Lecture 16: Attention and Transformers

Guest Lecture: Ali Kassir

Based on slides by Abigail See and everyone else she copied.

Overview

- Sequence-to-sequence models
- Attention
- Transformers

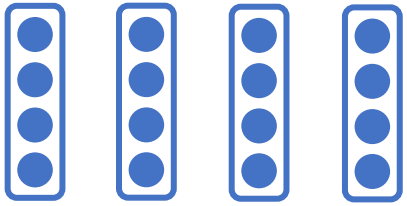
Sequence-to-sequence models

German: Representieren wir diesen Satz.

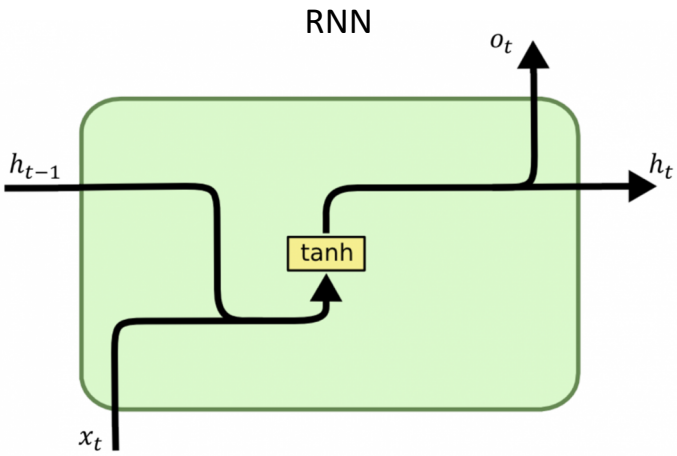
English: Let’s represent this sentence.

