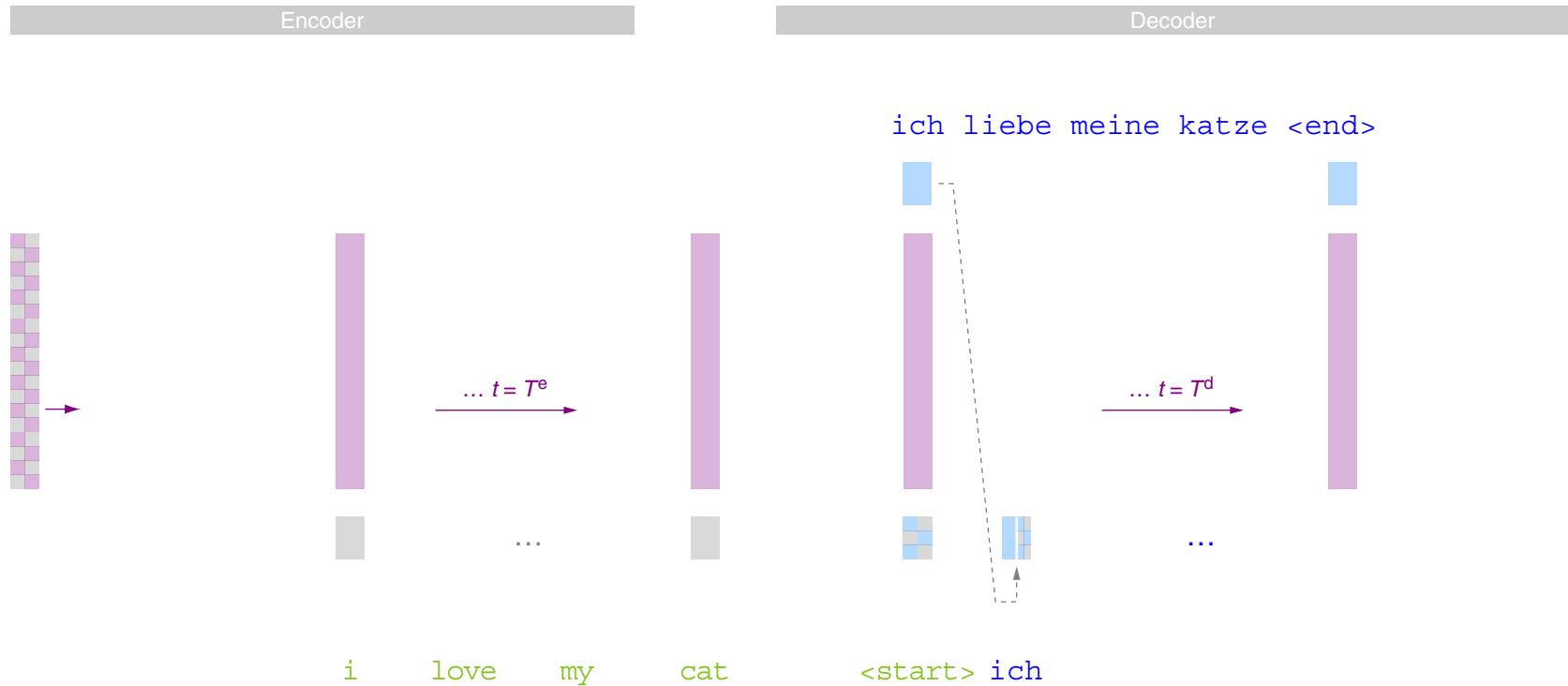
Attention Mechanism

Notation II (Omitting the Maths) (continued)



Attention Mechanism

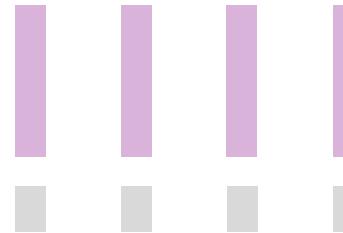
Notation II (Omitting the Maths) (continued)



Attention Mechanism

Notation II (Omitting the Maths) (continued)

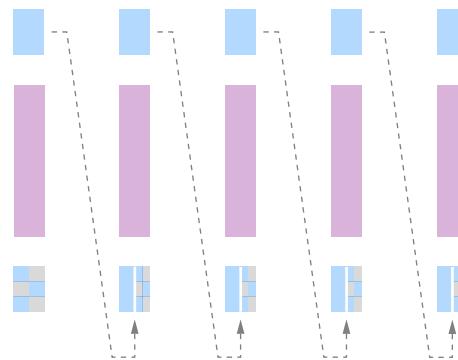
Encoder



i love my cat

Decoder

ich liebe meine katze <end>

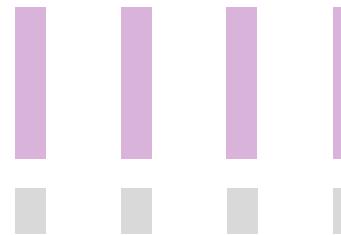


<start> ich liebe meine katze

Attention Mechanism

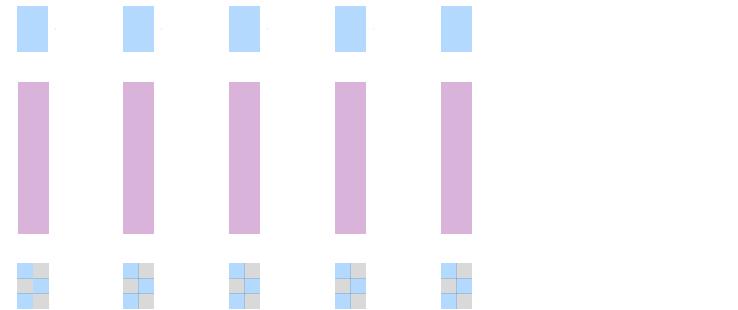
Notation II (Omitting the Maths) (continued)

