Convolutional Sequence to Sequence Learning

Denis Yarats

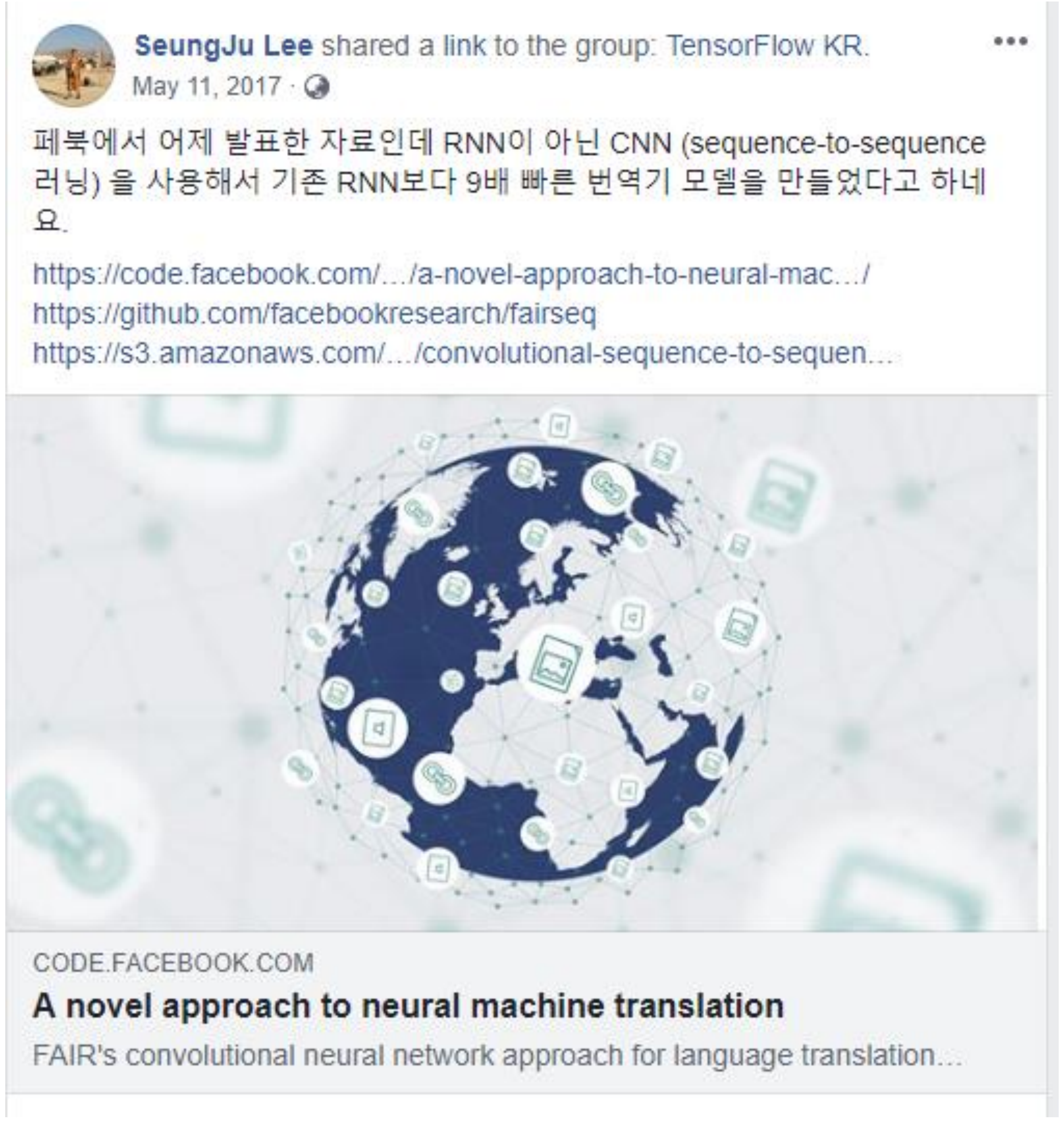
with Jonas Gehring, Michael Auli, David Grangier, Yann Dauphin

Facebook AI Research

발표자 : 모경현

History

2017년 5월 facebook에서 CNN 기반 Sequence to sequence 모델을 발표


SeungJu Lee shared a link to the group: TensorFlow KR.
May 11, 2017 ·

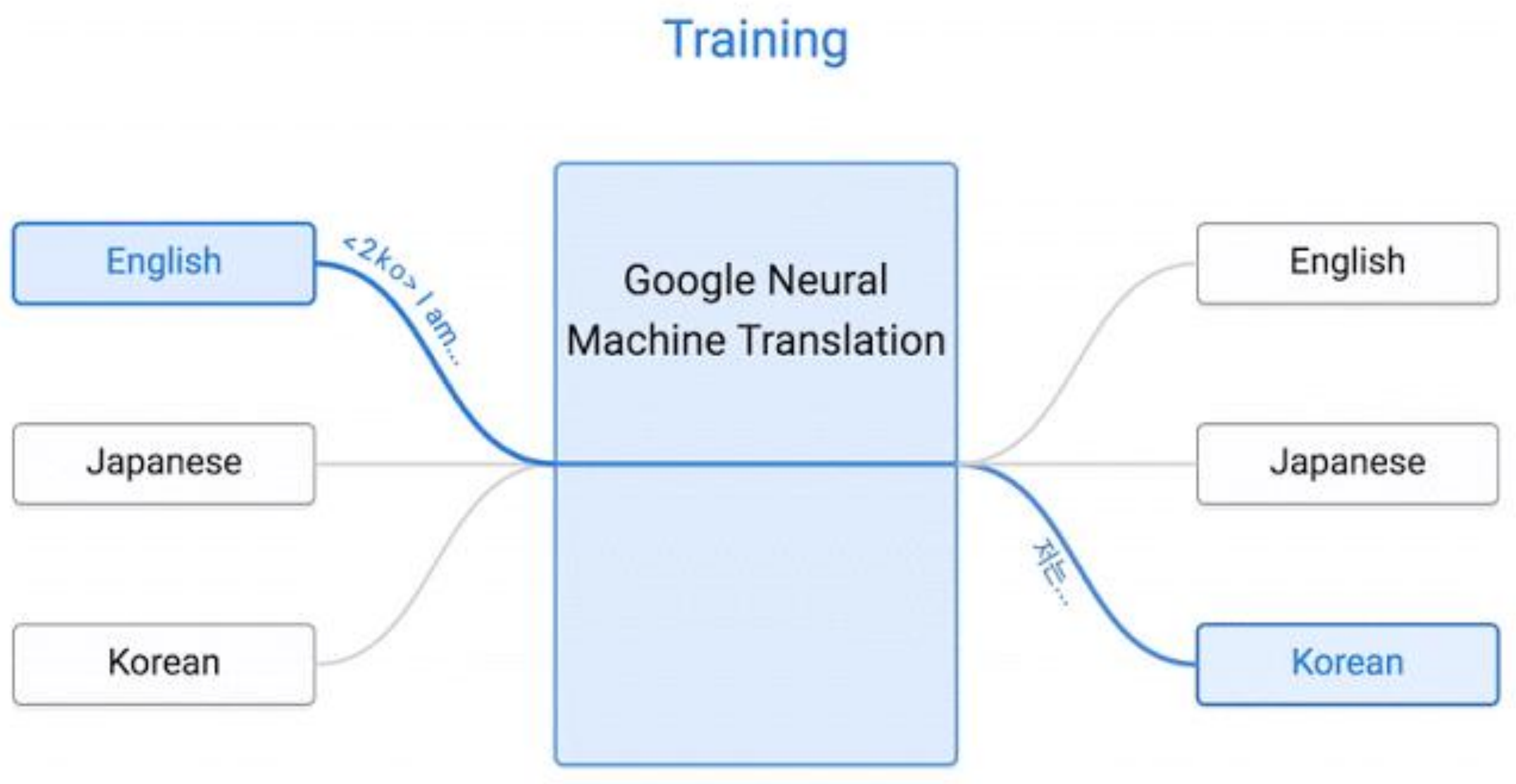
페북에서 어제 발표한 자료인데 RNN이 아닌 CNN (sequence-to-sequence 러닝) 을 사용해서 기존 RNN보다 9배 빠른 번역기 모델을 만들었다고 하네요.

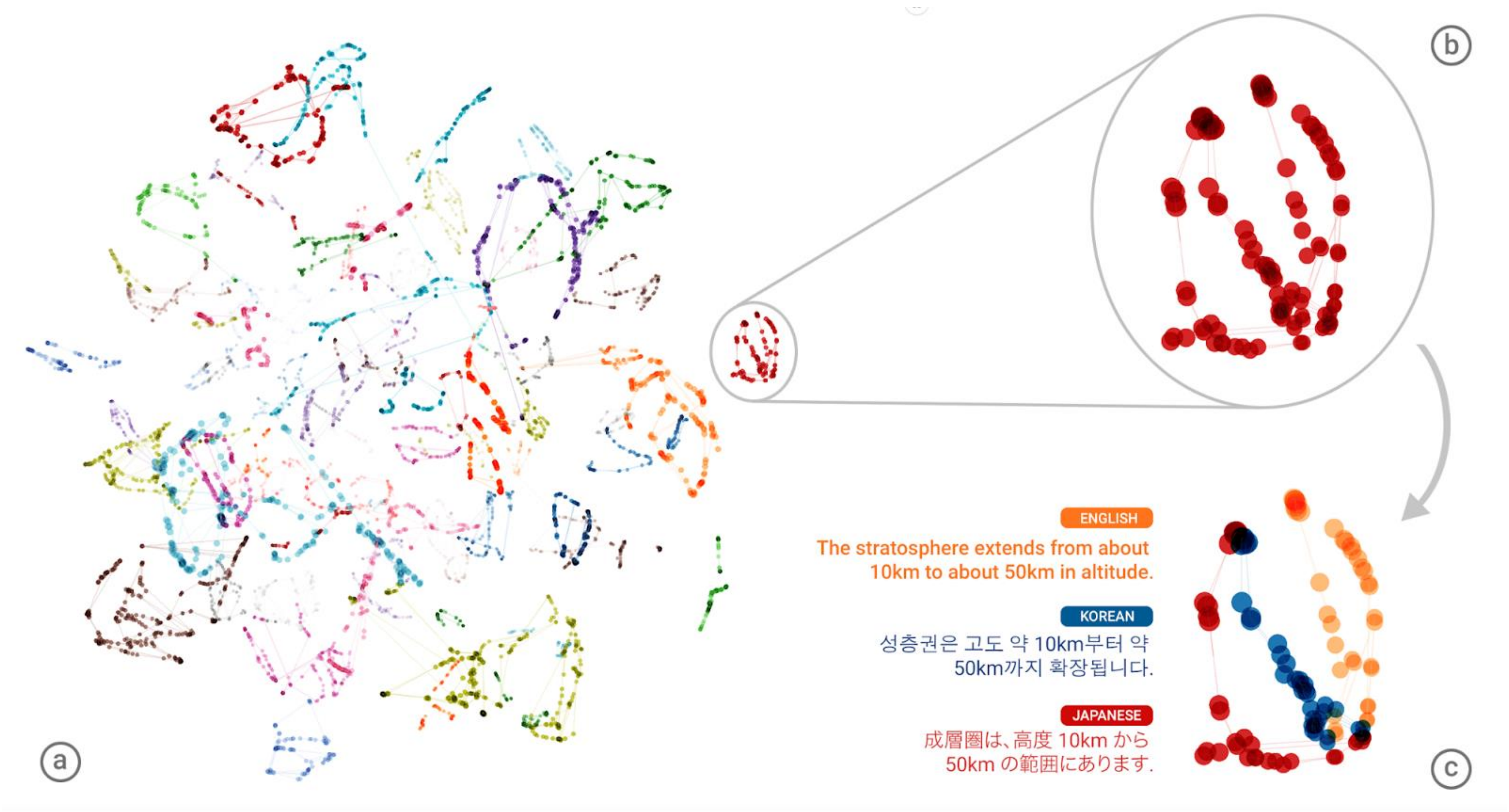
https://code.facebook.com/.../a-novel-approach-to-neural-mac.../
https://github.com/facebookresearch/fairseq
https://s3.amazonaws.com/.../convolutional-sequence-to-sequen...

CODE.FACEBOOK.COM
A novel approach to neural machine translation
FAIR's convolutional neural network approach for language translation...

History

기존에 SOTA 성능을 내고 있다고 알려진 Google의 GNMT 모델보다 높은 번역 성능을 낸다고 하여 화두가 됨

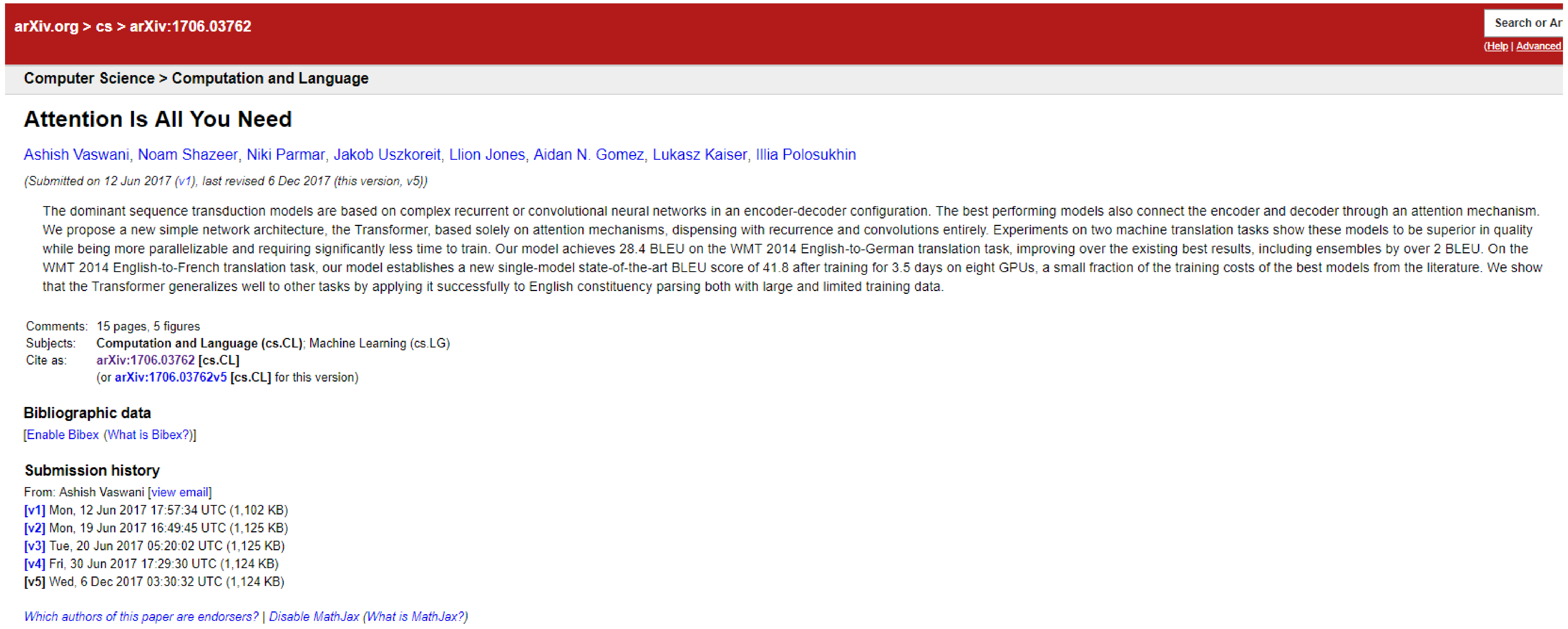




History

이후 한달만에 Attention is all you need라는 제목의 transformer 논문을 Google이 발표함

성능 평가척도에 Conv S2S를 언급하여 서로 경쟁하는 모습을 보임



arXiv.org > cs > arXiv:1706.03762

Search or Ar

(Help | Advanced

Computer Science > Computation and Language

Attention Is All You Need

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin

(Submitted on 12 Jun 2017 (v1), last revised 6 Dec 2017 (this version, v5))

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks in an encoder-decoder configuration. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.8 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data.

Comments:	15 pages, 5 figures
Subjects:	**Computation and Language (cs.CL)**; Machine Learning (cs.LG)
Cite as:	arXiv:1706.03762 [cs.CL]
	(or arXiv:1706.03762v5 [cs.CL] for this version)

Bibliographic data

[Enable Bibex (What is Bibex?)]

Submission history

From: Ashish Vaswani [view email]
[v1] Mon, 12 Jun 2017 17:57:34 UTC (1,102 KB)
[v2] Mon, 19 Jun 2017 16:49:45 UTC (1,125 KB)
[v3] Tue, 20 Jun 2017 05:20:02 UTC (1,125 KB)
[v4] Fri, 30 Jun 2017 17:29:30 UTC (1,124 KB)
[v5] Wed, 6 Dec 2017 03:30:32 UTC (1,124 KB)

Which authors of this paper are endorsers? | Disable MathJax (What is MathJax?)

Introduction

Introduction

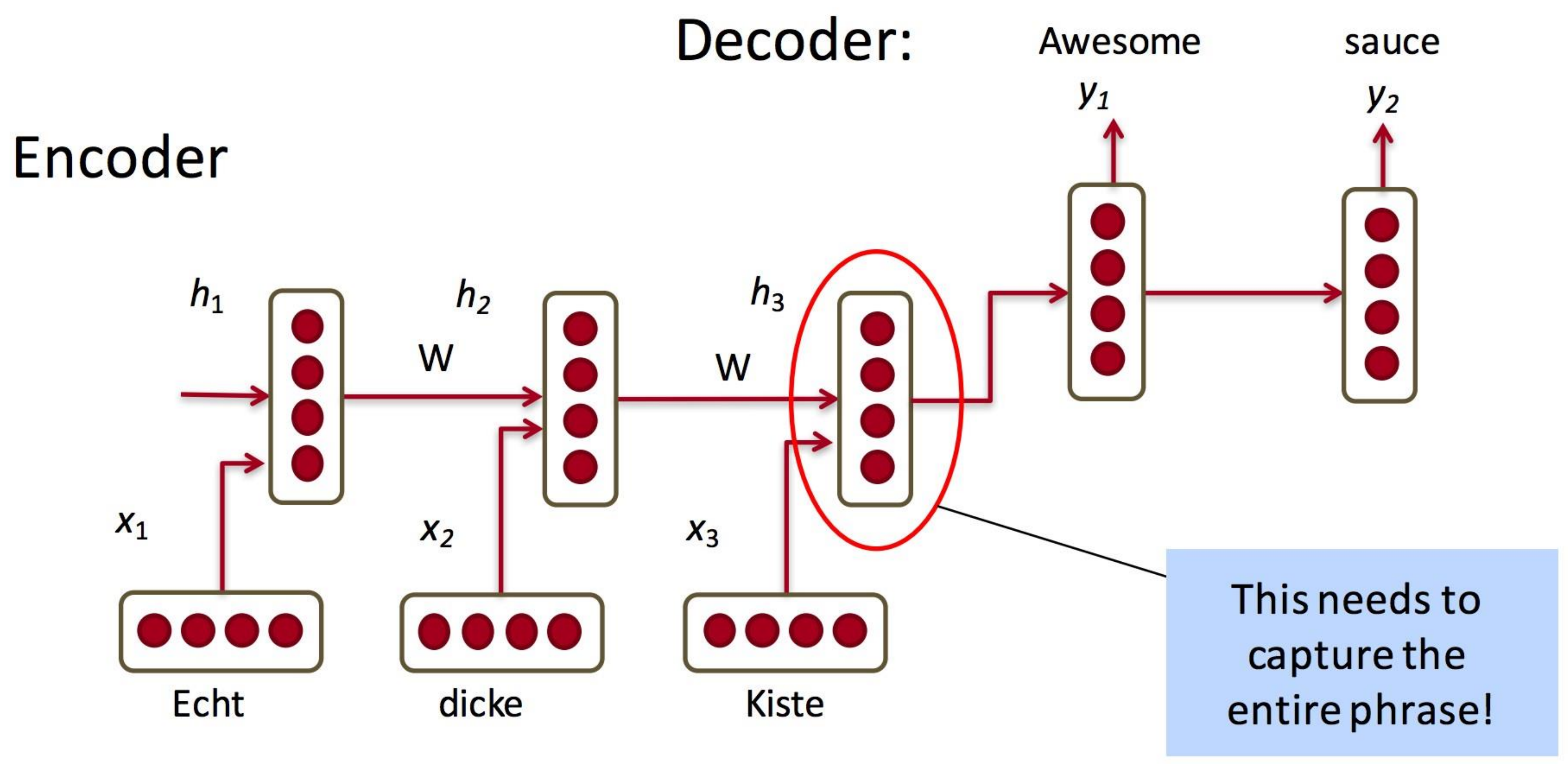
Sequence to sequence learning은 많은 task에서 좋은 성능을 보이는 것으로 알려짐