Let’s represent this sentence

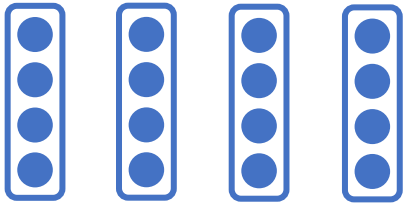
Source language
word embeddings



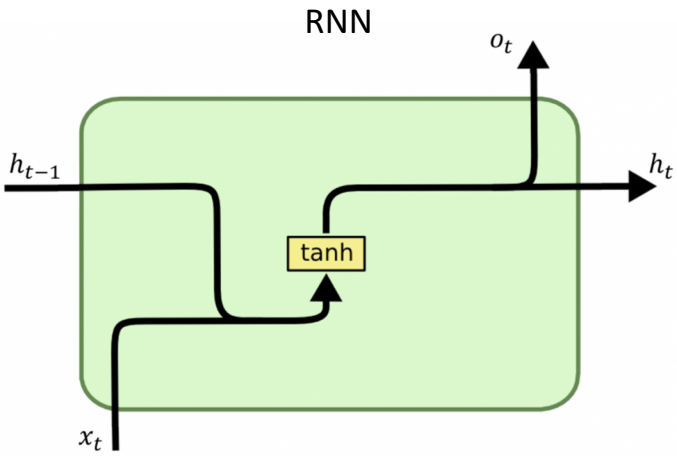
Let’s represent this sentence

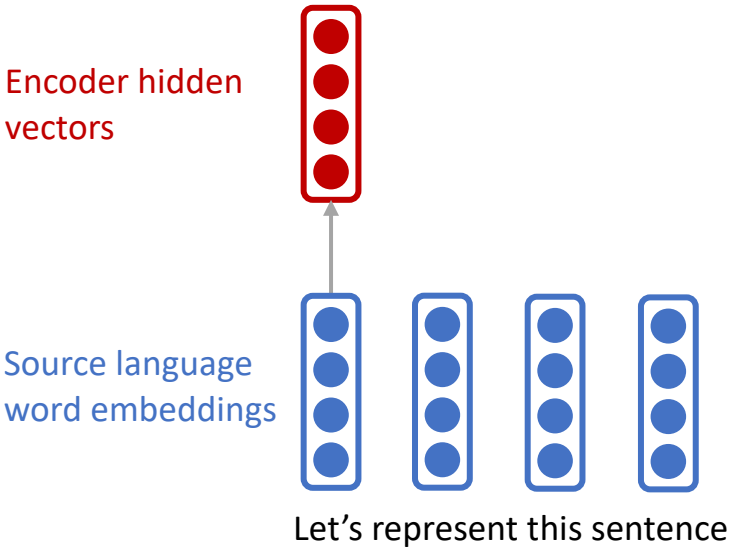


Source language word embeddings

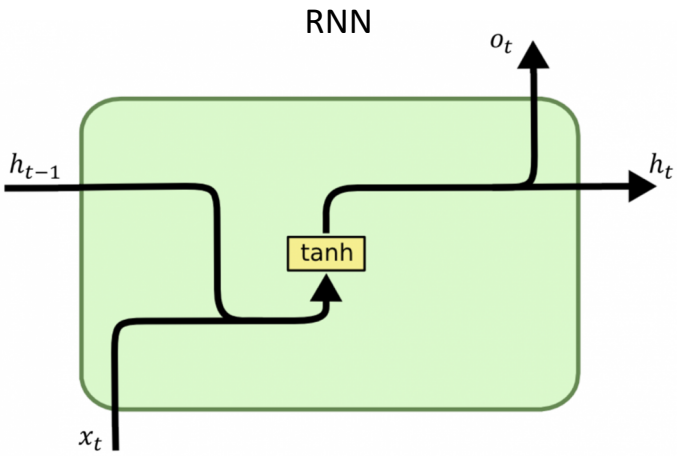


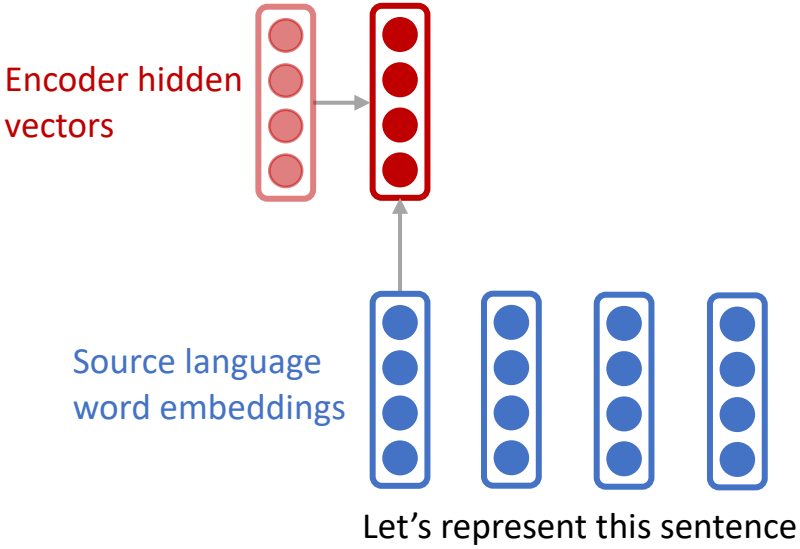
Let’s represent this sentence



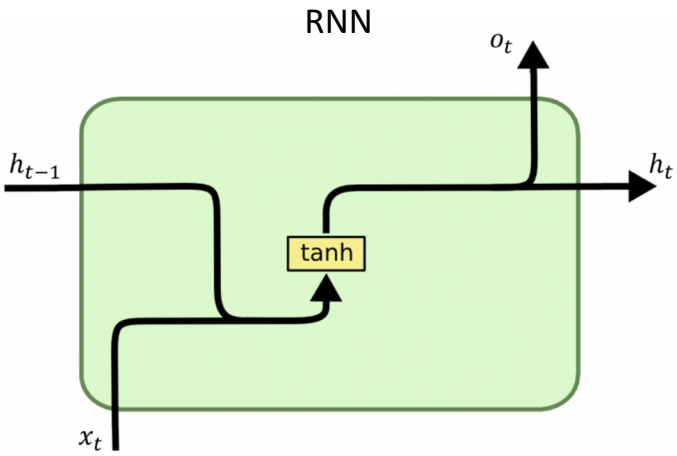


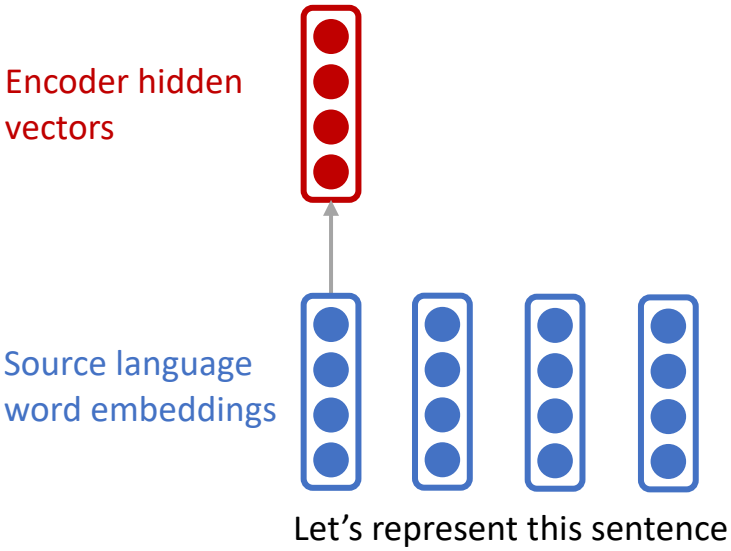
Let's represent this sentence





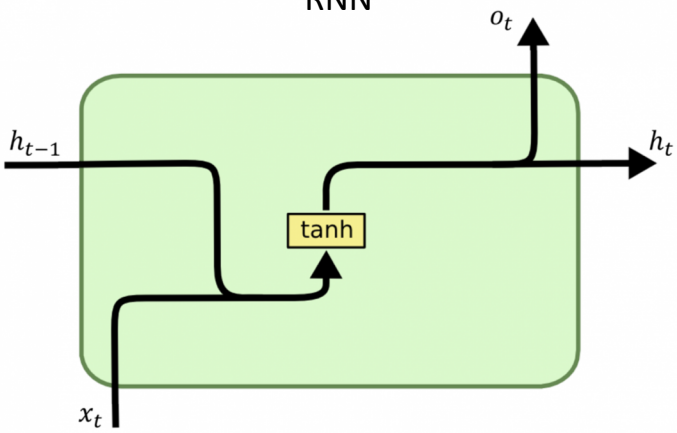
Let’s represent this sentence



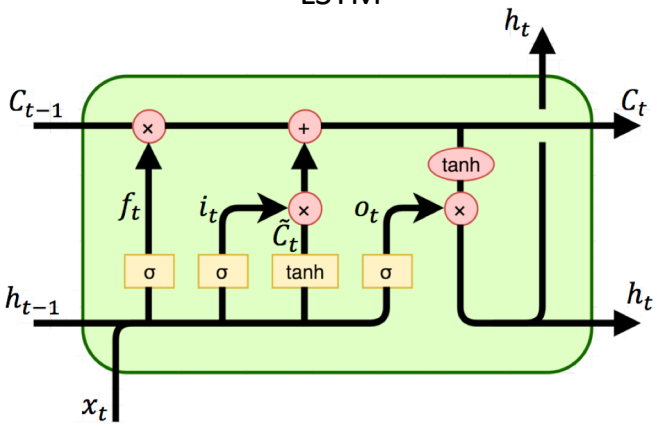


Let's represent this sentence

RNN

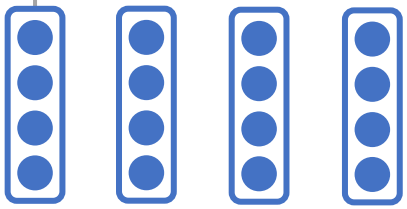


LSTM



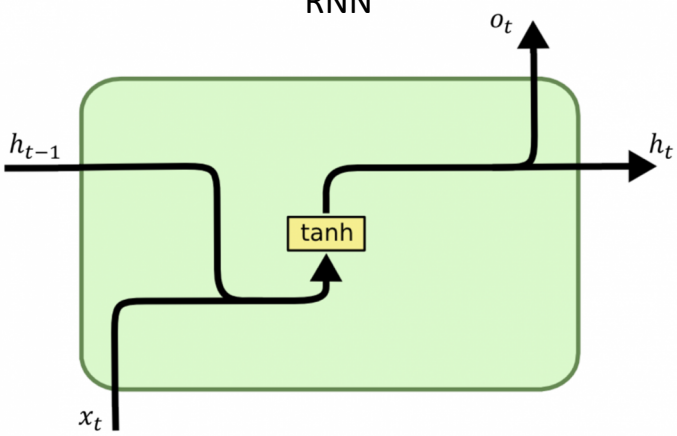
Encoder hidden vectors

Source language word embeddings

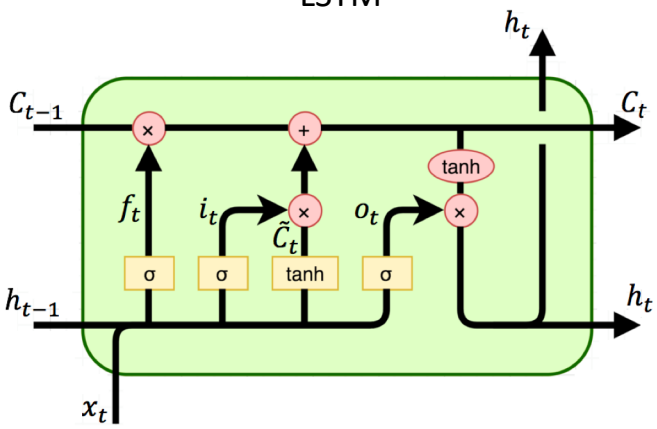


Let’s represent this sentence

RNN

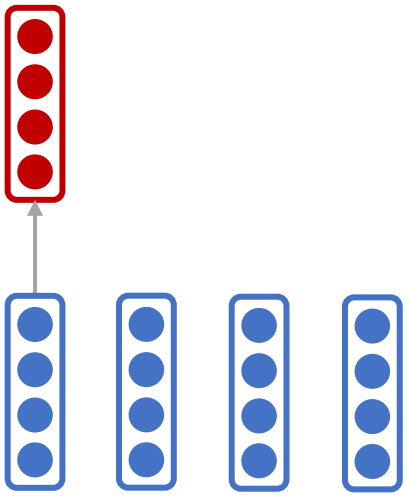


LSTM



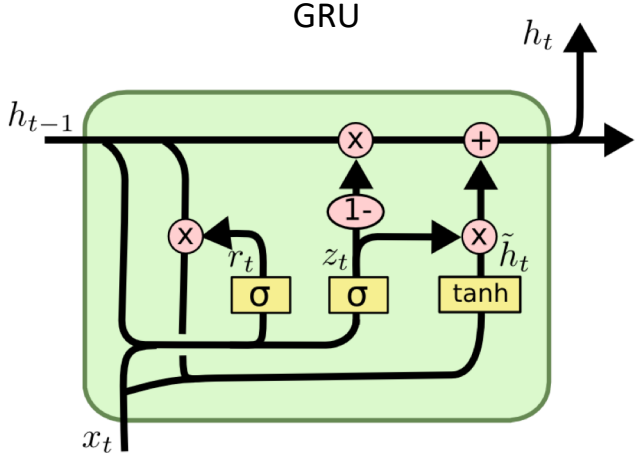
Encoder hidden vectors

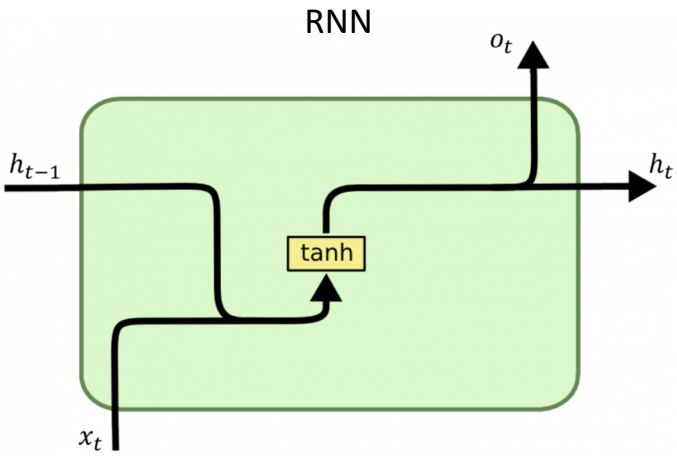
Source language word embeddings

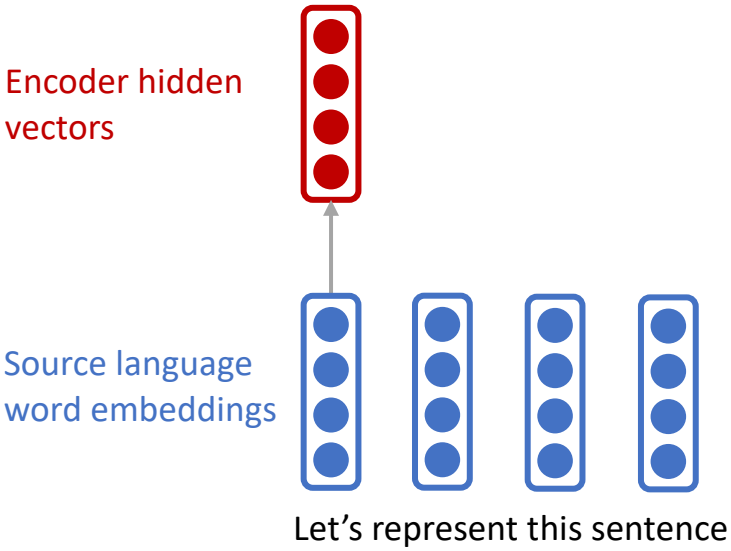


Let's represent this sentence

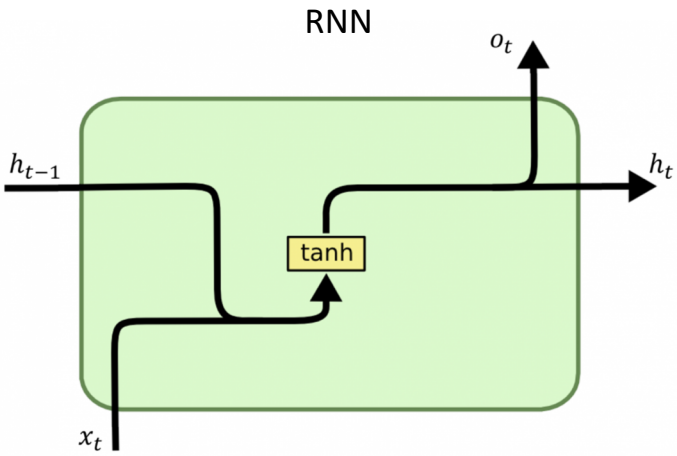
GRU

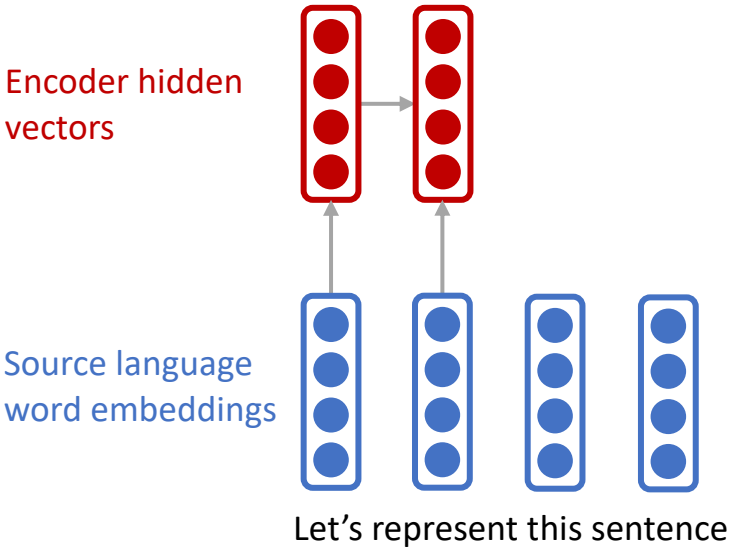




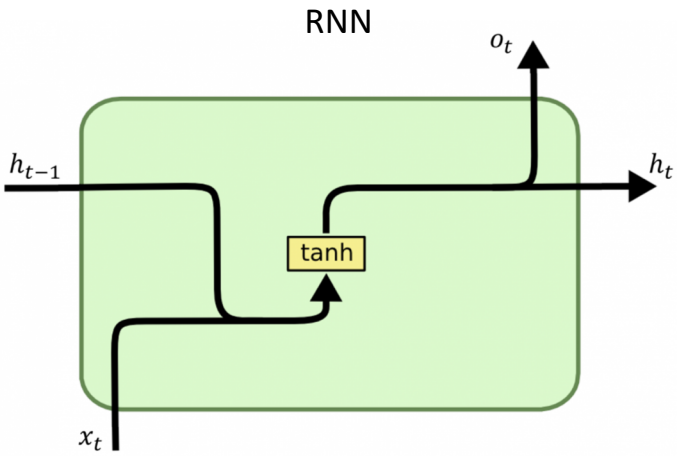


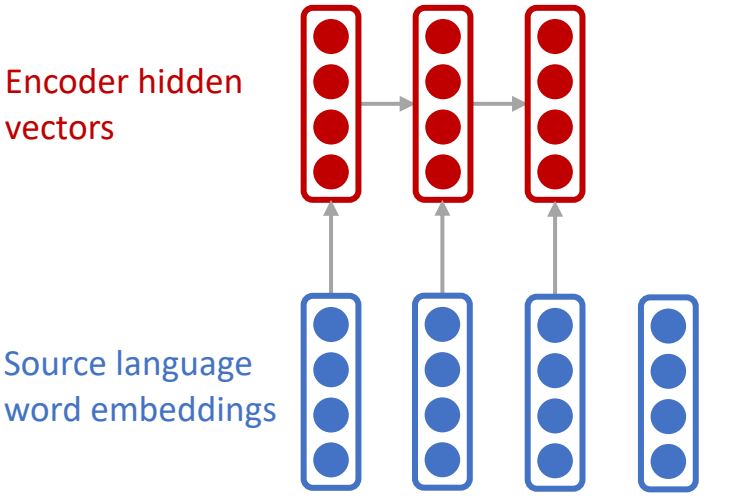
Let's represent this sentence



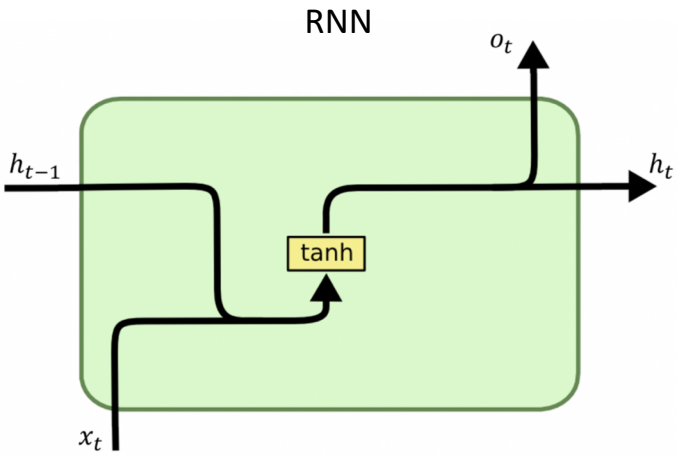


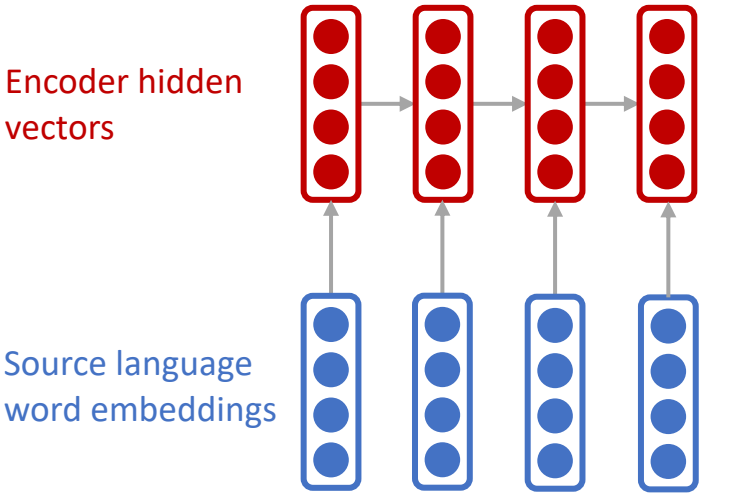
Let's represent this sentence



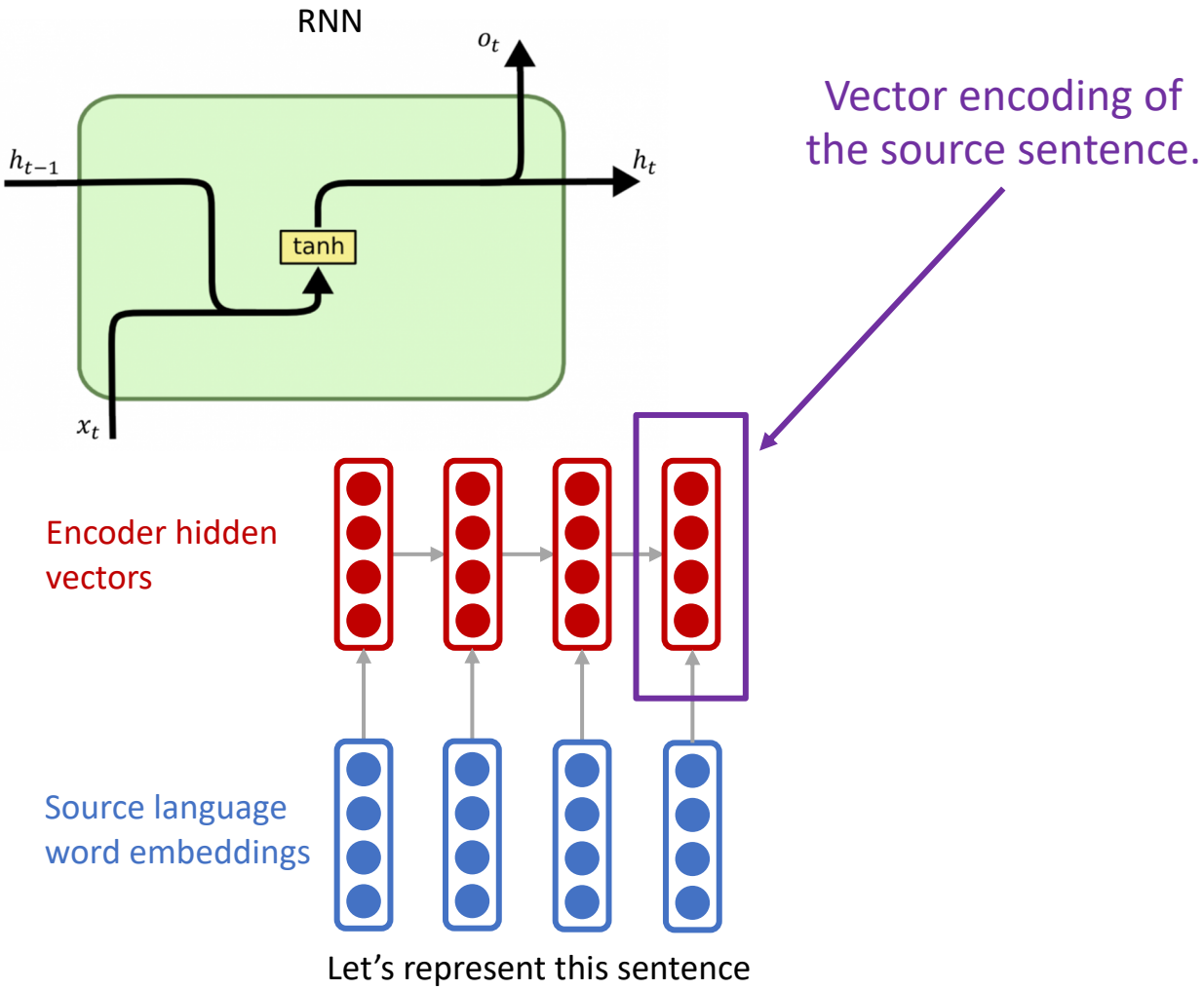


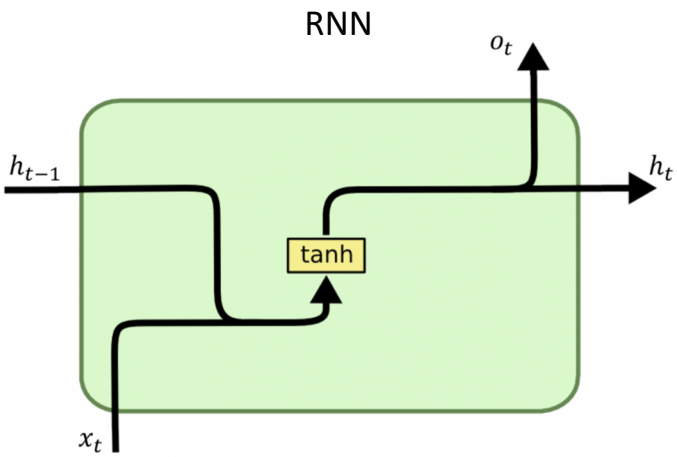
Let's represent this sentence

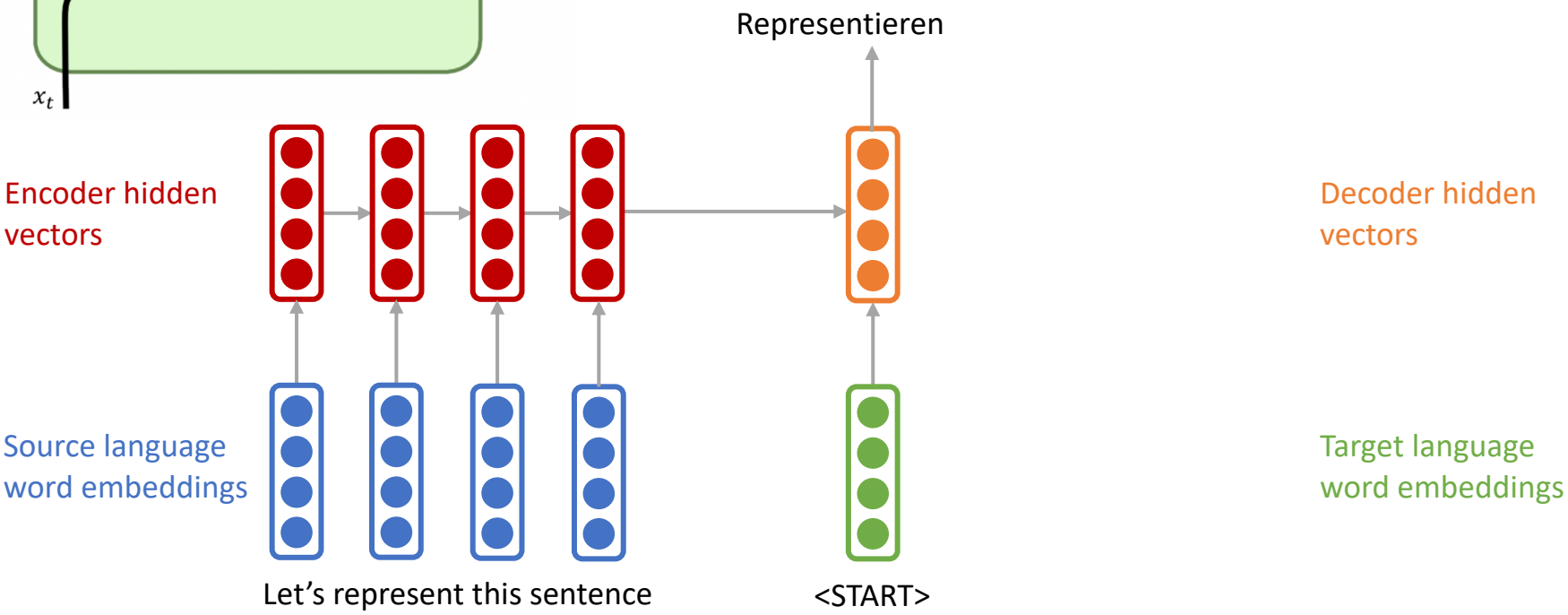


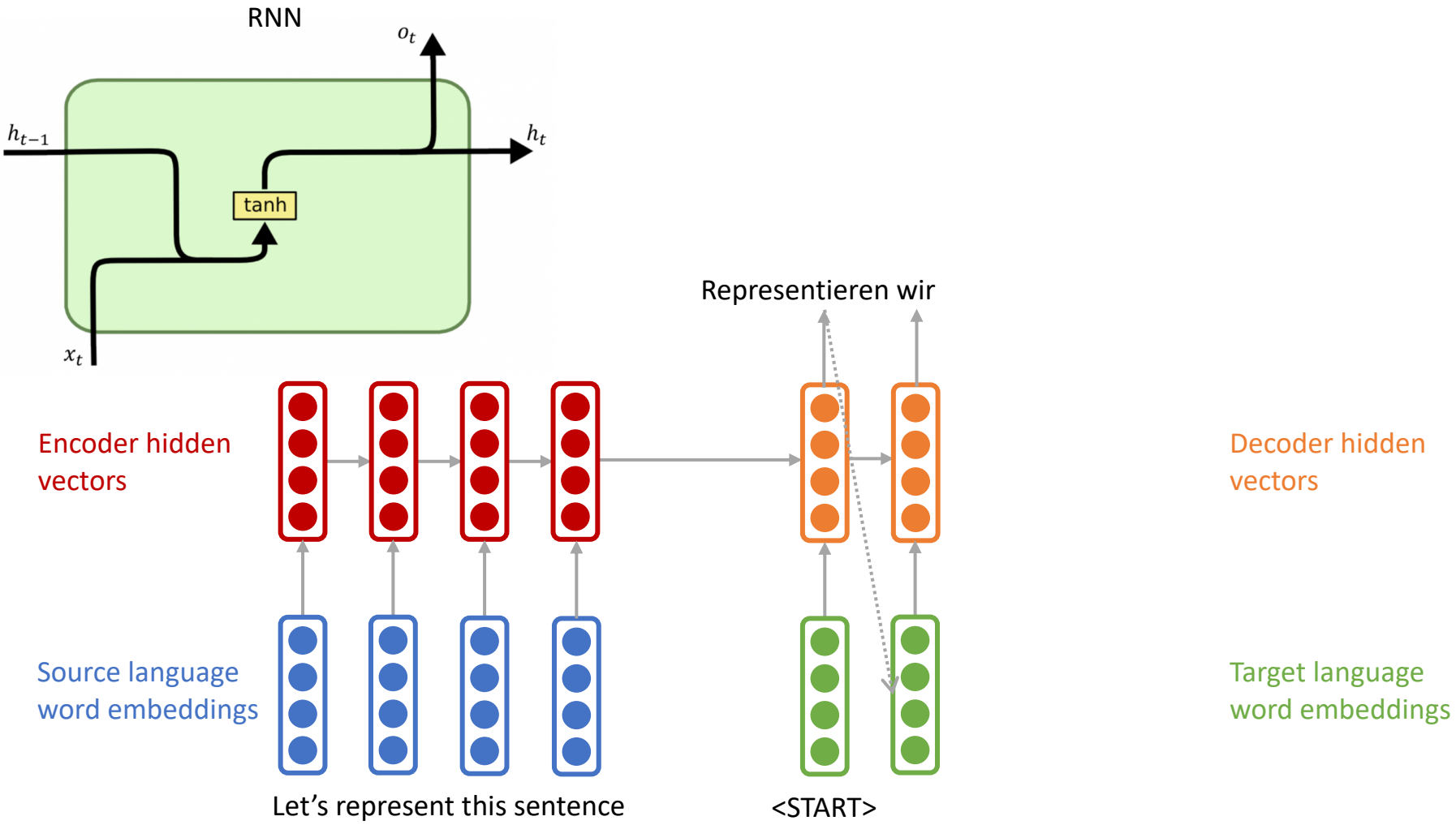


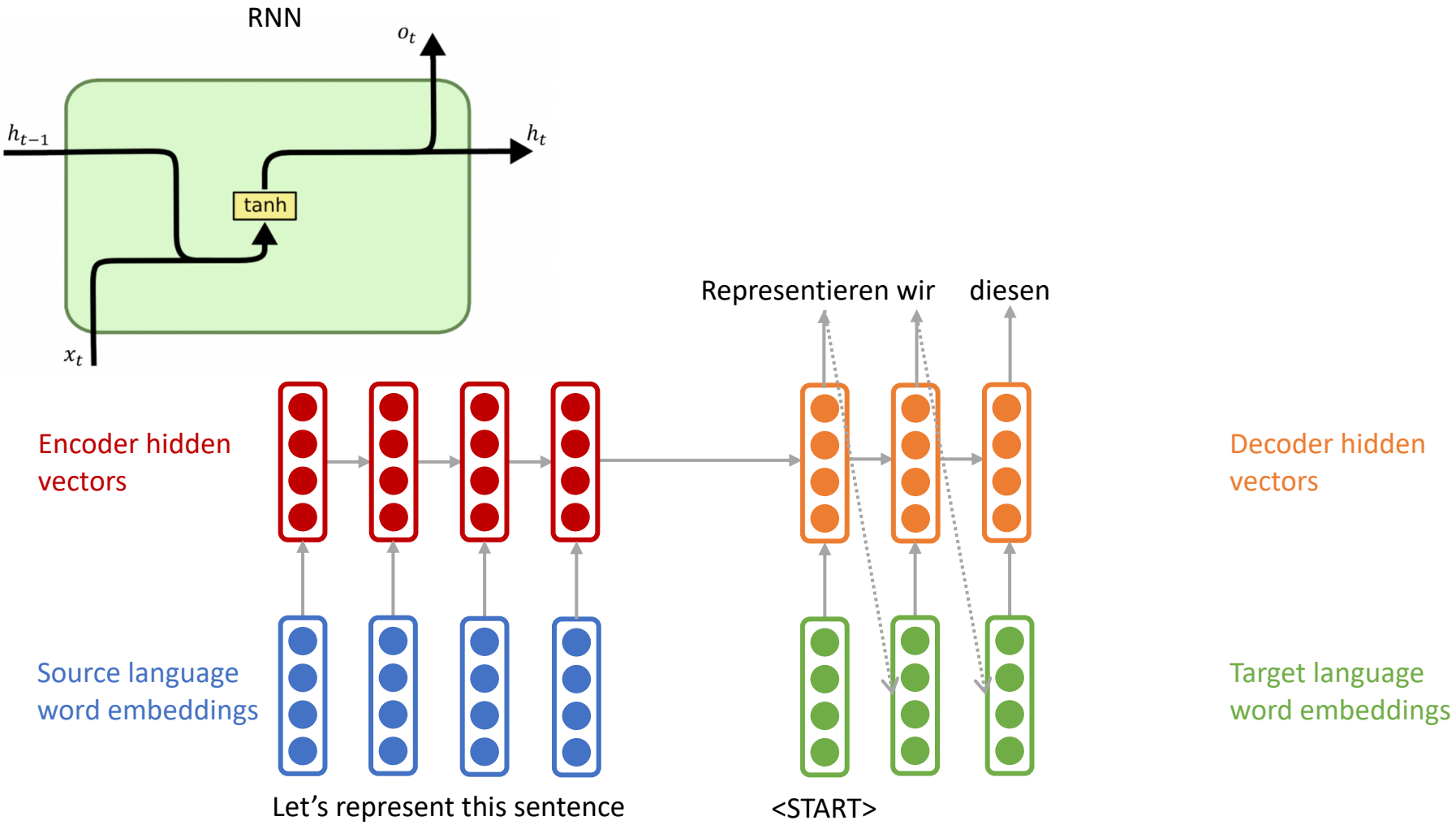
Let's represent this sentence

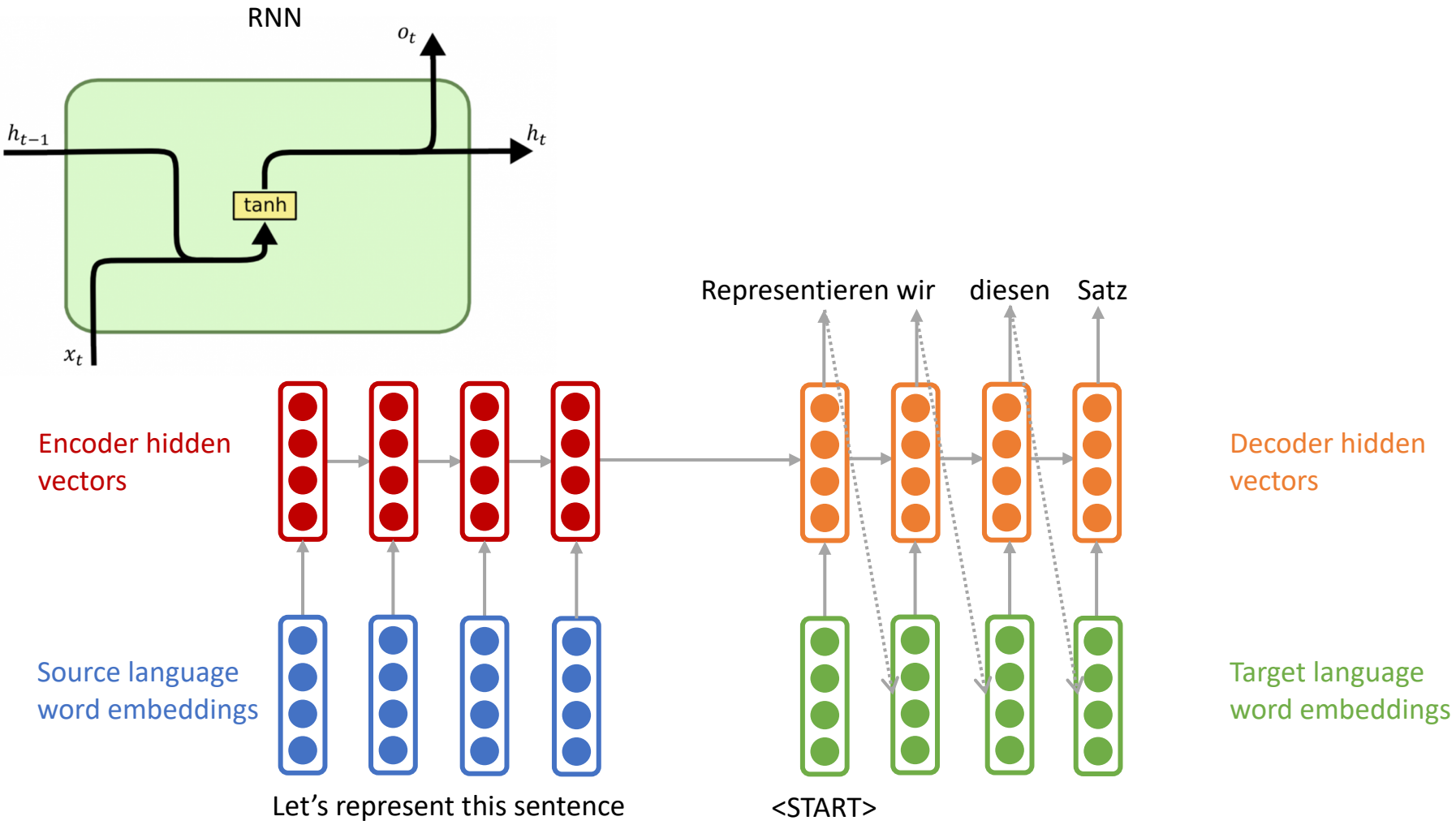

RNN
o_t
h_{t-1}
h_t
tanh
x_t
Vector encoding of
the source sentence.
Encoder hidden
vectors
Source language
word embeddings
Let's represent this sentence

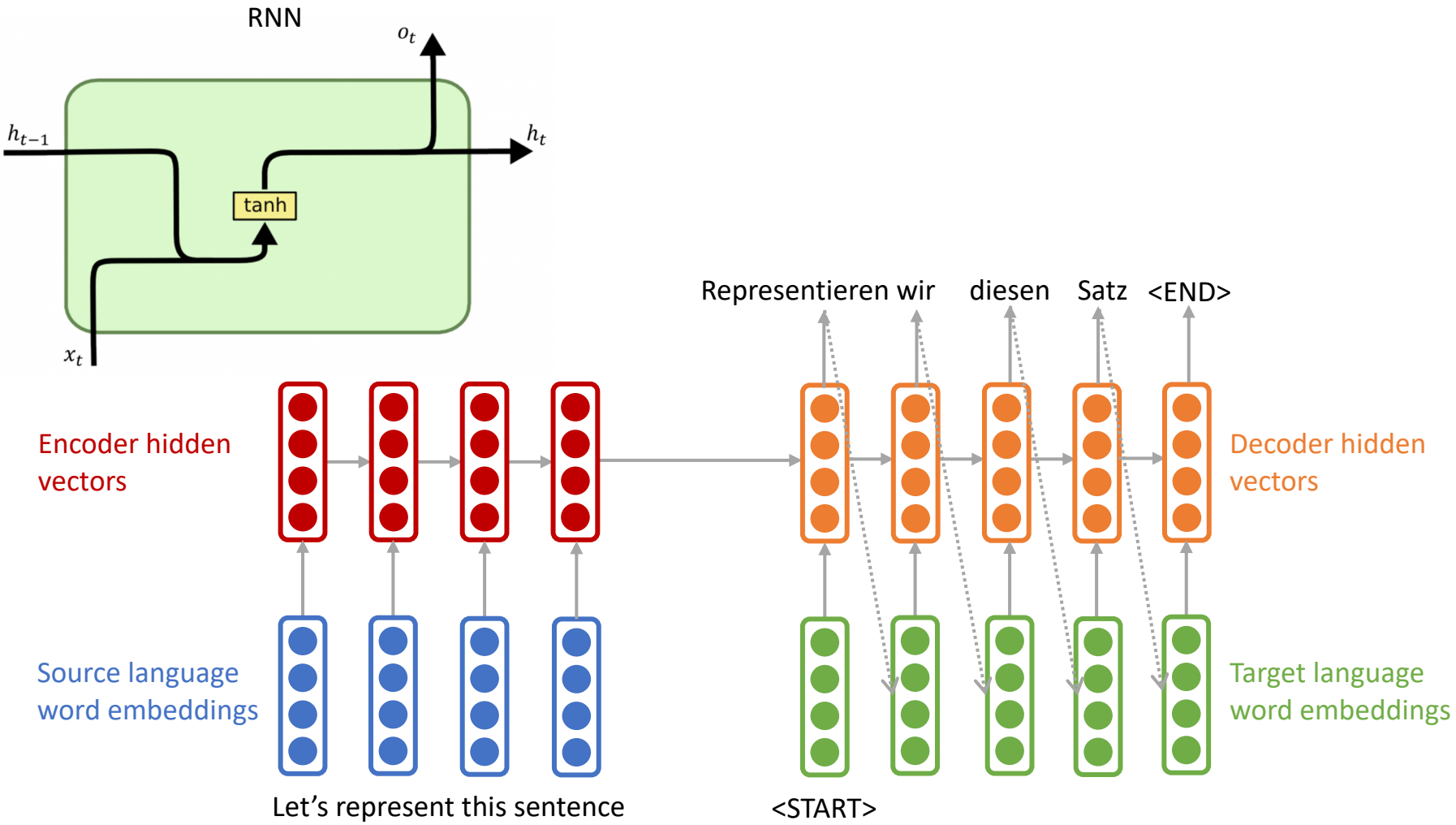

RNN
o_t
h_{t-1}
h_t
tanh
x_t


Representieren
Encoder hidden vectors
Decoder hidden vectors
Source language word embeddings
Target language word embeddings
Let's represent this sentence
<START>

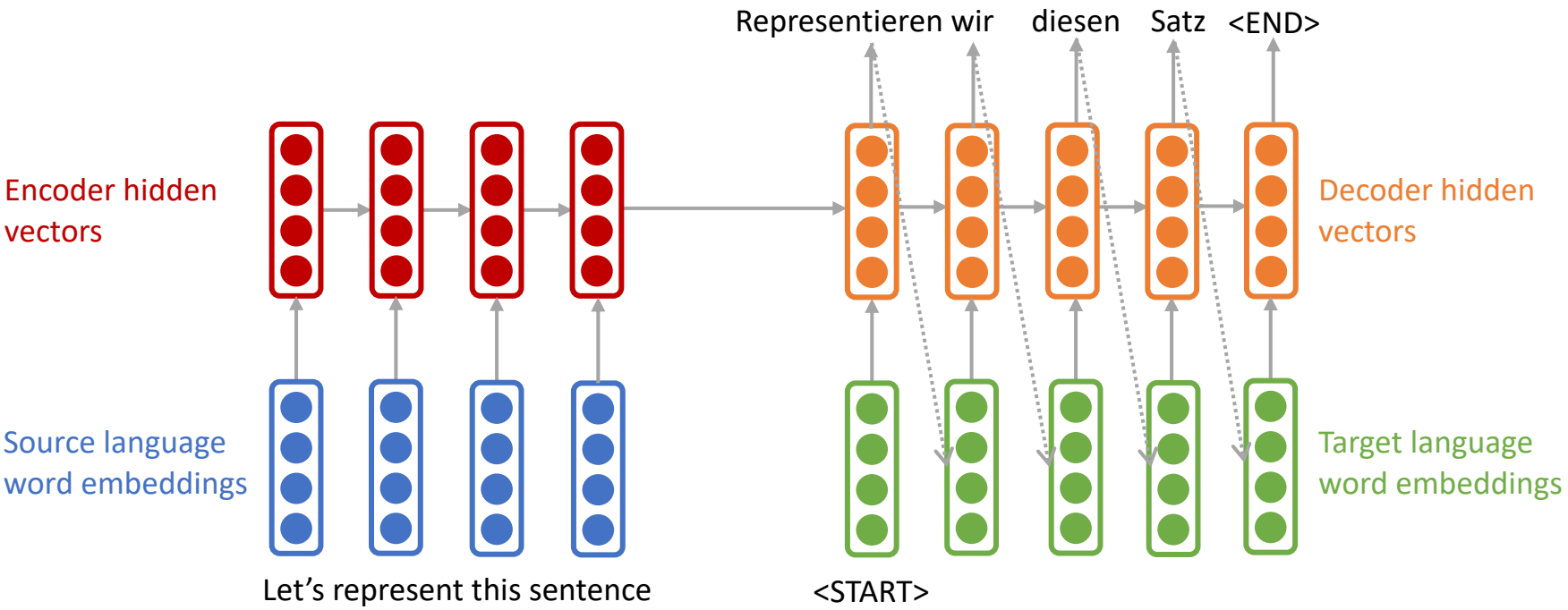

RNN
o_t
h_{t-1}
h_t
tanh
x_t
Representieren wir
Encoder hidden vectors
Decoder hidden vectors
Source language word embeddings
Target language word embeddings
Let’s represent this sentence
<START>

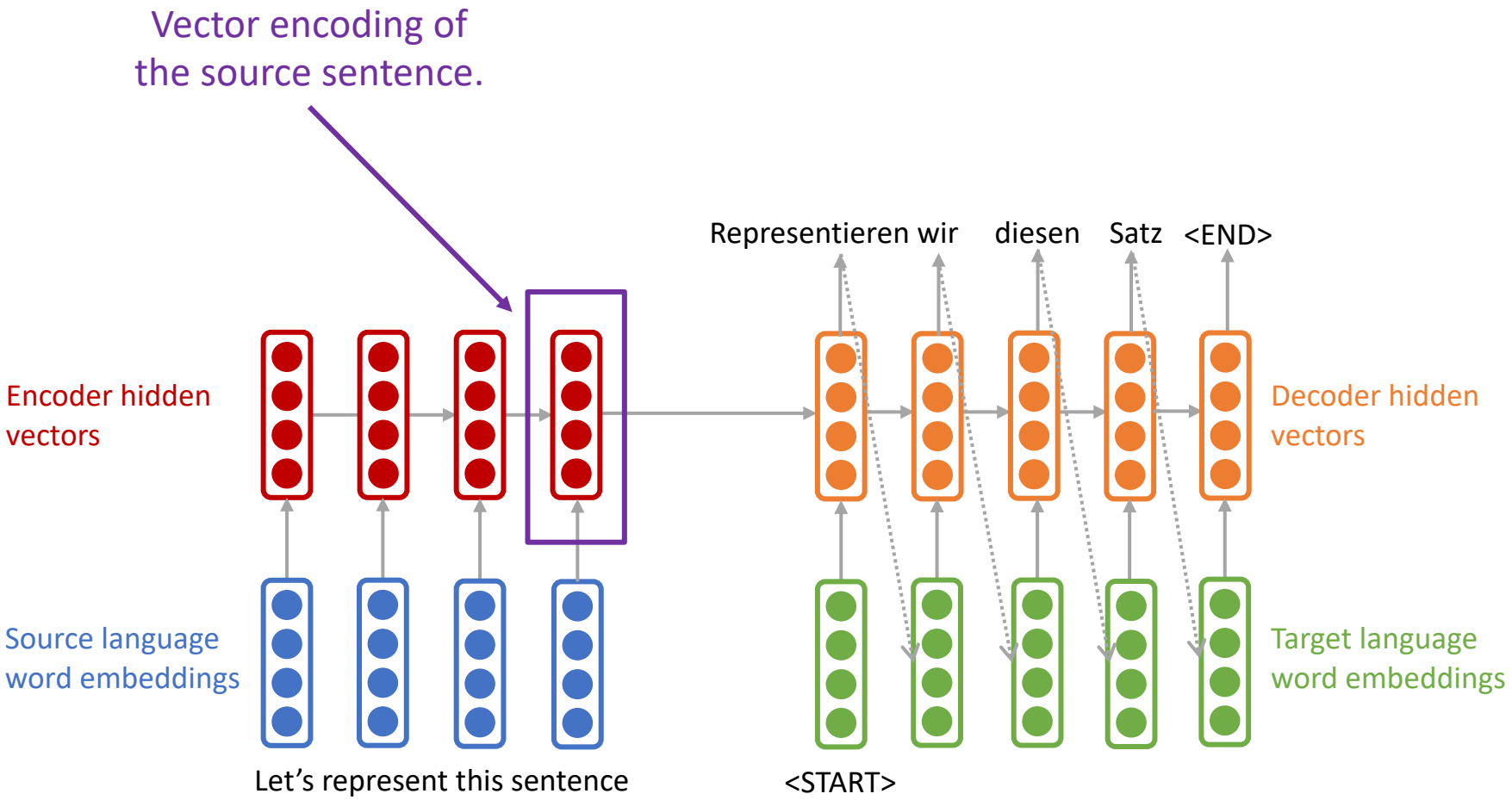

RNN
o_t
h_{t-1}
h_t
tanh
x_t
Representieren wir
diesen
Encoder hidden vectors
Decoder hidden vectors
Source language word embeddings
Target language word embeddings
Let's represent this sentence
<START>

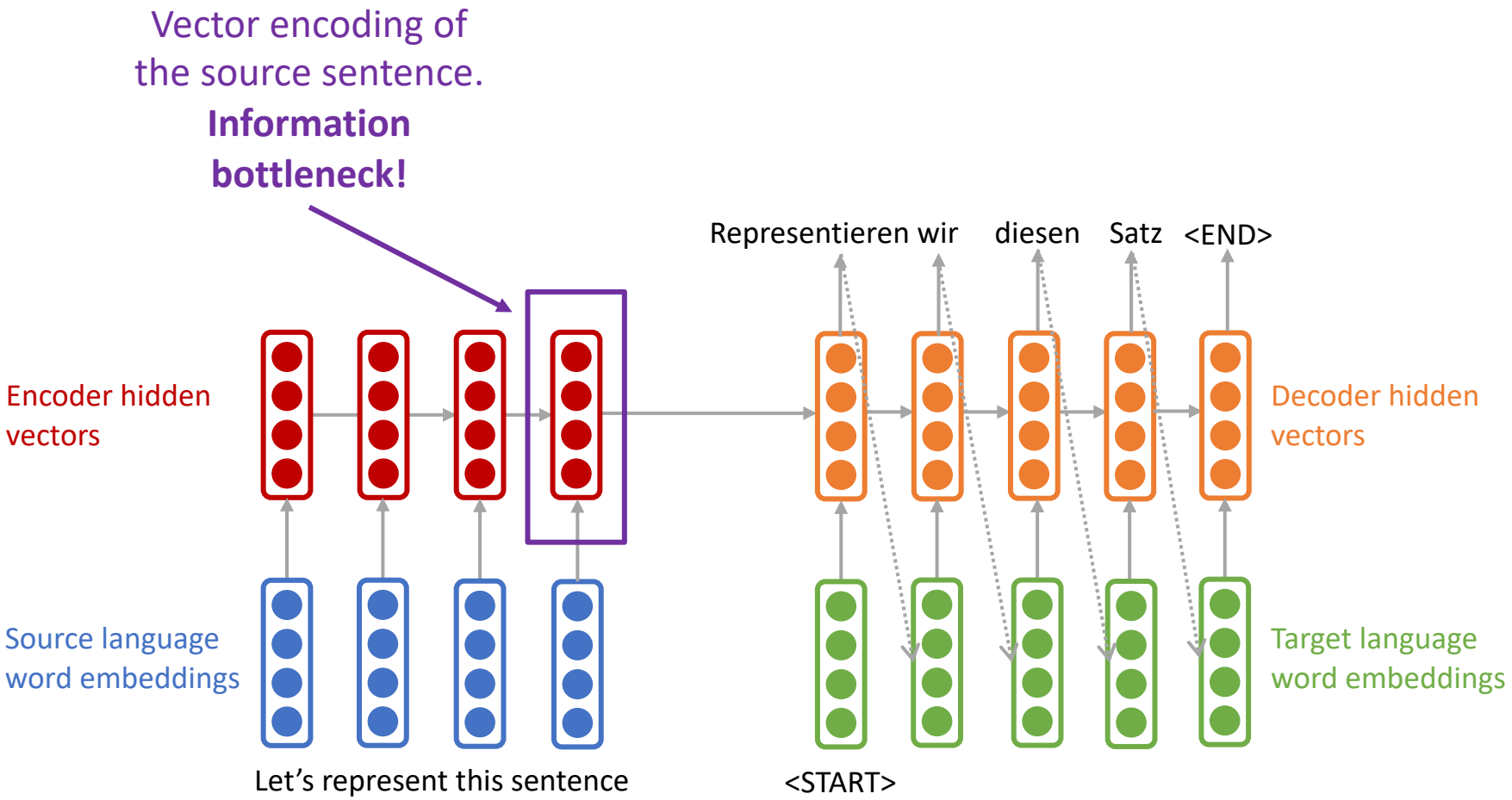

RNN
o_t
h_{t-1}
h_t
tanh
x_t
Representieren wir
diesen
Satz
Encoder hidden vectors
Decoder hidden vectors
Source language word embeddings
Target language word embeddings
Let's represent this sentence
<START>


RNN
o_t
h_{t-1}
h_t
tanh
x_t
Representieren wir
diesen
Satz
<END>
Encoder hidden vectors
Decoder hidden vectors
Source language word embeddings
Target language word embeddings
Let's represent this sentence
<START>

Any issues with this architecture?