Encoder



Decoder

ich liebe meine katze <end>
ich liebe meine katze <end>

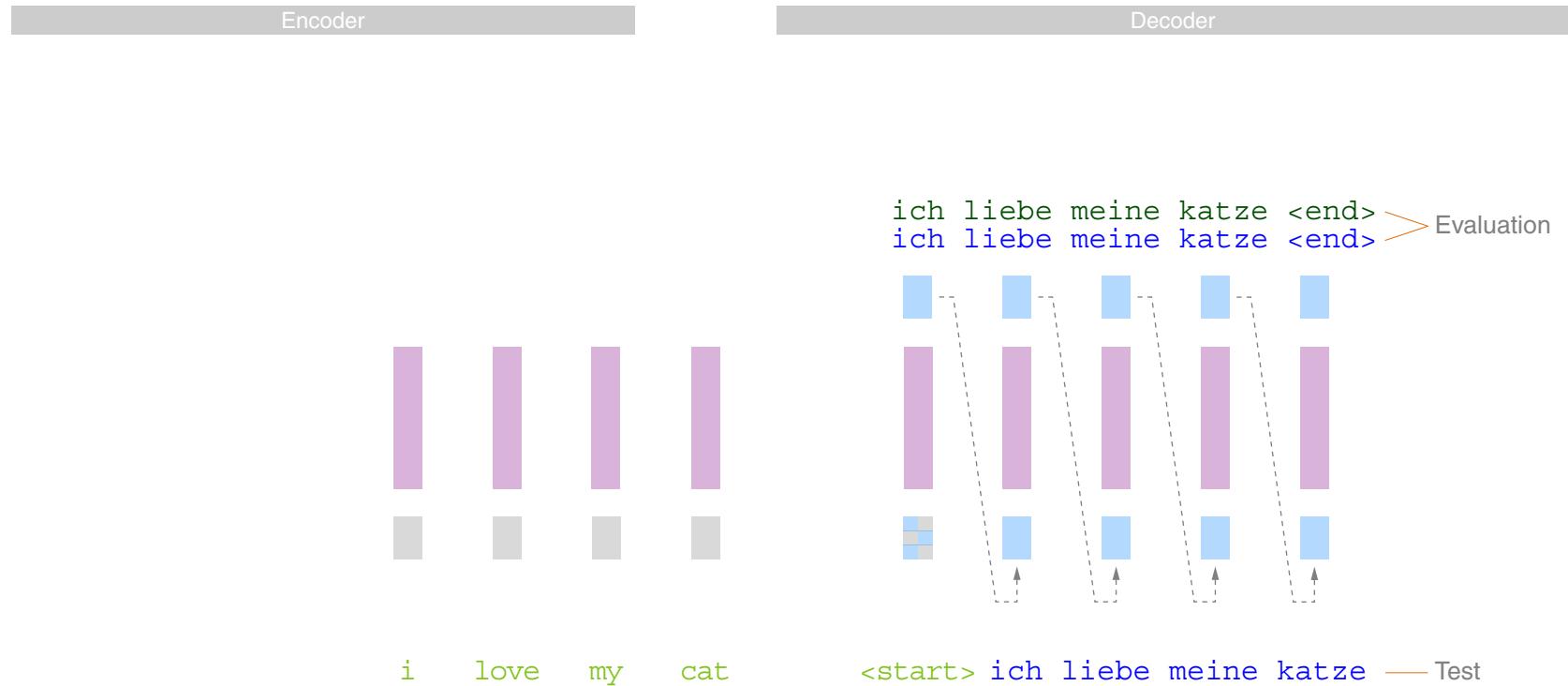


<start> ich liebe meine katze

Training

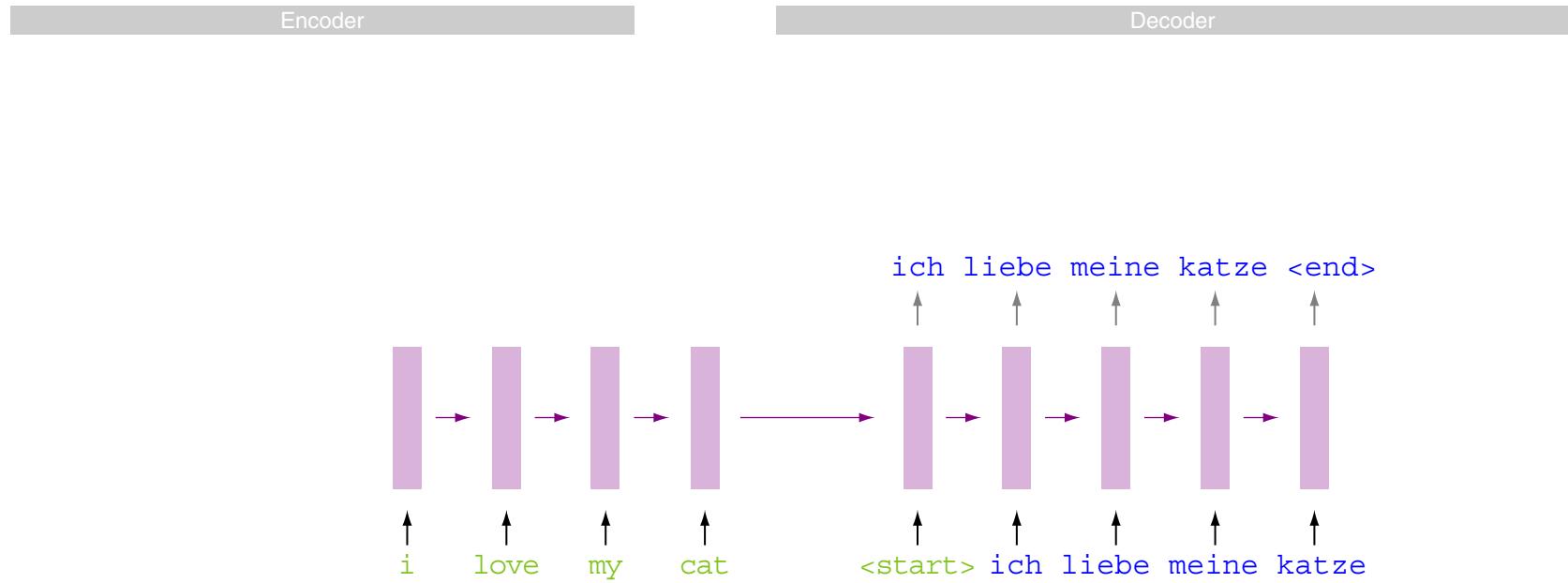
Attention Mechanism

Notation II (Omitting the Maths) (continued)