- Machine translation
- Speech recognition
- Text summarization
- …

Introduction

Sequence to sequence learning은 많은 task에서 좋은 성능을 보이는 것으로 알려짐

특히 attention을 활용한 RNN기반 encoder-decoder approach는 전통적인 방식들보다 높은 성능을 보여 그 효과를 입증함



Introduction

그러나 RNN 기반 encoder-decoder approach에는 몇 가지 문제점이 있음

- 장기 의존성(Long term dependencies)
- 병렬 처리 불가(prevent parallel)
- 그래디언트 손실(vanishing gradient)
- 큰 계산 복잡도 O(n)

Introduction

그러나 RNN 기반 encoder-decoder approach에는 몇 가지 문제점이 있음

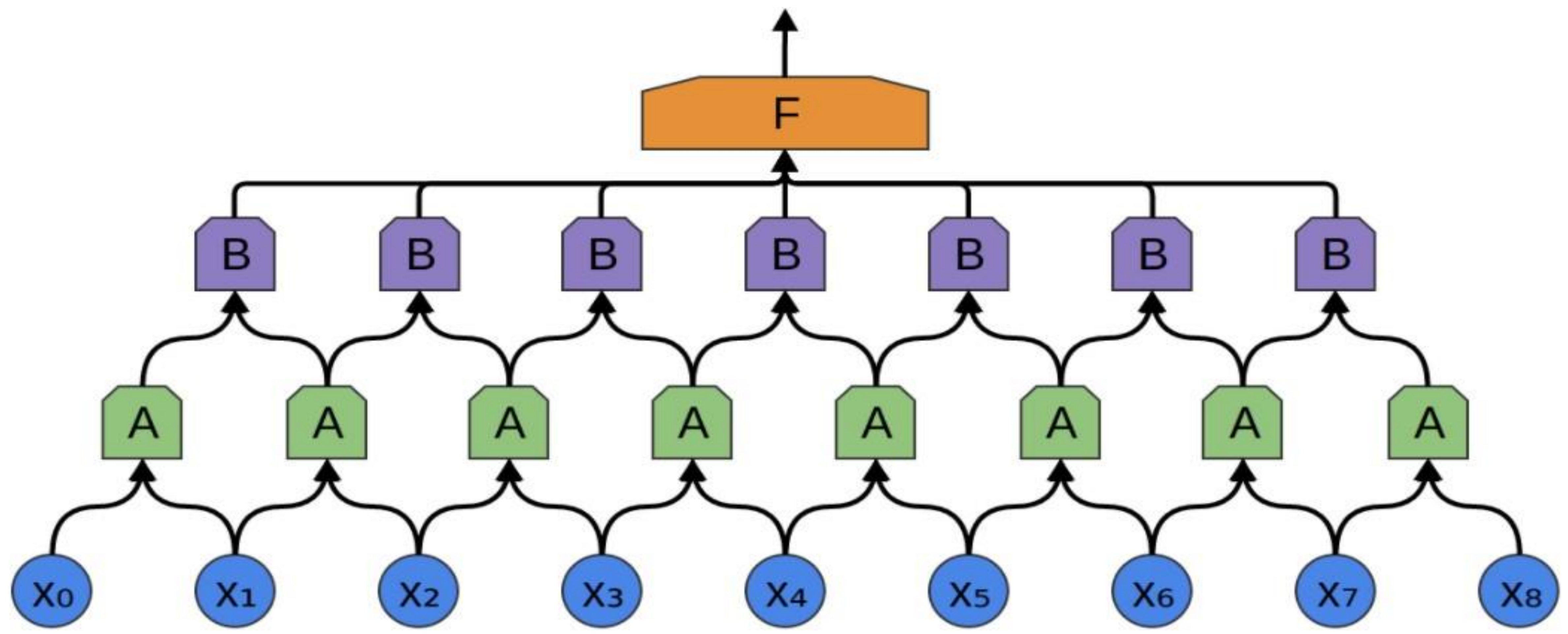
- 장기 의존성(Long term dependencies)
- 병렬 처리 불가(prevent parallel)
- 그래디언트 손실(vanishing gradient)
- 큰 계산 복잡도 O(n)

그러니 RNN 대신 CNN을 사용하자!

Introduction

CNN은 time step에 대한 의존성이 없음

병렬 연산이 가능함

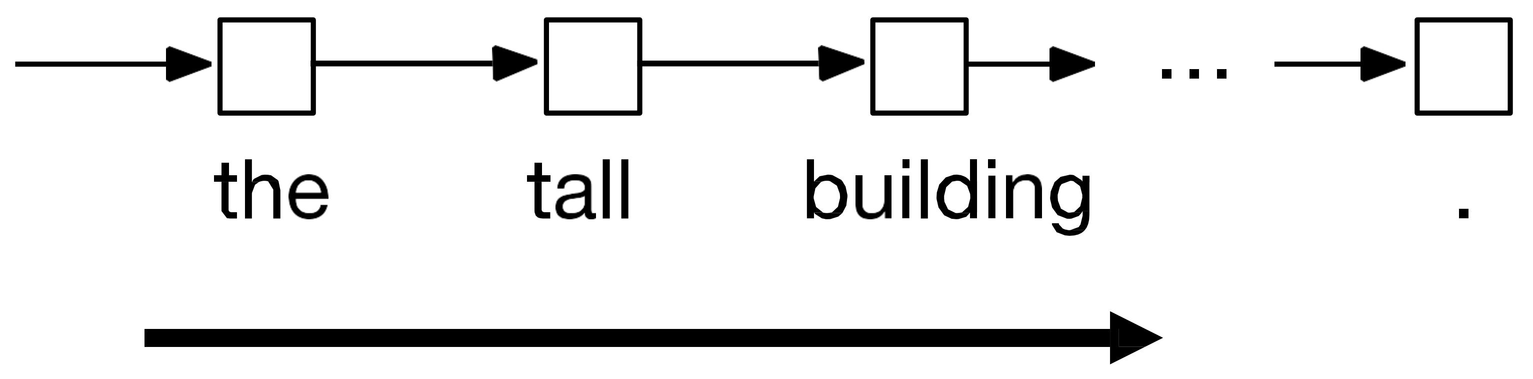


Introduction

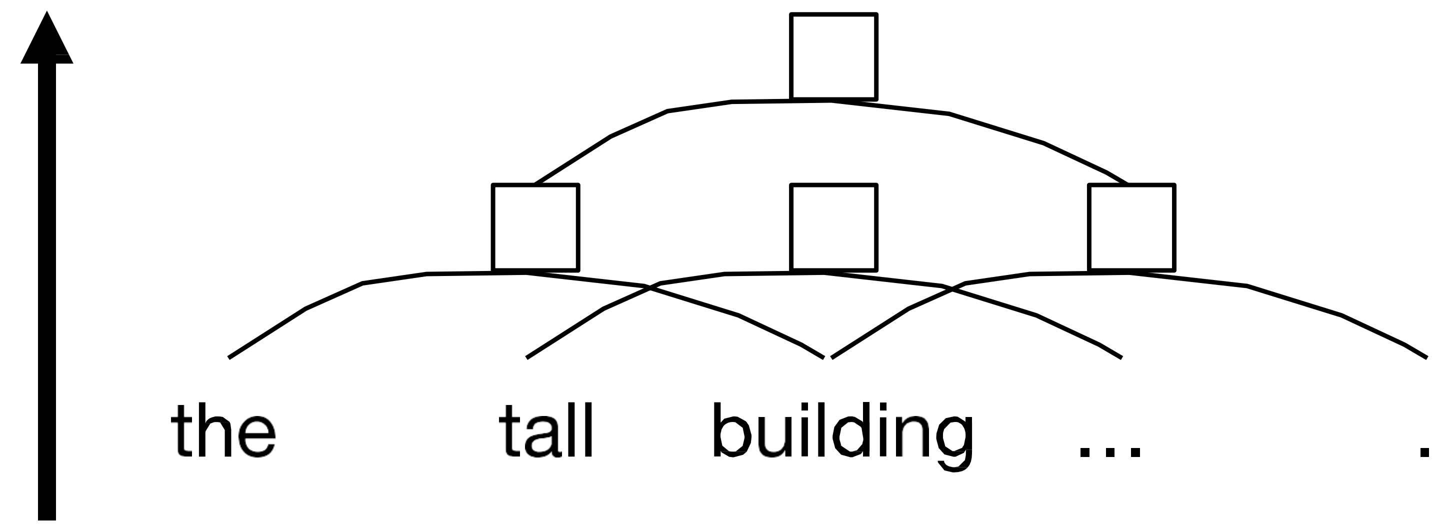
계층적 구조를 통해 RNN이 가진 chain structure 구조의 long term dependencies를 줄일 수 있음

보다 짧은 path로 전체 문장을 커버하는 것이 가능함

낮은 계산 복잡도를 가지게 됨 O($\frac{n}{k}$)



vs.



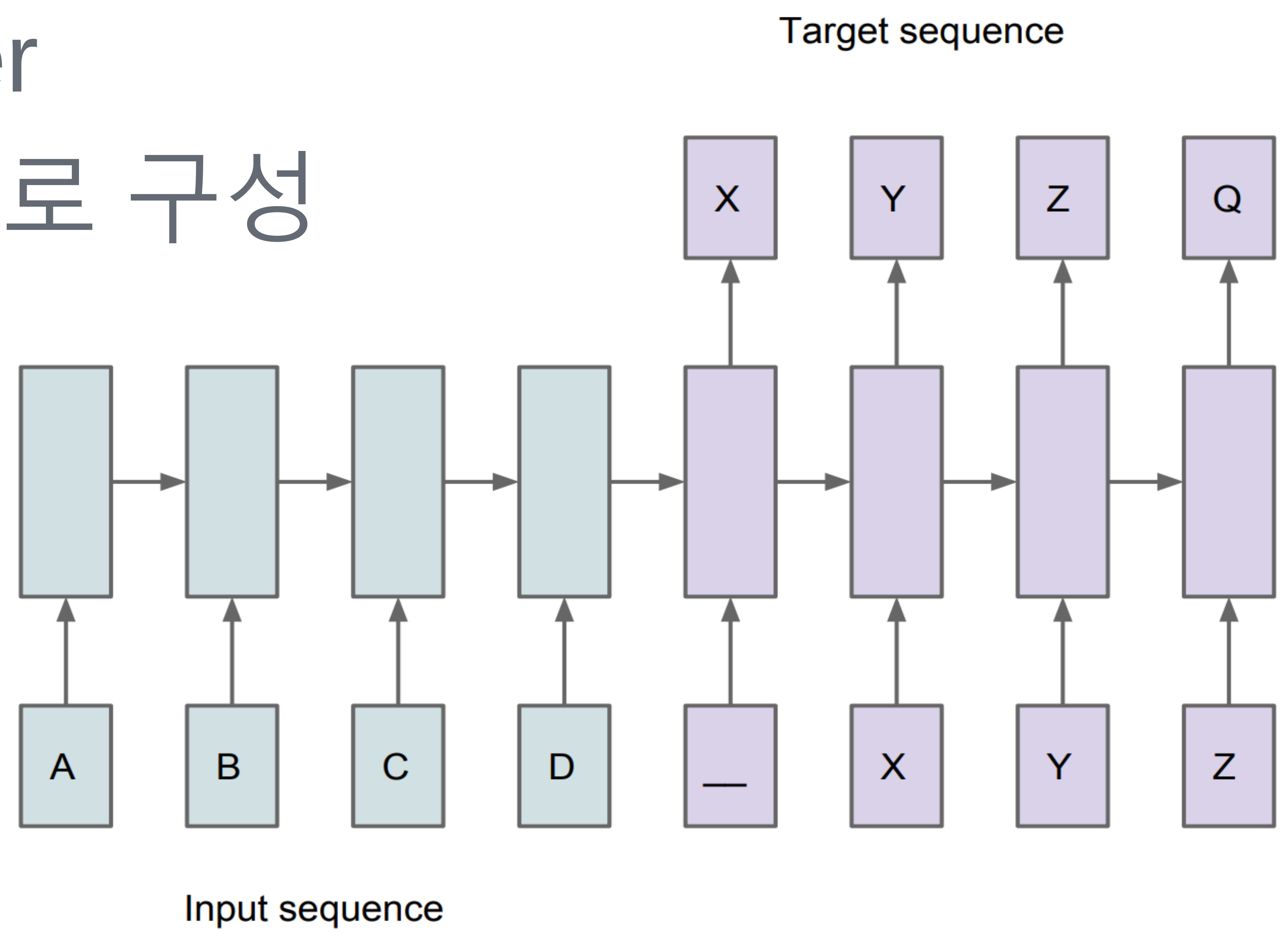
Attention

Attention

서로 다른 길이 문장을 처리하기 위해 2개 LSTM 또는 GRU를 사용

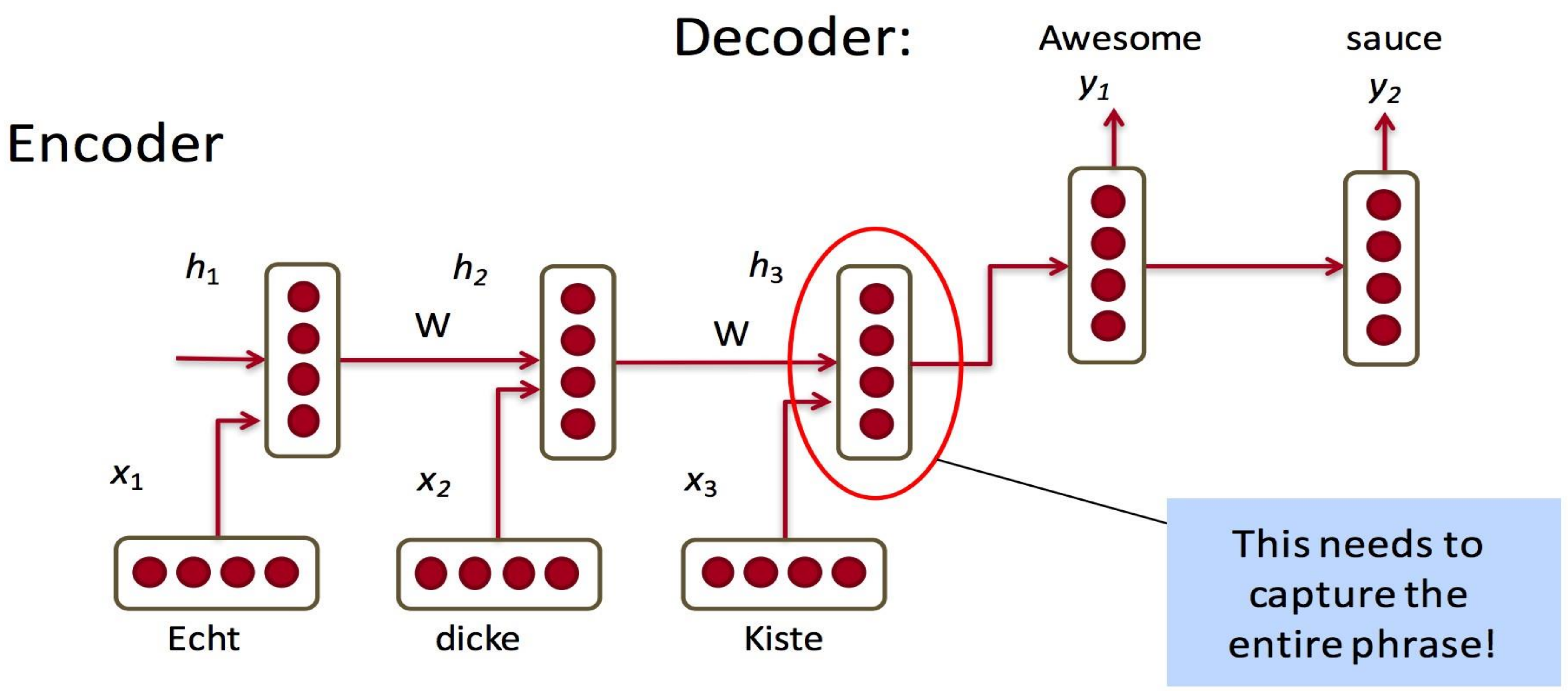
Source language를 처리하는 encoder
Target language를 처리하는 decoder로 구성

Sequence to sequence model
(Sutskever et al. 2014)



Attention

Encoder의 마지막 hidden state만을 이용해서 decoding을 수행하는 것은 병목현상을 일으킨다 주장



Attention

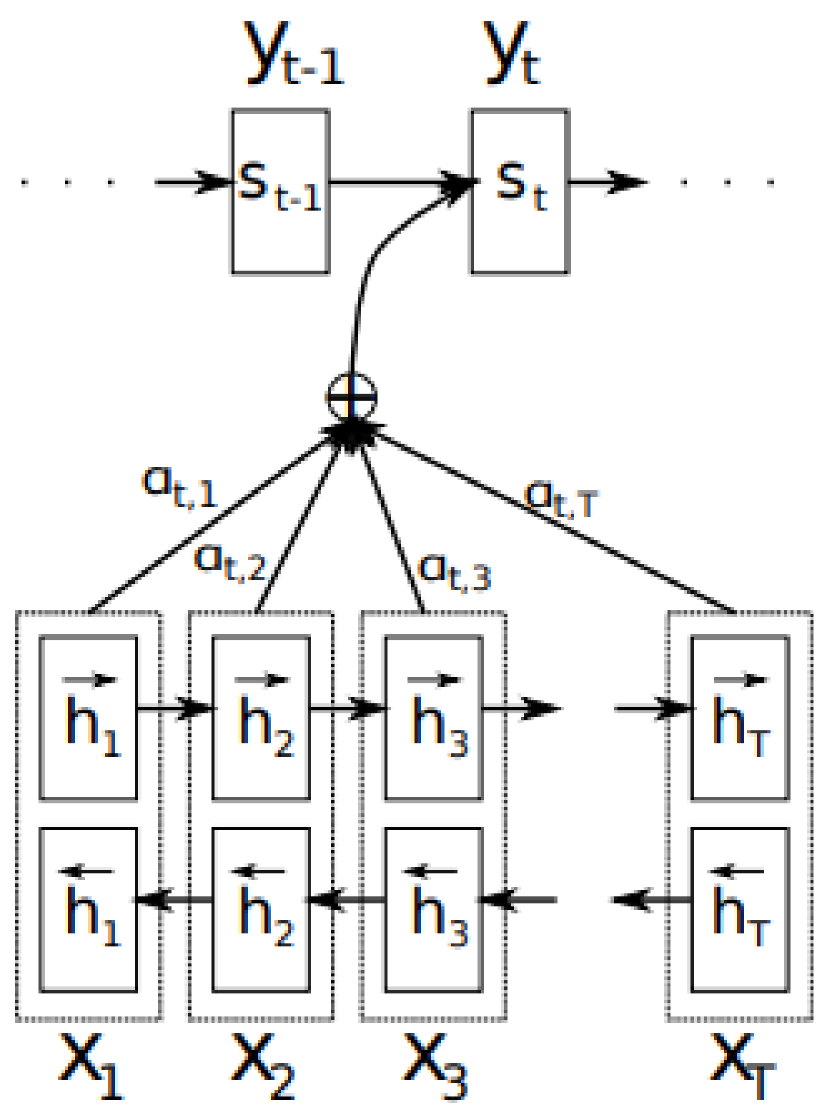
Bahdahau et al. 2014

Decoder hidden state가 next token을 예측할 때, 특정 encoder hidden state를 많이 참고하여 가중합을 활용하는 방식