Vector encoding of
the source sentence.
Representieren wir
diesen
Satz
<END>
Encoder hidden
vectors
Decoder hidden
vectors
Source language
word embeddings
Target language
word embeddings
Let's represent this sentence
<START>

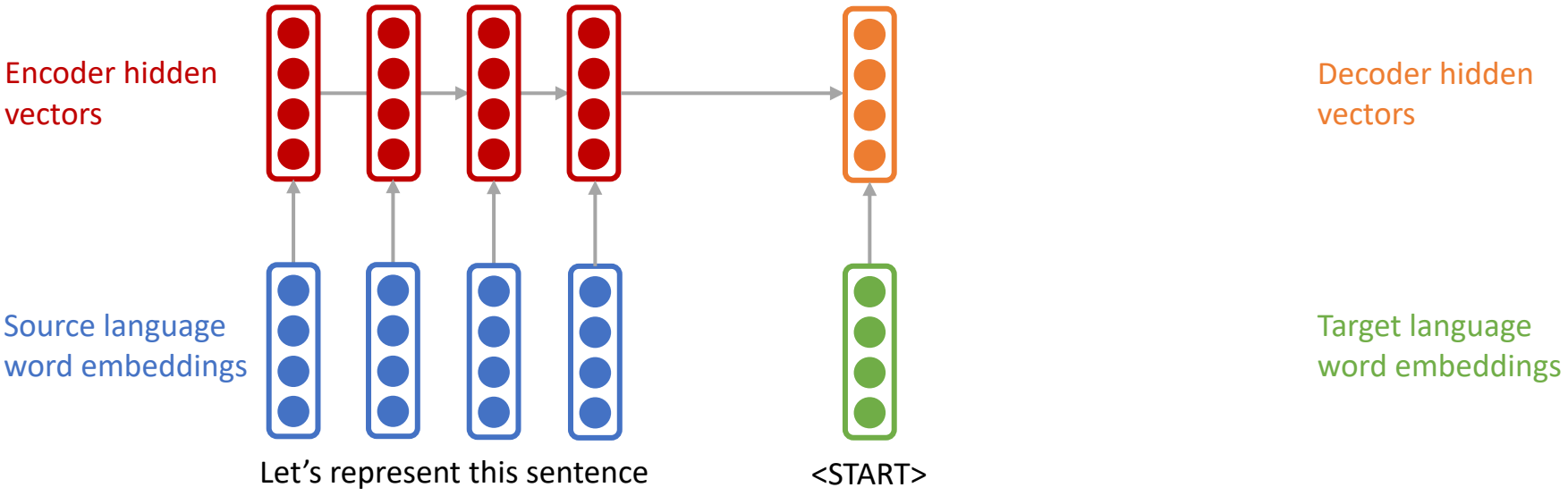

Vector encoding of
the source sentence.
Information
bottleneck!
Representieren wir
diesen
Satz
<END>
Encoder hidden
vectors
Decoder hidden
vectors
Source language
word embeddings
Target language
word embeddings
Let's represent this sentence
<START>

Sequence-to-sequence models with **attention**

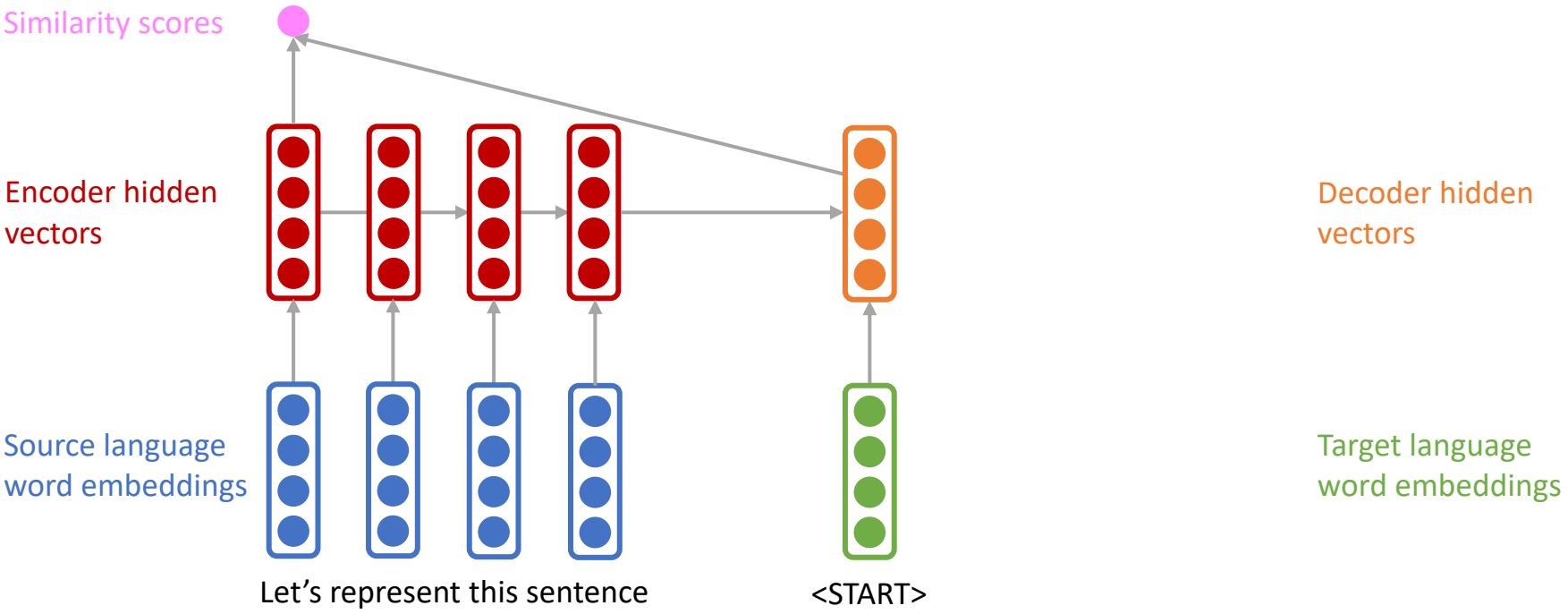
Attention: the motivation

- Provides a solution to the bottleneck problem.
- Allows the decoder to have direct access to all the hidden states of the encoder.
- At each step, the decoder focuses on the part of the input sentence that is most relevant.

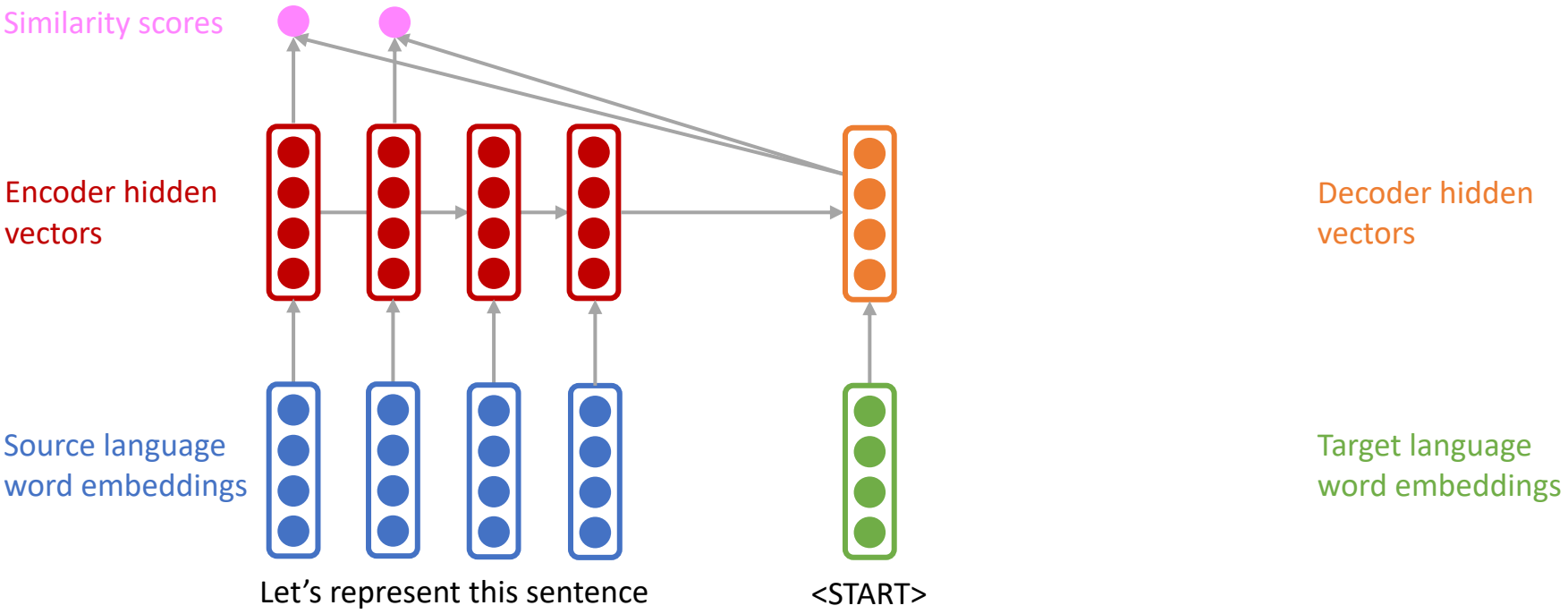
Generate 1st decoder hidden vector as before.



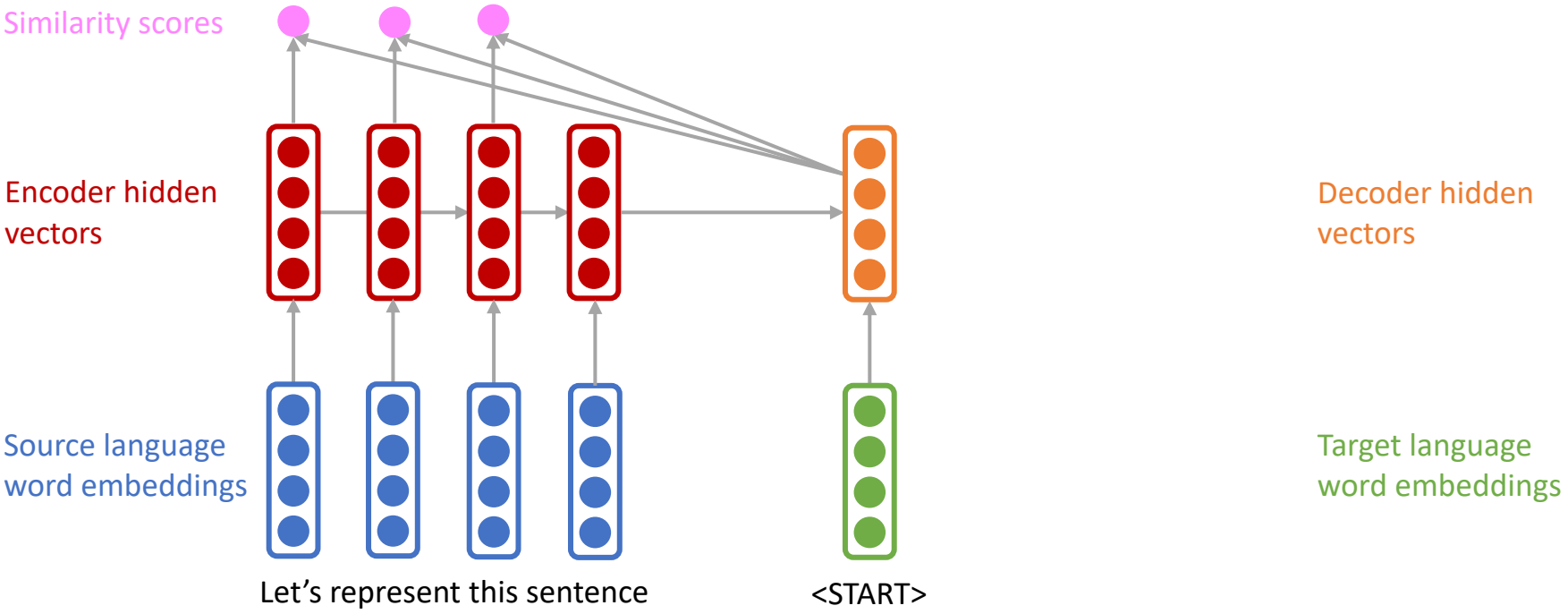
Calculate a **similarity score** between the decoder hidden vector and all the encoder hidden vectors.



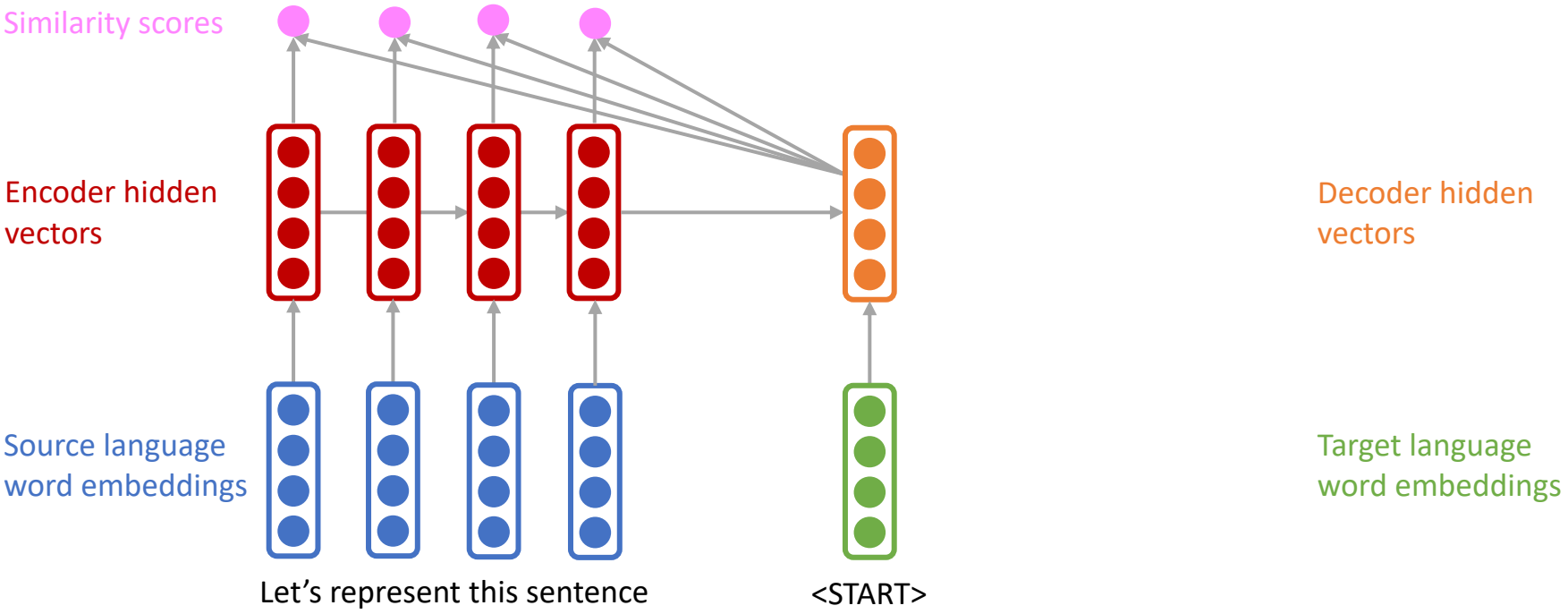
Calculate a **similarity score** between the decoder hidden vector and all the encoder hidden vectors.



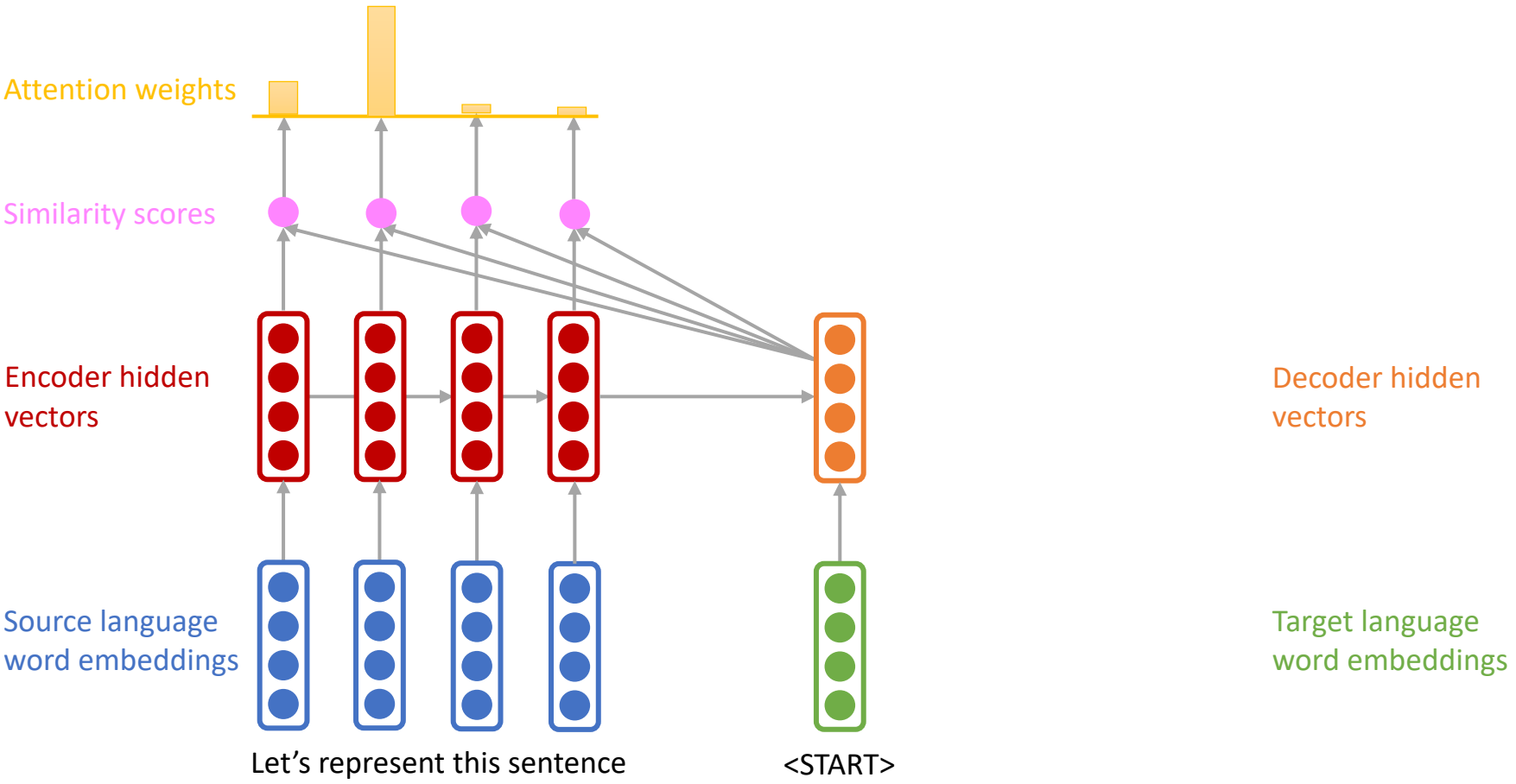
Calculate a **similarity score** between the decoder hidden vector and all the encoder hidden vectors.

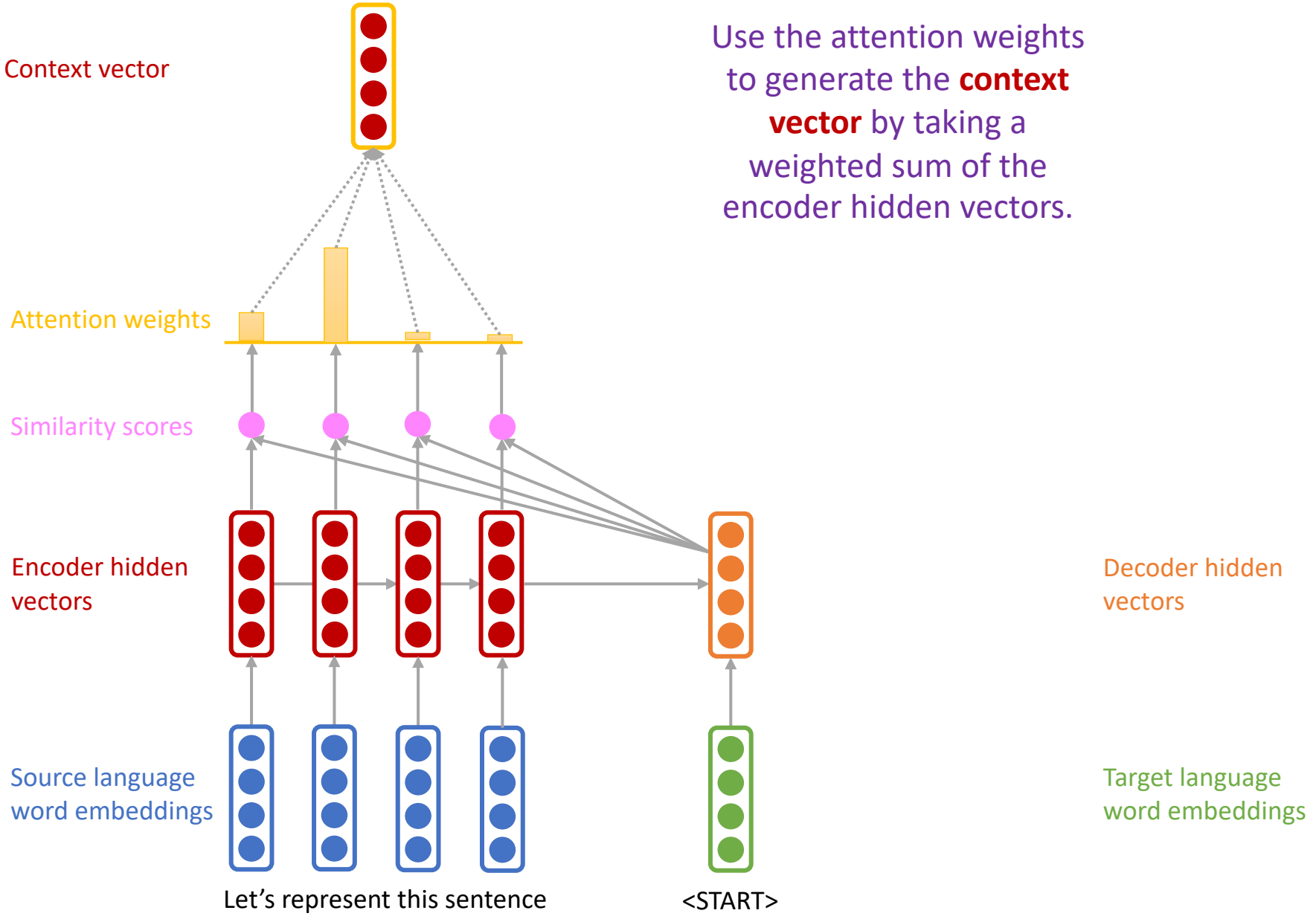


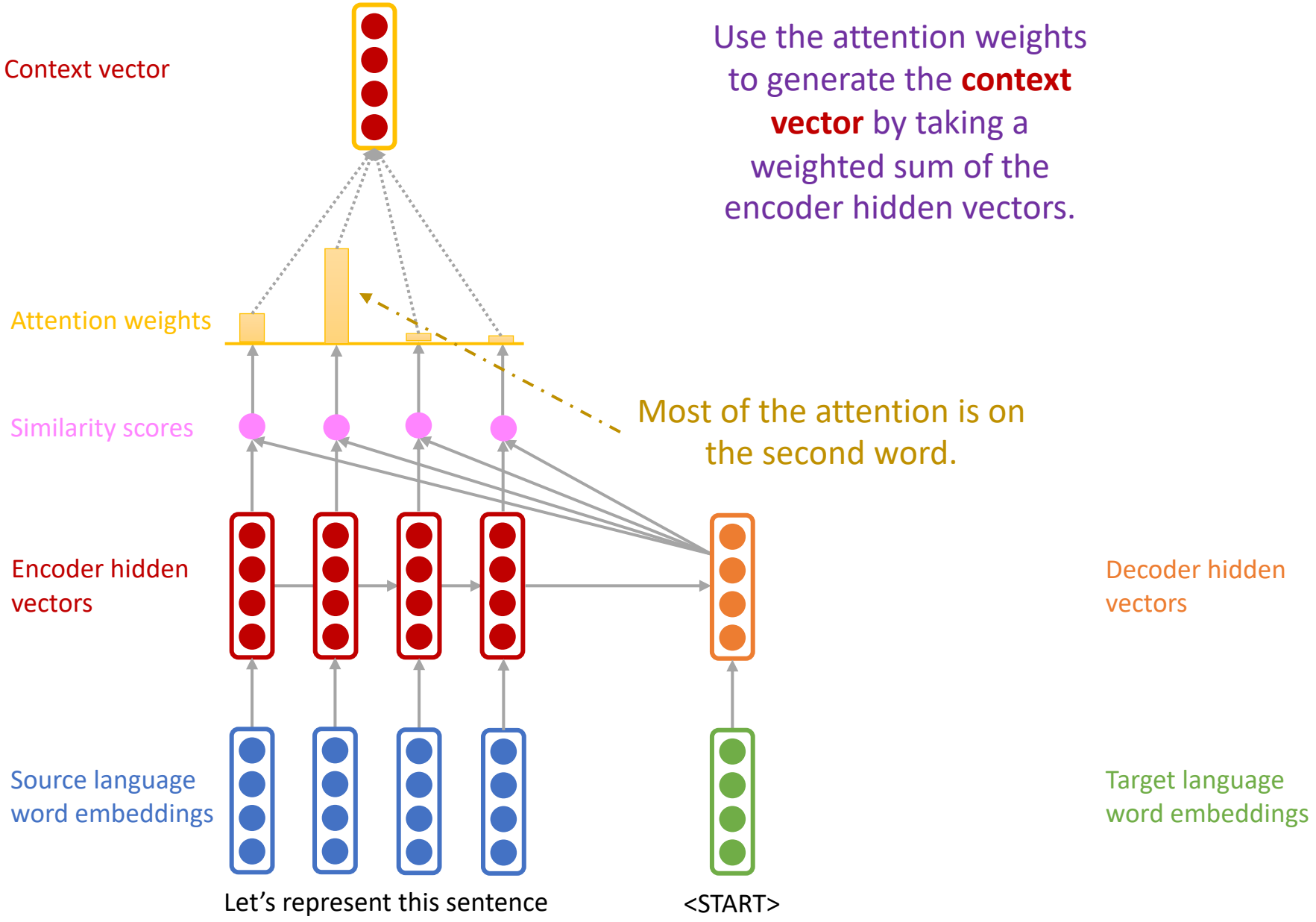
Calculate a **similarity score** between the decoder hidden vector and all the encoder hidden vectors.