Attention Mechanism

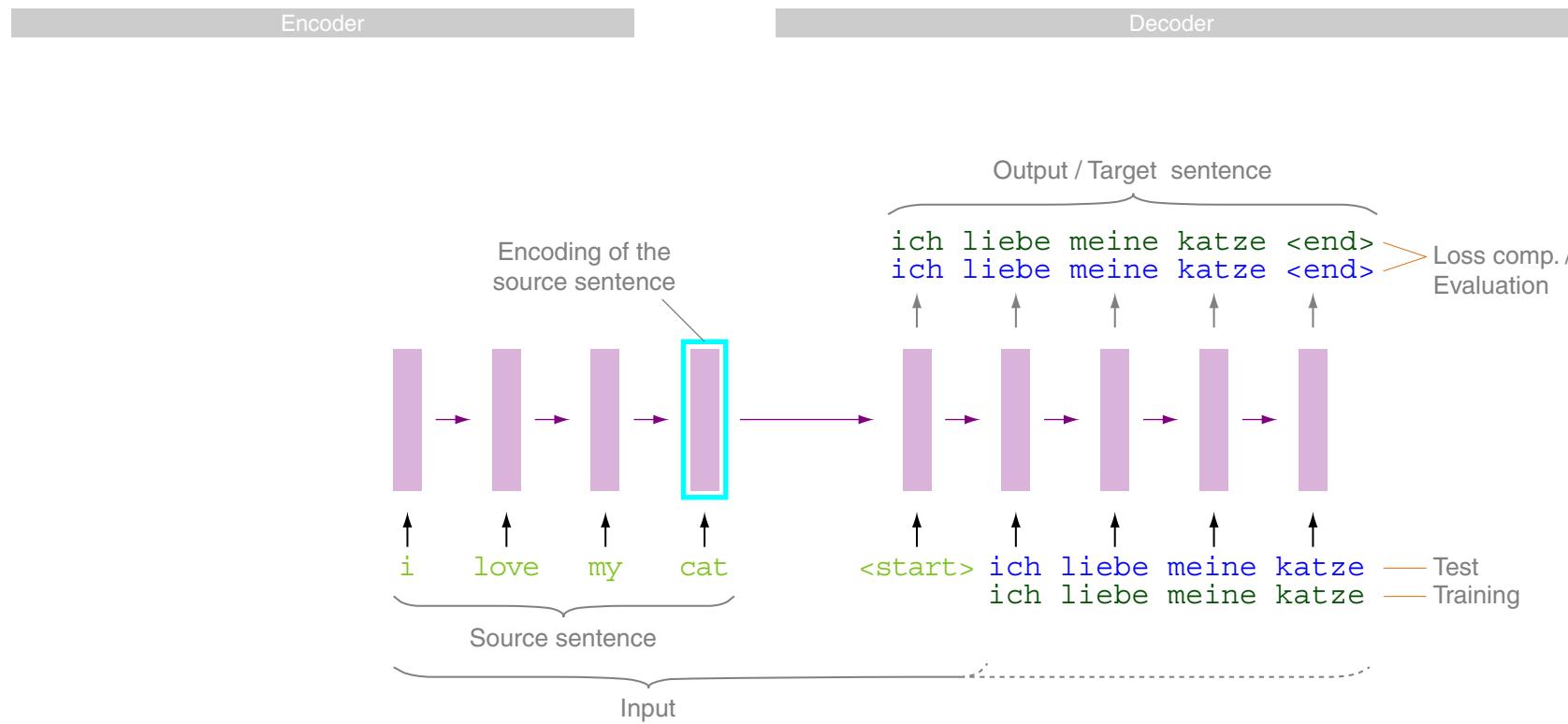
Notation II (Omitting the Maths) (continued)



This level of abstraction is used among others by [Manning 2021, lecture CS224N].

Attention Mechanism

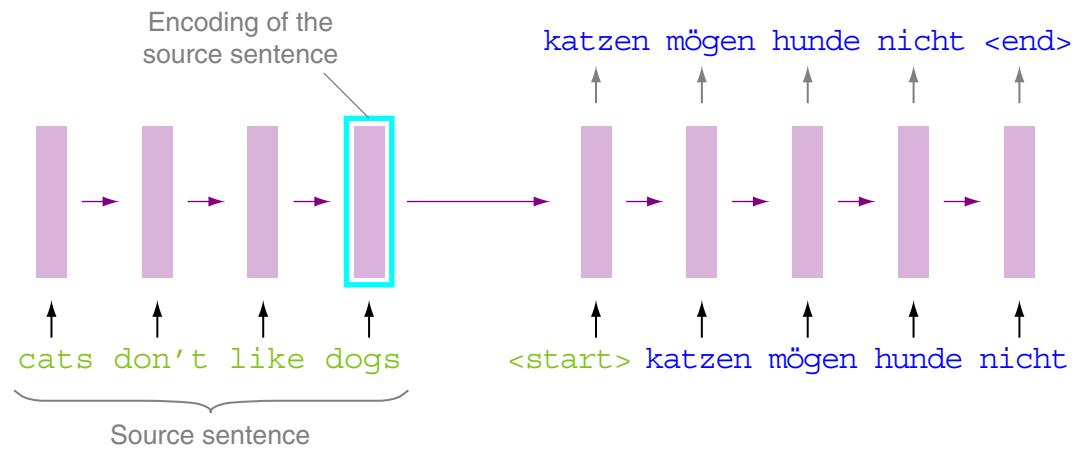
Notation II (Omitting the Maths) (continued)



This level of abstraction is used among others by [Manning 2021, lecture CS224N].

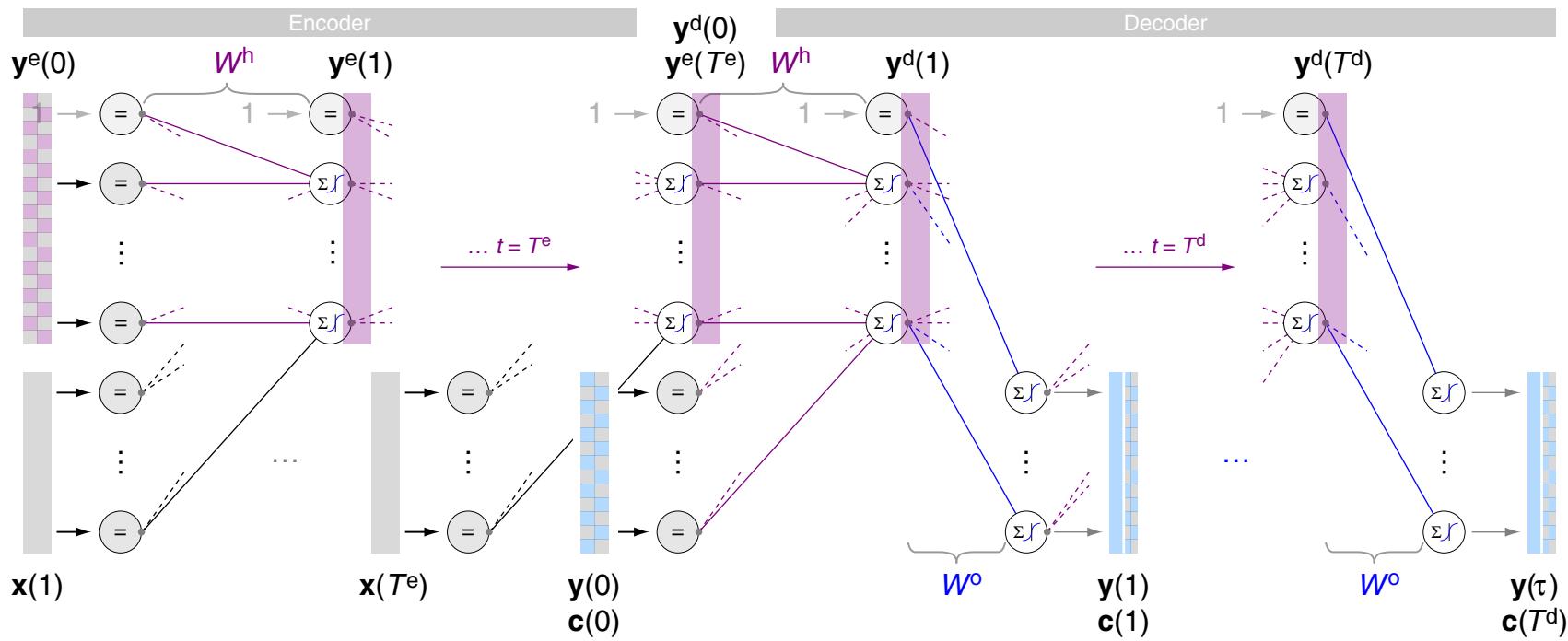
Attention Mechanism

Attention: Short Circuit of Decoder and Encoder



Attention Mechanism

RNN with Simple Attention



Input:

$$\mathbf{x}, \mathbf{y}^e(0), \mathbf{y}(t), \mathbf{y}(0) \hat{=} \langle \text{start} \rangle$$

Output:

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

Hidden:

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

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

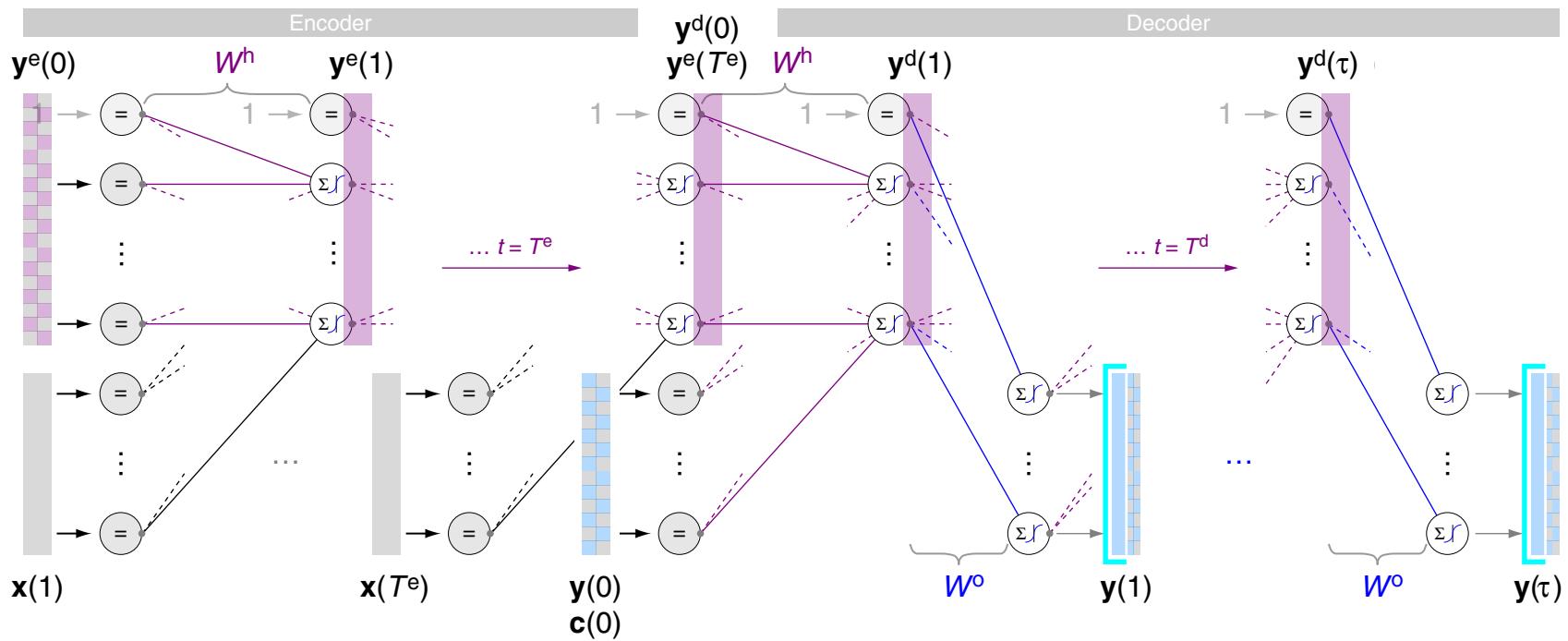
Target:

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

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

Attention Mechanism

RNN with Simple Attention (continued)



Output:

$$\mathbf{y}(t) = \sigma_1 (W^o \mathbf{y}^d(t))$$

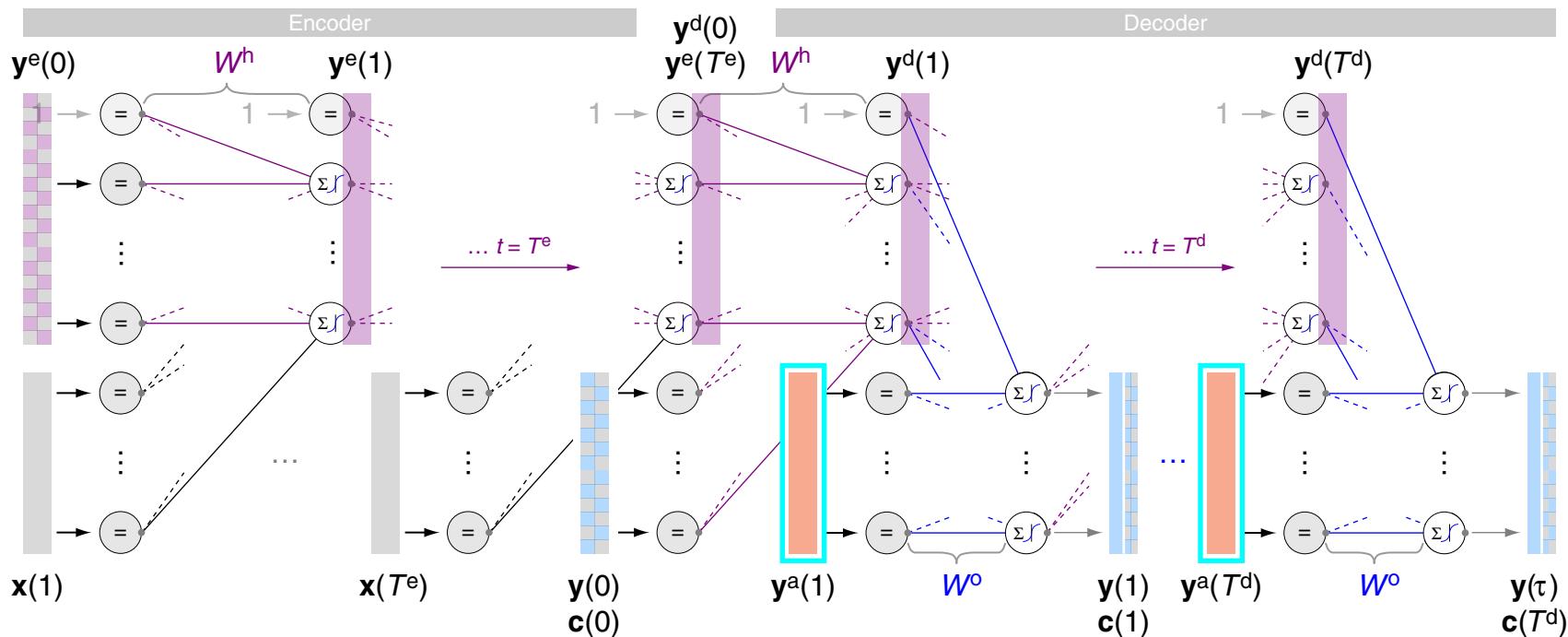
Attention:

$$\mathbf{y}^a(t) = [\mathbf{y}^e(1), \dots, \mathbf{y}^e(T^e)] \mathbf{a}(t)$$

$$\mathbf{a}(t) = [\mathbf{y}^e(1), \dots, \mathbf{y}^e(T^e)]^T \mathbf{y}^d(t), \quad t = 1, \dots, T^d$$

Attention Mechanism

RNN with Simple Attention (continued)



Output:

$$y(t) = \sigma_1 \left(W^o \begin{pmatrix} y^d(t) \\ y^a(t) \end{pmatrix} \right)$$

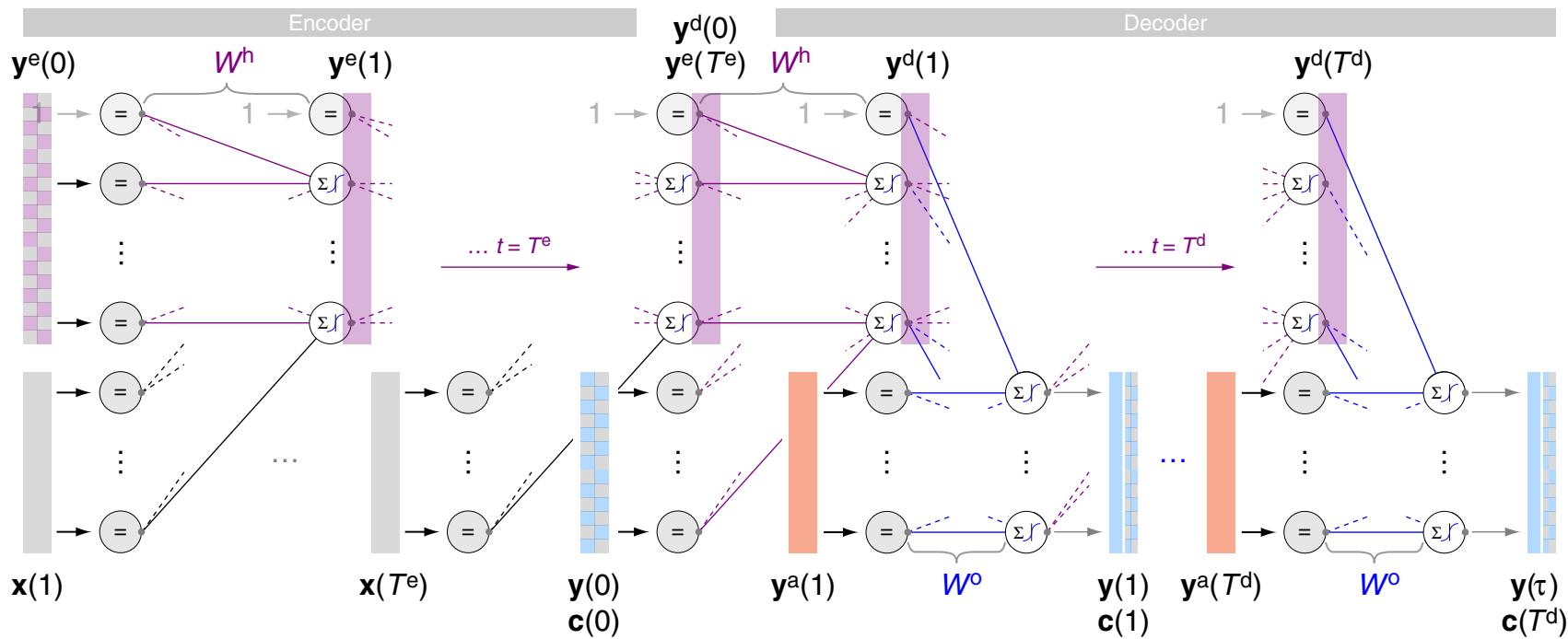
Attention:

$$y^a(t) = \left[y^e(1), \dots, y^e(T^e) \right]^T \alpha(t)$$

$$\alpha(t) = \text{softmax}(W^h y^d(t))$$

Attention Mechanism

RNN with Simple Attention (continued)



Output:

$$y(t) = \sigma_1 \left(W^o \begin{pmatrix} y^d(t) \\ y^a(t) \end{pmatrix} \right)$$

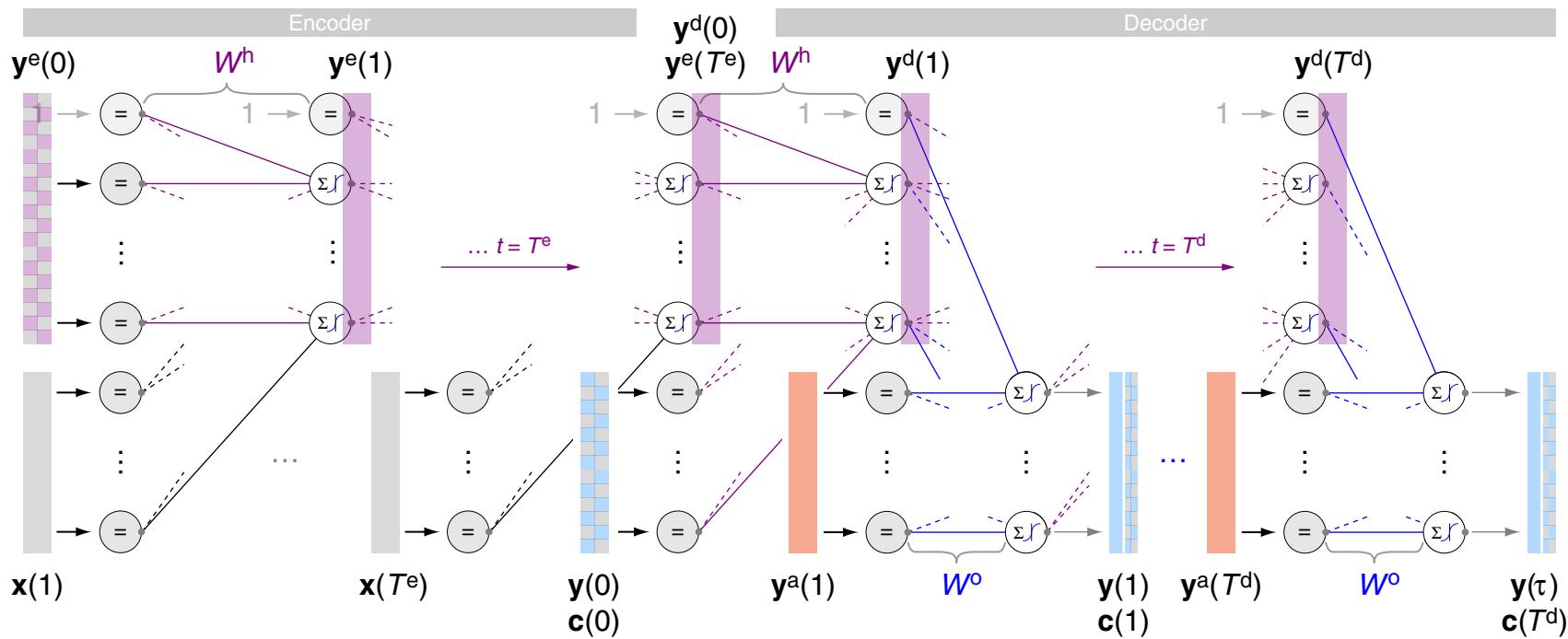
Attention:

$$y^a(t) = \left[y^e(1), \dots, y^e(T^e) \right] a(t)$$

$$a(t) = \left[y^e(1), \dots, y^e(T^e) \right]^T y^d(t), \quad t = 1, \dots, T^d$$

Attention Mechanism

RNN with Simple Attention (continued)



Output:

$$y(t) = \sigma_1 \left(W^o \begin{pmatrix} y^d(t) \\ y^a(t) \end{pmatrix} \right)$$

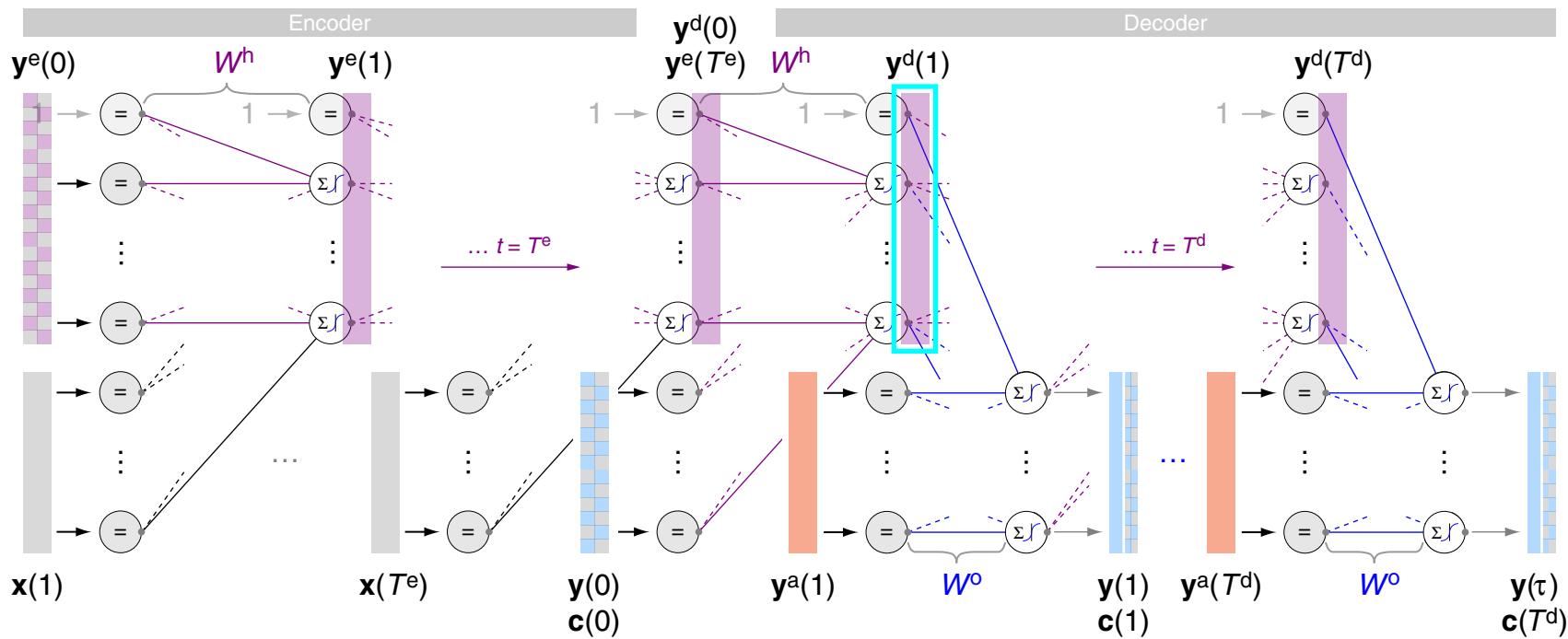
Attention:

$$y^a(t) = \left[y^e(1), \dots, y^e(T^e) \right] a_1(t), \quad a_1(t) = \sigma_1 \left(a(t) \right)$$

$$a(t) = \left[y^e(1), \dots, y^e(T^e) \right]^T y^d(t), \quad t = 1, \dots, T^d$$

Attention Mechanism

RNN with Simple Attention (continued)



Output:

$$\mathbf{y}(t) = \sigma_1 \left(W^o \begin{pmatrix} \mathbf{y}^d(t) \\ \mathbf{y}^a(t) \end{pmatrix} \right)$$

Attention:

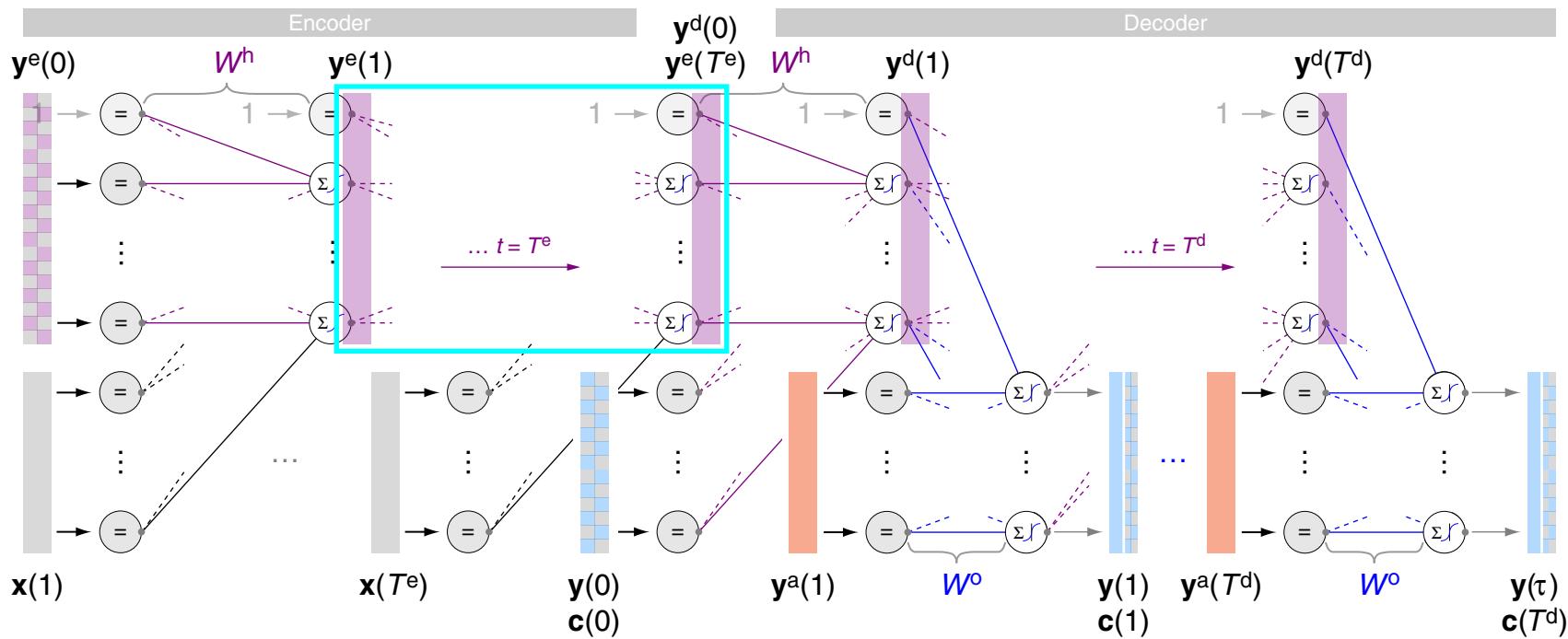
$$\mathbf{y}^a(t) = \left[\mathbf{y}^e(1), \dots, \mathbf{y}^e(T^e) \right] \mathbf{a}_1(t), \quad \mathbf{a}_1(t) = \sigma_1(\mathbf{a}(t))$$

$$\mathbf{a}(t) = \left[\mathbf{y}^e(1), \dots, \mathbf{y}^e(T^e) \right]^T \boxed{\mathbf{y}^d(t), \quad t = 1, \dots, T^d}$$

“Queries”

Attention Mechanism

RNN with Simple Attention (continued)



Output:

$$y(t) = \sigma_1 \left(W^o \begin{pmatrix} y^d(t) \\ y^a(t) \end{pmatrix} \right)$$

Attention:

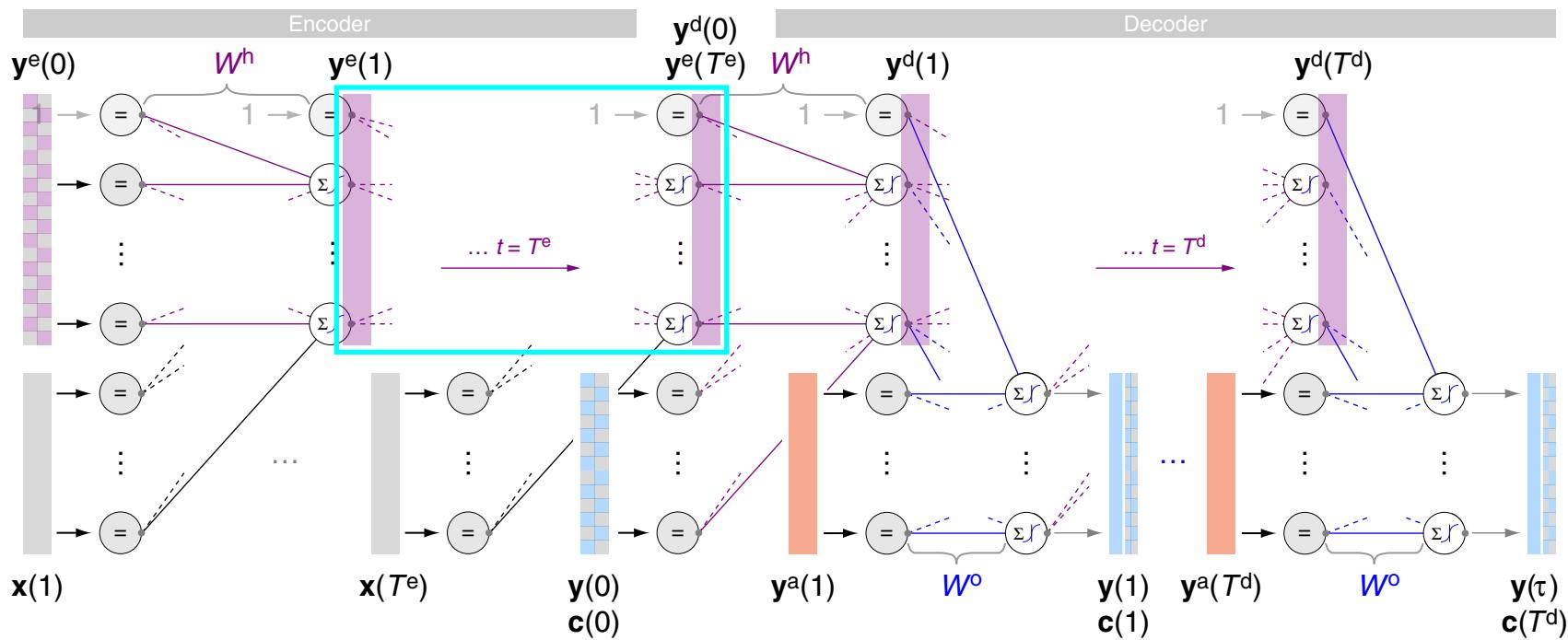
$$y^a(t) = \left[y^e(1), \dots, y^e(T^e) \right] a_1(t), \quad a_1(t) = \sigma_1 \left(a(t) \right)$$

$$a(t) = \left[y^e(1), \dots, y^e(T^e) \right]^T y^d(t), \quad t = 1, \dots, T^d$$

“Keys”

Attention Mechanism

RNN with Simple Attention (continued)



Output:

$$y(t) = \sigma_1 \left(W^o \begin{pmatrix} y^d(t) \\ y^a(t) \end{pmatrix} \right)$$