Alignment model $e_{ij} = a(s_{i-1}, h_j)$

Weight $\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}$

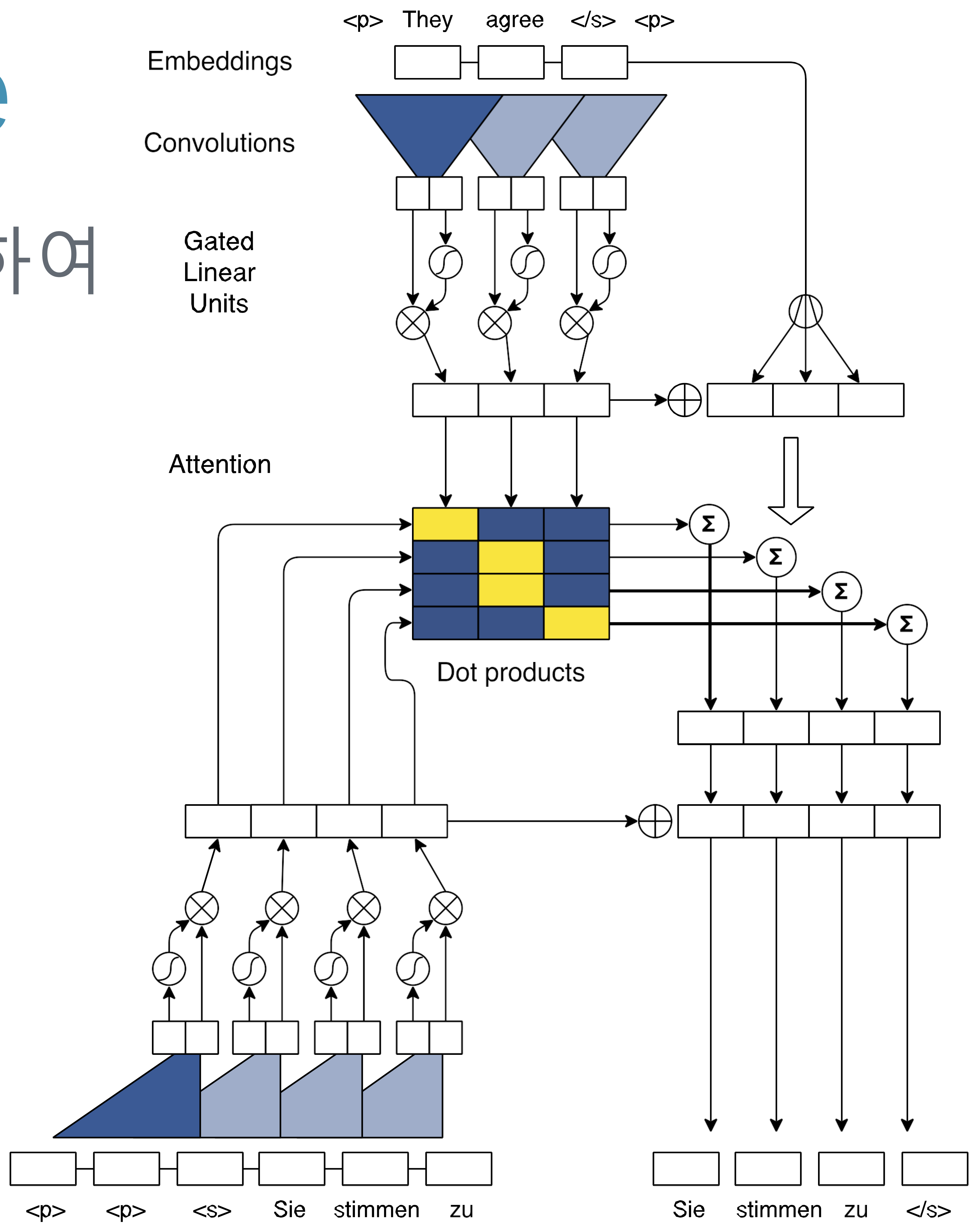
Context vector $c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j$


y_{t-1}
y_t
s_{t-1}
s_t
$\alpha_{t,1}$
$\alpha_{t,2}$
$\alpha_{t,3}$
$\alpha_{t,T}$
$\overrightarrow{h}_1$ $\overrightarrow{h}_2$ $\overrightarrow{h}_3$ $\overrightarrow{h}_T$
$\overleftarrow{h}_1$ $\overleftarrow{h}_2$ $\overleftarrow{h}_3$ $\overleftarrow{h}_T$
x_1 x_2 x_3 x_T