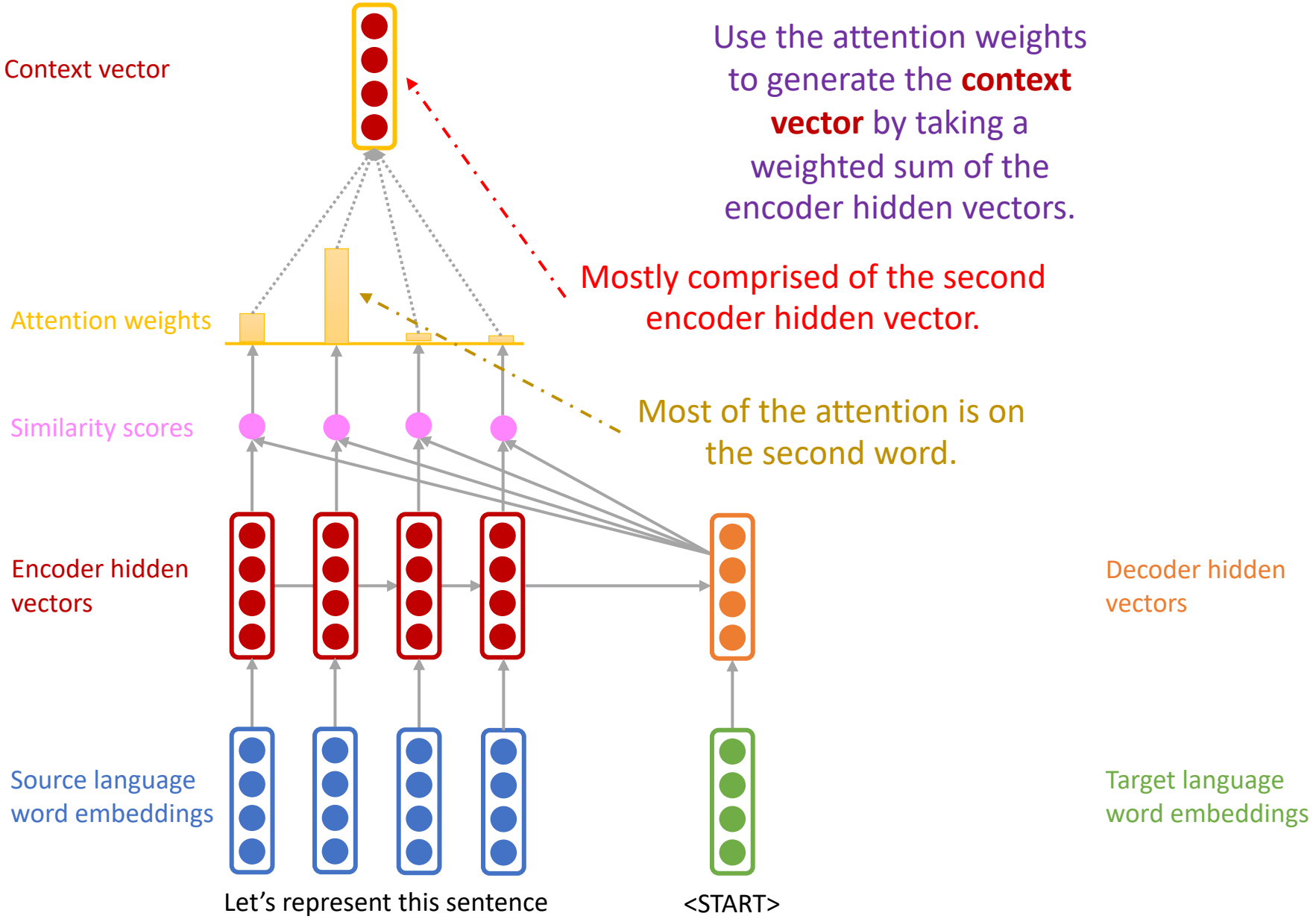


Pass similarity score vector through a softmax to generate **attention weights**.

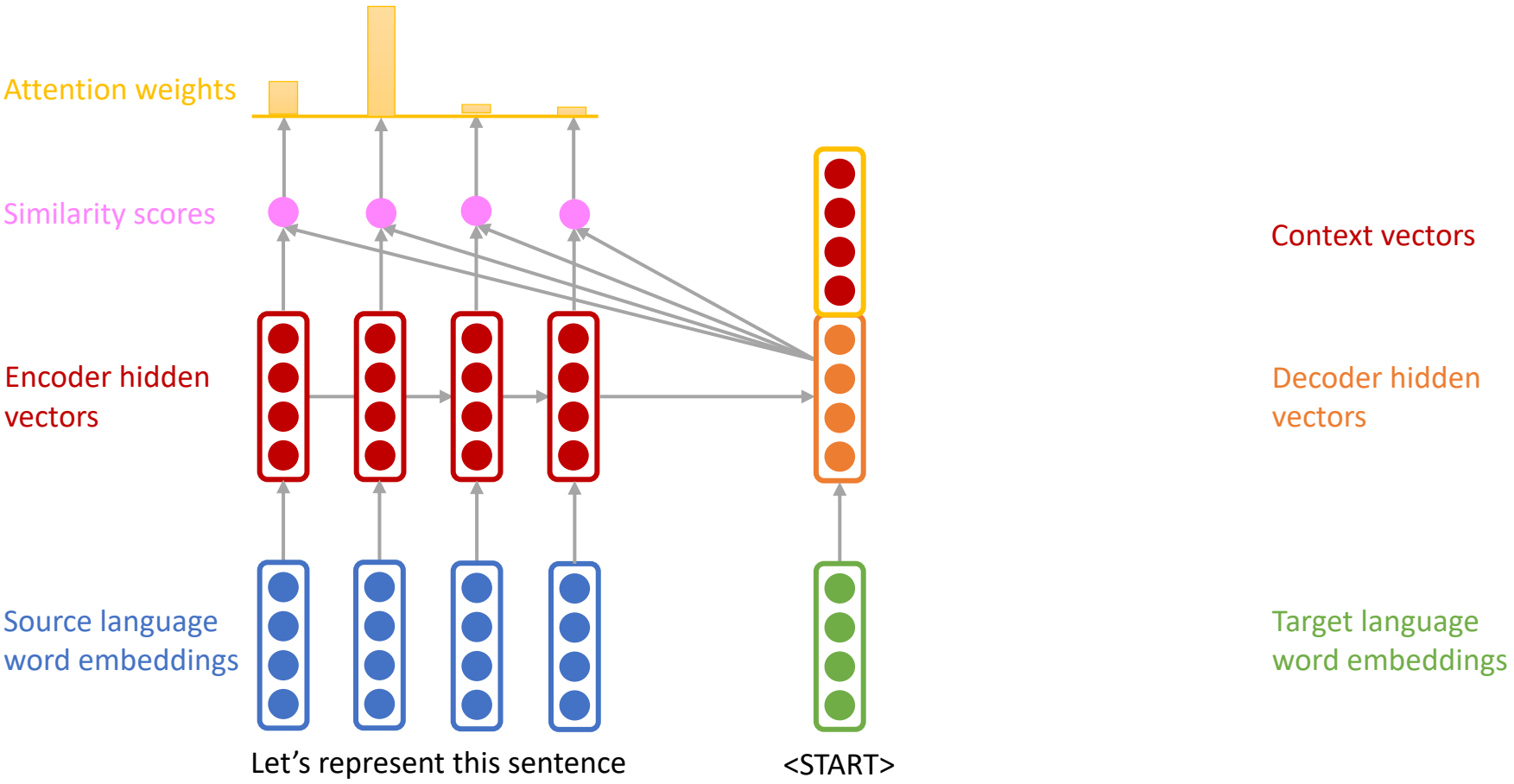



Context vector
Use the attention weights to generate the **context vector** by taking a weighted sum of the encoder hidden vectors.
Attention weights
Similarity scores
Encoder hidden vectors
Decoder hidden vectors
Source language word embeddings
Target language word embeddings
Let's represent this sentence
<START>

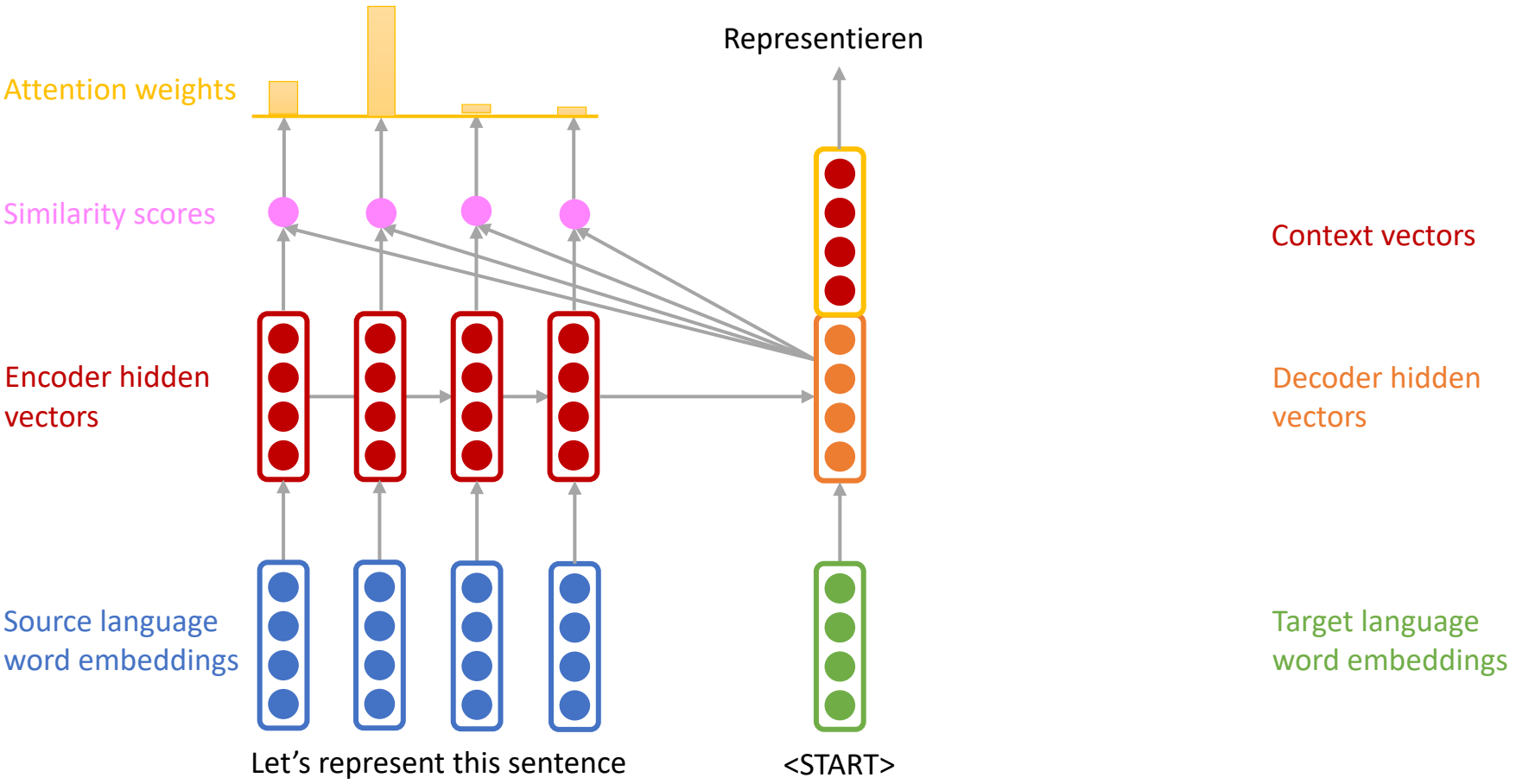

Context vector
Use the attention weights to generate the **context vector** by taking a weighted sum of the encoder hidden vectors.
Attention weights
Similarity scores
Most of the attention is on the second word.
Encoder hidden vectors
Decoder hidden vectors
Source language word embeddings
Target language word embeddings
Let's represent this sentence
<START>


Context vector
Use the attention weights to generate the **context vector** by taking a weighted sum of the encoder hidden vectors.
Mostly comprised of the second encoder hidden vector.
Attention weights
Most of the attention is on the second word.
Similarity scores
Encoder hidden vectors
Decoder hidden vectors
Source language word embeddings
Target language word embeddings
Let's represent this sentence
<START>

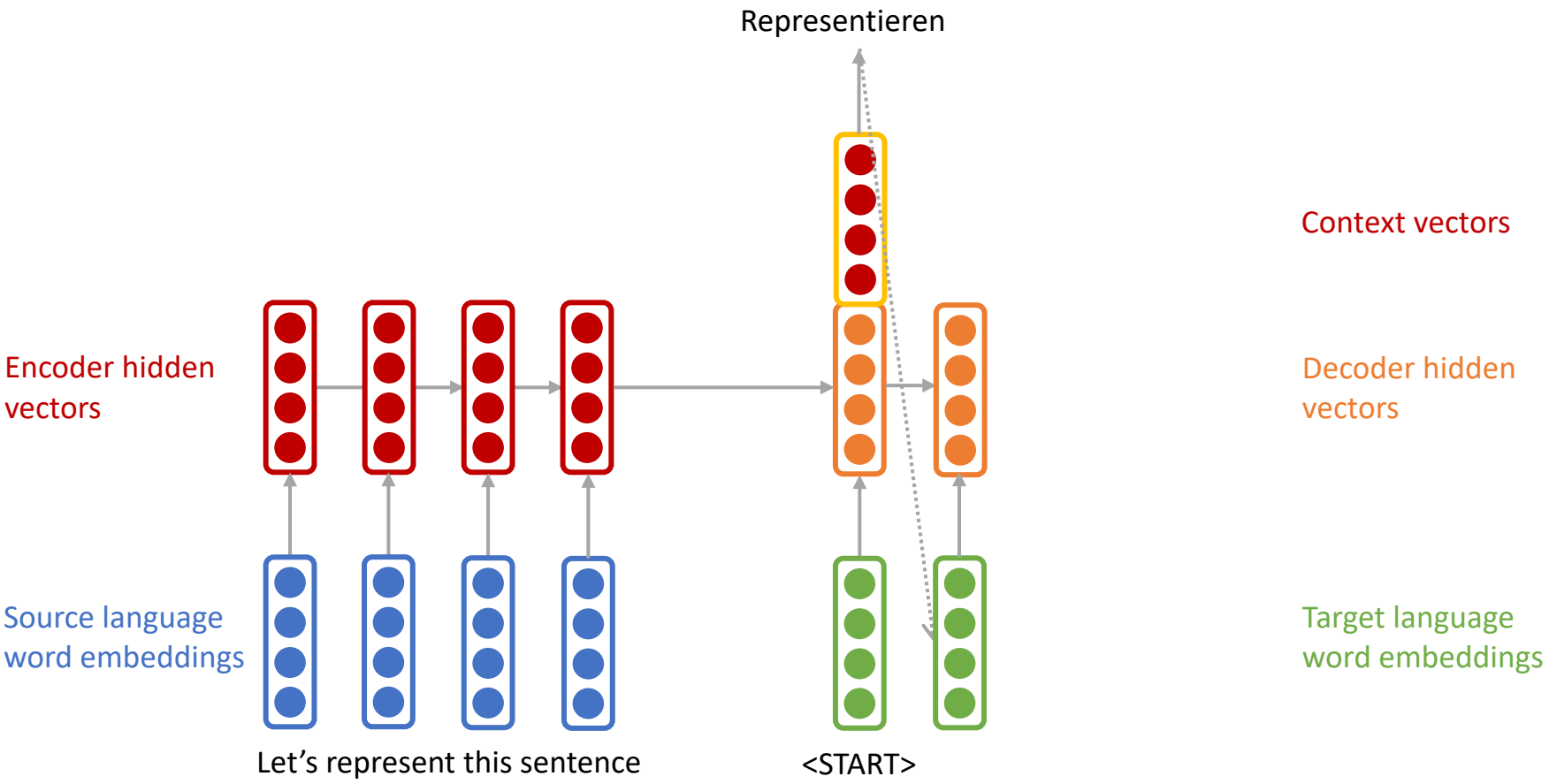
Concatenate the context vector to the decoder hidden vector



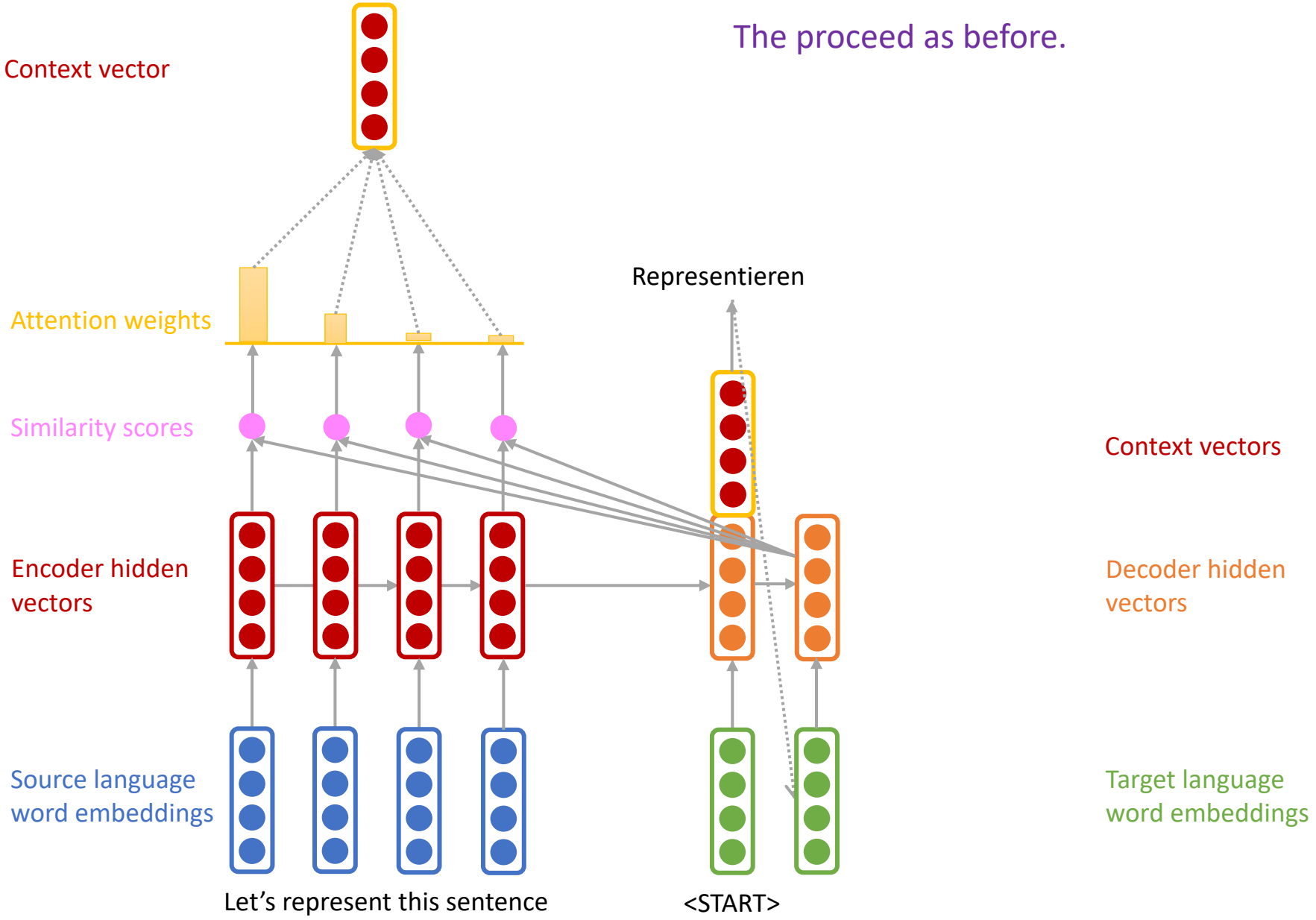
Concatenate the context vector to the decoder hidden vector and generate the first word.



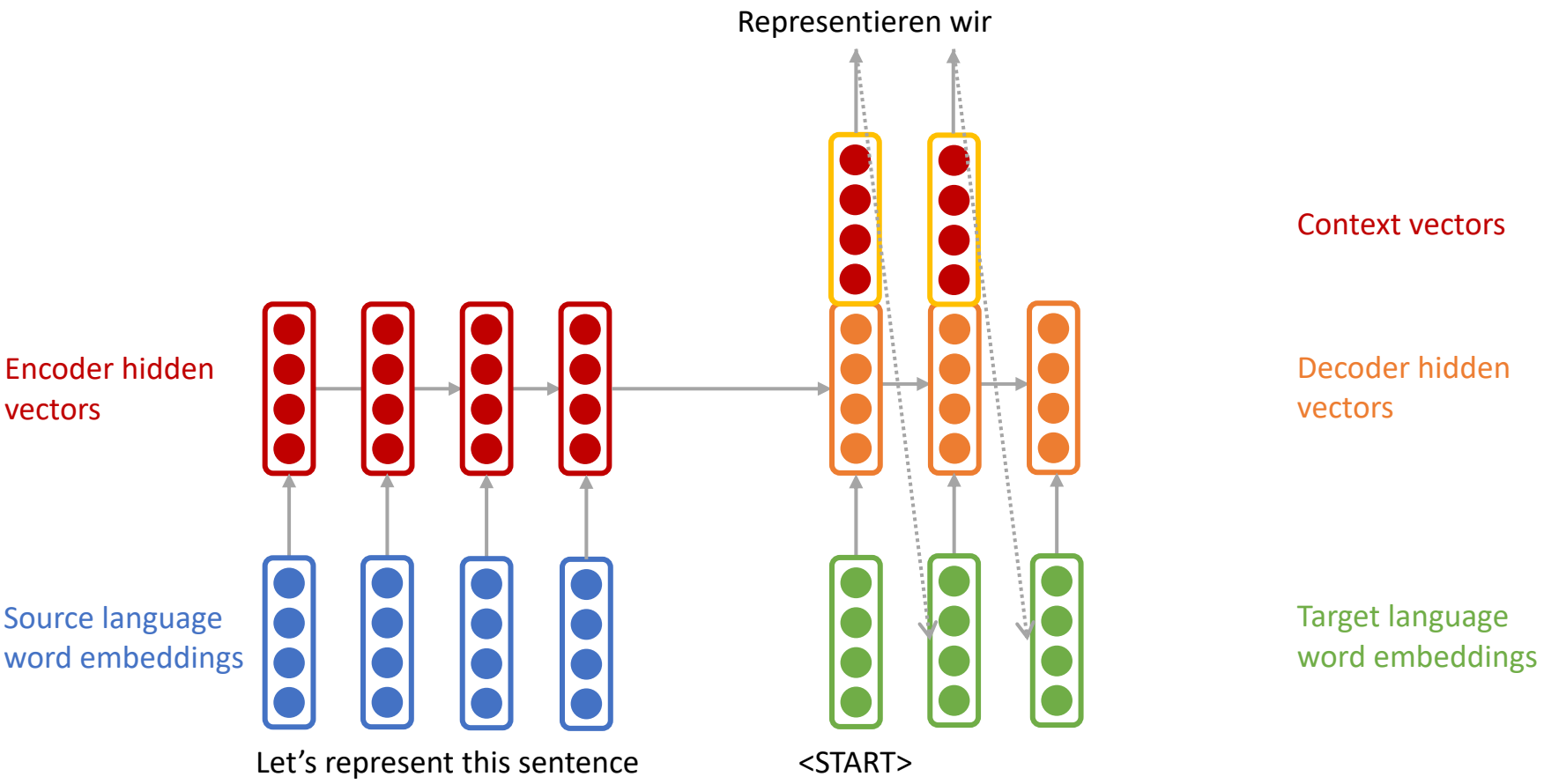
The proceed as before.



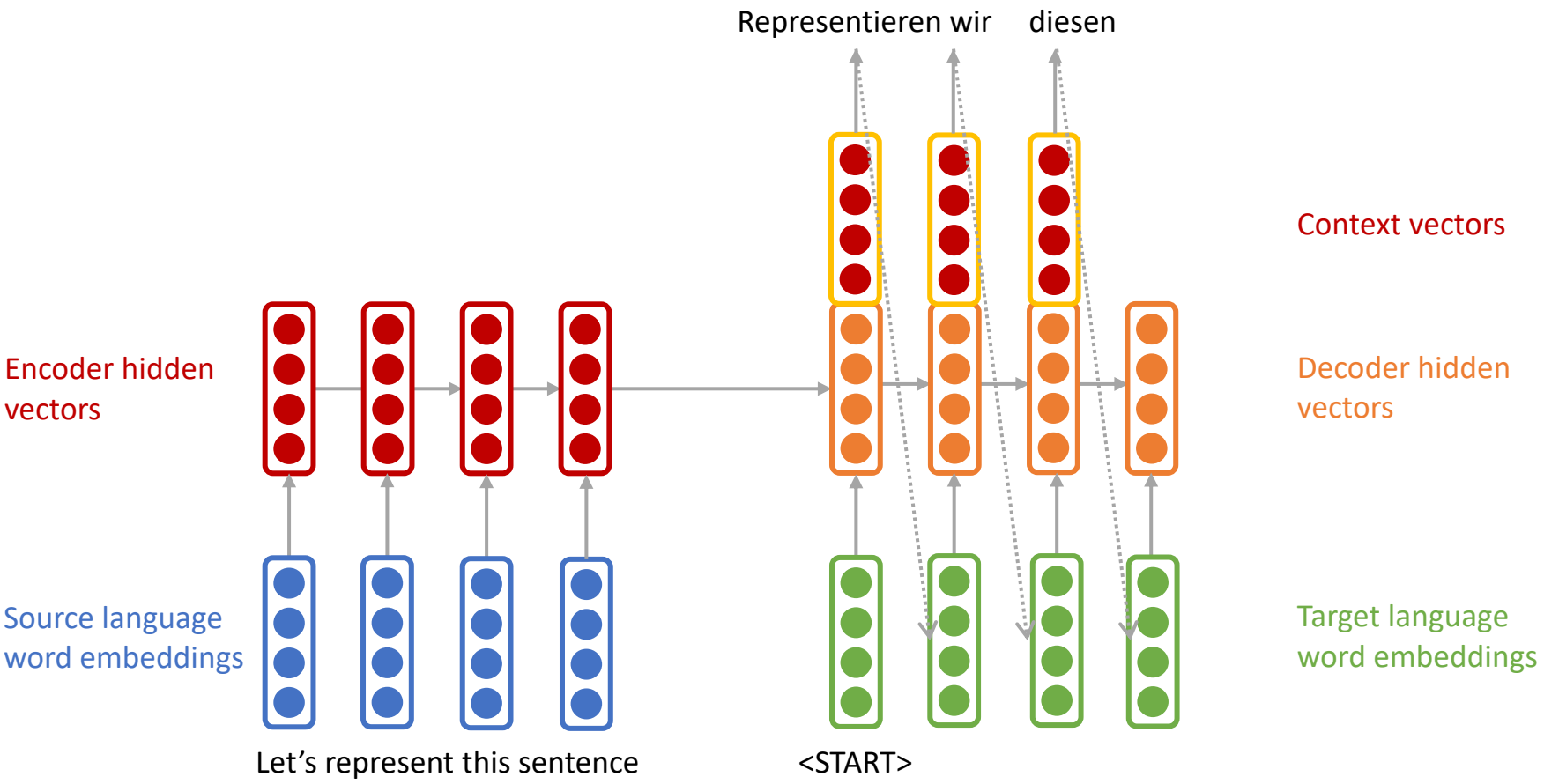
The proceed as before.



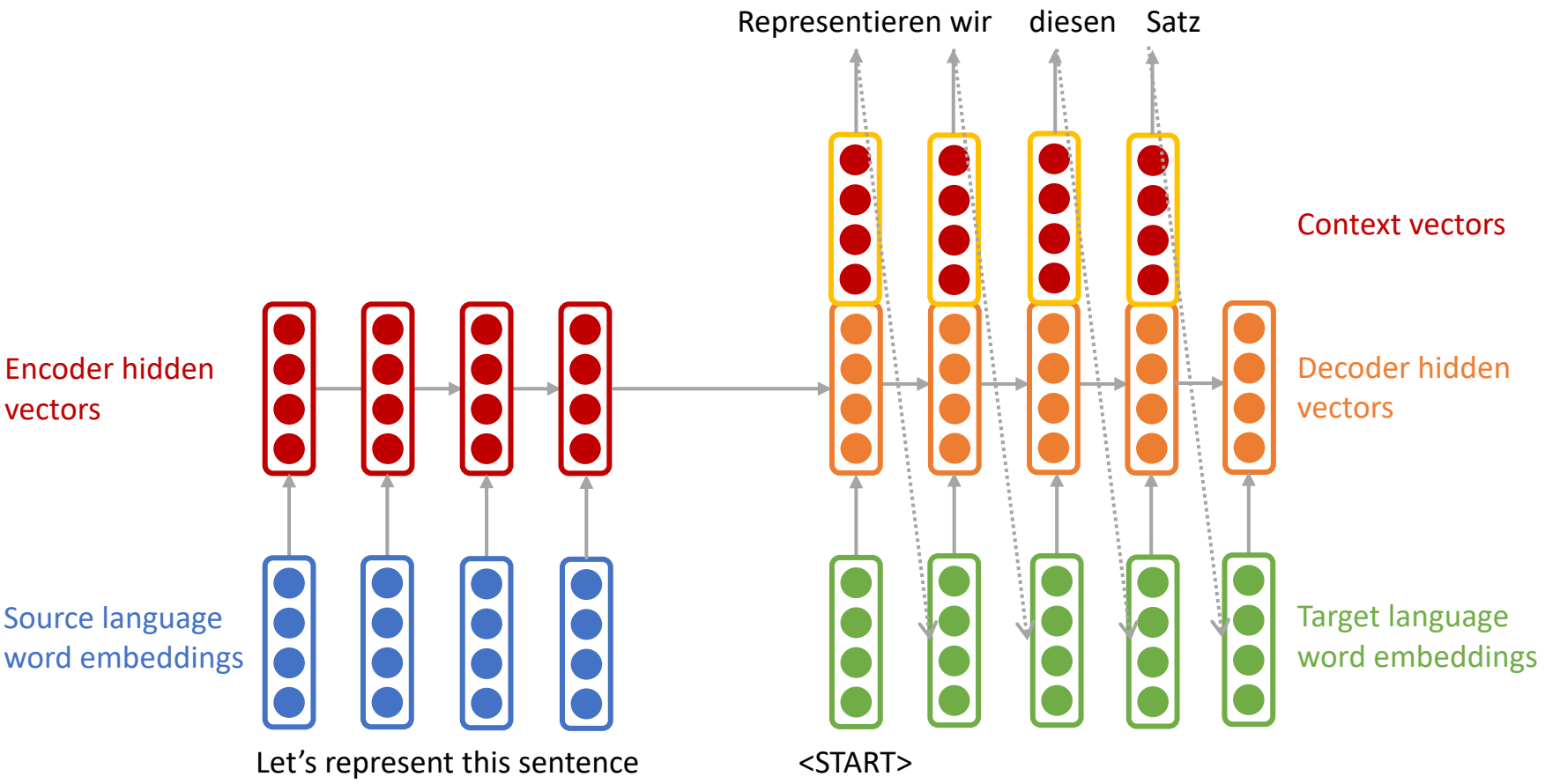
The proceed as before.