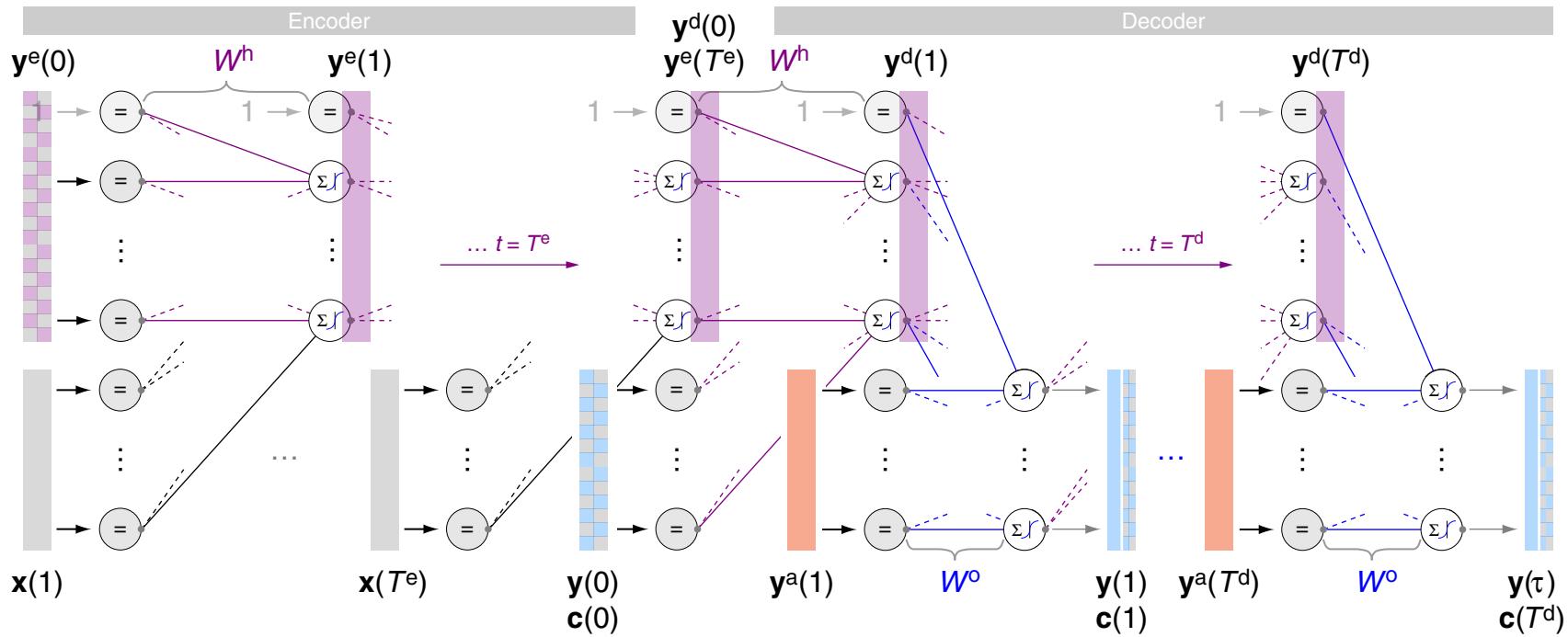
Attention:

$$y^a(t) = \boxed{\begin{bmatrix} y^e(1), \dots, y^e(T^e) \end{bmatrix}} \quad a_1(t), \quad a_1(t) = \sigma_1(a(t)) \quad \text{"Values"}$$

$$a(t) = \begin{bmatrix} y^e(1), \dots, y^e(T^e) \end{bmatrix}^T y^d(t), \quad t = 1, \dots, T^d$$

Attention Mechanism

RNN with Parameterized Attention



Output:

$$y(t) = \sigma_1 \left(W^o \begin{pmatrix} y^d(t) \\ y^a(t) \end{pmatrix} \right)$$

Attention:

$$y^a(t) = \left(W^v \begin{bmatrix} y^e(1), \dots, y^e(T^e) \end{bmatrix} \right) a_1(t), \quad a_1(t) = \sigma_1 \left(a(t) \right)$$

$$a(t) = \left(W^K \begin{bmatrix} y^e(1), \dots, y^e(T^e) \end{bmatrix} \right)^T \left(W^Q y^d(t) \right)$$

Remarks (attention calculus) :

- As before, $\sigma_1()$ denotes the softmax function.
- $a(t)$ is called vector of *attention scores*, $a_1(t)$ is the *attention distribution*. [Manning 2021]
- The i th component $a_i(t)$ of the attention score vector $a(t)$, $i = 1, \dots, T^e$, models the importance of the i th *encoder* hidden state $y^e(i)$ for the *decoder* hidden state $y^d(t)$: $a_i(t)$ is the scalar product of $y^e(i)$ and $y^d(t)$ (do not overlook the matrix transpose operation).
- $y^a(t)$ is the result of weighting the encoder hidden state sequence $[y^e(1), \dots, y^e(T^e)]$ with the attention distribution $a_1(t)$. I.e., each vector $y^e(i)$ is considered as a “value” that is weighted with the relative importance stored in the respective dimension (= time step) of $a_1(t)$.
 $y^a(t)$ is called *attention* [vector] for output vector $y(t)$ since it helps to pay attention to the most influential input states for $y(t)$.
- Consider $a(t)$. The scalar product of $y^e(i)$ and $y^d(t)$ becomes maximum if $y^e(i)$ and $y^d(t)$ are identical. The distribution $a_1(t)$ of the T^e weights of $a(t)$ reflects the distribution of absolute values among the y^e .
Consider $a_1(t)$. If some $y^e(i)$ has a high absolute value (compared to the other y^e) and if it has the same direction as $y^d(t)$, it will push the weight of the i th dimension of $a_1(t)$ towards 1 (and the others towards zero).
In the extreme case, $y^a(t)$ is equal to $y^e(i)$. I.e., as input for W^o only the i th *encoder* state, $y^e(i)$, along with the t th *decoder* state, $y^d(t)$, is used, say, $y^e(i)$ is passed directly to position t of the output sequence.

Remarks (parameterized attention) :

- $y^d(t)$ is also denoted as “query,” while the sequence of $y^e(i)$ are denoted as “keys” in this setting. Since $a(t)$ is normalized with a softmax operation it represents the importance of the T^e time steps as probabilities.
- *Parameterized* attention introduces three weight matrices, W^Q , W^K , and W^V , in order to learn a more sophisticated version of the simple attention vector $y^a(t)$. In this regard the matrices are called “query projection”, “key projection”, and “value projection” [matrix] respectively.
- Inspired by nature, the structure of the model function $y()$ has been developed in the form of a network that connects the matrices W^h , W^o , W^Q , W^K , and W^V in a particular manner. Note that the shown model function is specified completely by the set of parameters w , which are organized in the aforementioned matrices.