Conv S2S Architecture

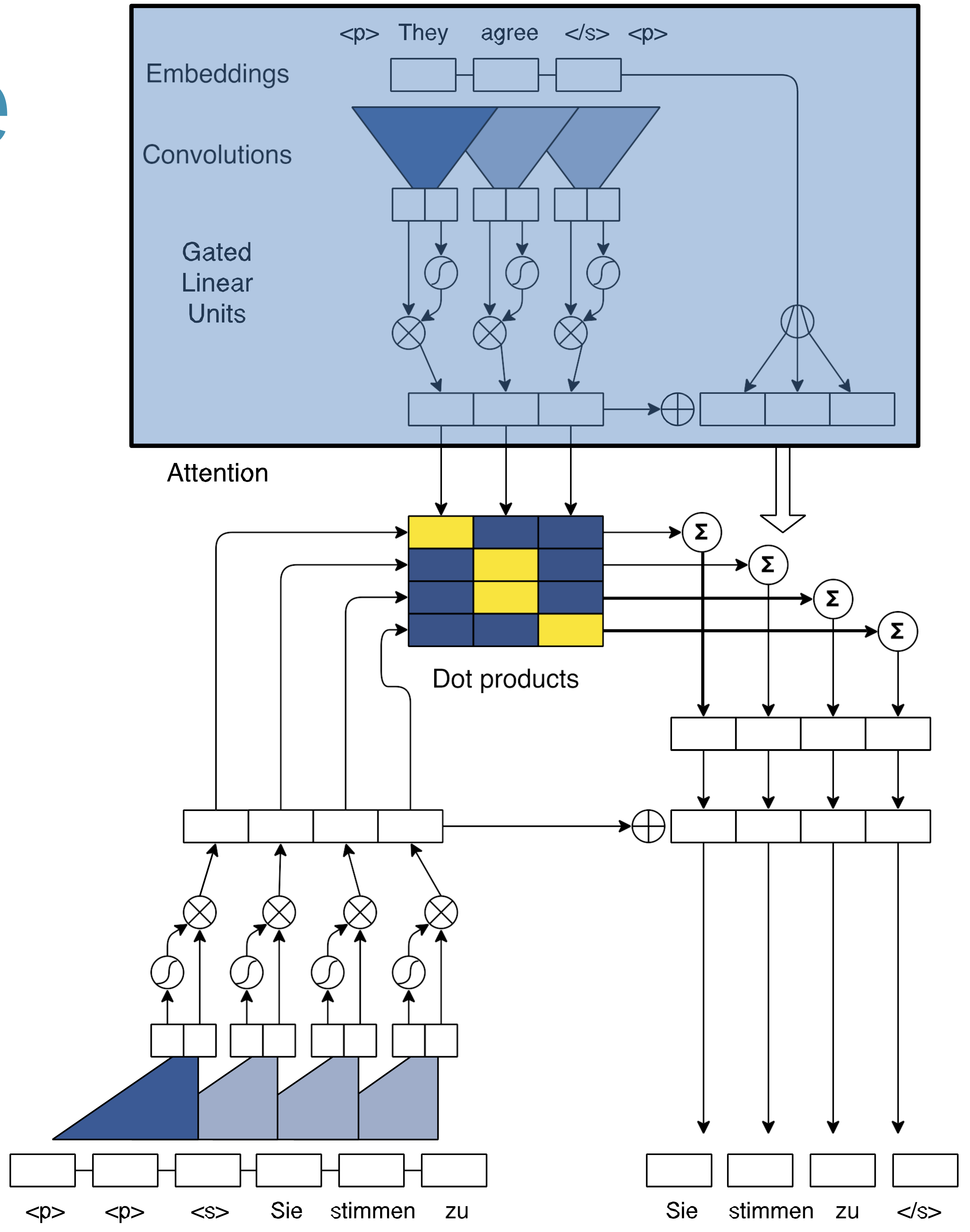
Conv S2S Architecture

Convolutional neural network를 사용하여 encoder와 decoder를 구성



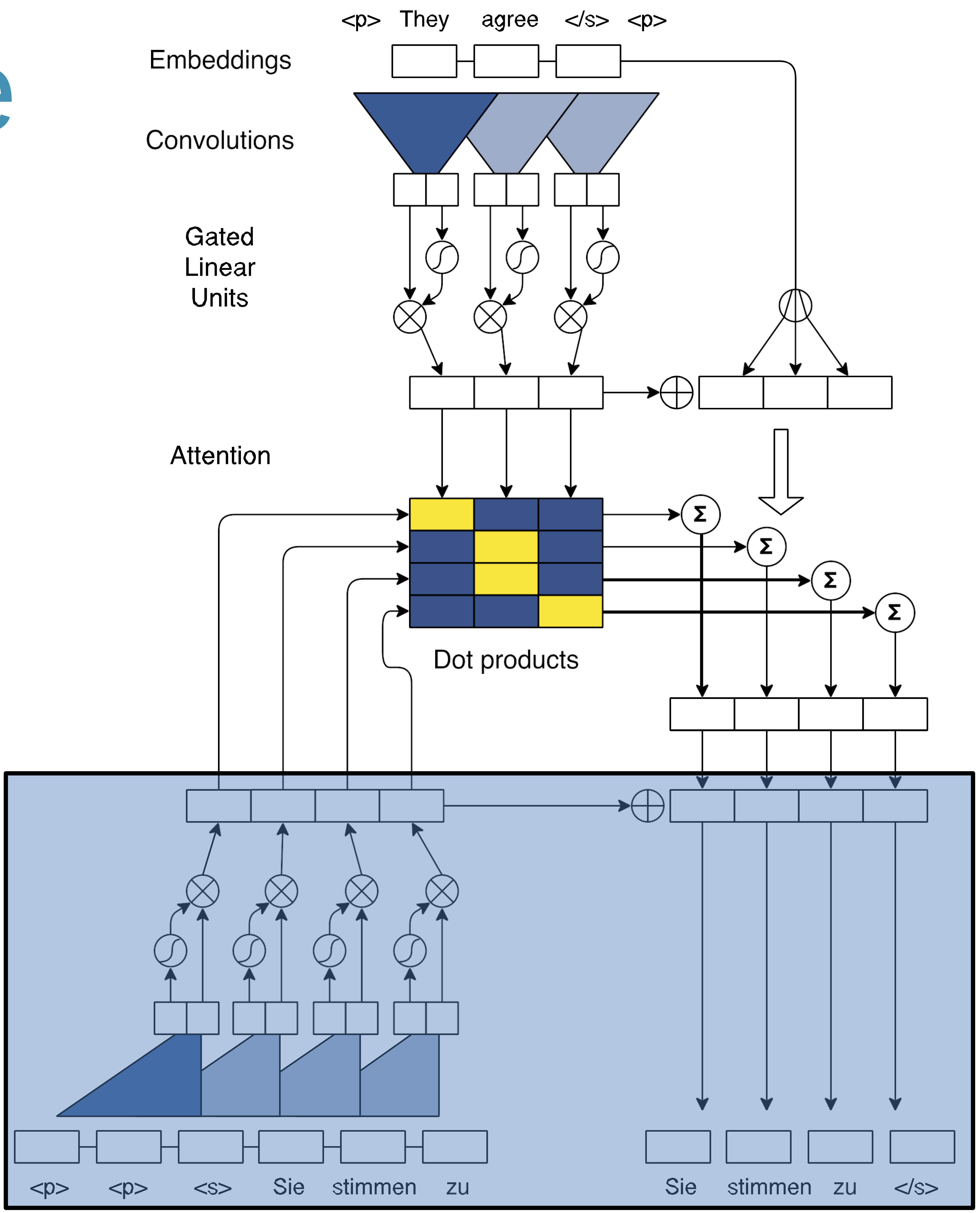
Conv S2S Architecture

Encoder



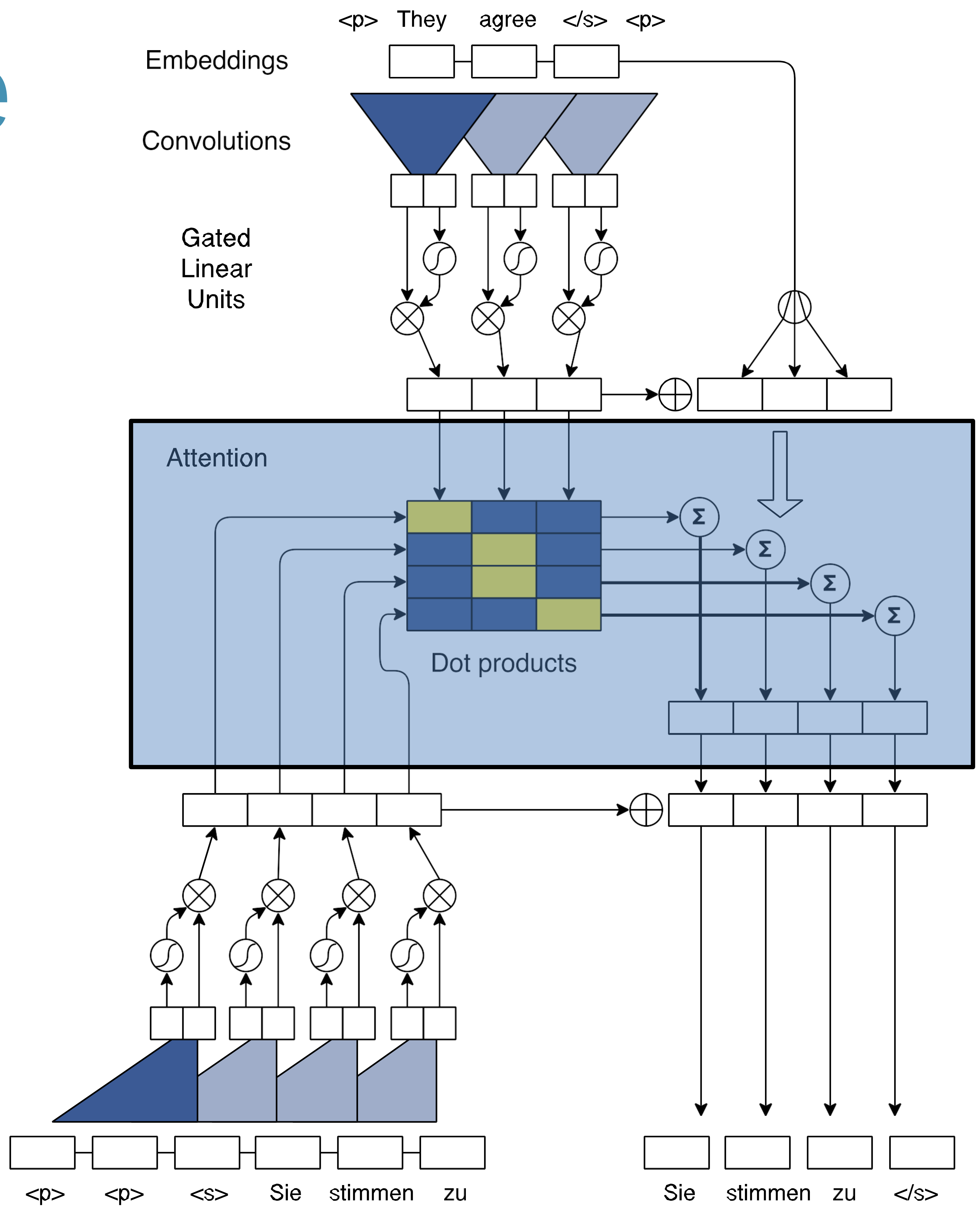
Conv S2S Architecture

Decoder



Conv S2S Architecture

Attention



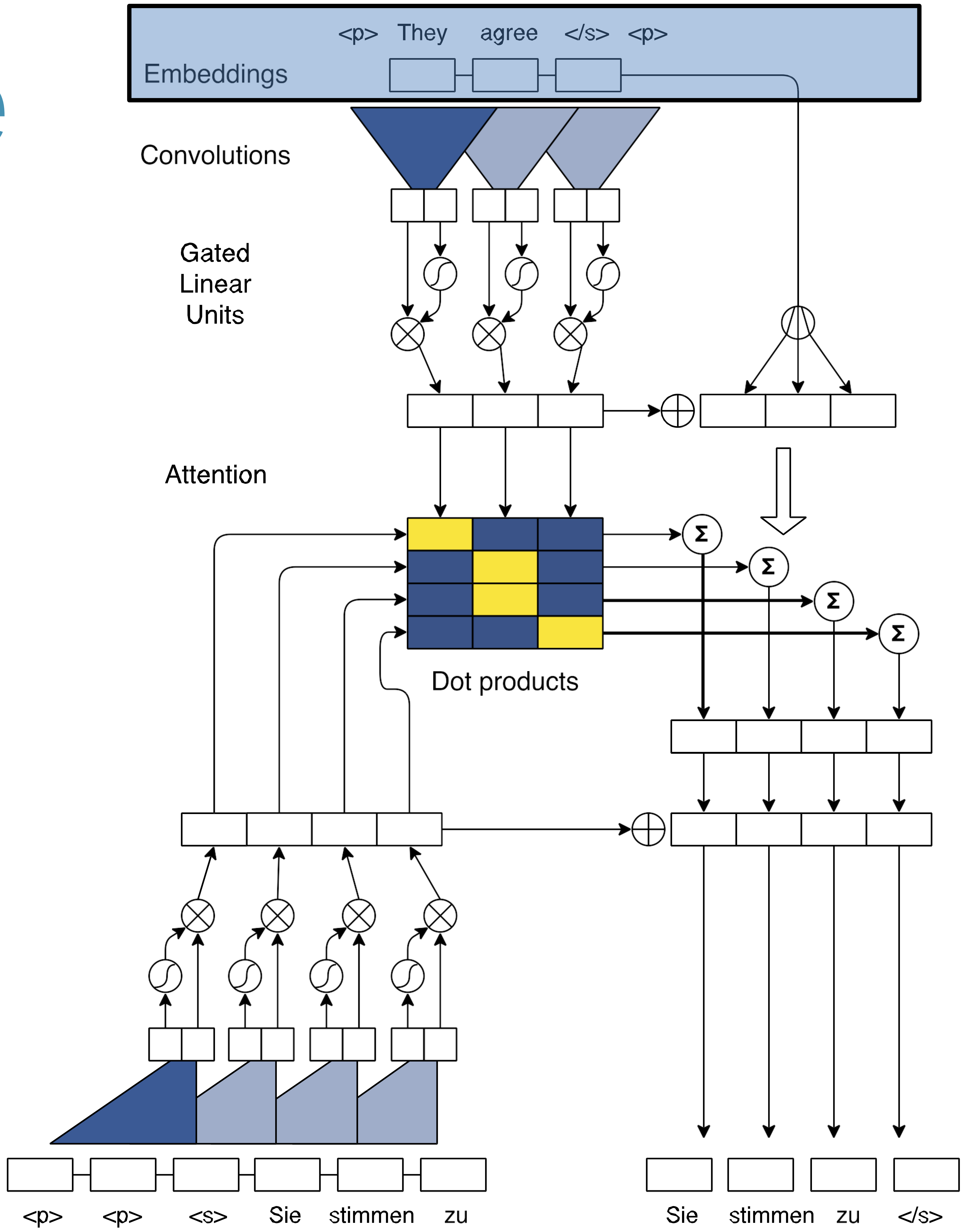
Conv S2S Architecture

Encoder

Word vector + position embedding

$$e = (w_1 + p_1, \dots, w_m + p_m)$$

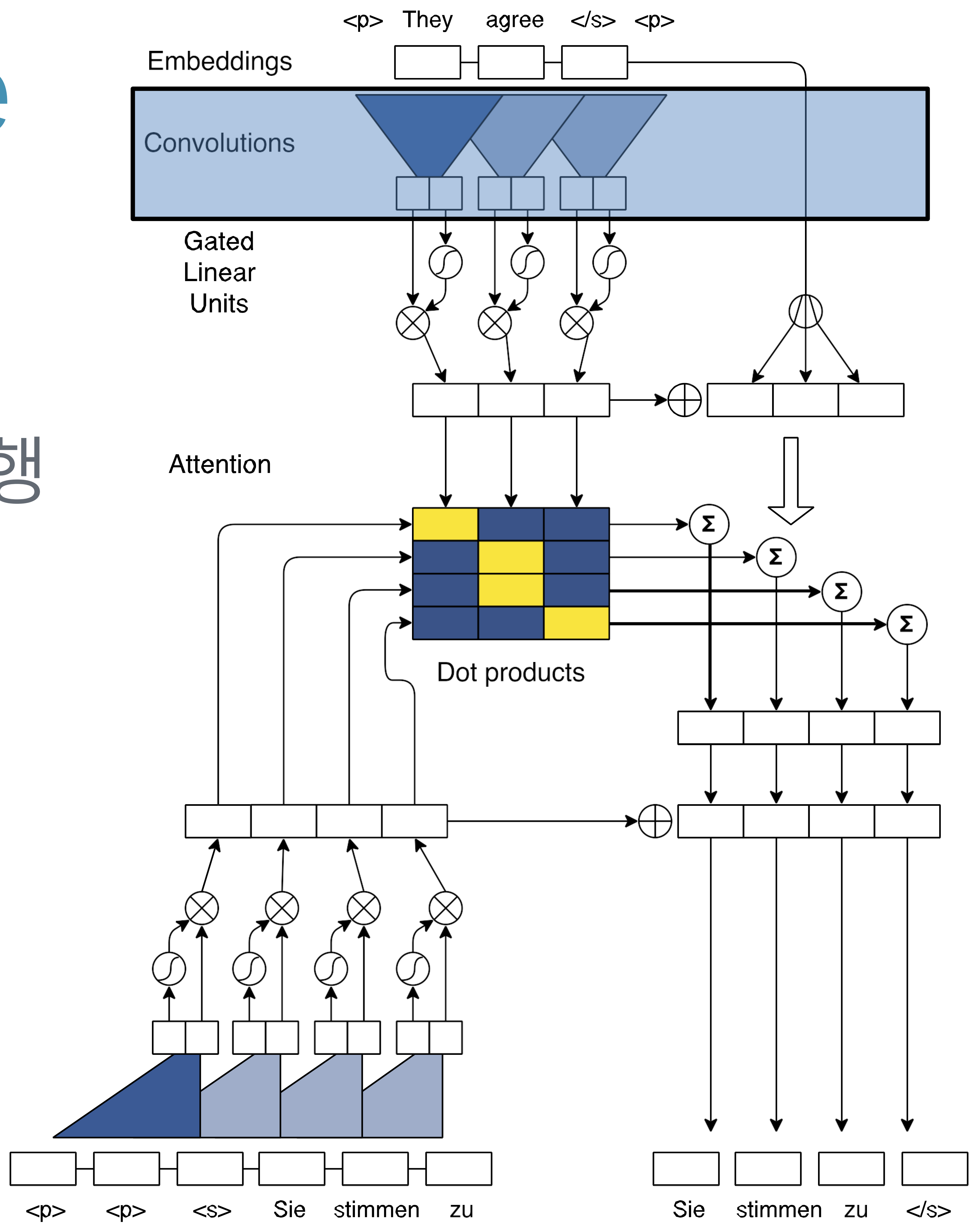
Fixed number of input elements



Conv S2S Architecture

Encoder

One dimensional convolution 연산 수행



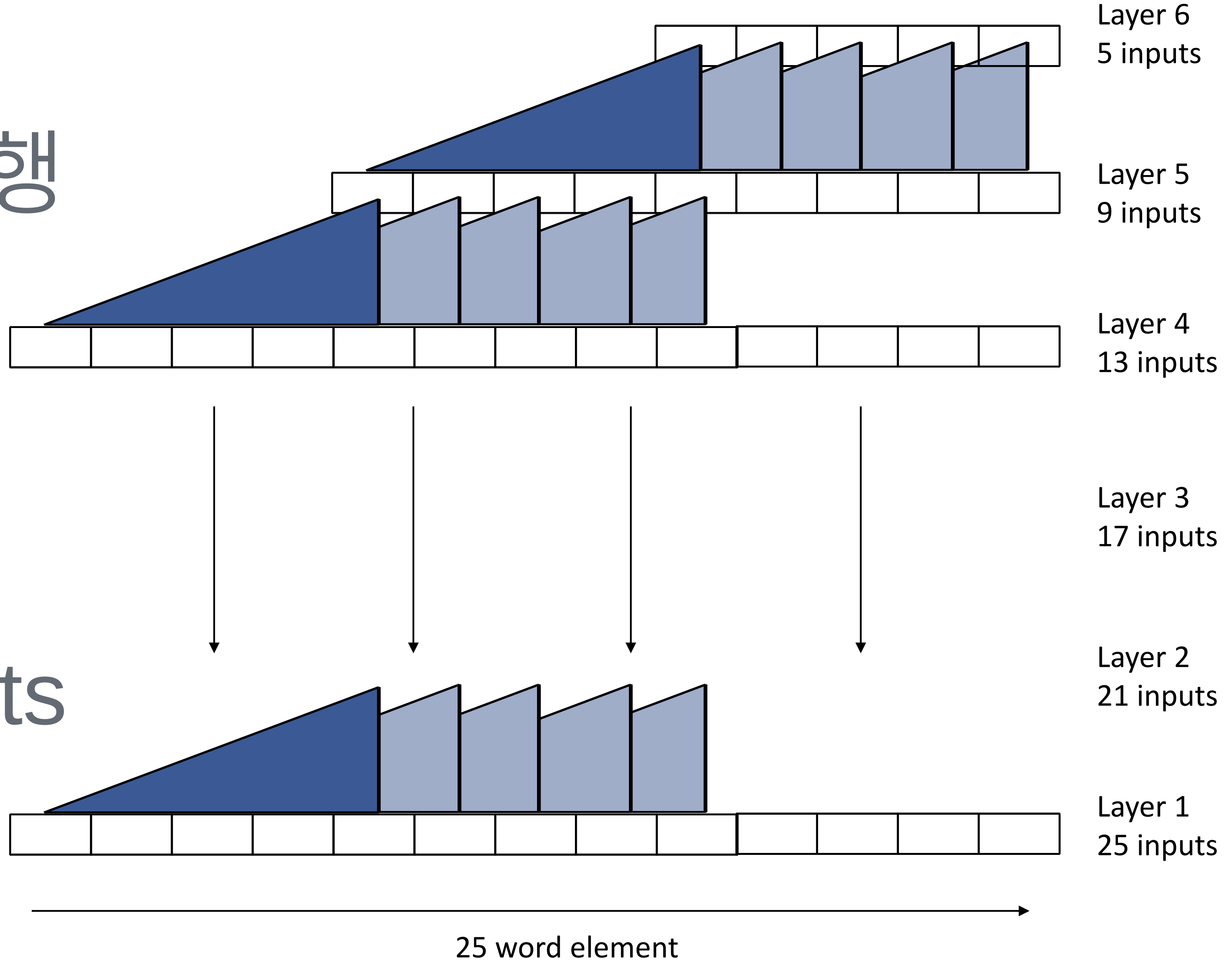
Conv S2S Architecture

Encoder

One dimensional convolution 연산 수행

크기 K인 kernel을 사용

- ex) stacking 6 block with k=5 :
  - each output depend on 25 inputs



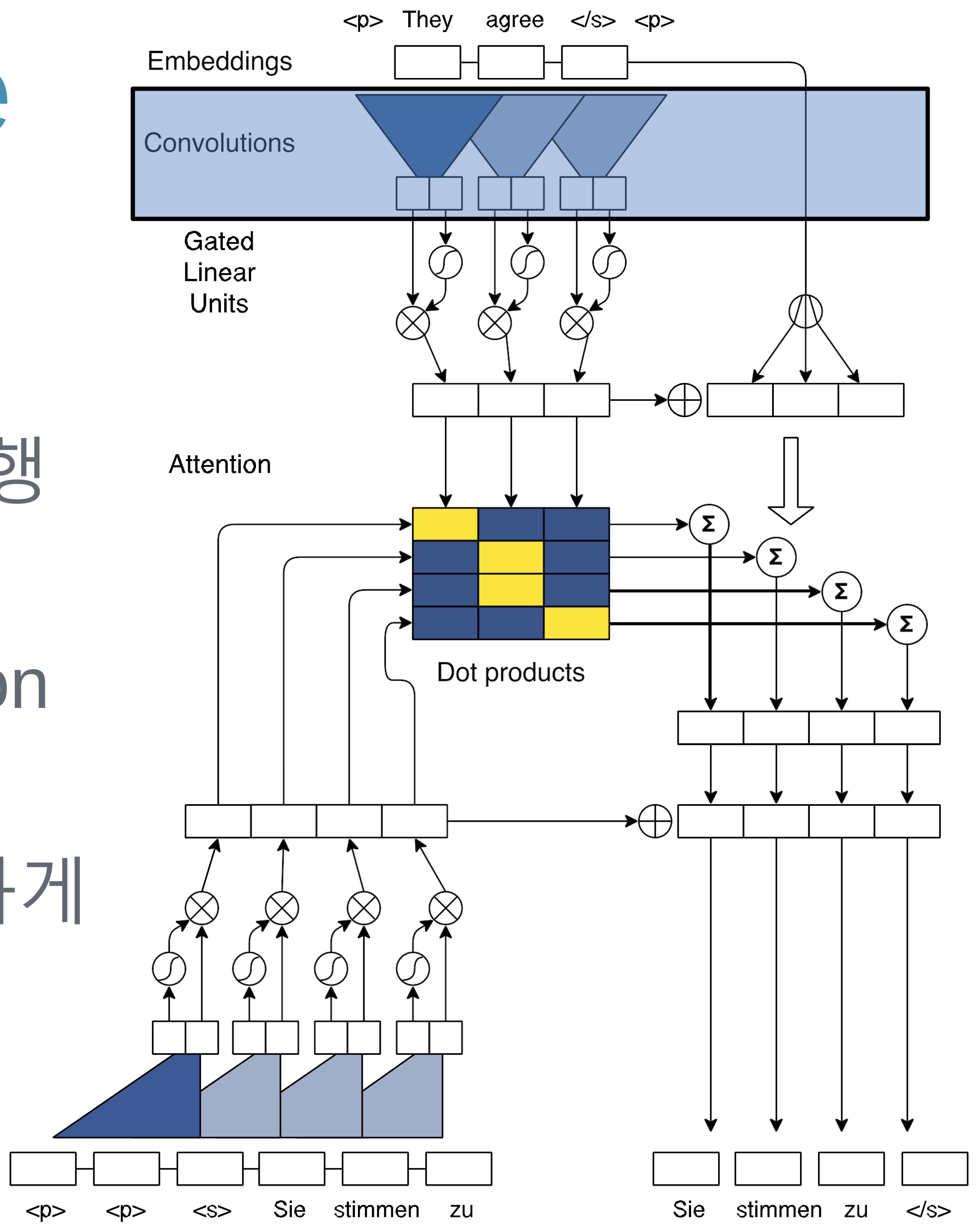
Conv S2S Architecture

Encoder

One dimensional convolution 연산 수행

Filter의 수가 2d개가 되어 2d dimension

Padding을 통해 input의 길이를 동일하게 유지

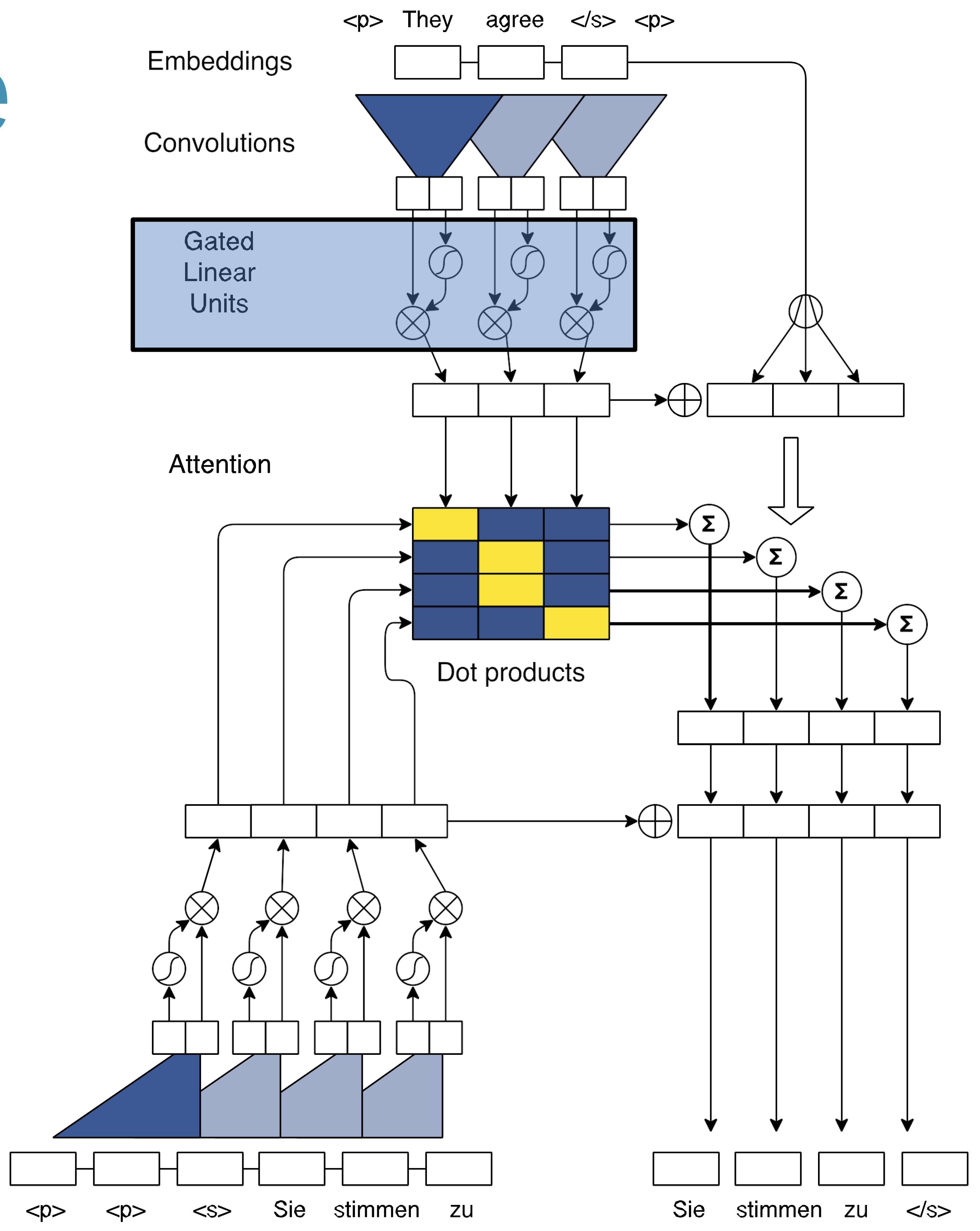


Conv S2S Architecture

Encoder

2d dimension에 GLU function 적용
(Dauphin et al., 2016)

$$v([A\ B]) = A \otimes \sigma(B)$$

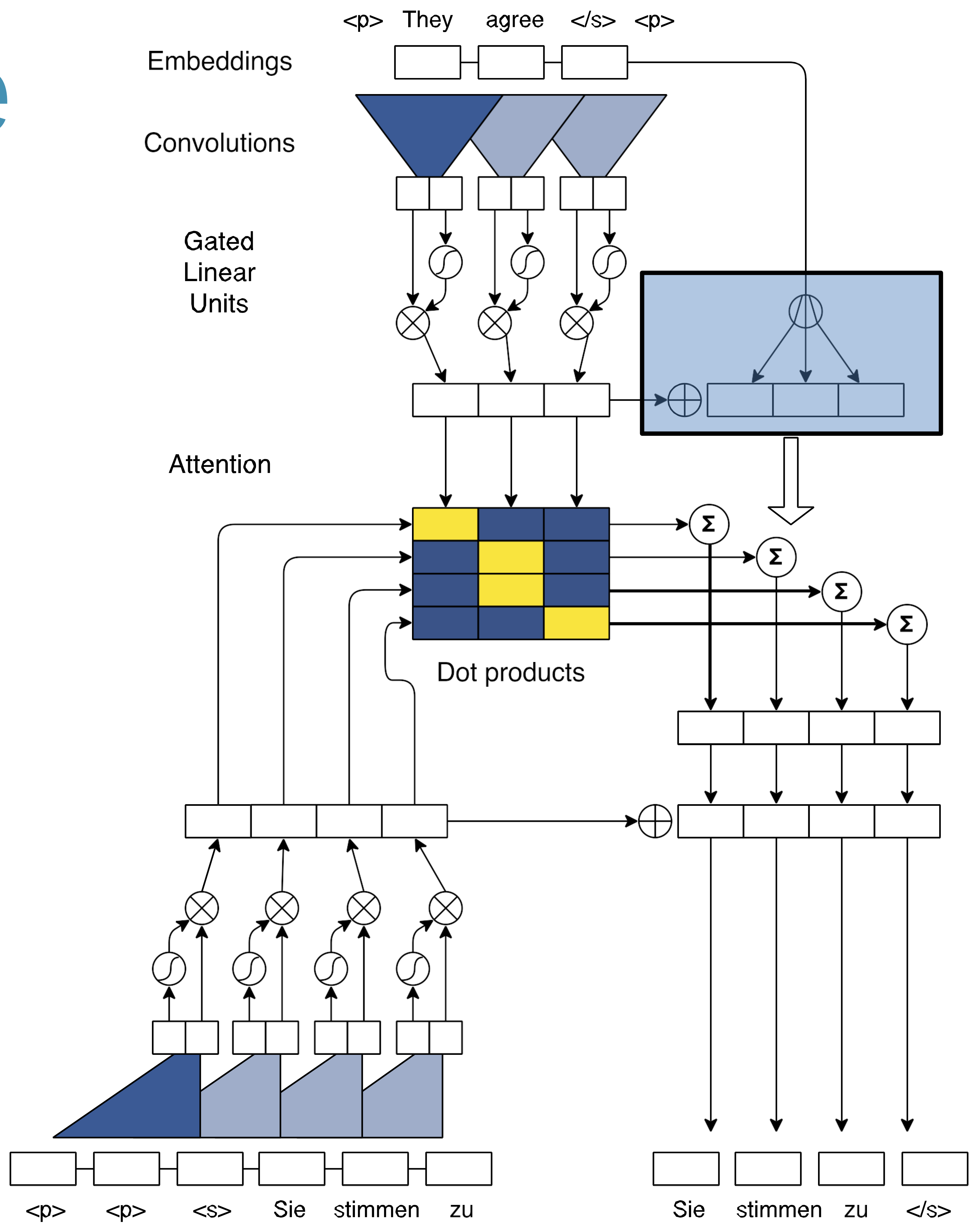


Conv S2S Architecture

Encoder

Residual Connection 적용
(He et al., 2016)

$$h_i^l = v([A\ B]) + h_i^{l-1}$$



Conv S2S Architecture

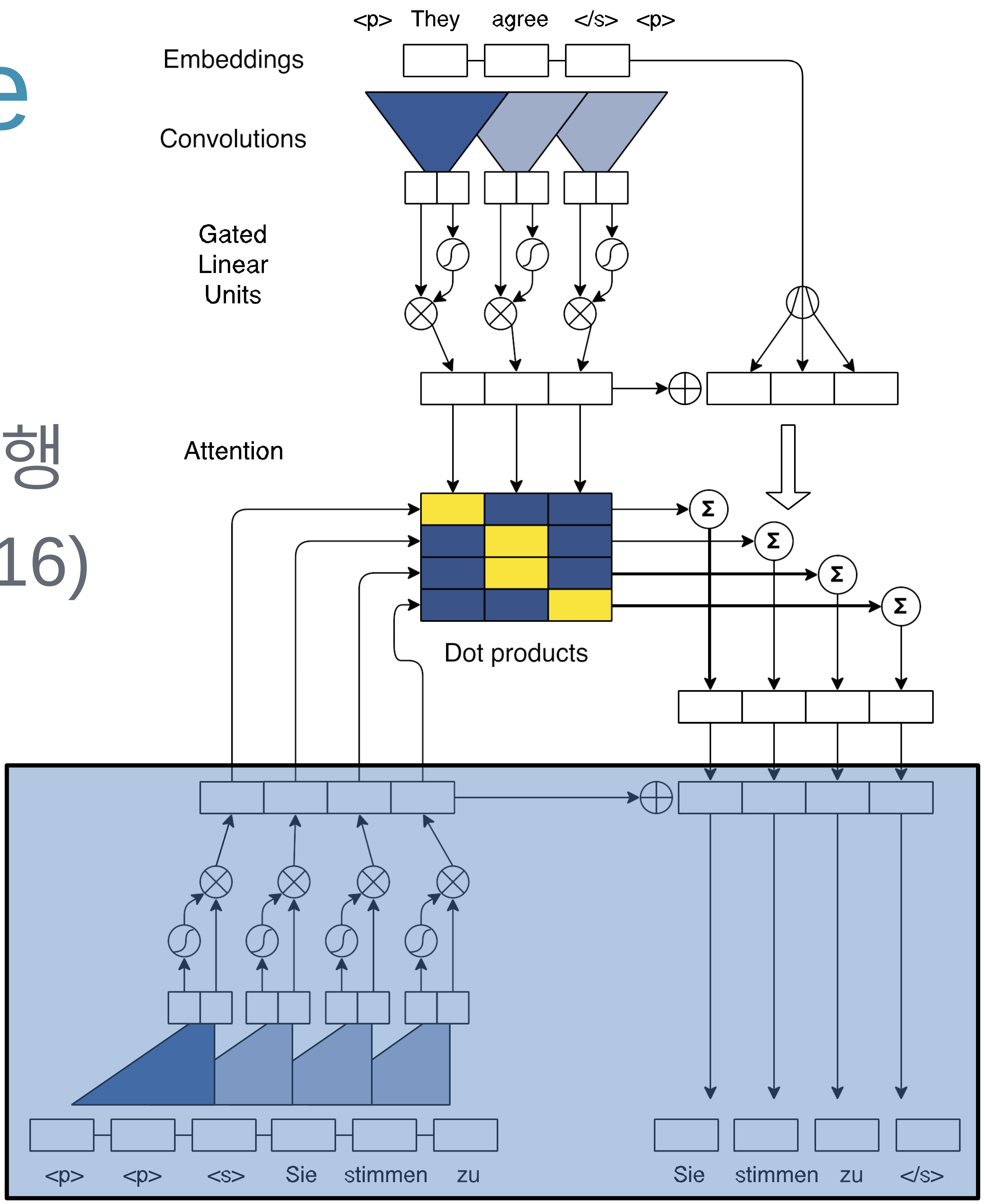
Decoder

One dimensional convolution 연산 수행

GLU function 적용(Dauphin et al., 2016)