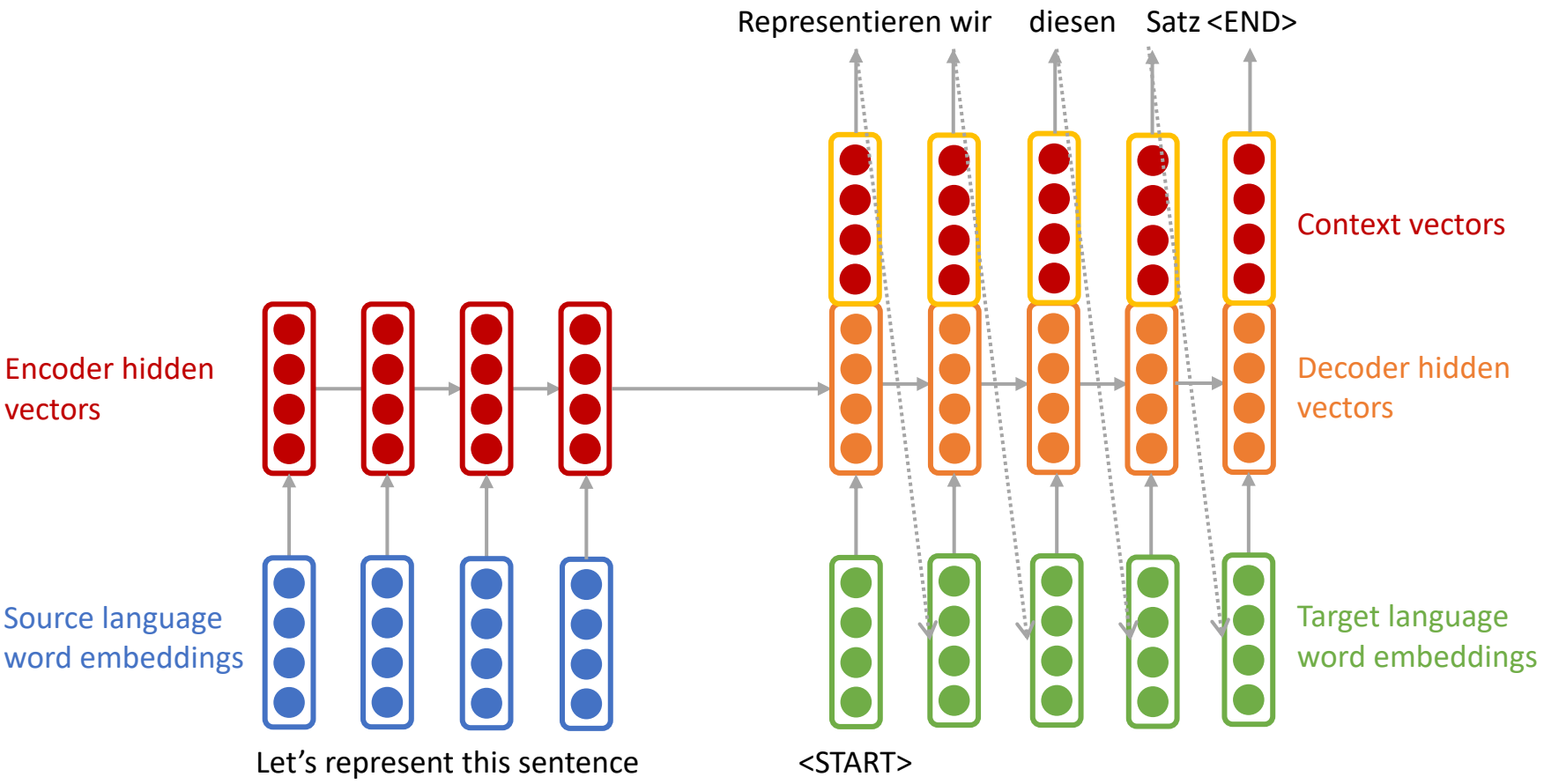
The proceed as before.



The proceed as before.



Translation is complete when the <END> token is generated.

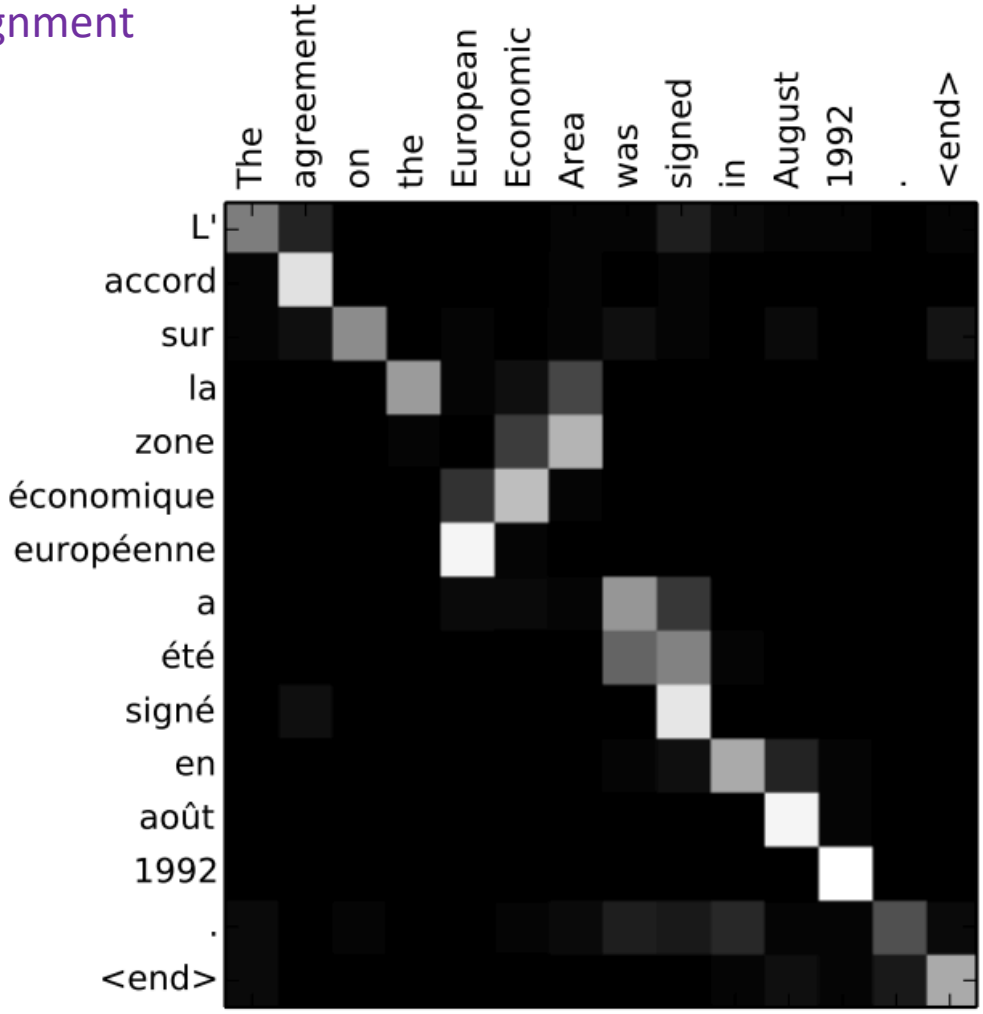


Attention: the equations

- Generate the encoder hidden states $h_1, \ldots, h_N \in \mathbb{R}^h$
- When decoding, generate hidden states $s_t \in \mathbb{R}^h$ on time step t
- Calculate the similarity scores using the dot product:
$$e^t = [s_t^T h_1, \ldots, s_t^T h_N] \in \mathbb{R}^N$$
- Calculate the attention weights by passing through softmax:
$$\alpha^t = \text{softmax}(e^t) \in \mathbb{R}^N$$
- Calculate the context vector as a weighted sum of the encoder hidden states:
$$a_t = \sum_{i=1}^{N} \alpha_i^t h_i \in \mathbb{R}^N$$
- Concatenate the context vector a_t with the decoder hidden state s_t and proceed as in the non-attention seq2seq model.

Attention: the interpretation

We get (soft) alignment for free.



Attention: the more general definition

- **Definition:** Given a set of value vectors and a query vector, attention is a technique to compute a weighted sum of the values, dependent on the query, using an attention function.

- The weighted sum is a selective summary of the information contained in the values, where the query determines which values to focus on.

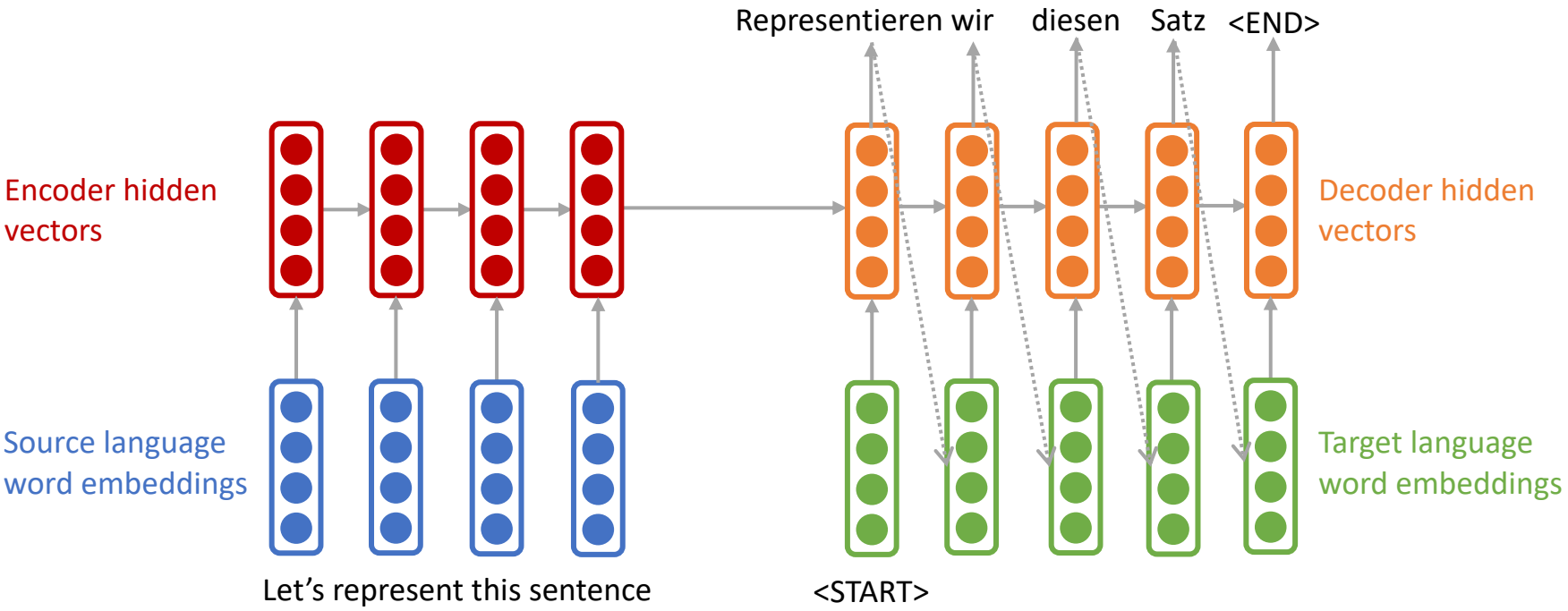
- There is a number of attention functions that are commonly used.

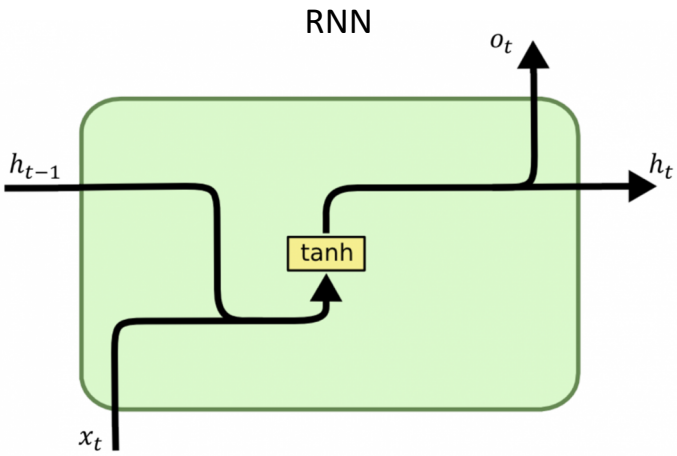
Attention: the many forms

- Dot-product attention $\boldsymbol{e}_i = \boldsymbol{s}^T \boldsymbol{h}_i \in \mathbb{R}$
- Multiplicative attention $\boldsymbol{e}_i = \boldsymbol{s}^T \boldsymbol{W} \boldsymbol{h}_i \in \mathbb{R}$
- Additive attention $\boldsymbol{e}_i = \boldsymbol{v}^T \tanh(\boldsymbol{W}_1 \boldsymbol{h}_i + \boldsymbol{W}_2 \boldsymbol{s}) \in \mathbb{R}$

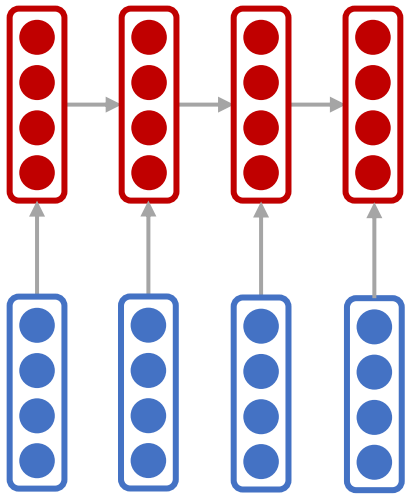
For more detail: https://ruder.io/deep-learning-nlp-best-practices/index.html#attention

Stacked sequence-to-sequence models with attention


Representieren wir
diesen
Satz
<END>
Encoder hidden vectors
Decoder hidden vectors
Source language word embeddings
Target language word embeddings
Let's represent this sentence
<START>

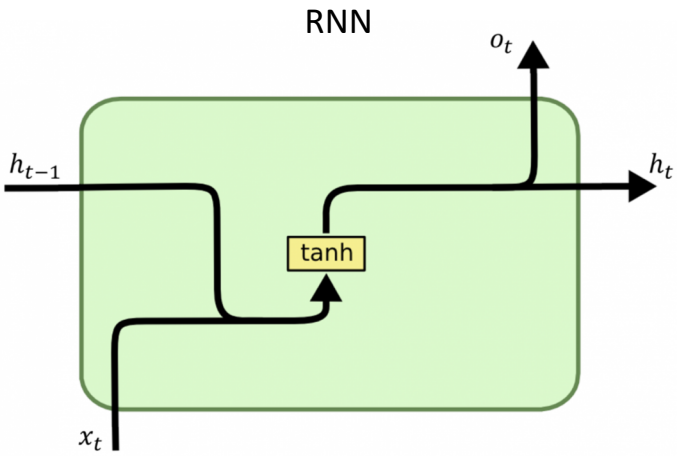


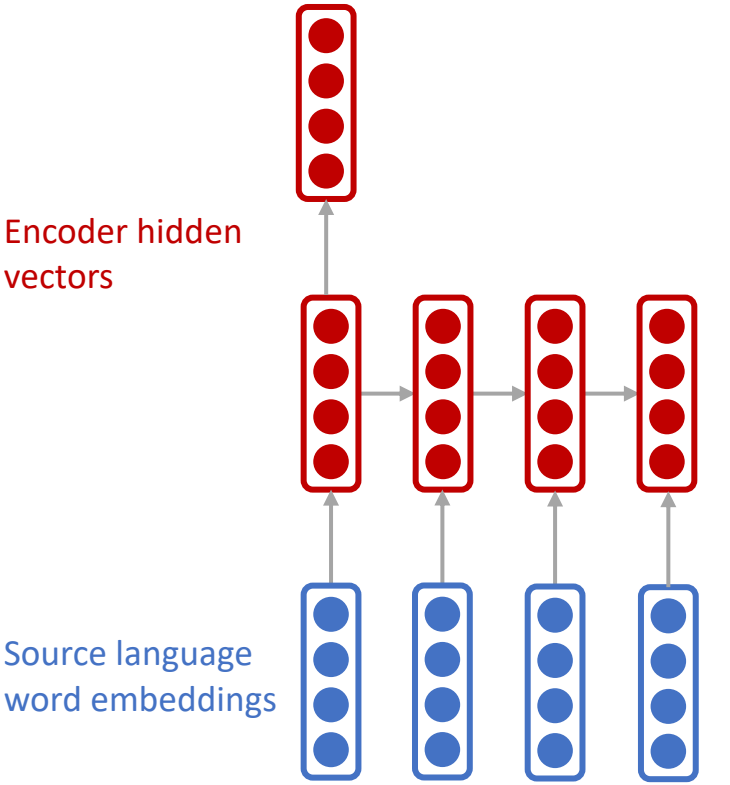
Encoder hidden vectors



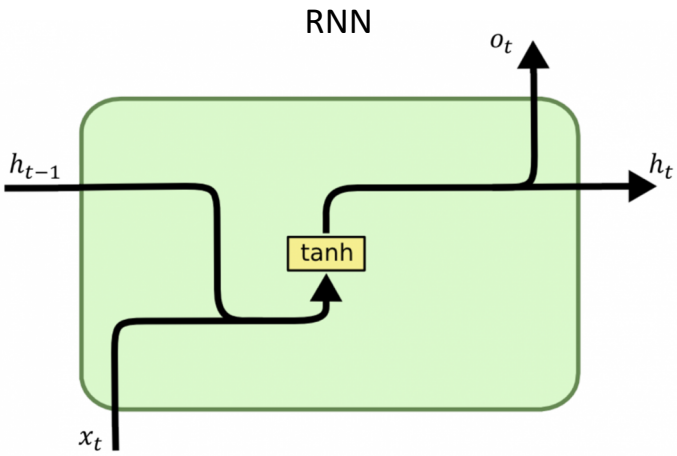
Source language word embeddings

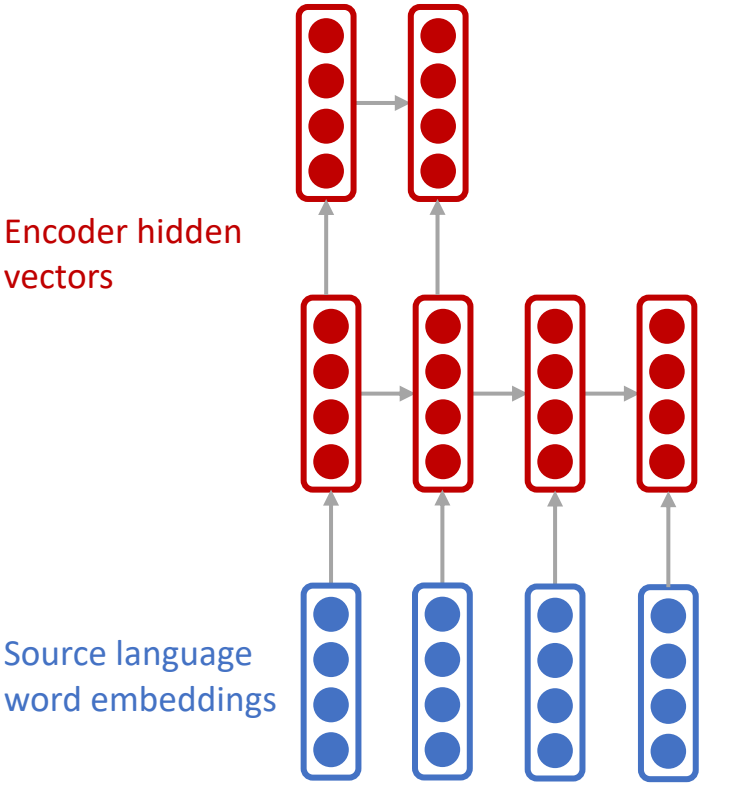
Let’s represent this sentence



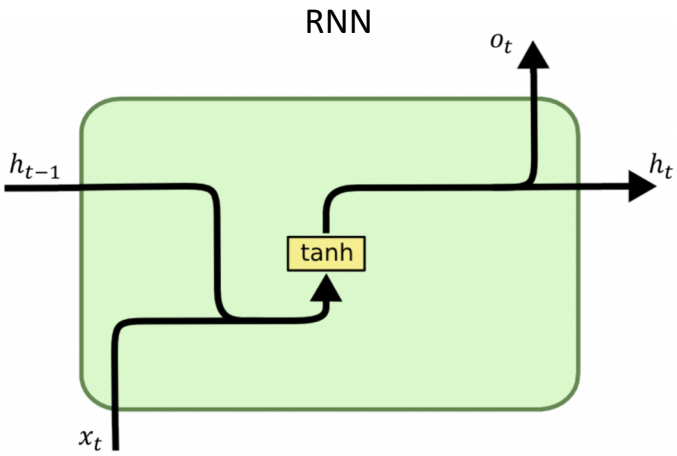


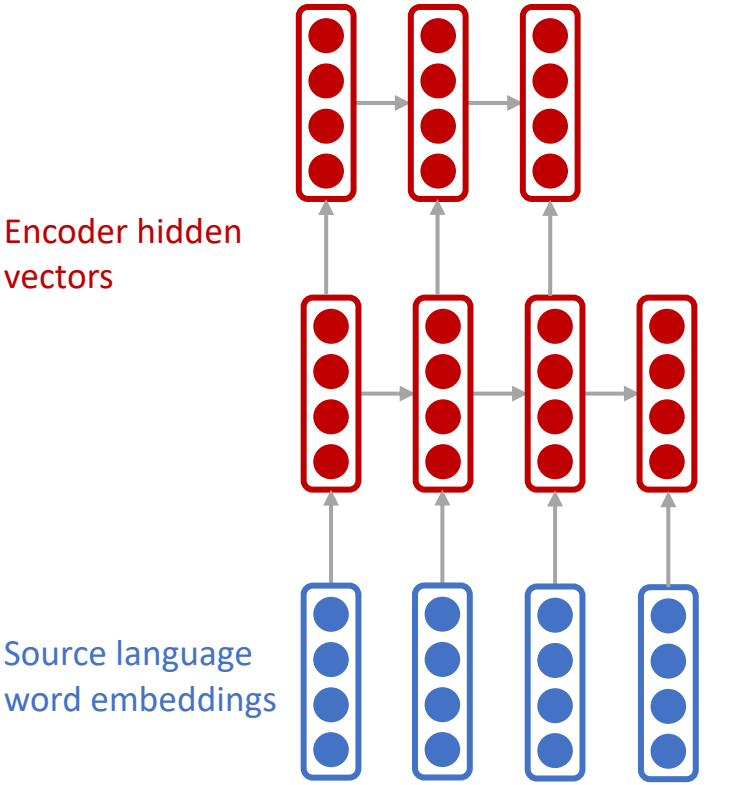
Let's represent this sentence



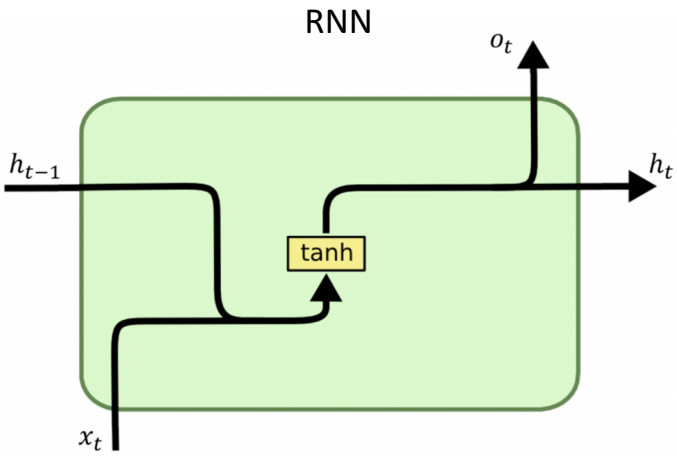


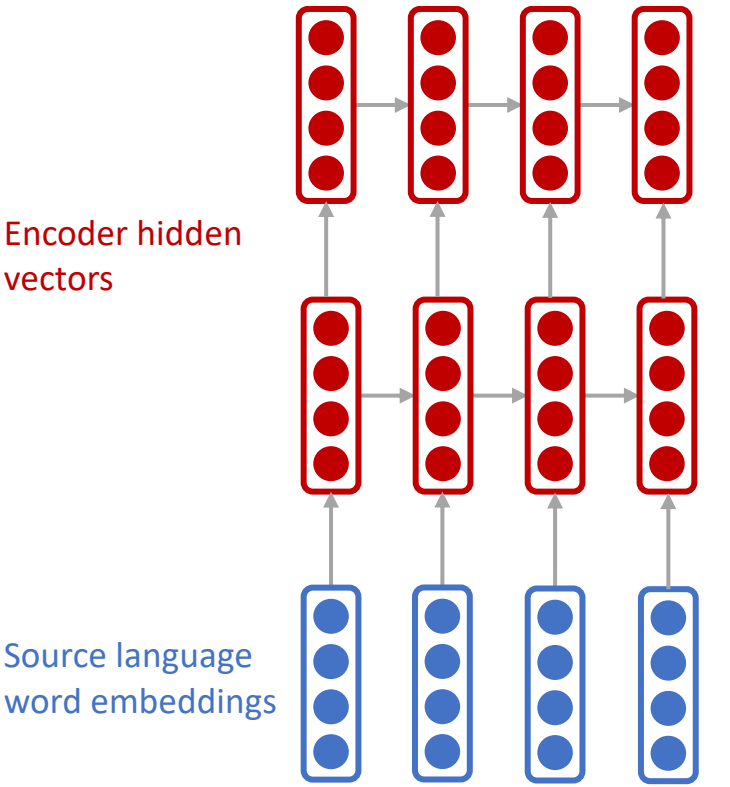
Let's represent this sentence



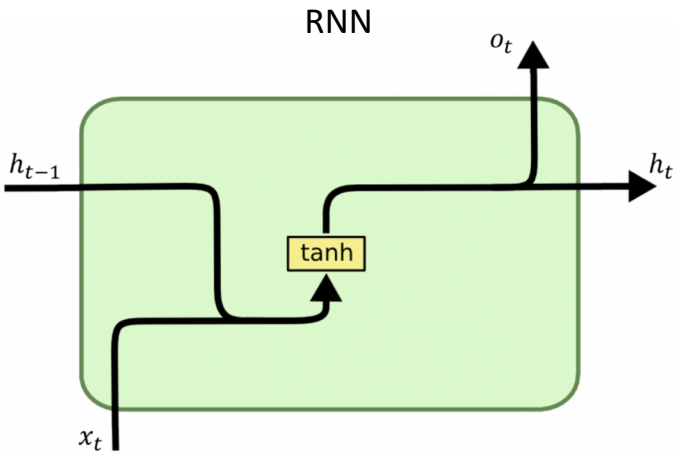


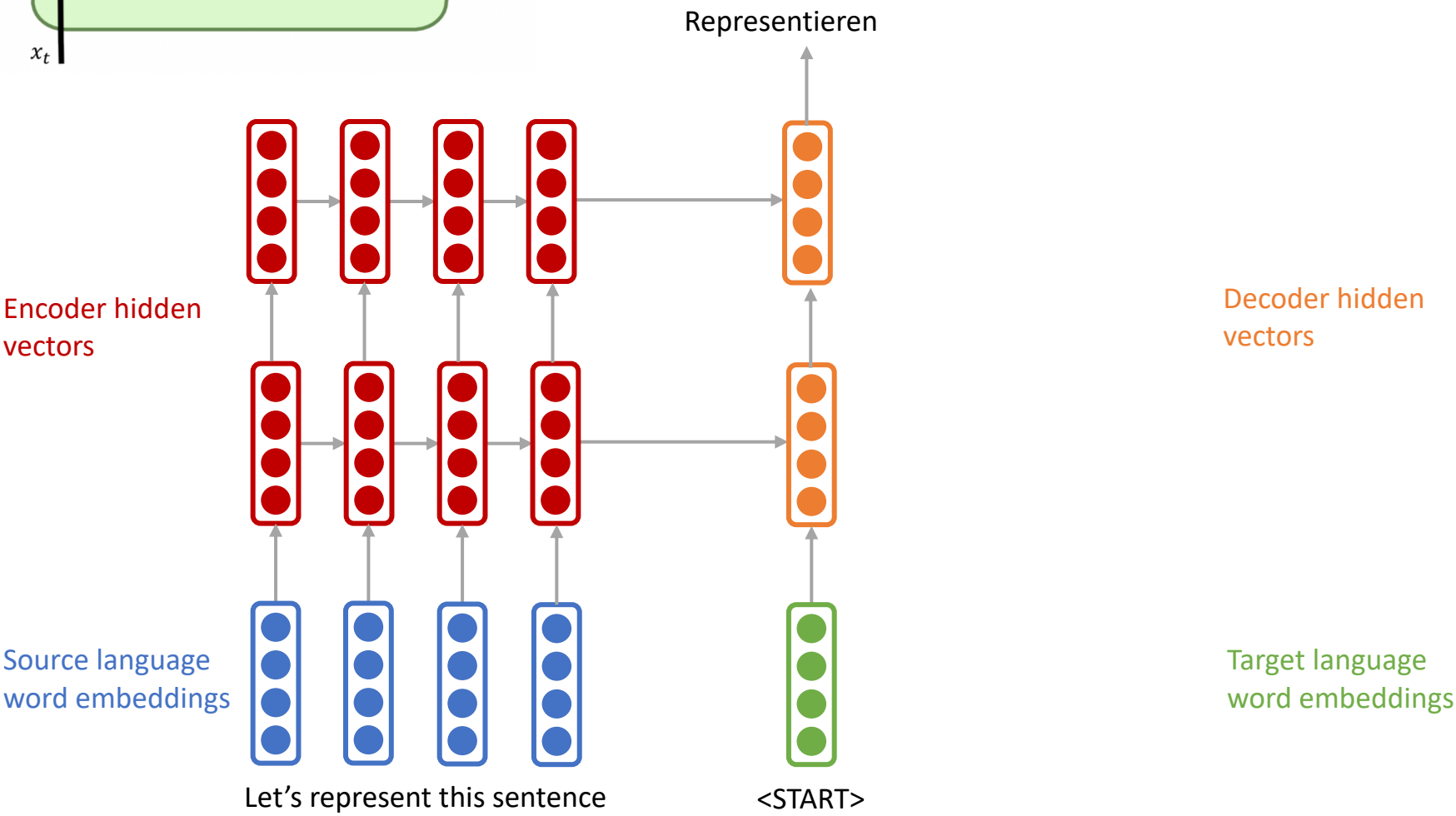
Let's represent this sentence

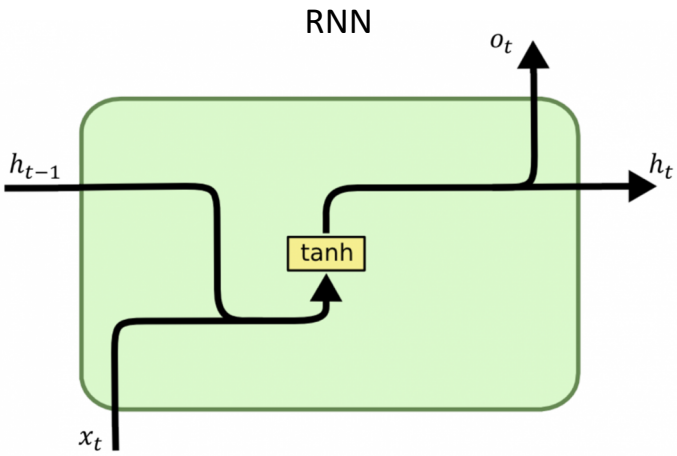




Let's represent this sentence


RNN
o_t
h_{t-1}
h_t
tanh
x_t


Representieren
Encoder hidden vectors
Decoder hidden vectors
Source language word embeddings
Target language word embeddings
Let’s represent this sentence
<START>