Decoder는 input 양쪽에 k-1 padding

output에서 k개 element를 제거



Conv S2S Architecture

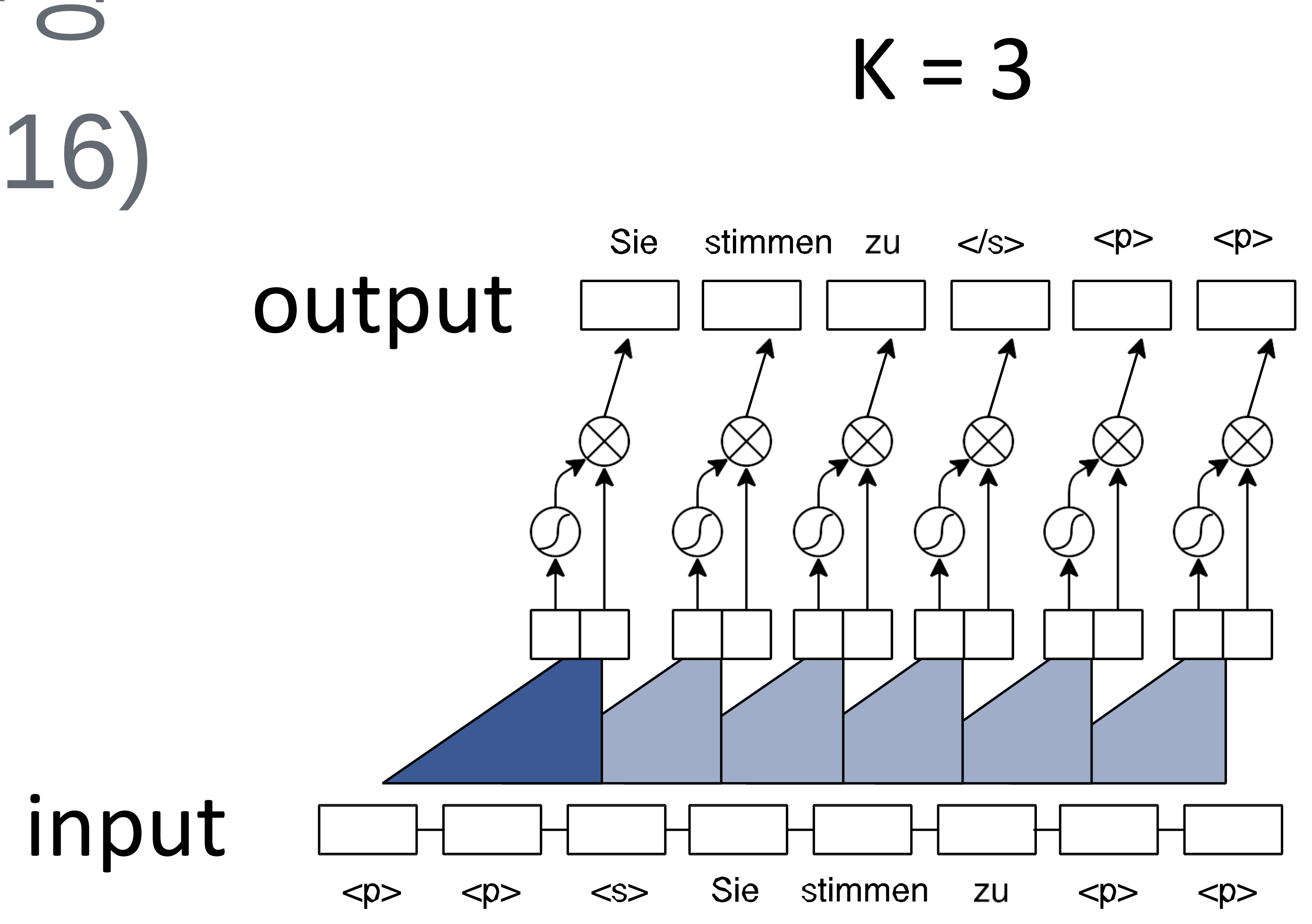
Decoder

One dimensional convolution 연산 수행

GLU function 적용(Dauphin et al., 2016)

Decoder는 input 양쪽에 k-1 padding

output에서 k개 element를 제거



Conv S2S Architecture

Decoder

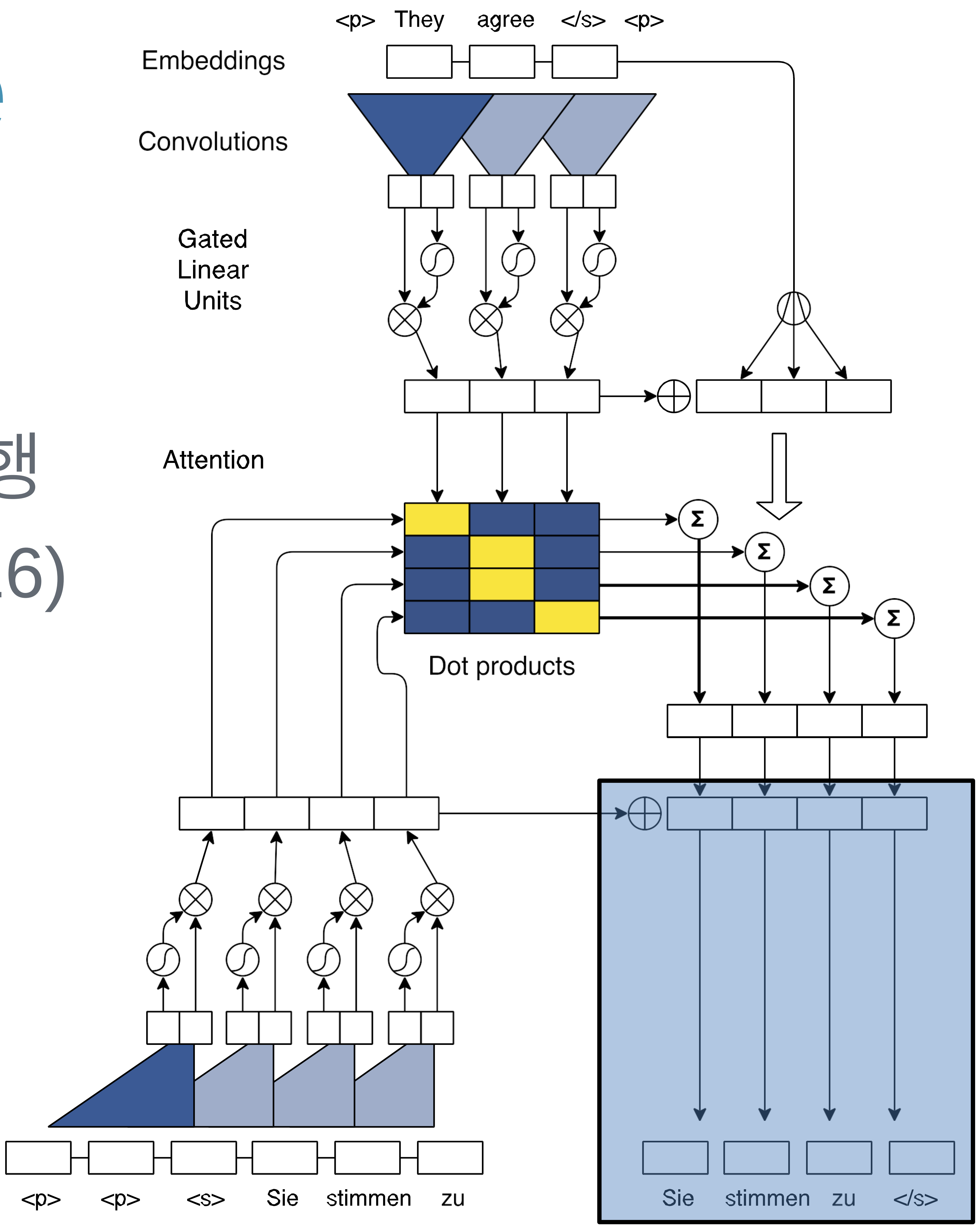
One dimensional convolution 연산 수행

GLU function 적용(Dauphin et al., 2016)

Decoder는 input 양쪽에 k-1 padding

output에서 k개 element를 제거

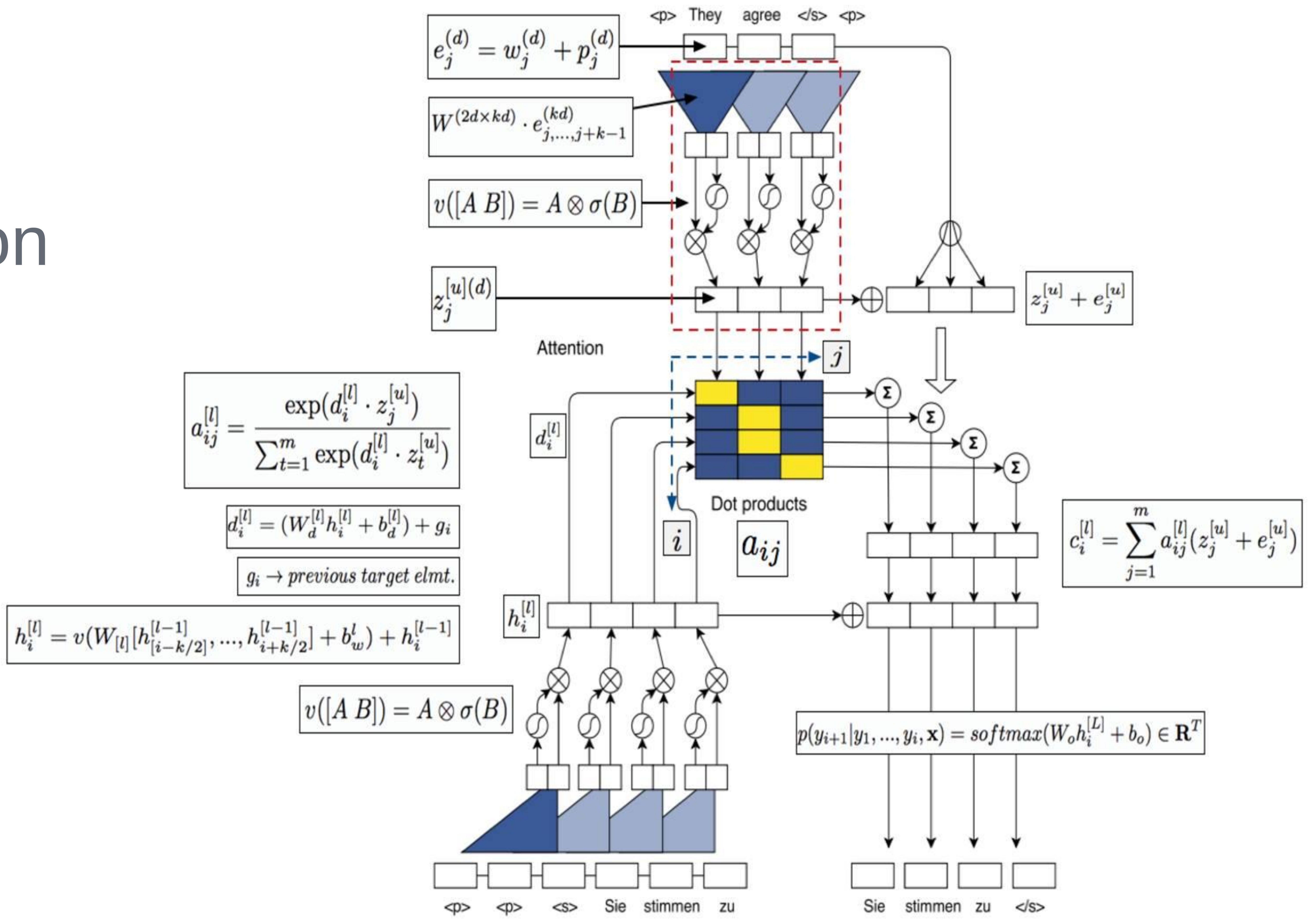
$$P(y_{i+1}|y_1, \ldots, y_i, x) = softmax(W_o h_i^L + b_0)$$



Conv S2S Architecture

Attention

Check notation position

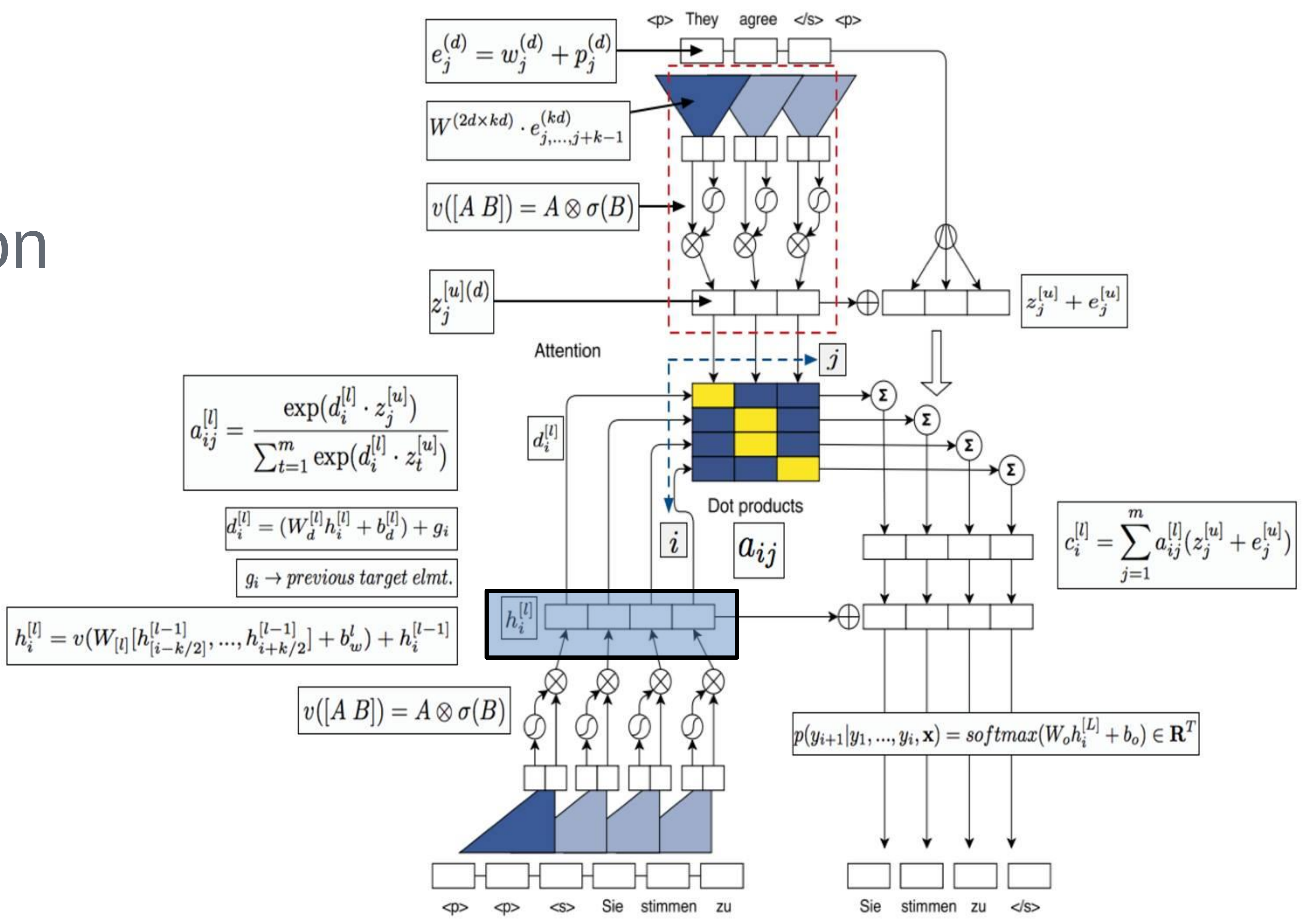


<p> They agree </s> <p>
$e_j^{(d)} = w_j^{(d)} + p_j^{(d)}$
$W^{(2d\times kd)} \cdot e^{(kd)}_{j,\dots,j+k-1}$
$v([A\ B]) = A \otimes \sigma(B)$
$z_j^{[u](d)}$
$z_j^{[u]} + e_j^{[u]}$
Attention
j
$a_{ij}^{[l]} = \frac{\exp(d_i^{[l]} \cdot z_j^{[u]})}{\sum_{t=1}^m \exp(d_i^{[l]} \cdot z_t^{[u]})}$
$d_i^{[l]}$
$d_i^{[l]} = (W_d^{[l]} h_i^{[l]} + b_d^{[l]}) + g_i$
$g_i \rightarrow previous\ target\ elmt.$
Dot products
i
a_{ij}
$c_i^{[l]} = \sum_{j=1}^m a_{ij}^{[l]}(z_j^{[u]} + e_j^{[u]})$
$h_i^{[l]}$
$h_i^{[l]} = v(W_{[l]}[h^{[l-1]}_{[i-k/2]}, \dots, h^{[l-1]}_{i+k/2}] + b^l_w) + h_i^{[l-1]}$
$v([A\ B]) = A \otimes \sigma(B)$
$p(y_{i+1}|y_1, \dots, y_i, \mathbf{x}) = softmax(W_o h_i^{[L]} + b_o) \in \mathbf{R}^T$
<p> <p> <s> Sie stimmen zu
Sie stimmen zu </s>