Representieren wir

Encoder hidden vectors

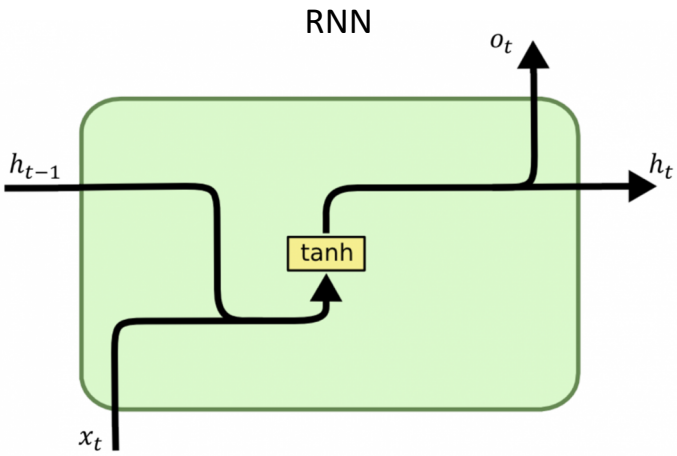
Decoder hidden vectors

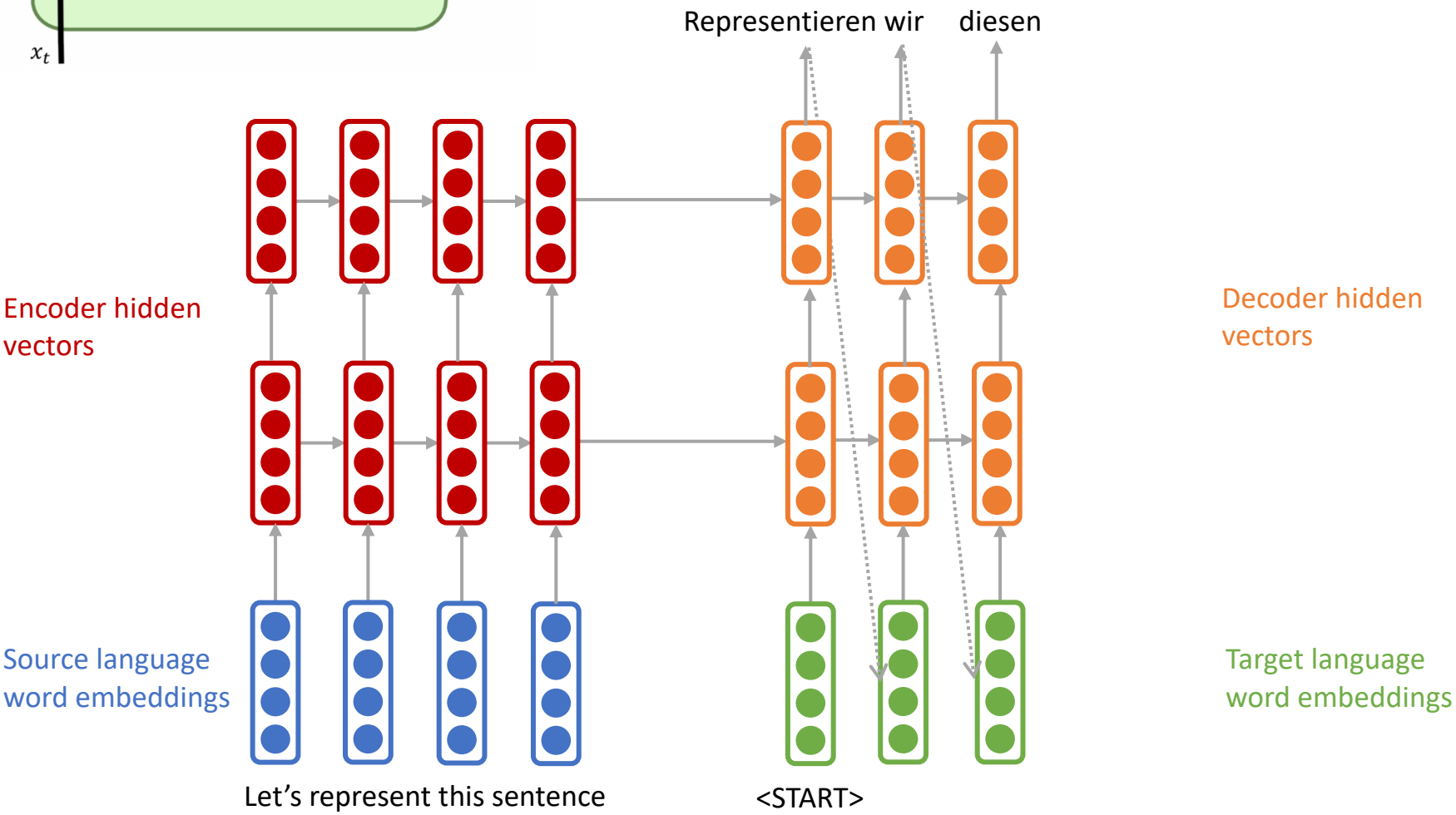
Source language word embeddings

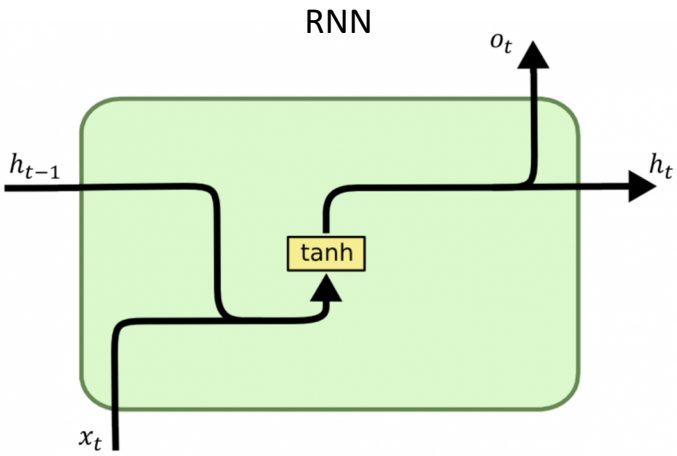
Target language word embeddings

Let’s represent this sentence

<START>


RNN
o_t
h_{t-1}
h_t
tanh
x_t


Representieren wir diesen
Encoder hidden vectors
Decoder hidden vectors
Source language word embeddings
Target language word embeddings
Let's represent this sentence
<START>



Representieren wir diesen Satz

Encoder hidden vectors

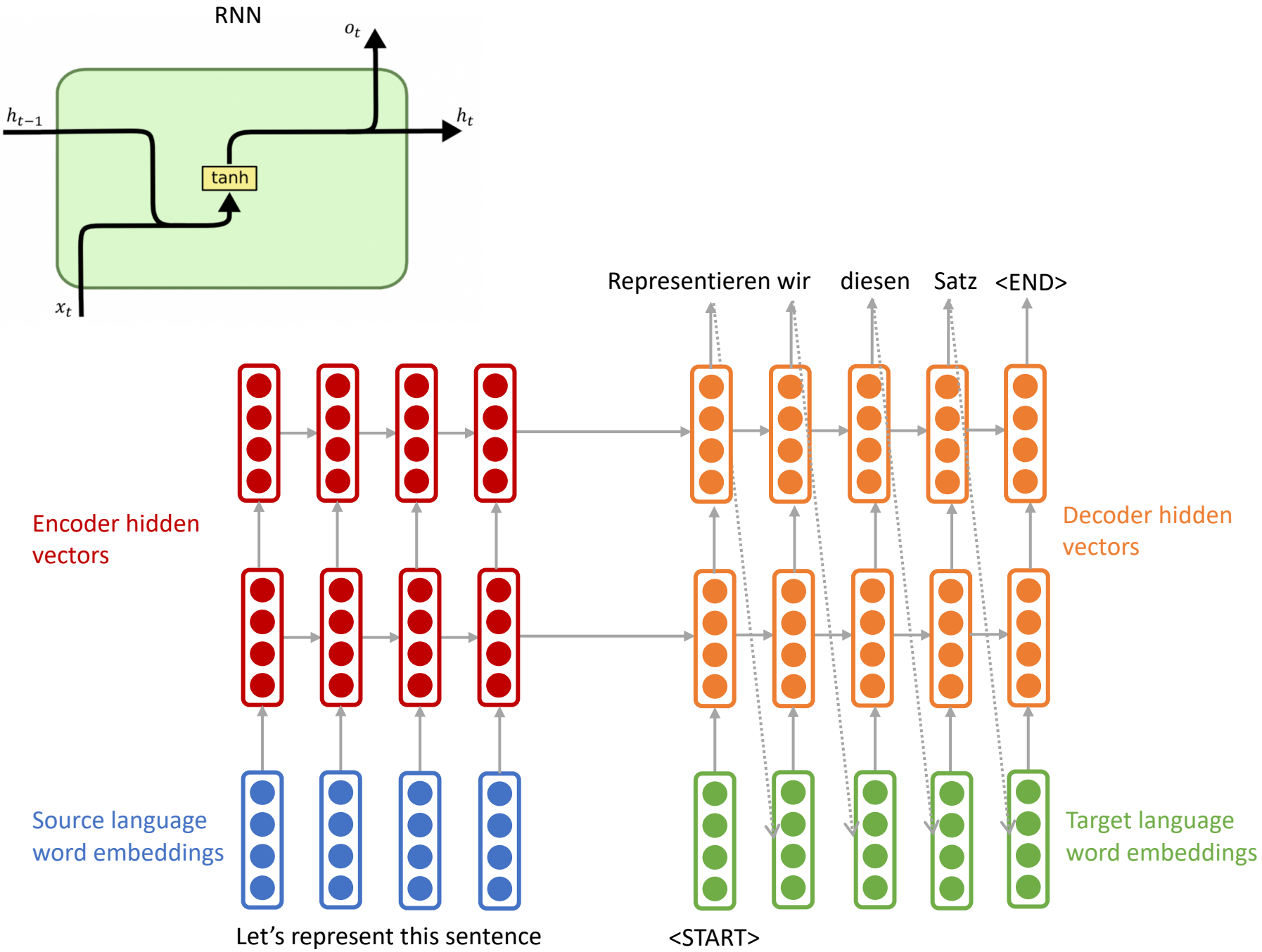
Decoder hidden vectors

Source language word embeddings

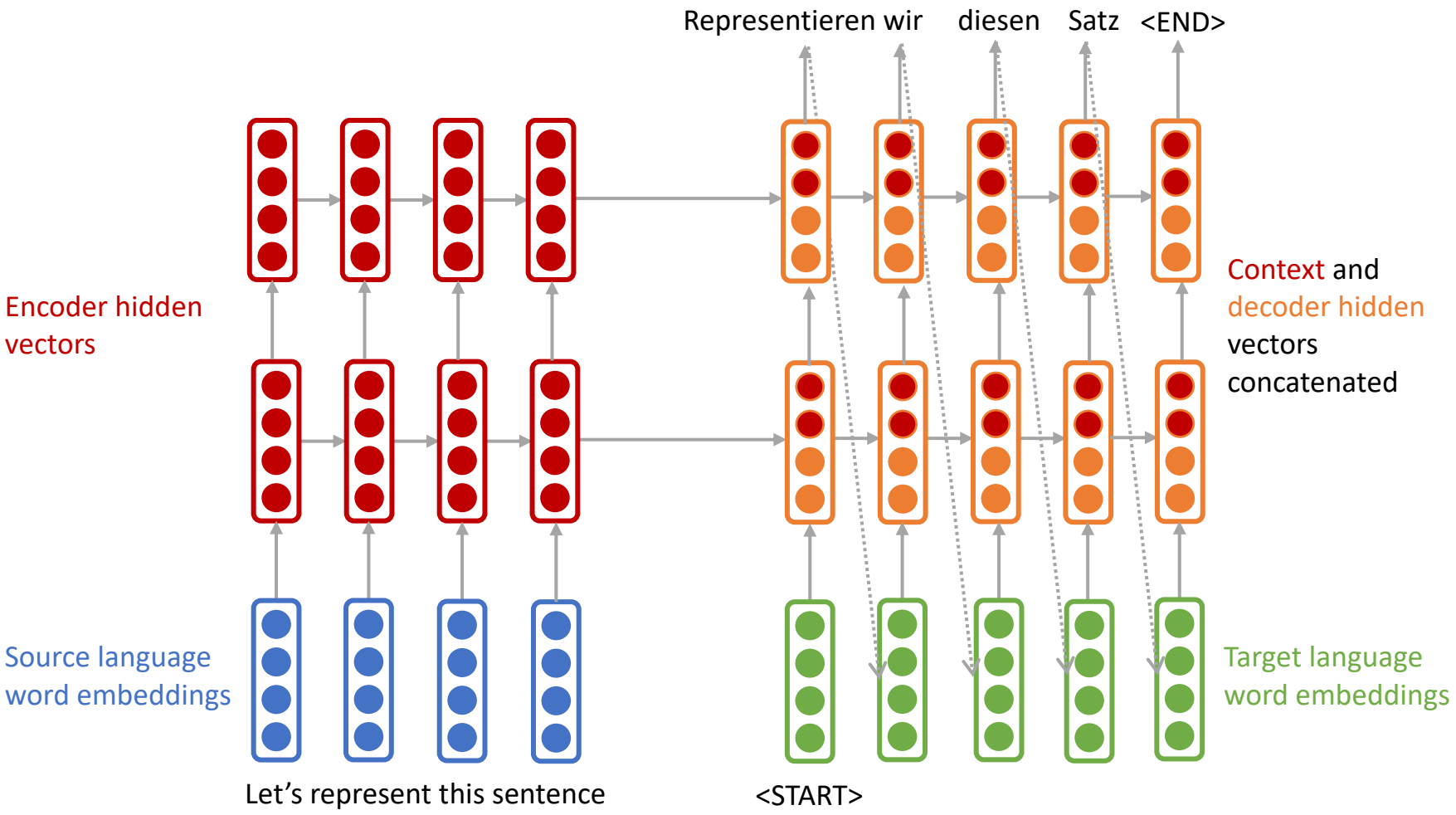
Target language word embeddings

Let's represent this sentence

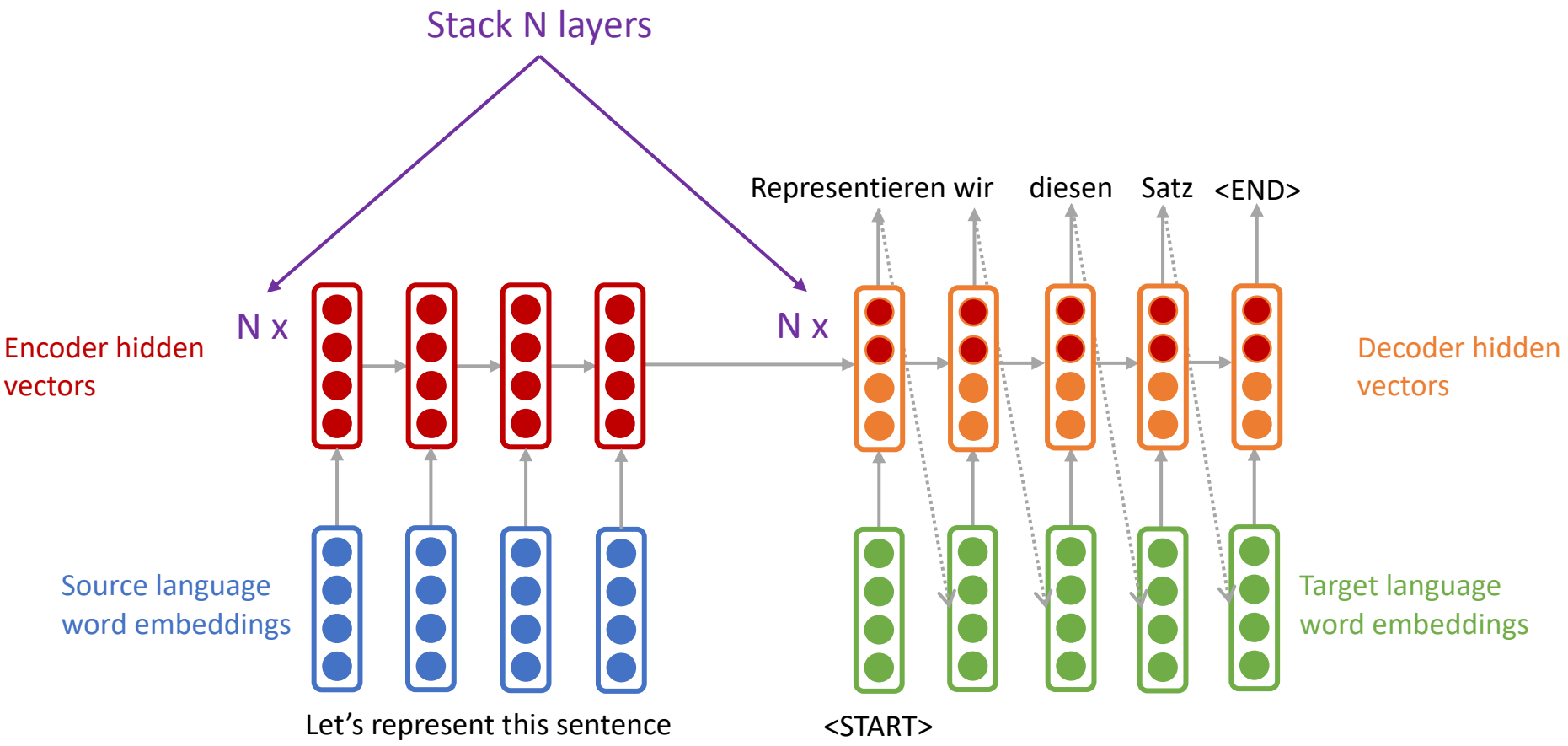
<START>


RNN
o_t
h_{t-1}
h_t
tanh
x_t
Representieren wir diesen Satz <END>
Encoder hidden vectors
Decoder hidden vectors
Source language word embeddings
Target language word embeddings
Let's represent this sentence
<START>

Stacked sequence-to-sequence with attention



Stacked sequence-to-sequence with attention

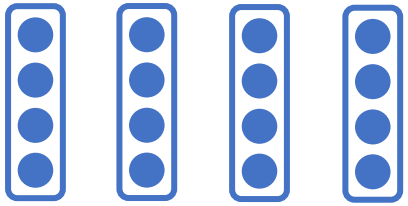


Transformers

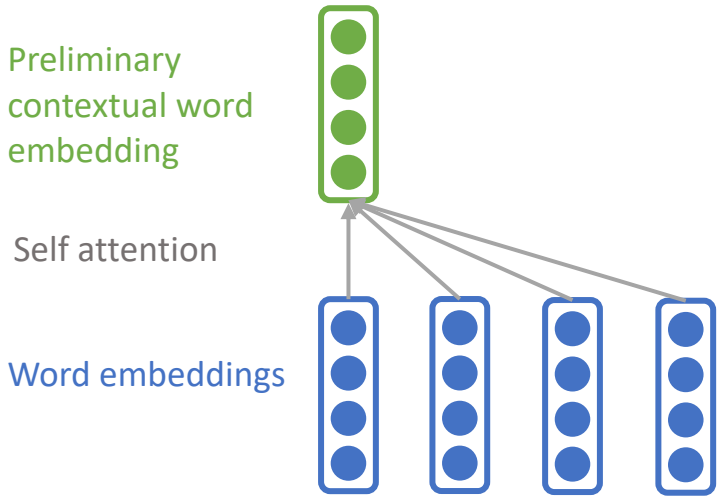
Transformers: the motivation

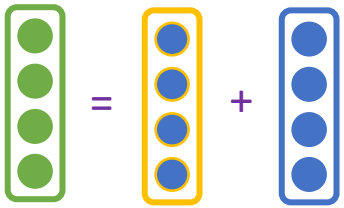
- The sequential nature of RNNs is not cheap.
- We want to be able to parallelize.

Word embeddings

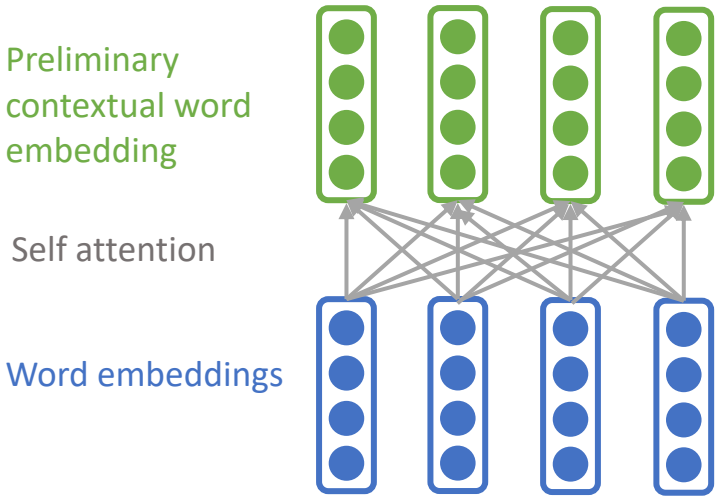


Let’s represent this sentence

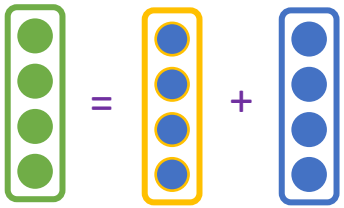




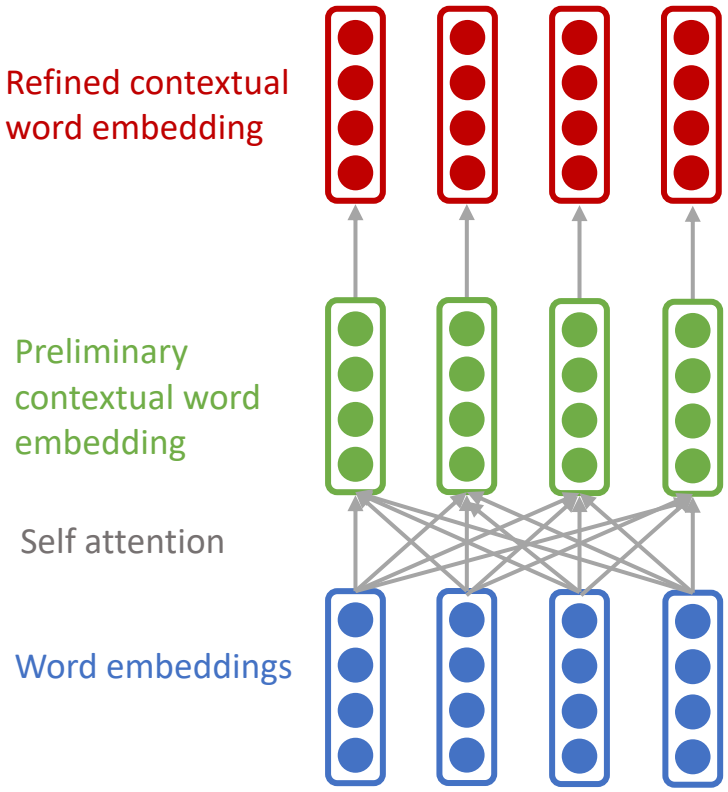
The preliminary contextual word embedding is the sum of the initial word embedding and the contextual word embedding from self attention



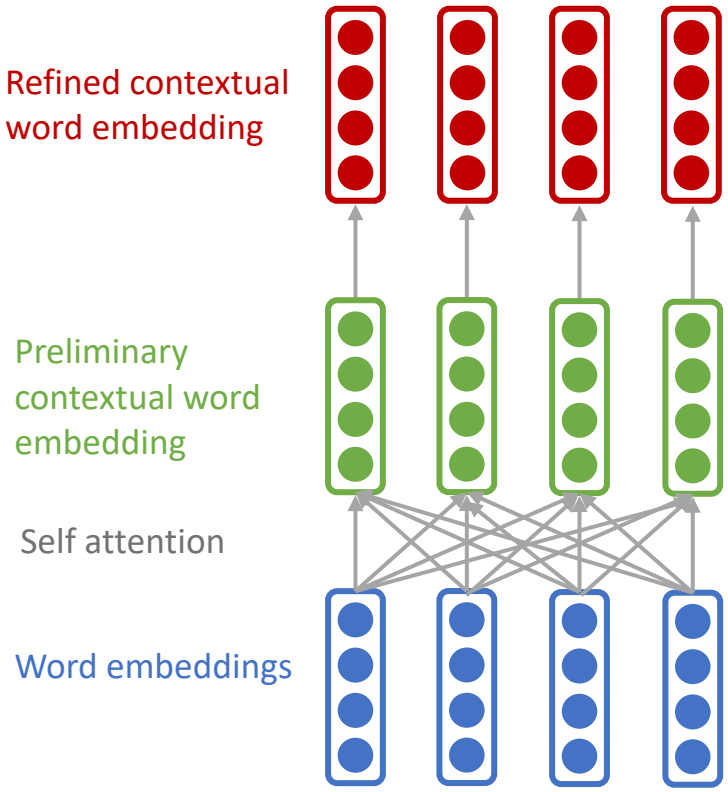
Let's represent this sentence



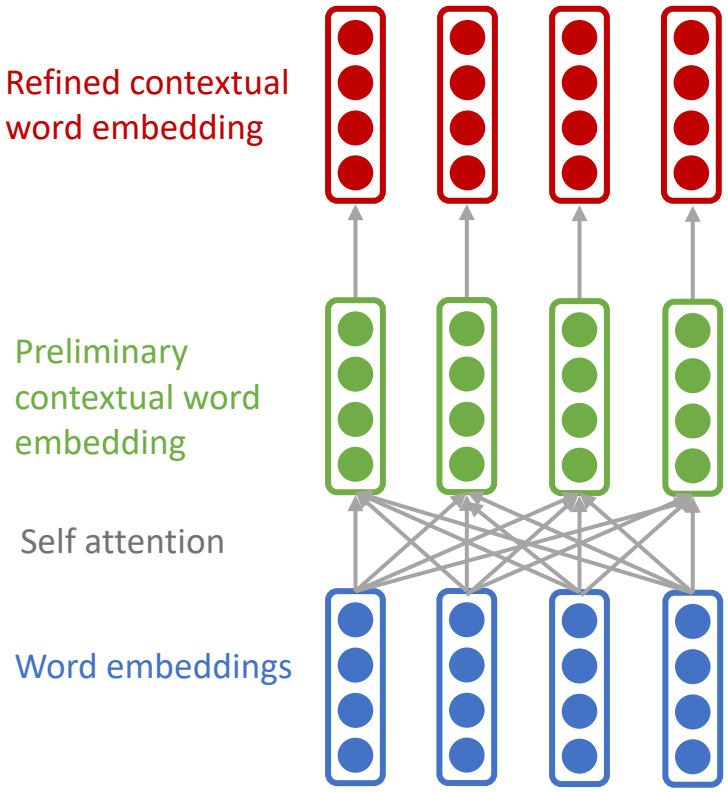
The preliminary contextual word embedding is the sum of the initial word embedding and the contextual word embedding from self attention



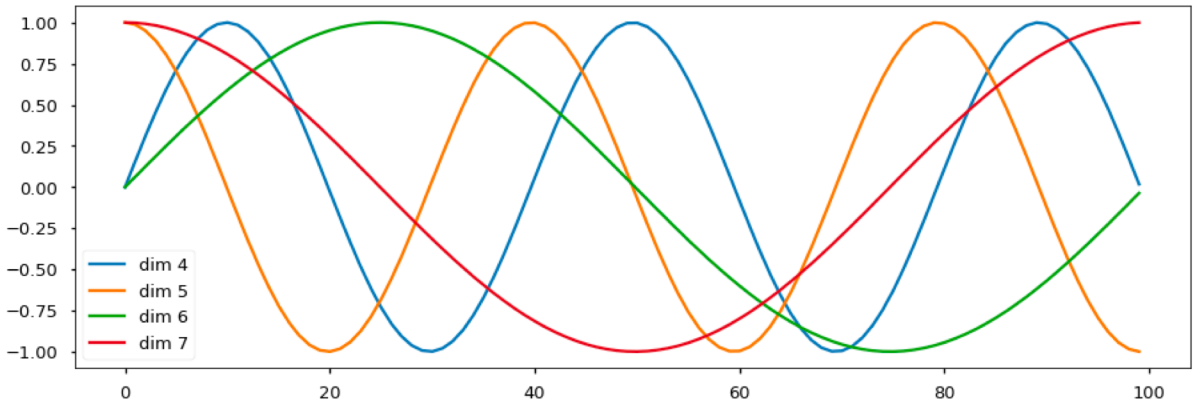
The preliminary contextual word embeddings are passed through feedforward layers to generate the refined contextual word embeddings.

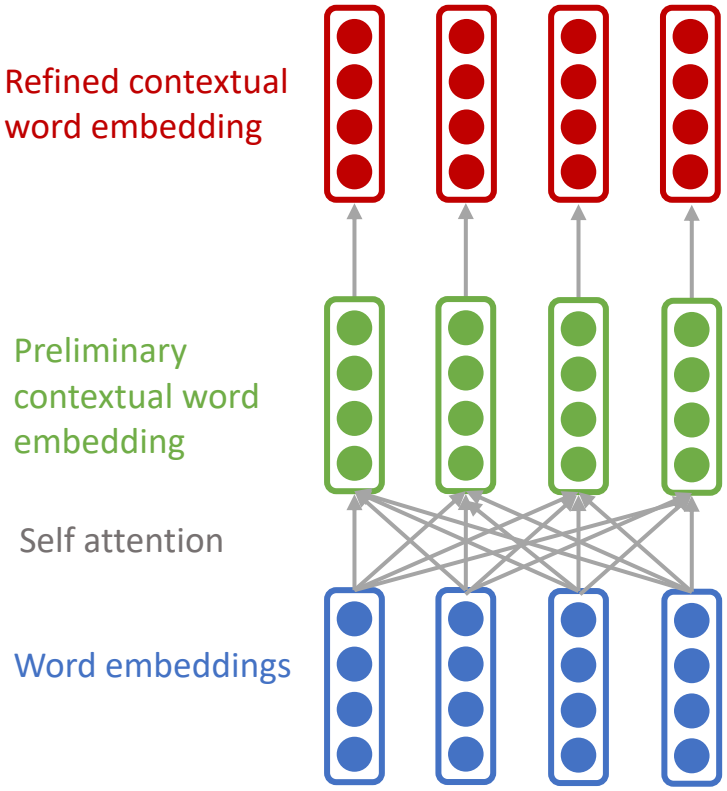

Refined contextual word embedding
Preliminary contextual word embedding
Self attention
Word embeddings
Let’s represent this sentence

Problem?



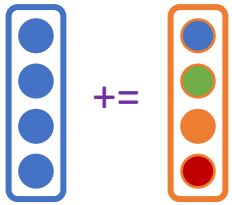
Problem: sequential structure is lost!





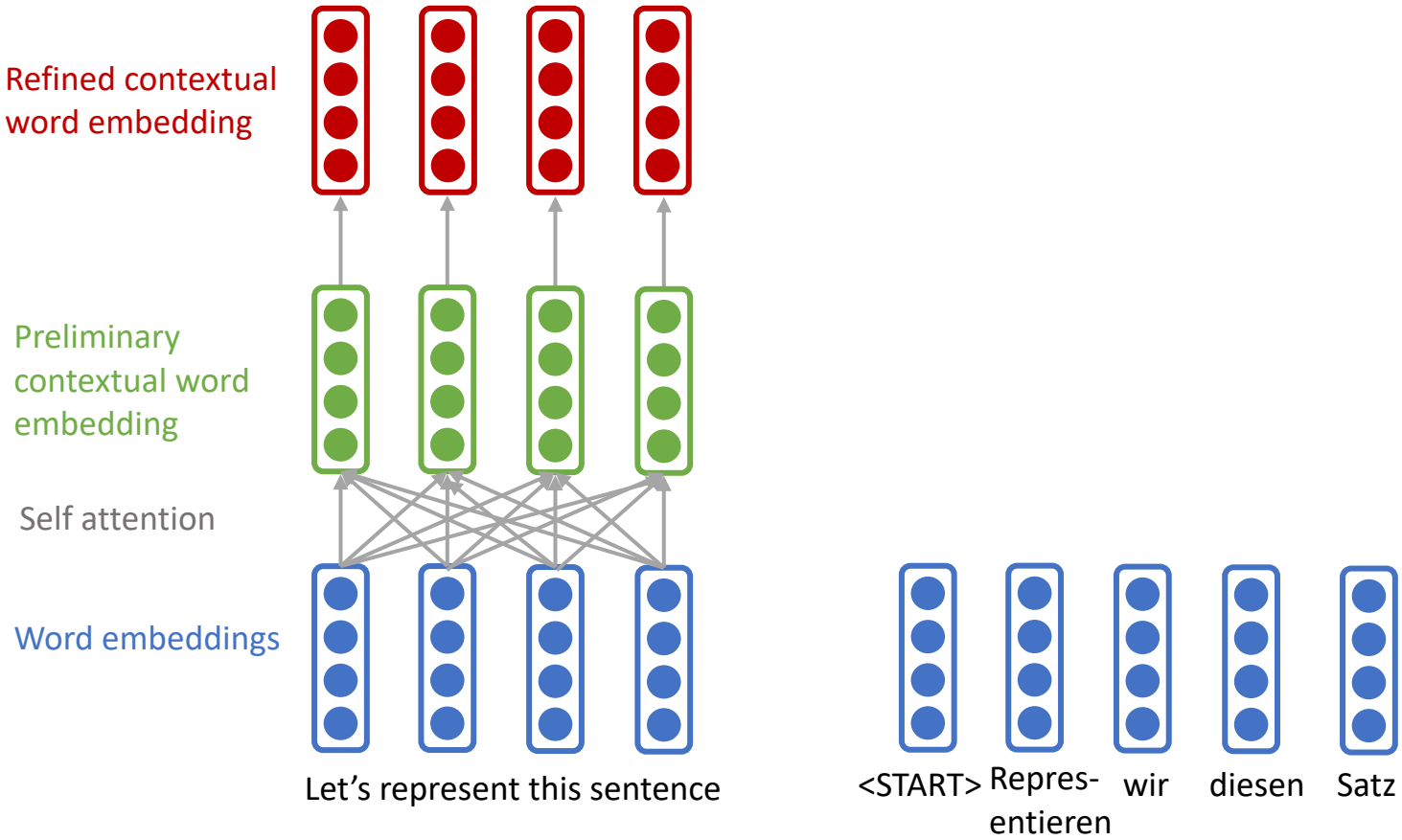
Let's represent this sentence

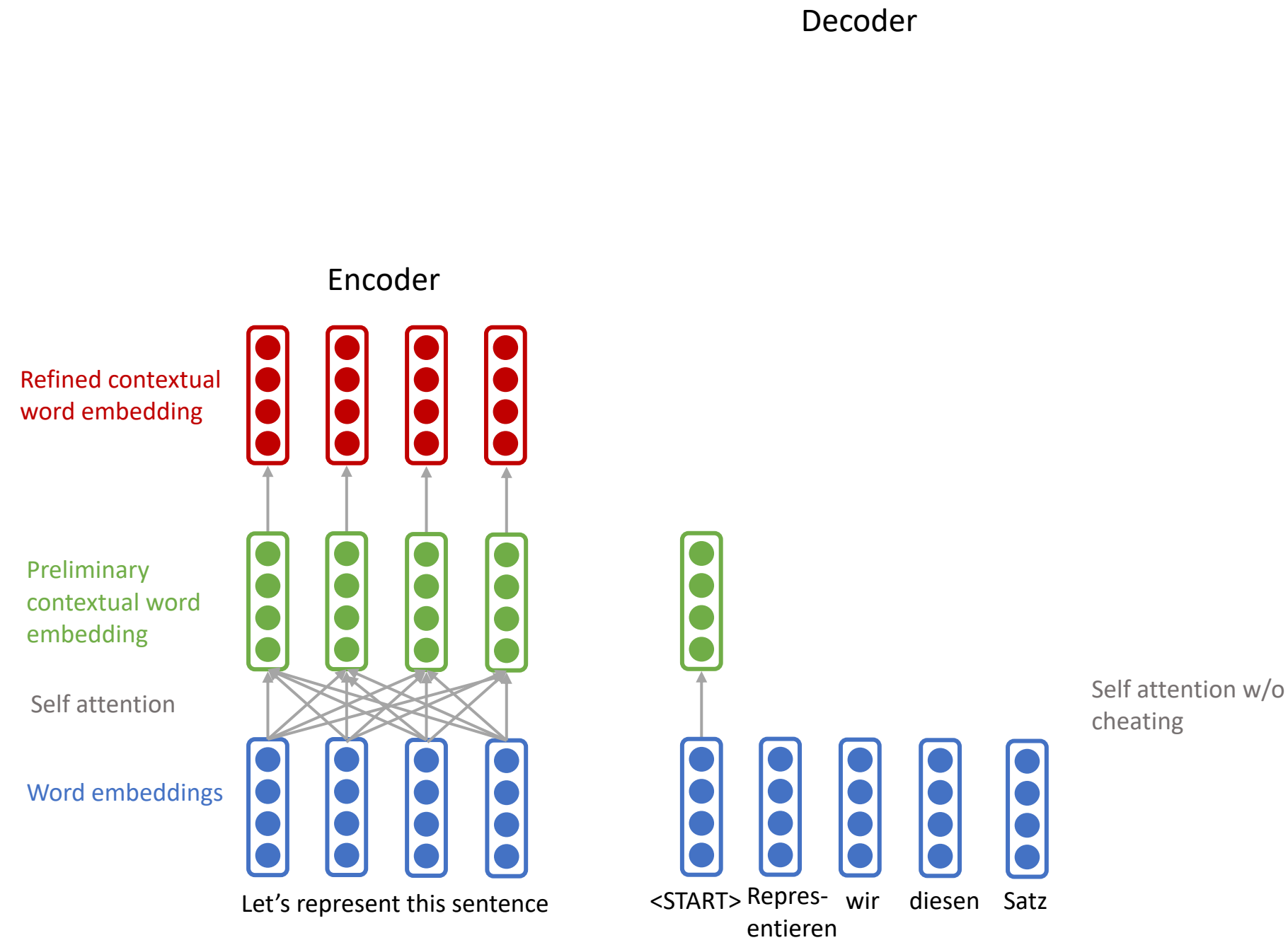
Problem: sequential structure is lost!
Solution: add positional encoding into the initial word embeddings.

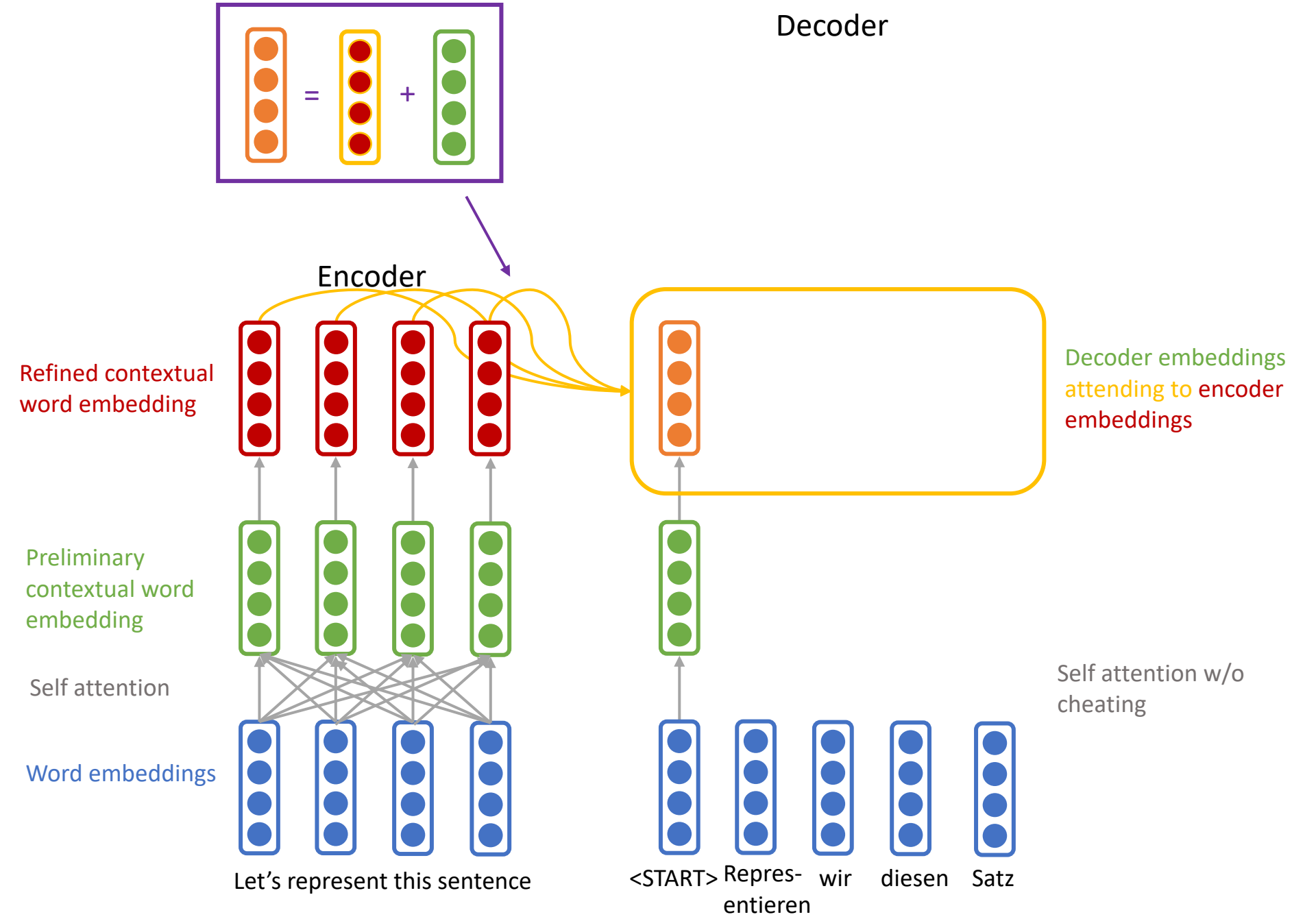


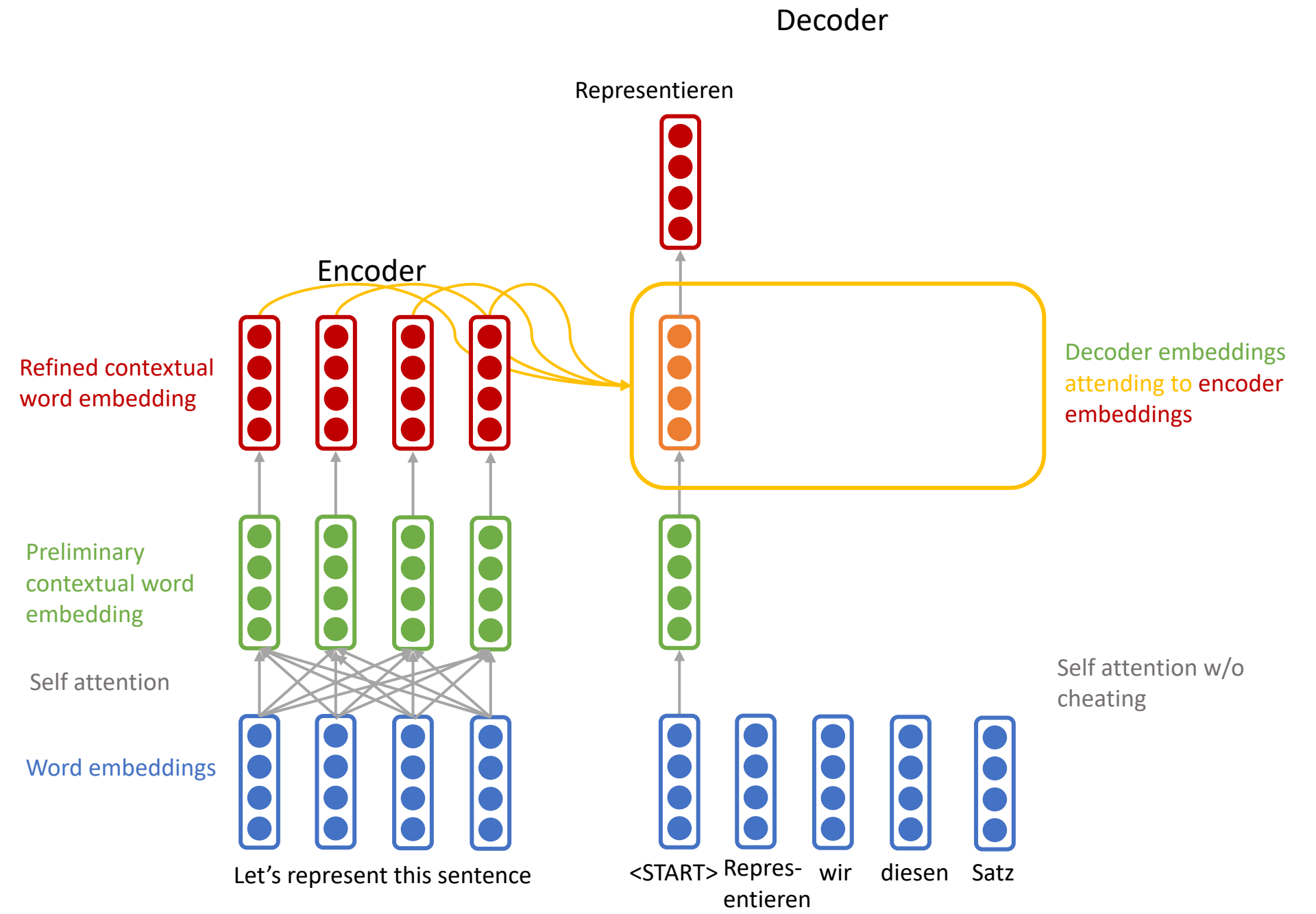
Decoder

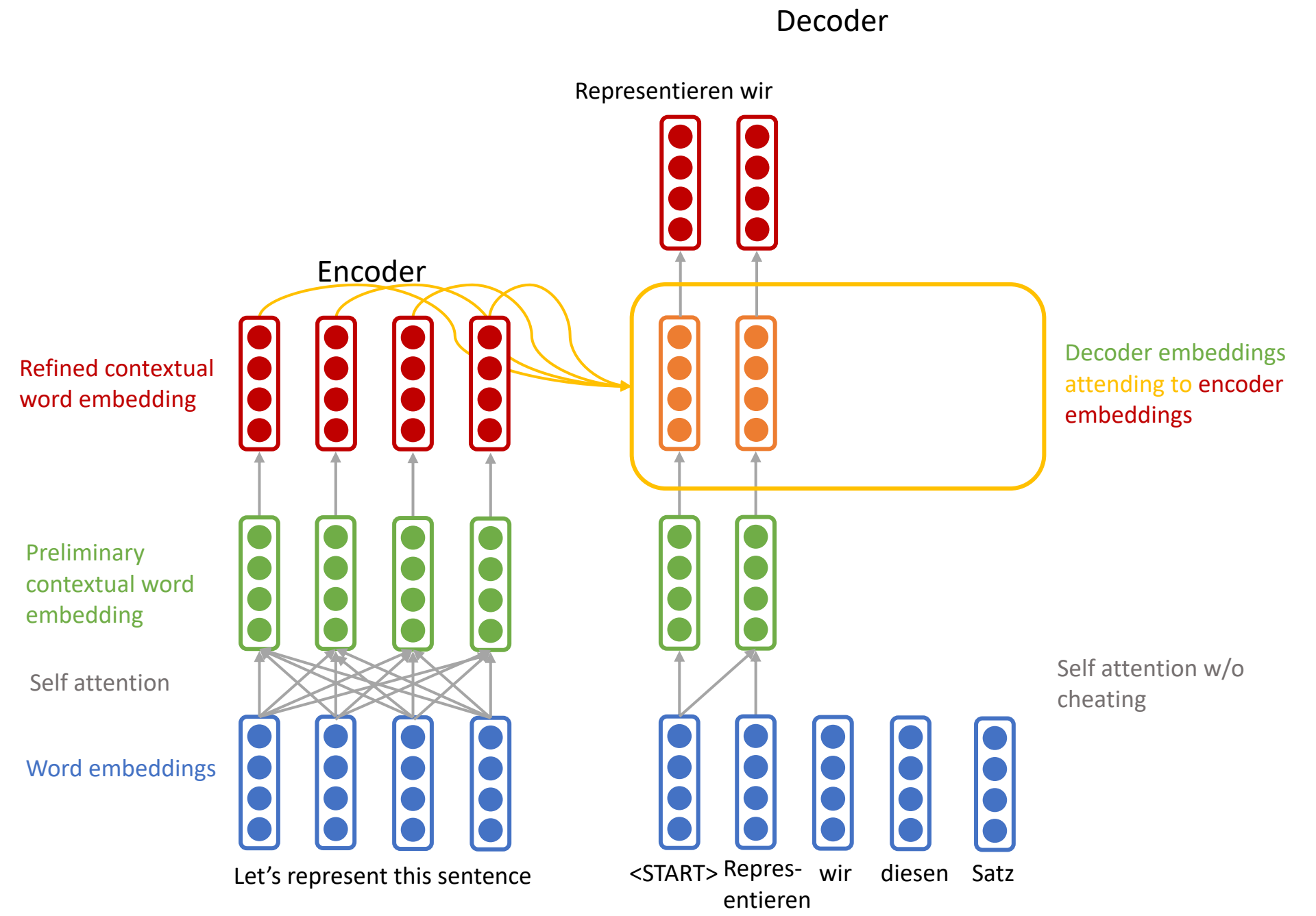
Encoder

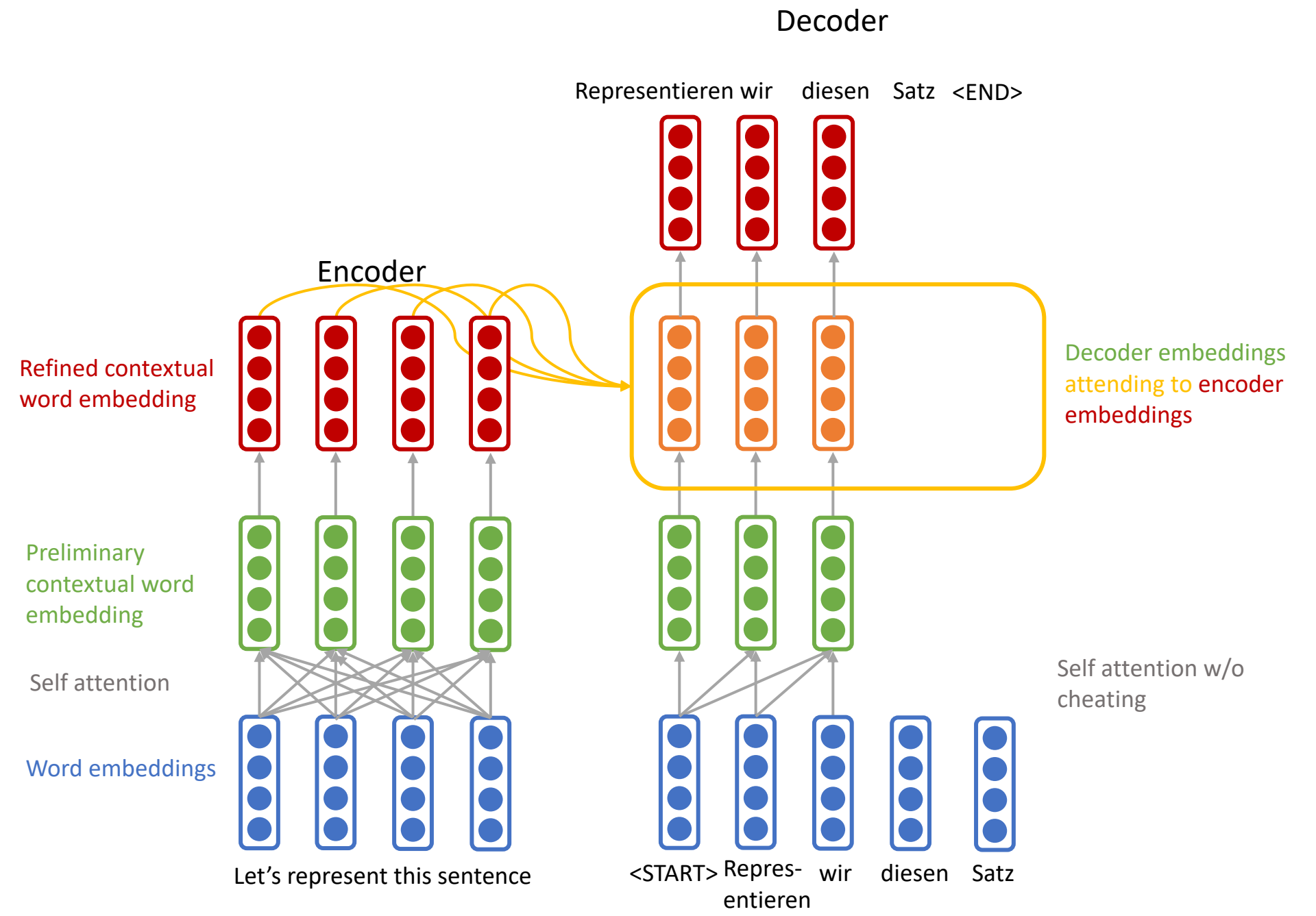


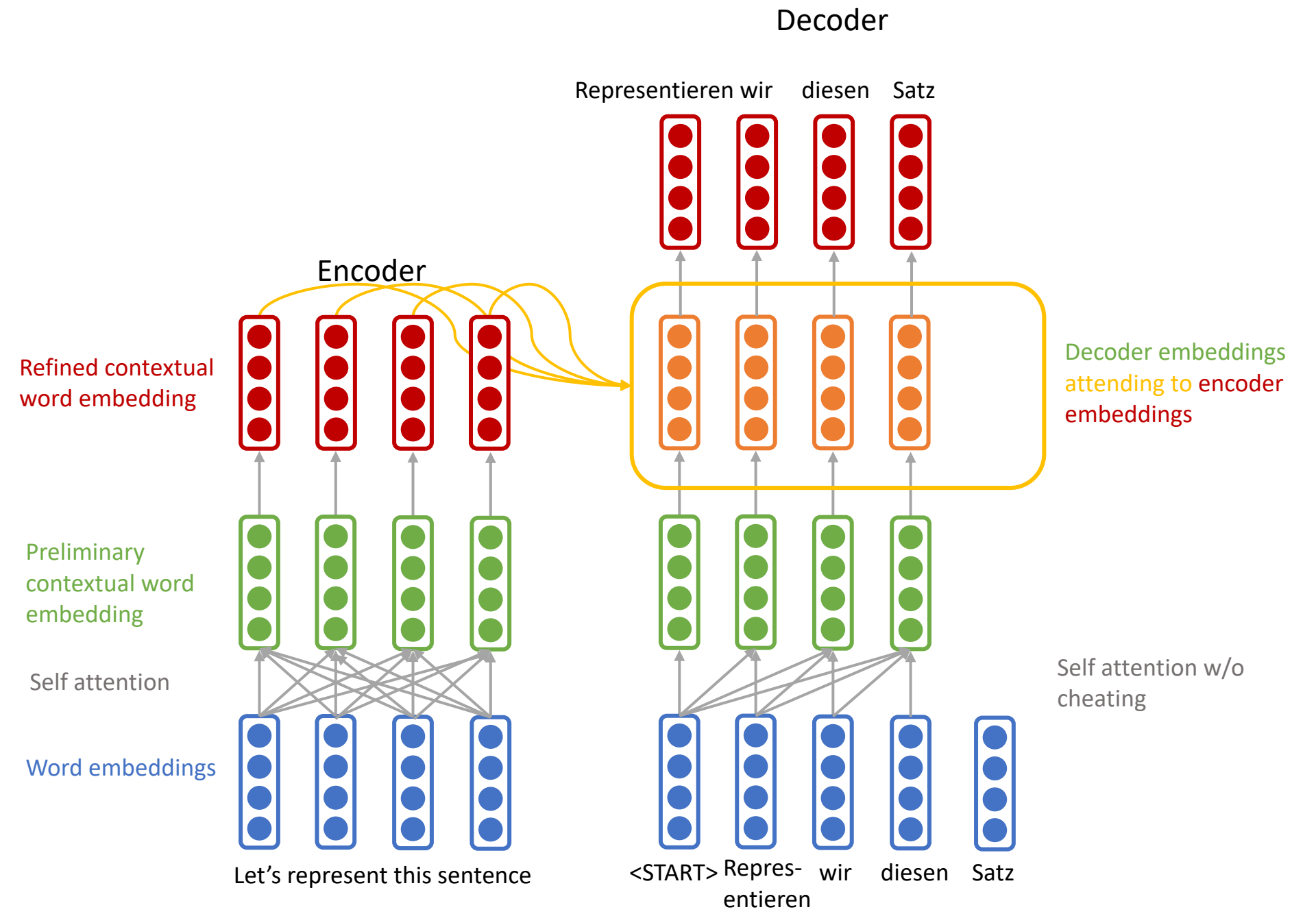

Decoder
Encoder
Refined contextual word embedding
Preliminary contextual word embedding
Self attention
Word embeddings
Let's represent this sentence
<START>
Repres-entieren
wir
diesen
Satz
Self attention w/o cheating