Conv S2S Architecture

Attention

Check notation position

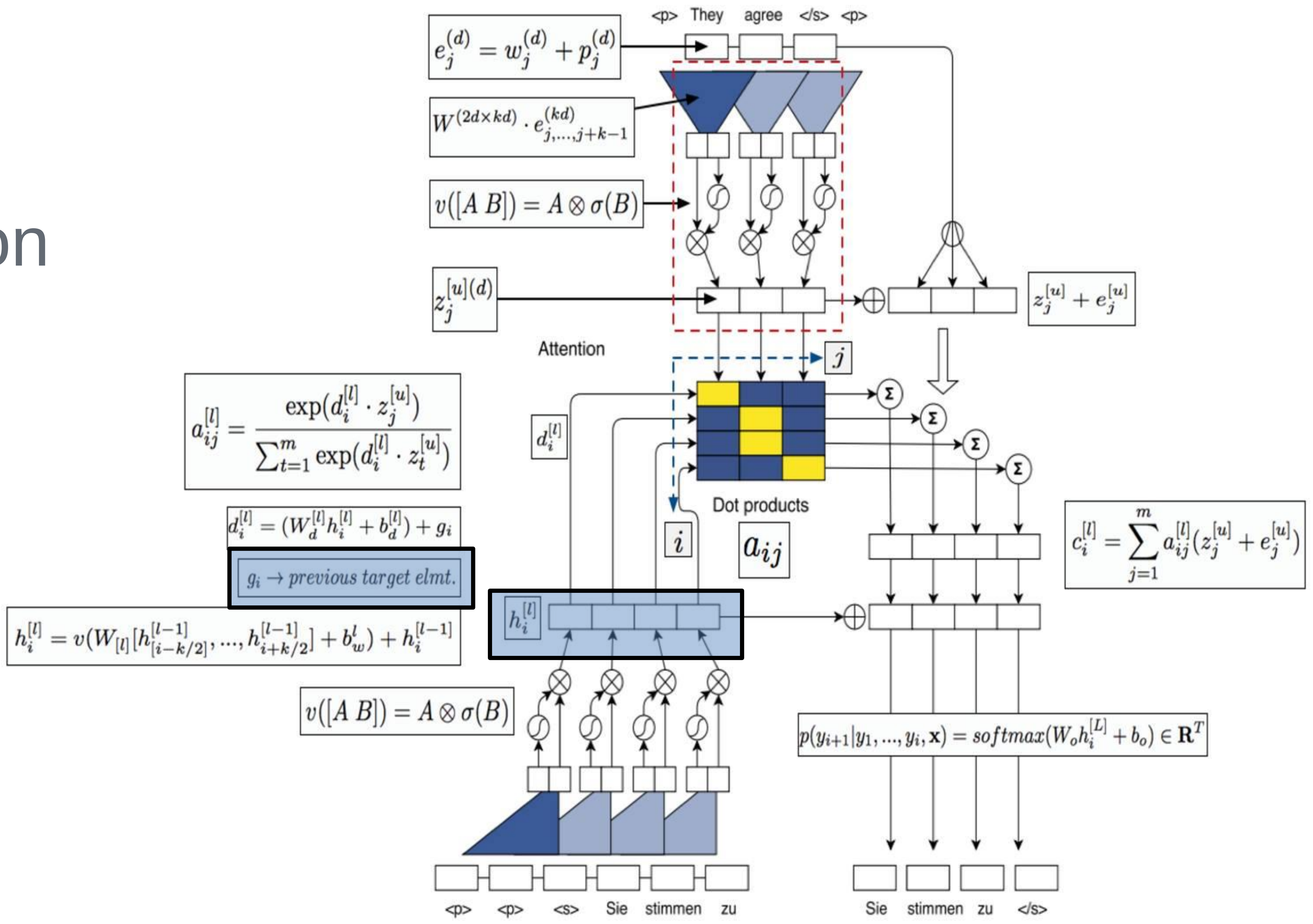


<p> They agree </s> <p>
$e_j^{(d)} = w_j^{(d)} + p_j^{(d)}$
$W^{(2d \times kd)} \cdot e_{j,\ldots,j+k-1}^{(kd)}$
$v([A\ B]) = A \otimes \sigma(B)$
$z_j^{[u](d)}$
$z_j^{[u]} + e_j^{[u]}$
Attention
j
$a_{ij}^{[l]} = \frac{\exp(d_i^{[l]} \cdot z_j^{[u]})}{\sum_{t=1}^{m} \exp(d_i^{[l]} \cdot z_t^{[u]})}$
$d_i^{[l]}$
$d_i^{[l]} = (W_d^{[l]} h_i^{[l]} + b_d^{[l]}) + g_i$
$g_i \rightarrow previous\ target\ elmt.$
Dot products
i
a_{ij}
$c_i^{[l]} = \sum_{j=1}^{m} a_{ij}^{[l]} (z_j^{[u]} + e_j^{[u]})$
$h_i^{[l]}$
$h_i^{[l]} = v(W_{[l]}[h_{[i-k/2]}^{[l-1]}, \ldots, h_{i+k/2}^{[l-1]}] + b_w^l) + h_i^{[l-1]}$
$v([A\ B]) = A \otimes \sigma(B)$
$p(y_{i+1}|y_1, \ldots, y_i, \mathbf{x}) = softmax(W_o h_i^{[L]} + b_o) \in \mathbf{R}^T$
<p> <p> <s> Sie stimmen zu
Sie stimmen zu </s>

Conv S2S Architecture

Attention

Check notation position

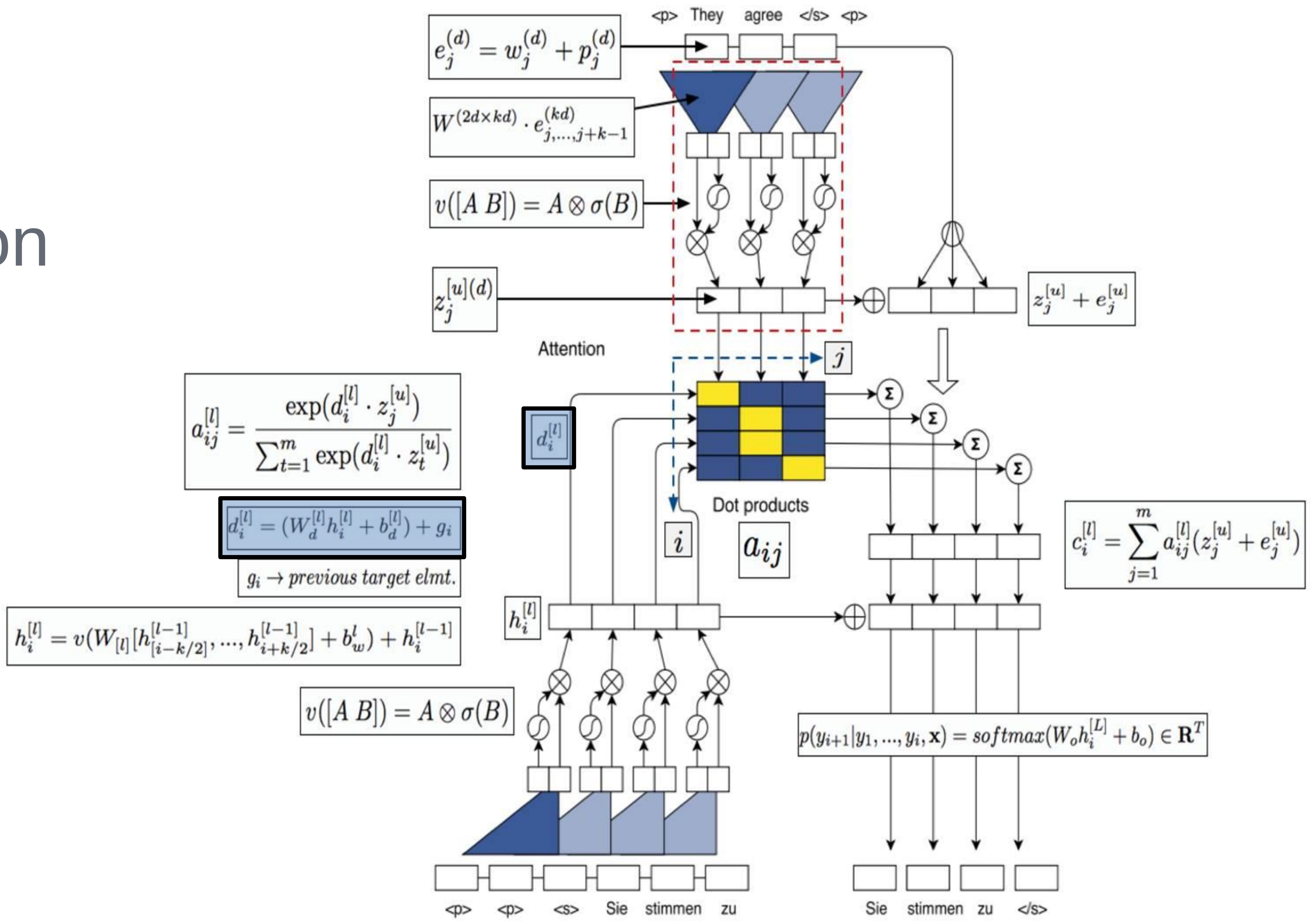


<p> They agree </s> <p>
$e_j^{(d)} = w_j^{(d)} + p_j^{(d)}$
$W^{(2d \times kd)} \cdot e^{(kd)}_{j,\dots,j+k-1}$
$v([A\ B]) = A \otimes \sigma(B)$
$z_j^{[u](d)}$
$z_j^{[u]} + e_j^{[u]}$
Attention
j
$a_{ij}^{[l]} = \frac{\exp(d_i^{[l]} \cdot z_j^{[u]})}{\sum_{t=1}^{m} \exp(d_i^{[l]} \cdot z_t^{[u]})}$
$d_i^{[l]}$
Dot products
i
a_{ij}
$d_i^{[l]} = (W_d^{[l]} h_i^{[l]} + b_d^{[l]}) + g_i$
$g_i \rightarrow$ previous target elmt.
$c_i^{[l]} = \sum_{j=1}^{m} a_{ij}^{[l]} (z_j^{[u]} + e_j^{[u]})$
$h_i^{[l]}$
$h_i^{[l]} = v(W_{[l]}[h^{[l-1]}_{[i-k/2]}, \dots, h^{[l-1]}_{i+k/2}] + b^l_w) + h_i^{[l-1]}$
$v([A\ B]) = A \otimes \sigma(B)$
$p(y_{i+1}|y_1, \dots, y_i, \mathbf{x}) = softmax(W_o h_i^{[L]} + b_o) \in \mathbf{R}^T$
<p> <p> <s> Sie stimmen zu
Sie stimmen zu </s>

Conv S2S Architecture

Attention

Check notation position

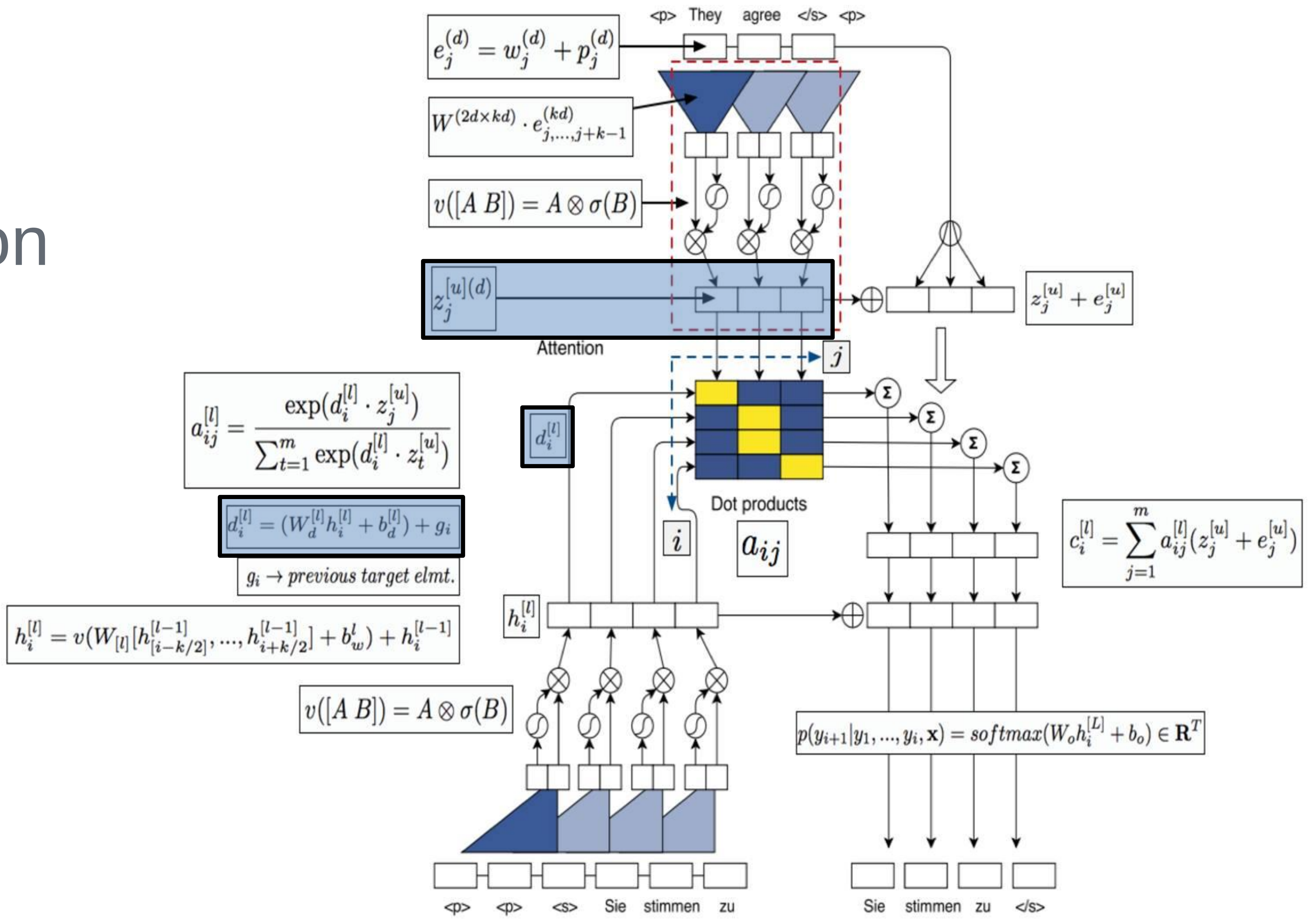


<p> They agree </s> <p>
$e_j^{(d)} = w_j^{(d)} + p_j^{(d)}$
$W^{(2d \times kd)} \cdot e^{(kd)}_{j,\dots,j+k-1}$
$v([A\ B]) = A \otimes \sigma(B)$
$z_j^{[u](d)}$
$z_j^{[u]} + e_j^{[u]}$
Attention
j
$a_{ij}^{[l]} = \frac{\exp(d_i^{[l]} \cdot z_j^{[u]})}{\sum_{t=1}^{m} \exp(d_i^{[l]} \cdot z_t^{[u]})}$
$d_i^{[l]}$
$d_i^{[l]} = (W_d^{[l]} h_i^{[l]} + b_d^{[l]}) + g_i$
$g_i \rightarrow previous\ target\ elmt.$
Dot products
i
a_{ij}
$c_i^{[l]} = \sum_{j=1}^{m} a_{ij}^{[l]} (z_j^{[u]} + e_j^{[u]})$
$h_i^{[l]}$
$h_i^{[l]} = v(W_{[l]}[h^{[l-1]}_{[i-k/2]}, \dots, h^{[l-1]}_{i+k/2}] + b_w^l) + h_i^{[l-1]}$
$v([A\ B]) = A \otimes \sigma(B)$
$p(y_{i+1}|y_1, \dots, y_i, \mathbf{x}) = softmax(W_o h_i^{[L]} + b_o) \in \mathbf{R}^T$
<p> <p> <s> Sie stimmen zu
Sie stimmen zu </s>

Conv S2S Architecture

Attention

Check notation position

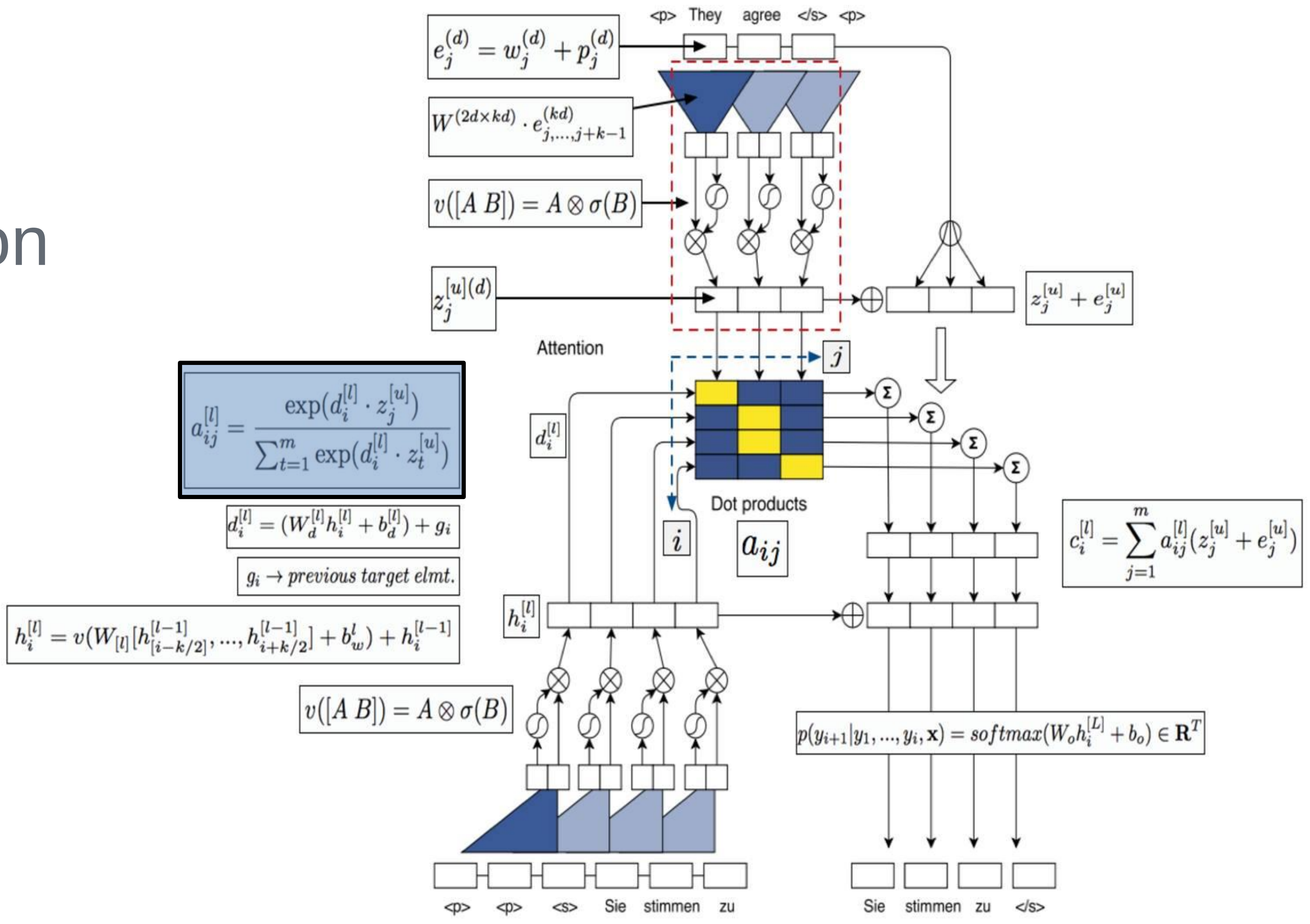


<p> They agree </s> <p>
$e_j^{(d)} = w_j^{(d)} + p_j^{(d)}$
$W^{(2d\times kd)} \cdot e_{j,\dots,j+k-1}^{(kd)}$
$v([A\ B]) = A \otimes \sigma(B)$
$z_j^{[u](d)}$
$z_j^{[u]} + e_j^{[u]}$
Attention
j
$a_{ij}^{[l]} = \frac{\exp(d_i^{[l]} \cdot z_j^{[u]})}{\sum_{t=1}^{m} \exp(d_i^{[l]} \cdot z_t^{[u]})}$
$d_i^{[l]}$
$d_i^{[l]} = (W_d^{[l]} h_i^{[l]} + b_d^{[l]}) + g_i$
$g_i \rightarrow previous\ target\ elmt.$
Dot products
i
a_{ij}
$c_i^{[l]} = \sum_{j=1}^{m} a_{ij}^{[l]}(z_j^{[u]} + e_j^{[u]})$
$h_i^{[l]}$
$h_i^{[l]} = v(W_{[l]}[h_{[i-k/2]}^{[l-1]}, \dots, h_{i+k/2}^{[l-1]}] + b_w^l) + h_i^{[l-1]}$
$v([A\ B]) = A \otimes \sigma(B)$
$p(y_{i+1}|y_1, \dots, y_i, \mathbf{x}) = softmax(W_o h_i^{[L]} + b_o) \in \mathbf{R}^T$
<p> <p> <s> Sie stimmen zu
Sie stimmen zu </s>

Conv S2S Architecture

Attention

Check notation position

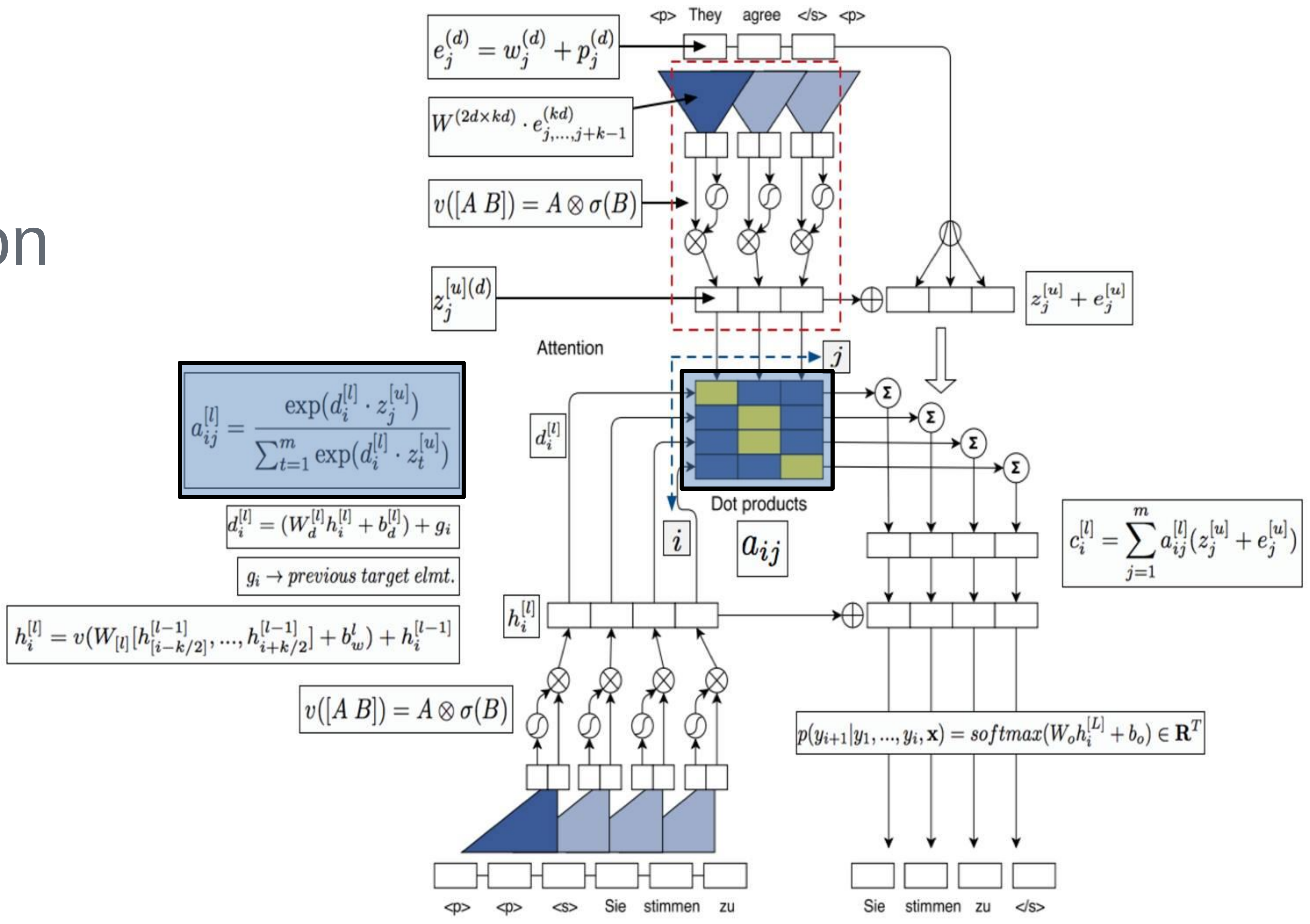


<p> They agree </s> <p>
$e_j^{(d)} = w_j^{(d)} + p_j^{(d)}$
$W^{(2d\times kd)} \cdot e^{(kd)}_{j,\dots,j+k-1}$
$v([A\ B]) = A \otimes \sigma(B)$
$z_j^{[u](d)}$
$z_j^{[u]} + e_j^{[u]}$
Attention
j
$a_{ij}^{[l]} = \frac{\exp(d_i^{[l]} \cdot z_j^{[u]})}{\sum_{t=1}^{m} \exp(d_i^{[l]} \cdot z_t^{[u]})}$
$d_i^{[l]}$
Dot products
i
a_{ij}
$c_i^{[l]} = \sum_{j=1}^{m} a_{ij}^{[l]}(z_j^{[u]} + e_j^{[u]})$
$d_i^{[l]} = (W_d^{[l]} h_i^{[l]} + b_d^{[l]}) + g_i$
$g_i \rightarrow previous\ target\ elmt.$
$h_i^{[l]} = v(W_{[l]}[h^{[l-1]}_{[i-k/2]}, \dots, h^{[l-1]}_{i+k/2}] + b_w^{l}) + h_i^{[l-1]}$
$h_i^{[l]}$
$v([A\ B]) = A \otimes \sigma(B)$
$p(y_{i+1}|y_1,\dots,y_i,\mathbf{x}) = softmax(W_o h_i^{[L]} + b_o) \in \mathbf{R}^T$
<p> <p> <s> Sie stimmen zu
Sie stimmen zu </s>

Conv S2S Architecture

Attention

Check notation position

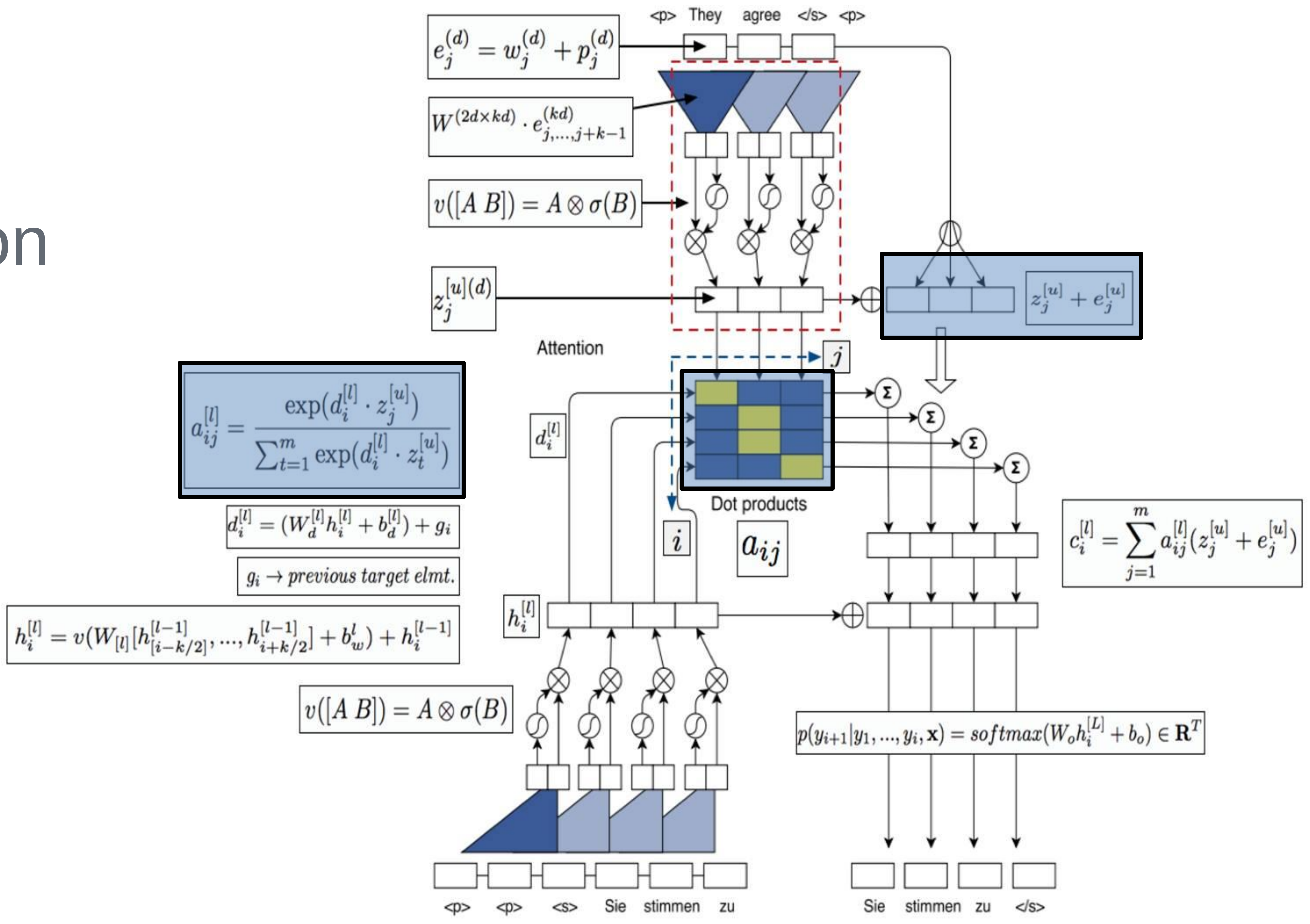


<p> They agree </s> <p>
$e_j^{(d)} = w_j^{(d)} + p_j^{(d)}$
$W^{(2d\times kd)} \cdot e^{(kd)}_{j,\dots,j+k-1}$
$v([A\ B]) = A \otimes \sigma(B)$
$z_j^{[u](d)}$
$z_j^{[u]} + e_j^{[u]}$
Attention
j
$a_{ij}^{[l]} = \frac{\exp(d_i^{[l]} \cdot z_j^{[u]})}{\sum_{t=1}^{m} \exp(d_i^{[l]} \cdot z_t^{[u]})}$
$d_i^{[l]}$
Dot products
i
a_{ij}
$d_i^{[l]} = (W_d^{[l]} h_i^{[l]} + b_d^{[l]}) + g_i$
$g_i \rightarrow previous\ target\ elmt.$
$c_i^{[l]} = \sum_{j=1}^{m} a_{ij}^{[l]}(z_j^{[u]} + e_j^{[u]})$
$h_i^{[l]} = v(W_{[l]}[h^{[l-1]}_{[i-k/2]}, \dots, h^{[l-1]}_{i+k/2}] + b_w^l) + h_i^{[l-1]}$
$h_i^{[l]}$
$v([A\ B]) = A \otimes \sigma(B)$
$p(y_{i+1}|y_1, \dots, y_i, \mathbf{x}) = softmax(W_o h_i^{[L]} + b_o) \in \mathbf{R}^T$
<p> <p> <s> Sie stimmen zu
Sie stimmen zu </s>

Conv S2S Architecture

Attention

Check notation position

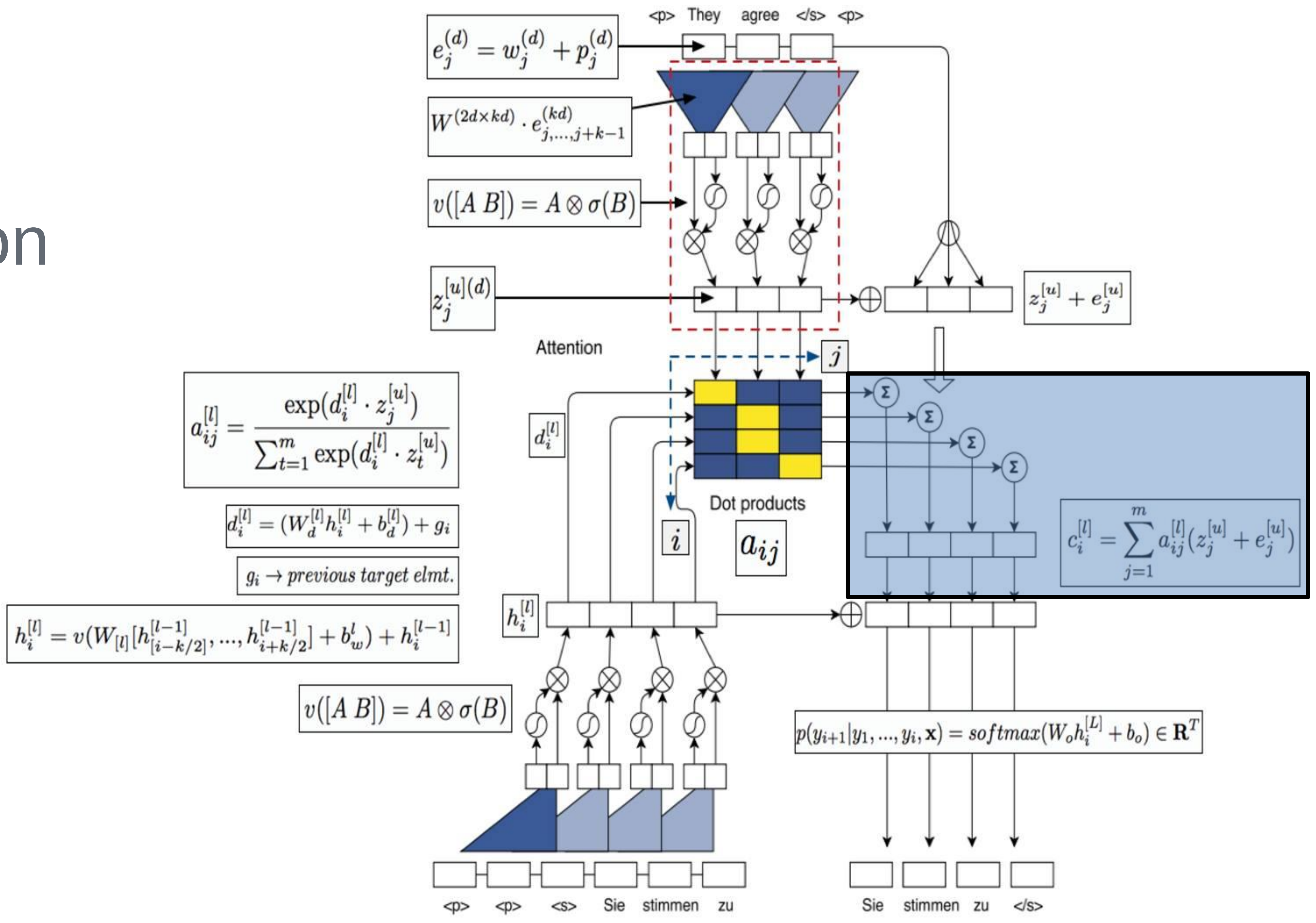


<p> They agree </s> <p>
$e_j^{(d)} = w_j^{(d)} + p_j^{(d)}$
$W^{(2d\times kd)} \cdot e^{(kd)}_{j,\dots,j+k-1}$
$v([A\ B]) = A \otimes \sigma(B)$
$z_j^{[u](d)}$
$z_j^{[u]} + e_j^{[u]}$
Attention
j
$a_{ij}^{[l]} = \frac{\exp(d_i^{[l]} \cdot z_j^{[u]})}{\sum_{t=1}^{m} \exp(d_i^{[l]} \cdot z_t^{[u]})}$
$d_i^{[l]}$
Dot products
i
a_{ij}
$d_i^{[l]} = (W_d^{[l]} h_i^{[l]} + b_d^{[l]}) + g_i$
$g_i \rightarrow previous\ target\ elmt.$
$c_i^{[l]} = \sum_{j=1}^{m} a_{ij}^{[l]} (z_j^{[u]} + e_j^{[u]})$
$h_i^{[l]}$
$h_i^{[l]} = v(W_{[l]}[h^{[l-1]}_{[i-k/2]}, \dots, h^{[l-1]}_{i+k/2}] + b_w^l) + h_i^{[l-1]}$
$v([A\ B]) = A \otimes \sigma(B)$
$p(y_{i+1}|y_1, \dots, y_i, \mathbf{x}) = softmax(W_o h_i^{[L]} + b_o) \in \mathbf{R}^T$
<p> <p> <s> Sie stimmen zu
Sie stimmen zu </s>

Conv S2S Architecture

Attention

Check notation position

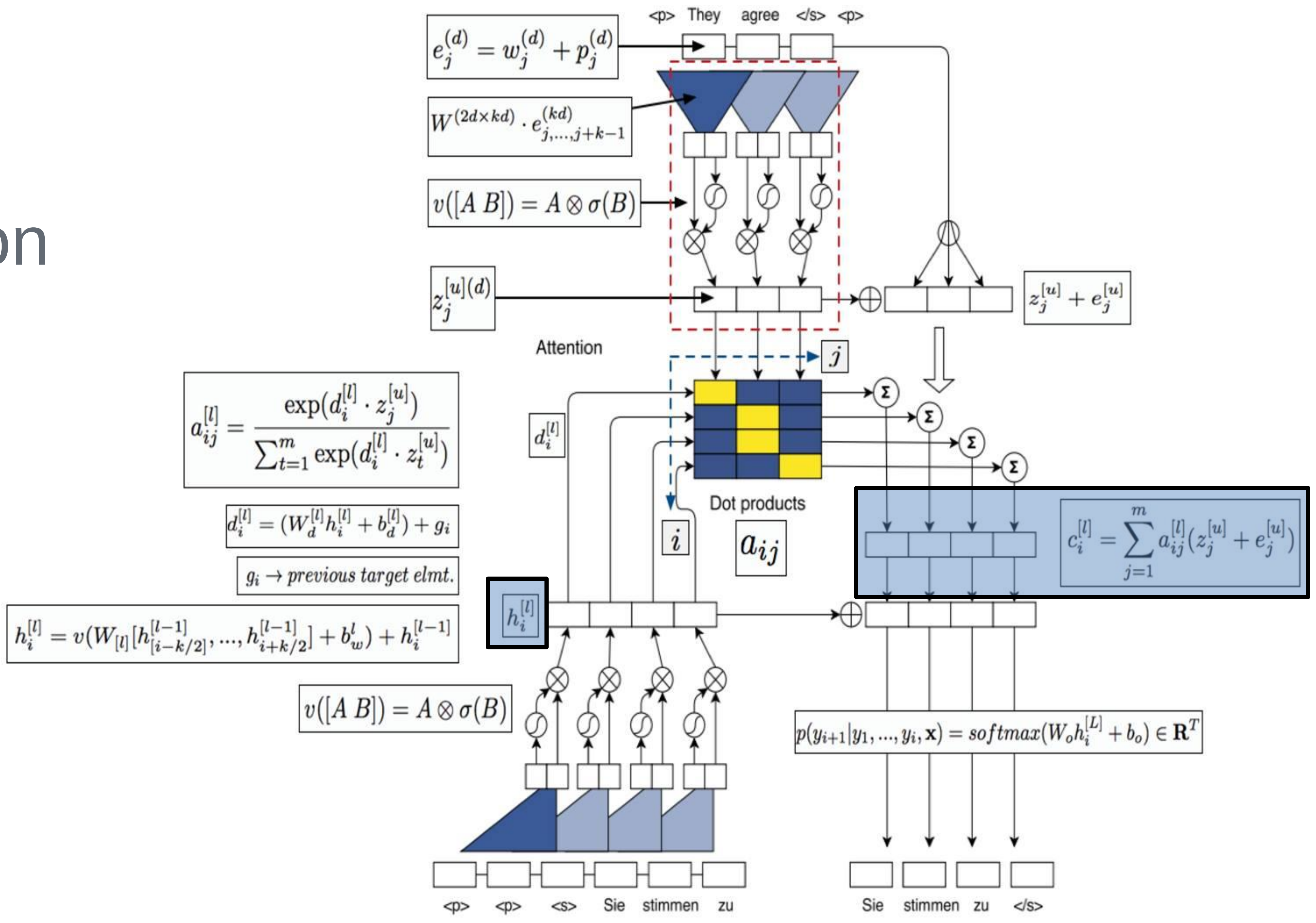


<p> They agree </s> <p>
$e_j^{(d)} = w_j^{(d)} + p_j^{(d)}$
$W^{(2d\times kd)} \cdot e^{(kd)}_{j,\dots,j+k-1}$
$v([A\ B]) = A \otimes \sigma(B)$
$z_j^{[u](d)}$
$z_j^{[u]} + e_j^{[u]}$
Attention
j
$a_{ij}^{[l]} = \frac{\exp(d_i^{[l]} \cdot z_j^{[u]})}{\sum_{t=1}^{m} \exp(d_i^{[l]} \cdot z_t^{[u]})}$
$d_i^{[l]}$
Σ
$c_i^{[l]} = \sum_{j=1}^{m} a_{ij}^{[l]}(z_j^{[u]} + e_j^{[u]})$
$d_i^{[l]} = (W_d^{[l]} h_i^{[l]} + b_d^{[l]}) + g_i$
Dot products
i
a_{ij}
$g_i \rightarrow previous\ target\ elmt.$
$h_i^{[l]}$
$h_i^{[l]} = v(W_{[l]}[h^{[l-1]}_{[i-k/2]}, \dots, h^{[l-1]}_{i+k/2}] + b_w^{l}) + h_i^{[l-1]}$
$v([A\ B]) = A \otimes \sigma(B)$
$p(y_{i+1}|y_1, \dots, y_i, \mathbf{x}) = softmax(W_o h_i^{[L]} + b_o) \in \mathbf{R}^T$
<p> <p> <s> Sie stimmen zu
Sie stimmen zu </s>