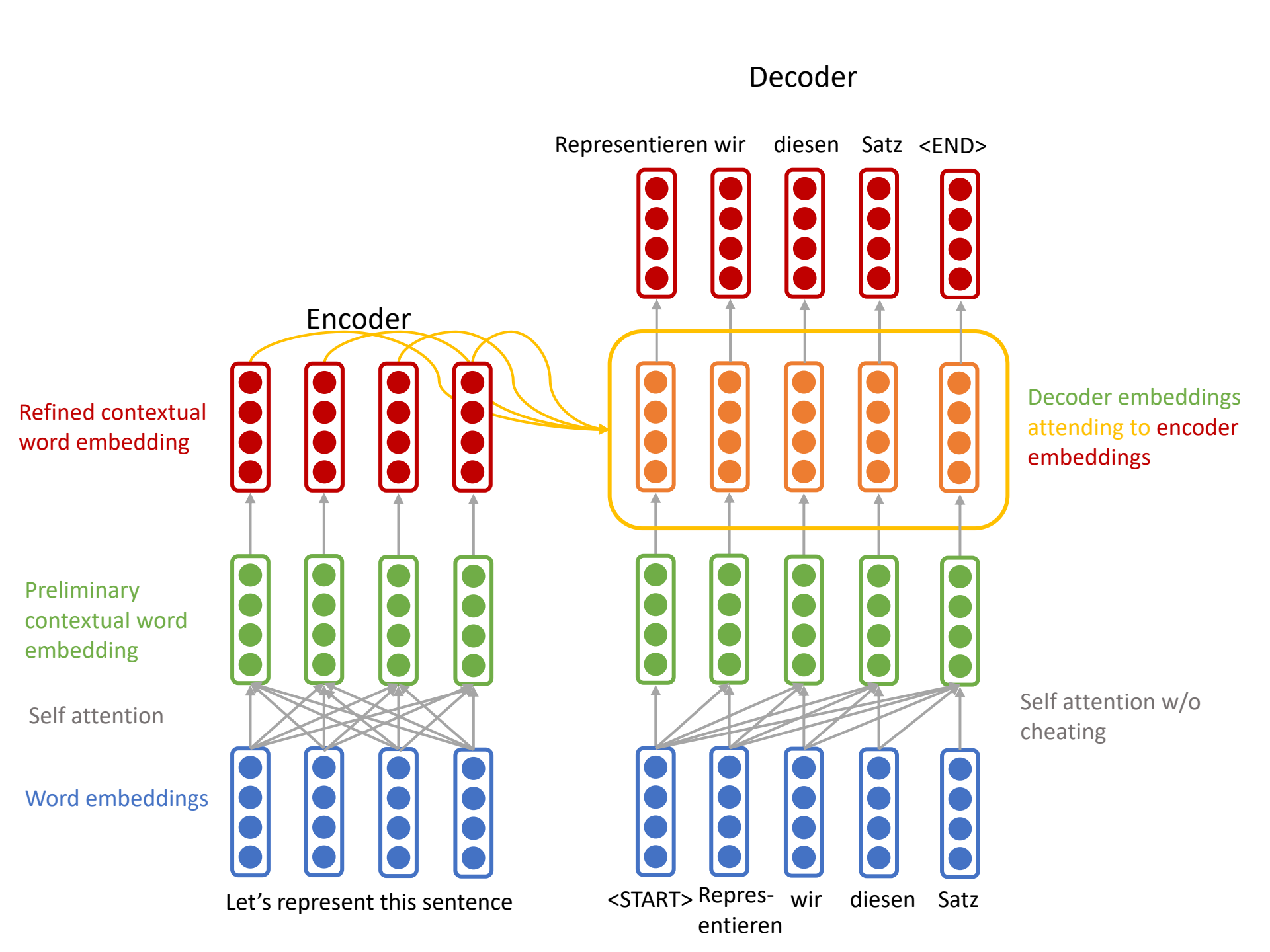

=
+
Decoder
Encoder
Refined contextual word embedding
Decoder embeddings attending to encoder embeddings
Preliminary contextual word embedding
Self attention
Self attention w/o cheating
Word embeddings
Let's represent this sentence
<START>
Repres-entieren
wir
diesen
Satz

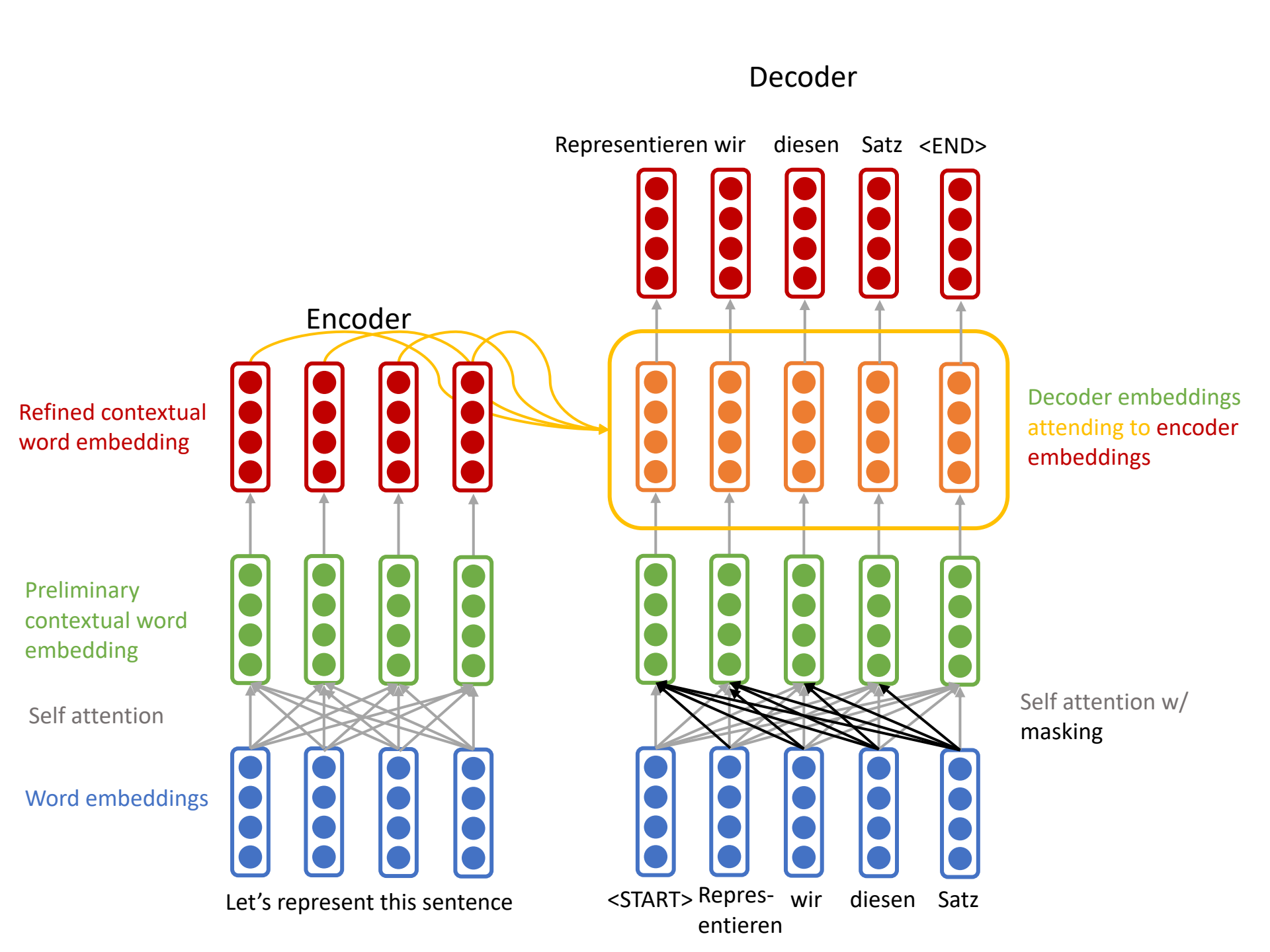

Decoder
Representieren
Encoder
Refined contextual word embedding
Decoder embeddings attending to encoder embeddings
Preliminary contextual word embedding
Self attention
Self attention w/o cheating
Word embeddings
Let's represent this sentence
<START>
Repres-entieren
wir
diesen
Satz

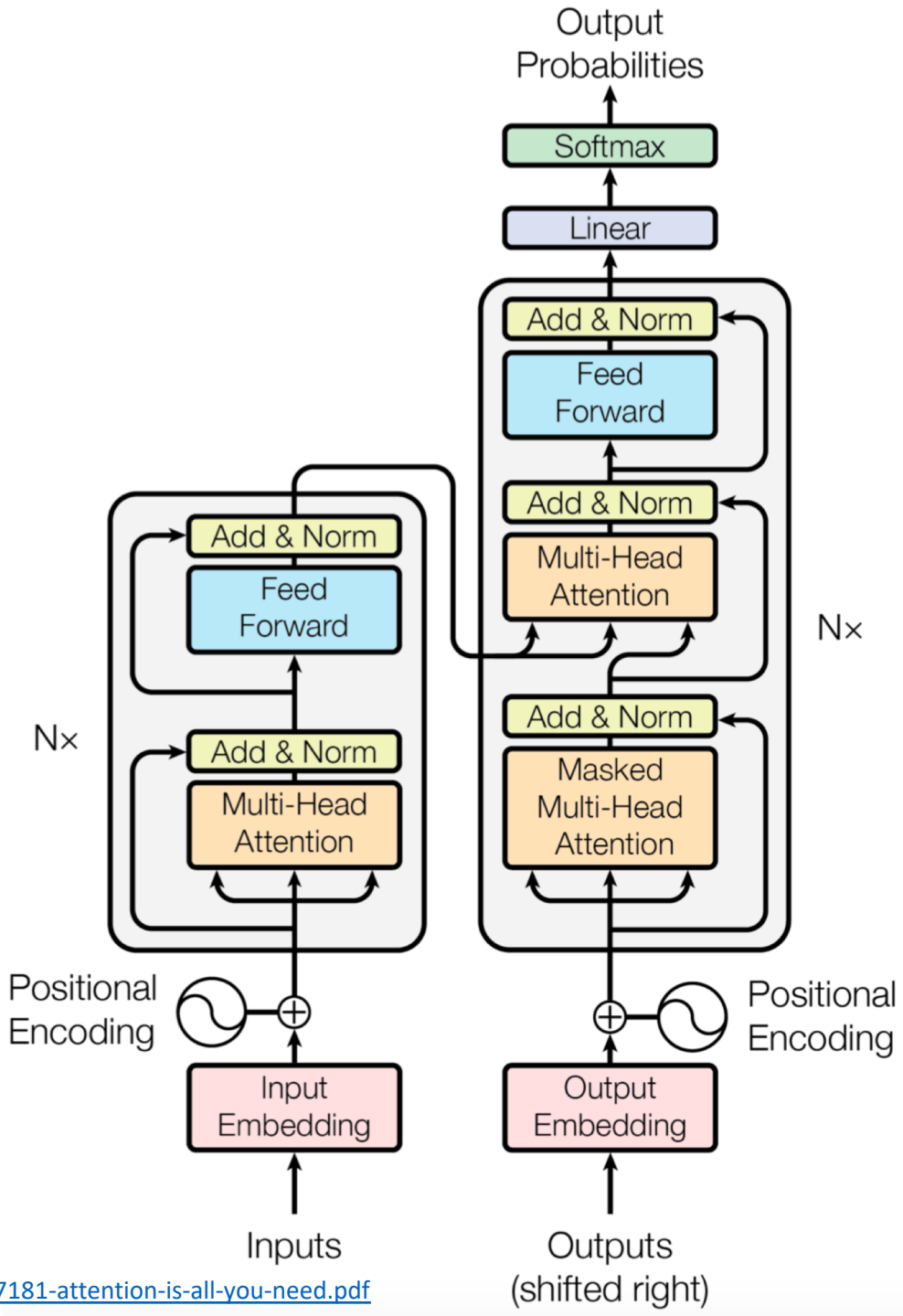

Decoder
Representieren wir
Encoder
Refined contextual word embedding
Decoder embeddings attending to encoder embeddings
Preliminary contextual word embedding
Self attention
Self attention w/o cheating
Word embeddings
Let's represent this sentence
<START>
Repres-entieren
wir
diesen
Satz


Decoder
Representieren wir diesen Satz <END>
Encoder
Refined contextual word embedding
Decoder embeddings attending to encoder embeddings
Preliminary contextual word embedding
Self attention
Self attention w/o cheating
Word embeddings
Let's represent this sentence
<START> Repres- entieren wir diesen Satz

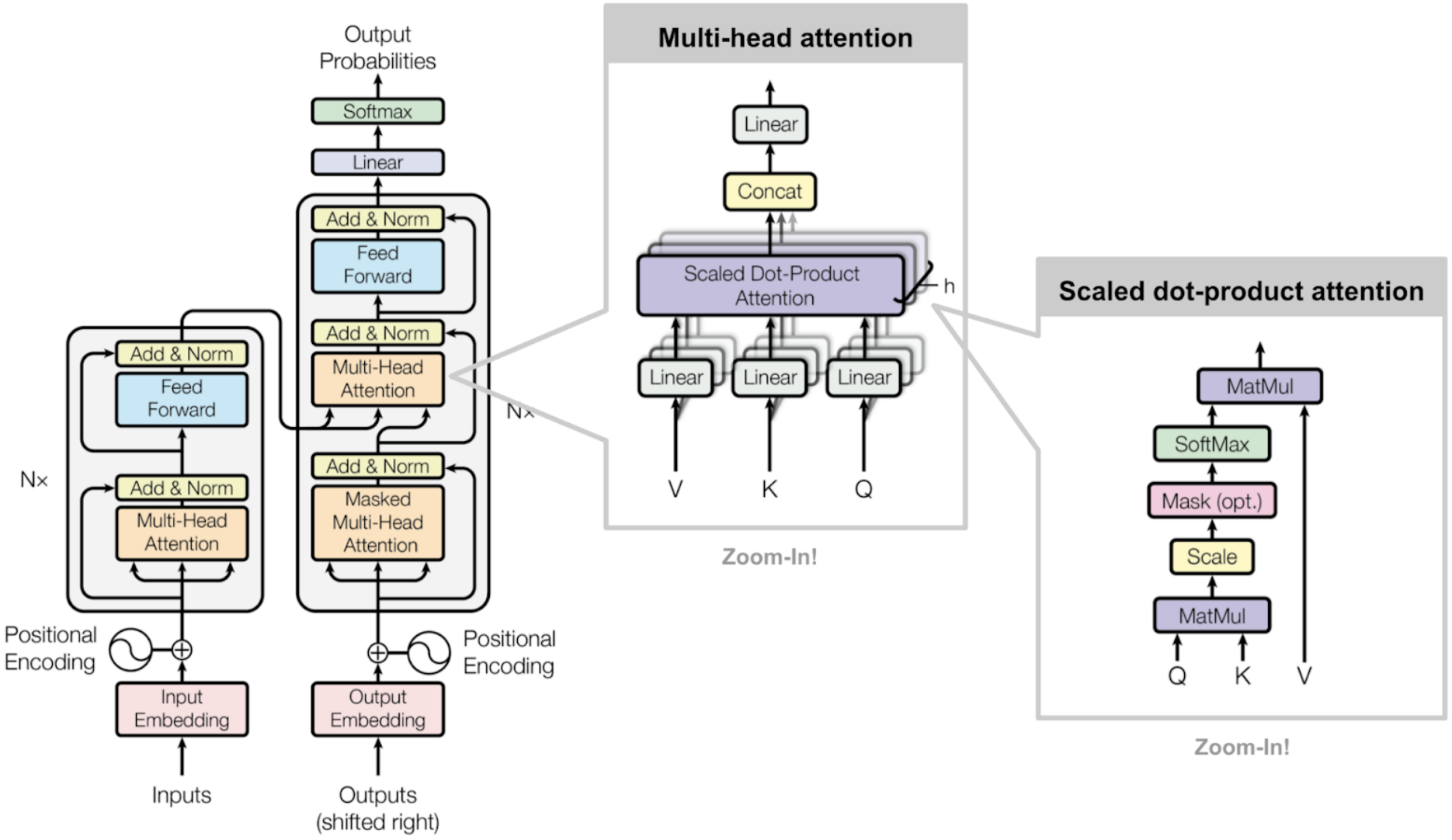

Decoder
Representieren wir diesen Satz
Encoder
Refined contextual word embedding
Decoder embeddings attending to encoder embeddings
Preliminary contextual word embedding
Self attention
Self attention w/o cheating
Word embeddings
Let's represent this sentence
<START> Repres- entieren wir diesen Satz


Decoder
Representieren wir diesen Satz <END>
Encoder
Refined contextual word embedding
Decoder embeddings attending to encoder embeddings
Preliminary contextual word embedding
Self attention
Self attention w/o cheating
Word embeddings
Let's represent this sentence
<START> Repres- entieren wir diesen Satz

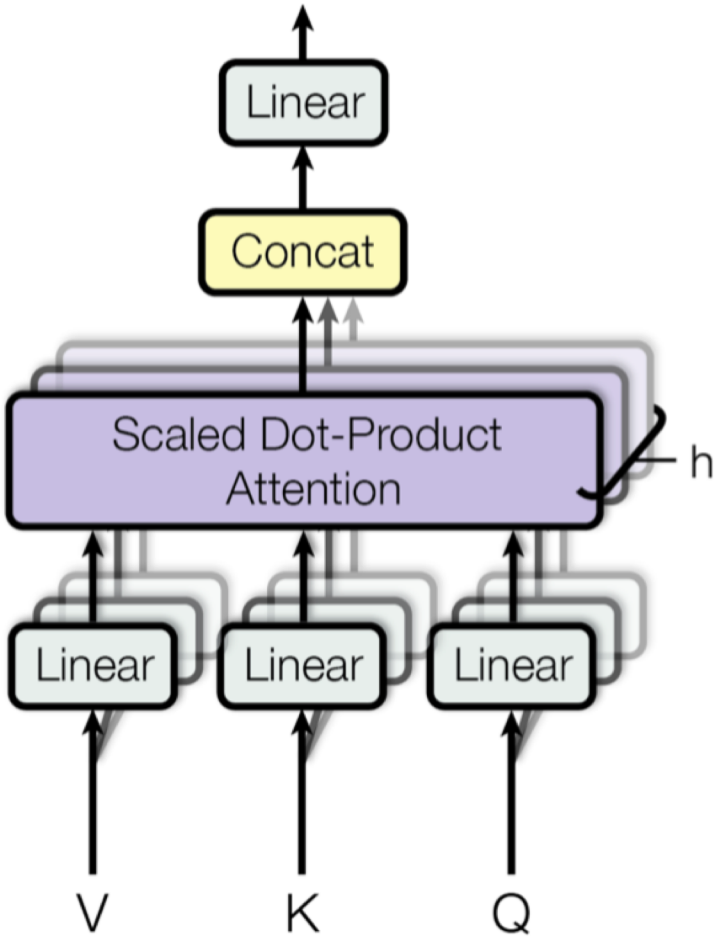

Decoder
Representieren wir diesen Satz <END>
Encoder
Refined contextual word embedding
Decoder embeddings attending to encoder embeddings
Preliminary contextual word embedding
Self attention
Self attention w/ masking
Word embeddings
Let's represent this sentence
<START> Repres- entieren wir diesen Satz

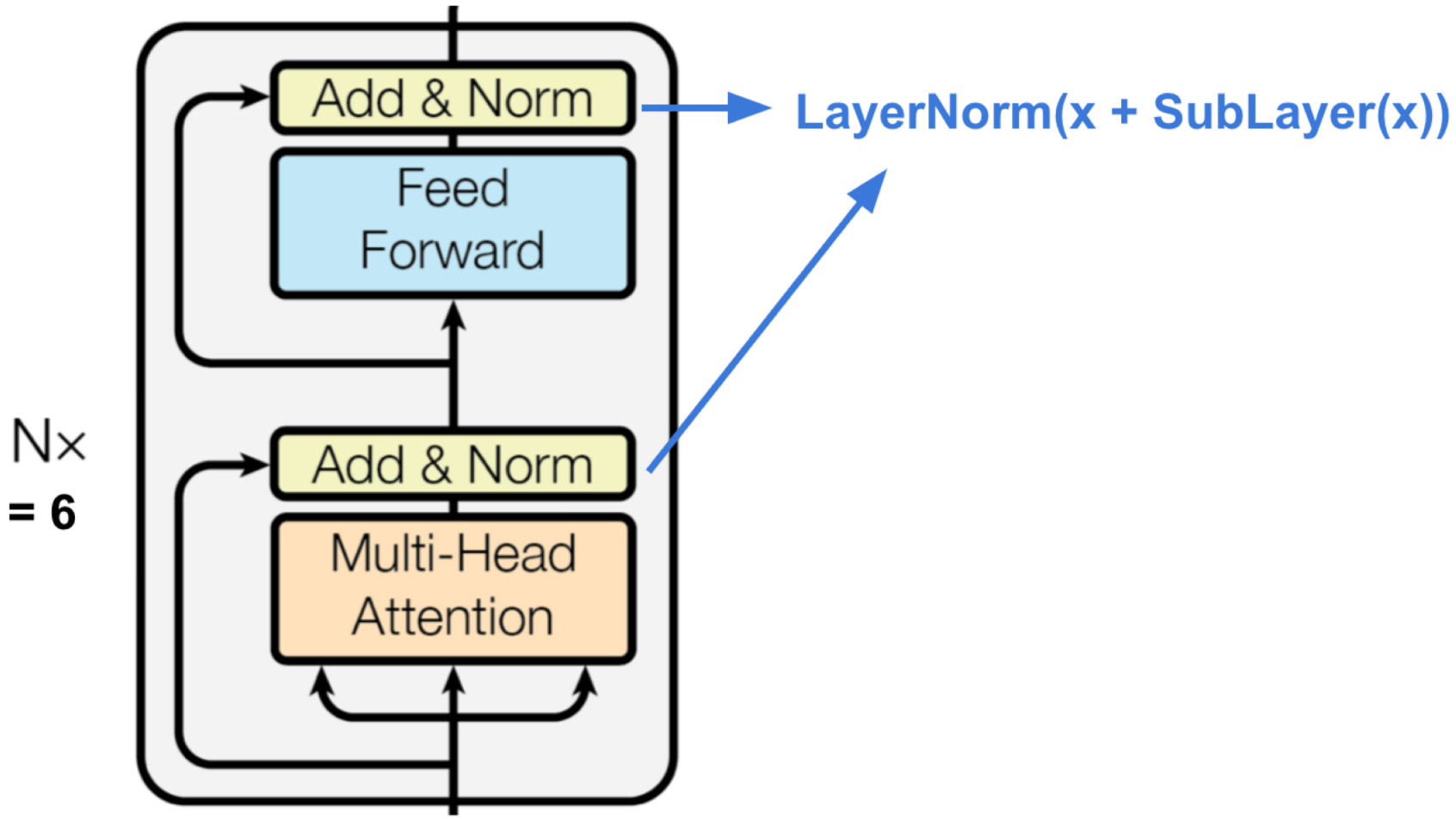


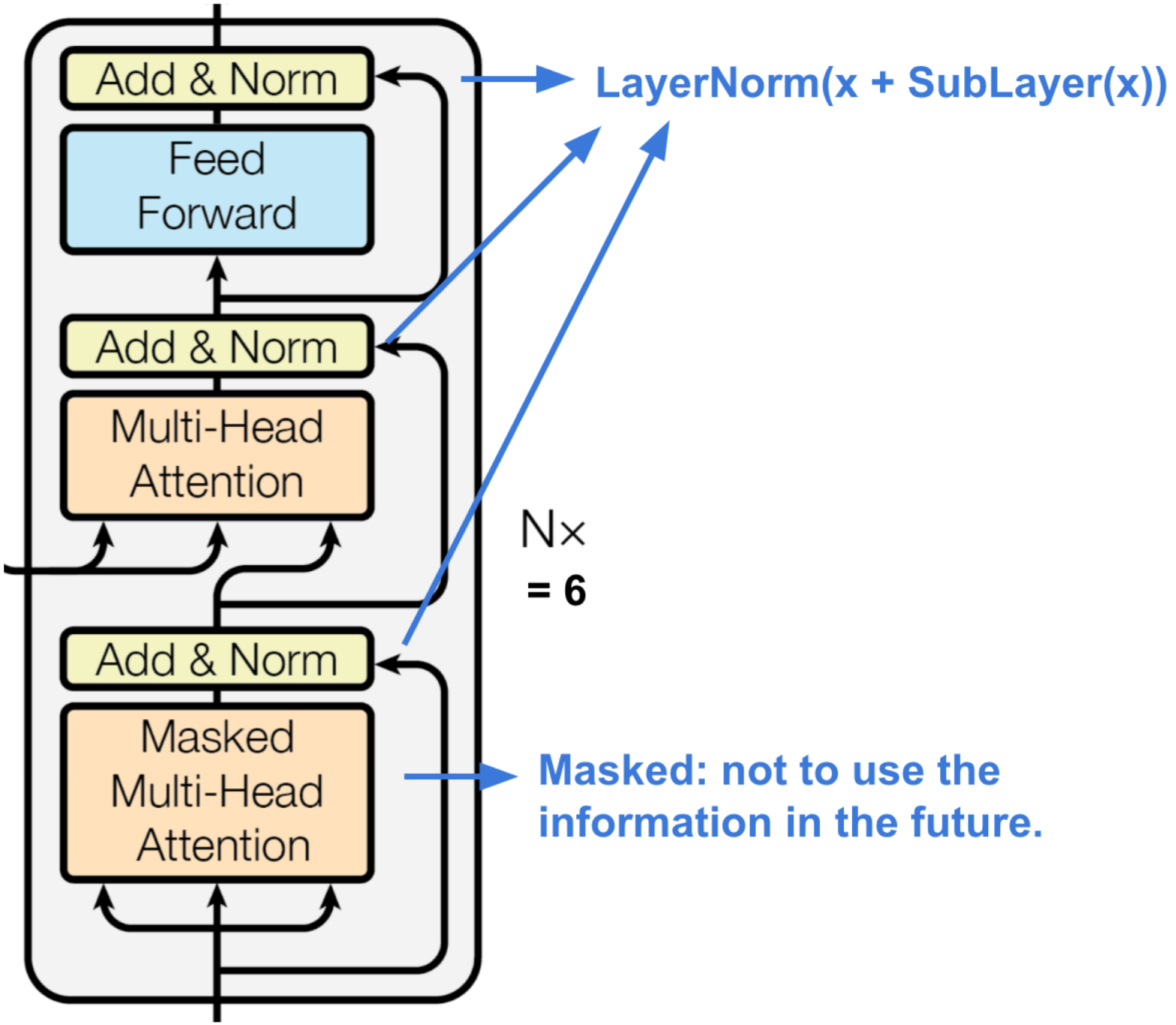
Source: http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf


Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
Nx
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
Positional Encoding
Positional Encoding
Input Embedding
Output Embedding
Inputs
Outputs (shifted right)
Multi-head attention
Linear
Concat
Scaled Dot-Product Attention
h
Linear
Linear
Linear
V
K
Q
Zoom-In!
Scaled dot-product attention
MatMul
SoftMax
Mask (opt.)
Scale
MatMul
Q
K
V
Zoom-In!

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{n}})\mathbf{V}$$




Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
= 6
LayerNorm(x + SubLayer(x))


Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
LayerNorm(x + SubLayer(x))
Nx
= 6
Masked: not to use the information in the future.

Summary

- Discussed the bottleneck problem with the vanilla sequence-to-sequence model
- Introduced attention as a solution
- Introduced the transformer architecture

References

- http://web.stanford.edu/class/cs224n/slides/cs224n-2019-lecture08-nmt.pdf
- http://web.stanford.edu/class/cs224n/slides/cs224n-2019-lecture14-transformers.pdf
- https://lilianweng.github.io/lil-log/2018/06/24/attention-attention.html
- http://nlp.seas.harvard.edu/2018/04/03/attention.html