Conv S2S Architecture

Attention

Check notation position

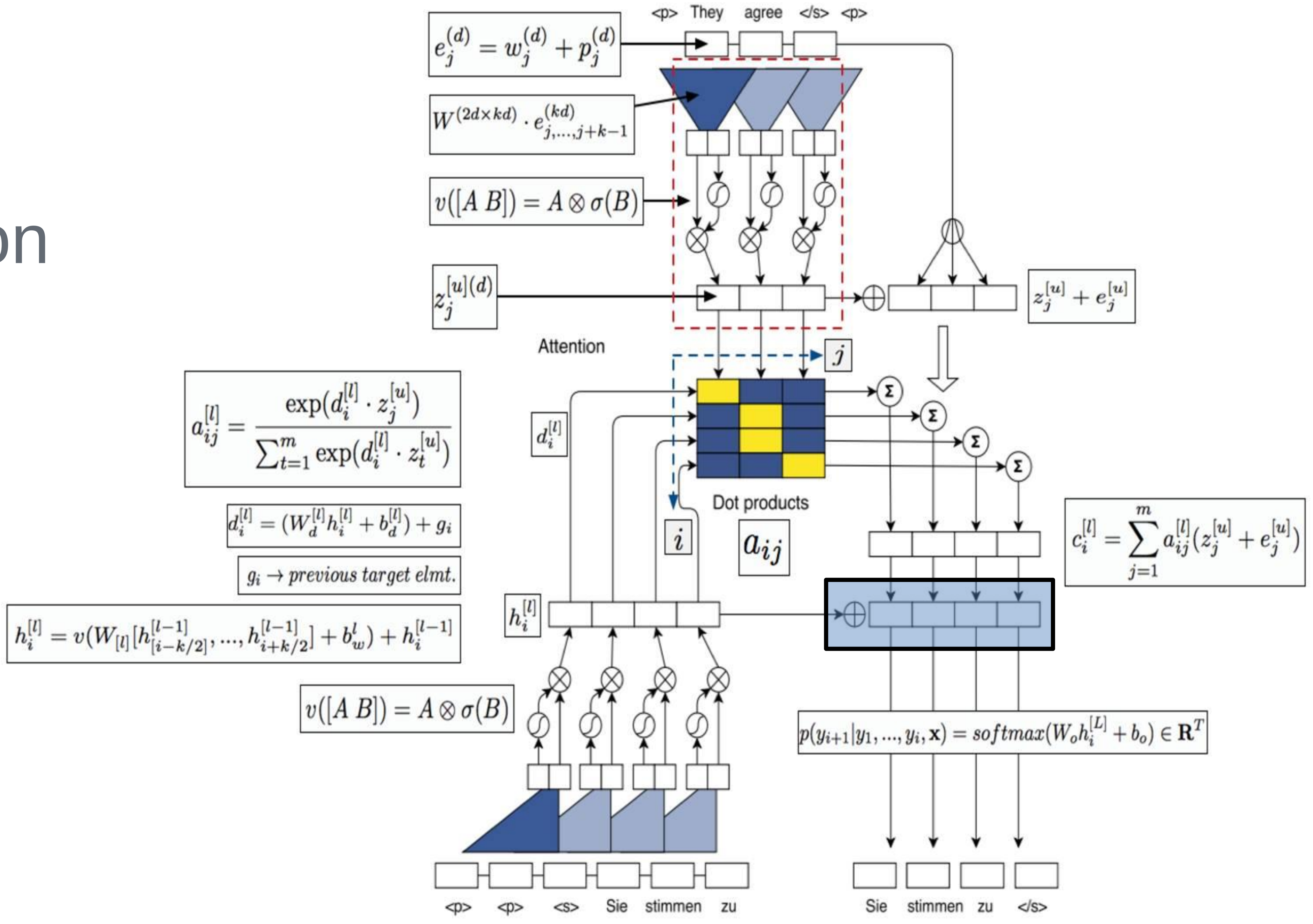


<p> They agree </s> <p>
$e_j^{(d)} = w_j^{(d)} + p_j^{(d)}$
$W^{(2d\times kd)} \cdot e^{(kd)}_{j,\dots,j+k-1}$
$v([A\ B]) = A \otimes \sigma(B)$
$z_j^{[u](d)}$
$z_j^{[u]} + e_j^{[u]}$
Attention
j
$a_{ij}^{[l]} = \frac{\exp(d_i^{[l]} \cdot z_j^{[u]})}{\sum_{t=1}^{m} \exp(d_i^{[l]} \cdot z_t^{[u]})}$
$d_i^{[l]}$
$d_i^{[l]} = (W_d^{[l]} h_i^{[l]} + b_d^{[l]}) + g_i$
$g_i \rightarrow$ previous target elmt.
Dot products
i
a_{ij}
$c_i^{[l]} = \sum_{j=1}^{m} a_{ij}^{[l]} (z_j^{[u]} + e_j^{[u]})$
$h_i^{[l]}$
$h_i^{[l]} = v(W_{[l]}[h^{[l-1]}_{[i-k/2]}, \dots, h^{[l-1]}_{i+k/2}] + b^l_w) + h_i^{[l-1]}$
$v([A\ B]) = A \otimes \sigma(B)$
$p(y_{i+1}|y_1, \dots, y_i, \mathbf{x}) = softmax(W_o h_i^{[L]} + b_o) \in \mathbf{R}^T$
<p> <p> <s> Sie stimmen zu
Sie stimmen zu </s>

Conv S2S Architecture

Attention

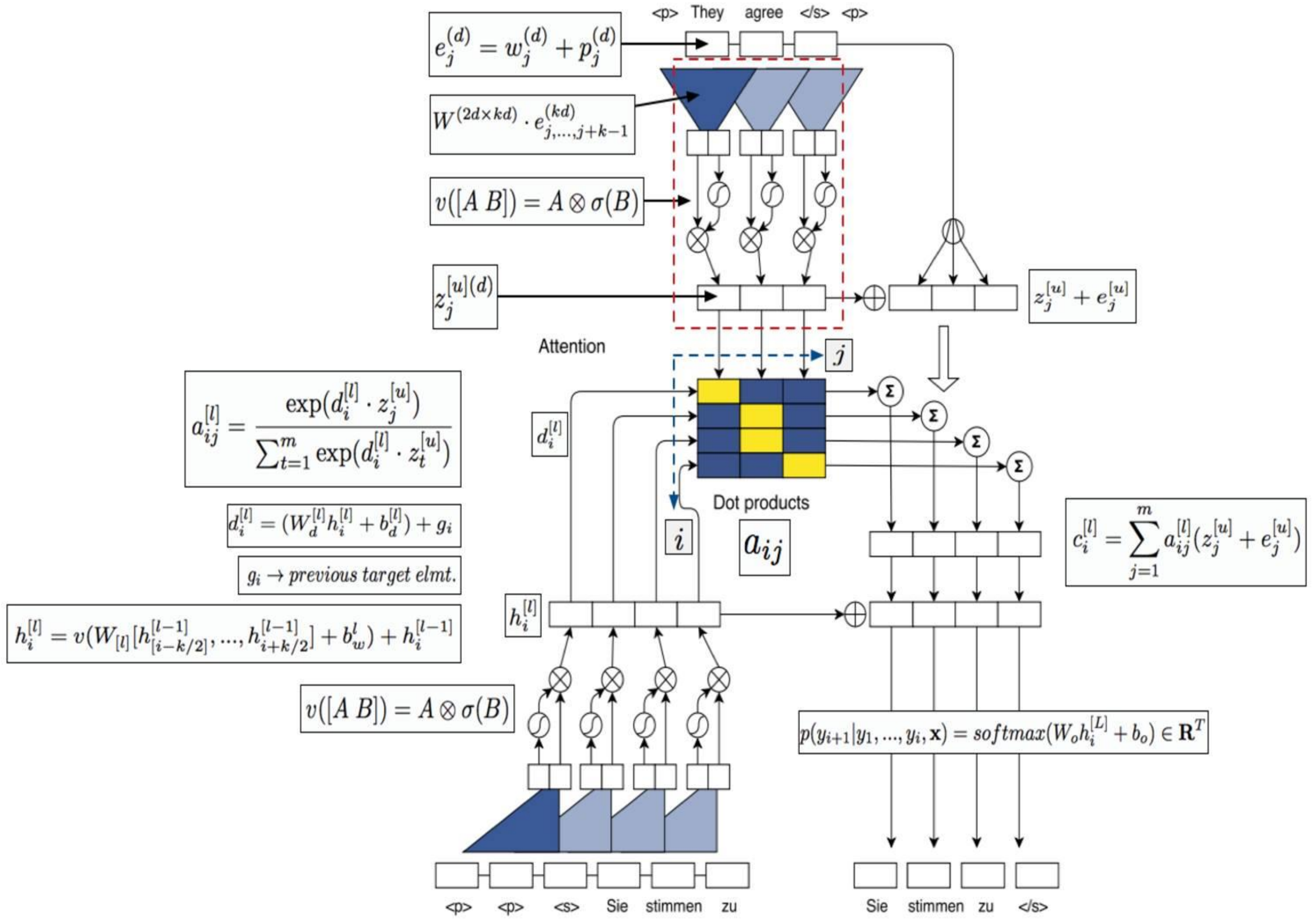
Check notation position



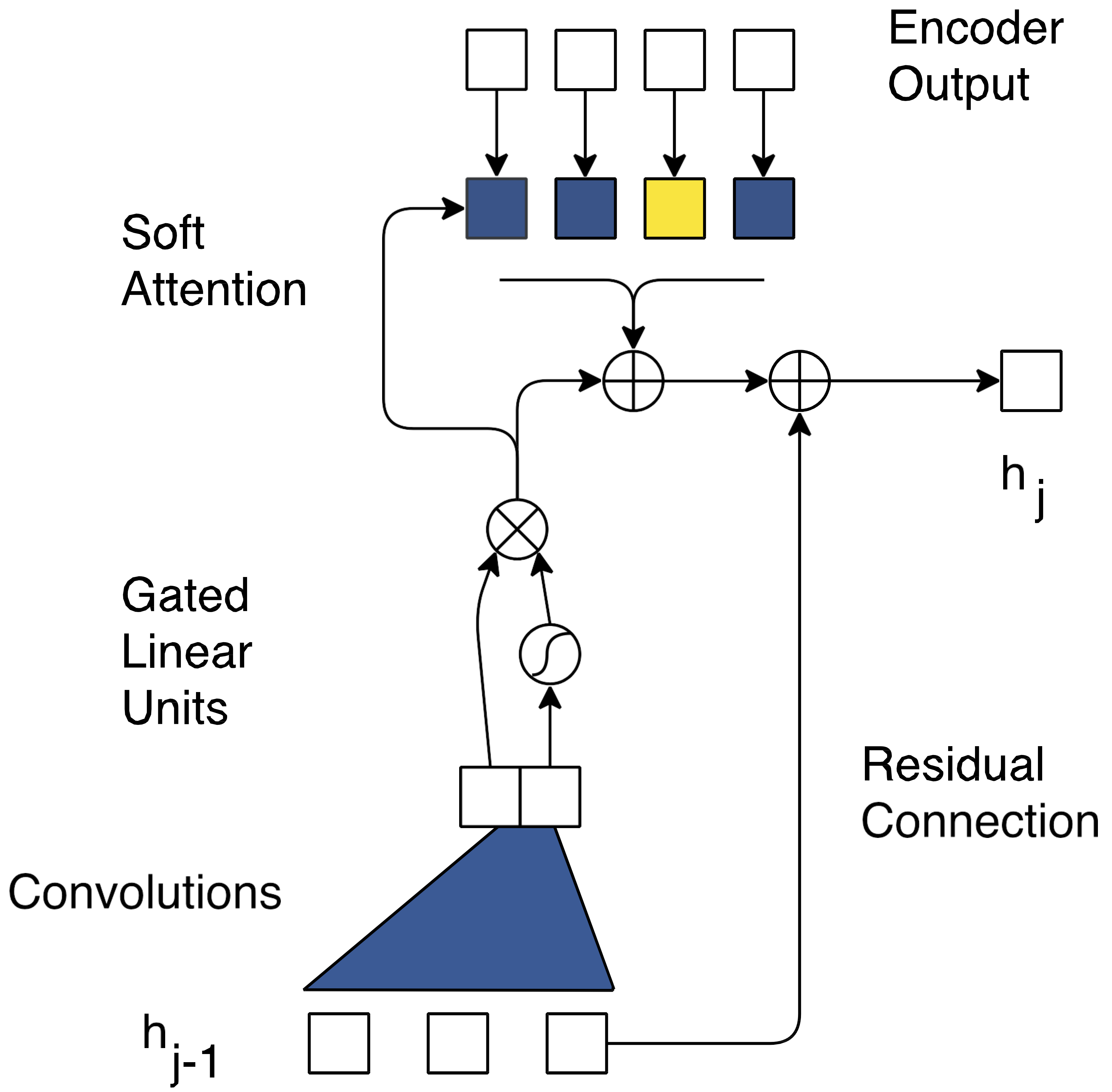
<p> They agree </s> <p>
$e_j^{(d)} = w_j^{(d)} + p_j^{(d)}$
$W^{(2d\times kd)} \cdot e_{j,\ldots,j+k-1}^{(kd)}$
$v([A\ B]) = A \otimes \sigma(B)$
$z_j^{[u](d)}$
$z_j^{[u]} + e_j^{[u]}$
Attention
j
$a_{ij}^{[l]} = \frac{\exp(d_i^{[l]} \cdot z_j^{[u]})}{\sum_{t=1}^{m} \exp(d_i^{[l]} \cdot z_t^{[u]})}$
$d_i^{[l]}$
$d_i^{[l]} = (W_d^{[l]} h_i^{[l]} + b_d^{[l]}) + g_i$
$g_i \rightarrow previous\ target\ elmt.$
Dot products
i
a_{ij}
$c_i^{[l]} = \sum_{j=1}^{m} a_{ij}^{[l]}(z_j^{[u]} + e_j^{[u]})$
$h_i^{[l]}$
$h_i^{[l]} = v(W_{[l]}[h_{[i-k/2]}^{[l-1]}, \ldots, h_{i+k/2}^{[l-1]}] + b_w^l) + h_i^{[l-1]}$
$v([A\ B]) = A \otimes \sigma(B)$
$p(y_{i+1}|y_1, \ldots, y_i, \mathbf{x}) = softmax(W_o h_i^{[L]} + b_o) \in \mathbf{R}^T$
<p> <p> <s> Sie stimmen zu
Sie stimmen zu </s>

Conv S2S Architecture

Attention with multiple hops

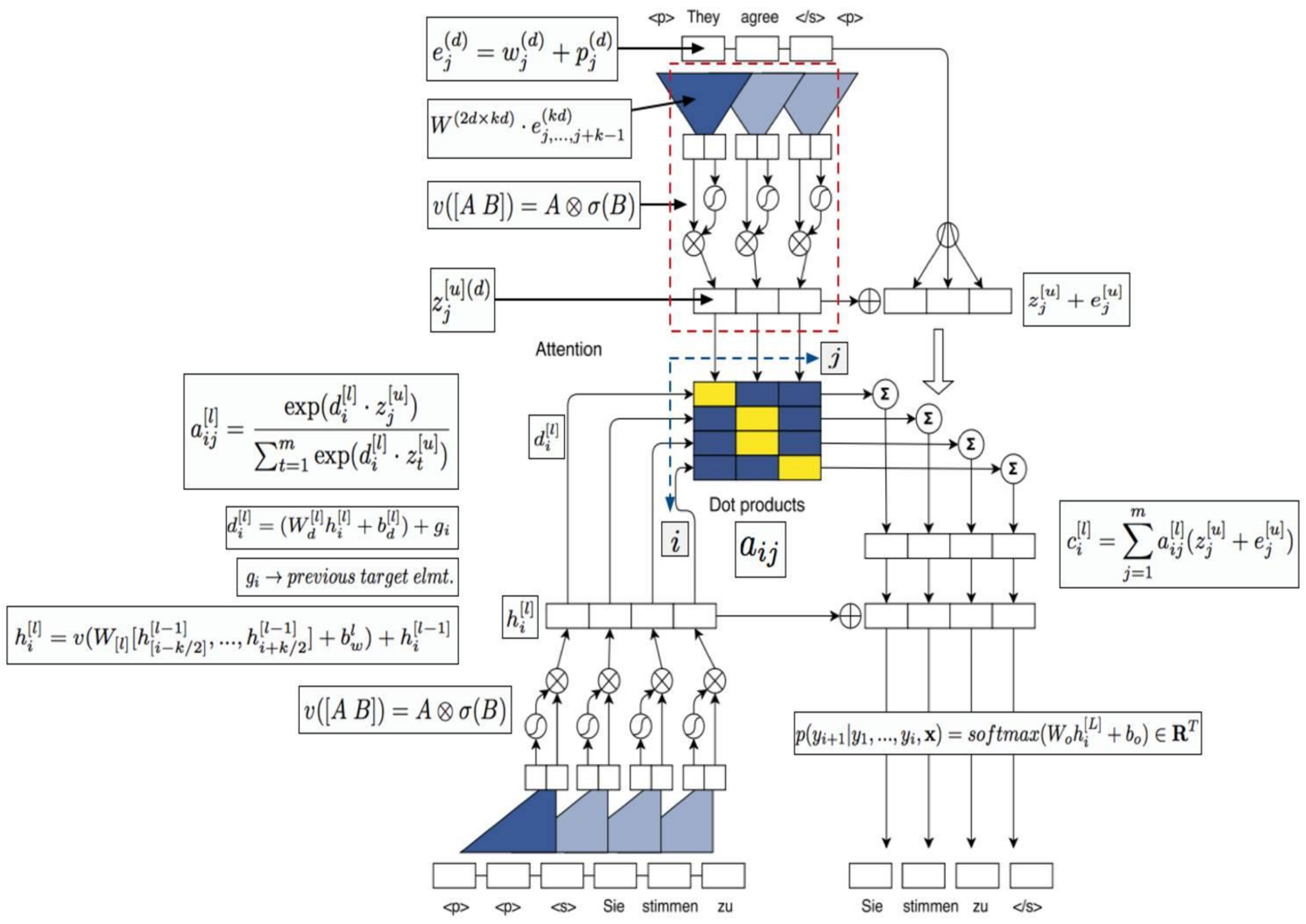


<p> They agree </s> <p>
$e_j^{(d)} = w_j^{(d)} + p_j^{(d)}$
$W^{(2d\times kd)} \cdot e^{(kd)}_{j,\ldots,j+k-1}$
$v([A\ B]) = A \otimes \sigma(B)$
$z_j^{[u](d)}$
$z_j^{[u]} + e_j^{[u]}$
Attention
j
$a_{ij}^{[l]} = \frac{\exp(d_i^{[l]} \cdot z_j^{[u]})}{\sum_{t=1}^{m} \exp(d_i^{[l]} \cdot z_t^{[u]})}$
$d_i^{[l]}$
$d_i^{[l]} = (W_d^{[l]} h_i^{[l]} + b_d^{[l]}) + g_i$
$g_i \rightarrow previous\ target\ elmt.$
i
Dot products
a_{ij}
$c_i^{[l]} = \sum_{j=1}^{m} a_{ij}^{[l]}(z_j^{[u]} + e_j^{[u]})$
$h_i^{[l]}$
$h_i^{[l]} = v(W_{[l]}[h_{[i-k/2]}^{[l-1]}, \ldots, h_{i+k/2}^{[l-1]}] + b_w^l) + h_i^{[l-1]}$
$v([A\ B]) = A \otimes \sigma(B)$
$p(y_{i+1}|y_1, \ldots, y_i, \mathbf{x}) = softmax(W_o h_i^{[L]} + b_o) \in \mathbf{R}^T$
<p> <p> <s> Sie stimmen zu
Sie stimmen zu </s>

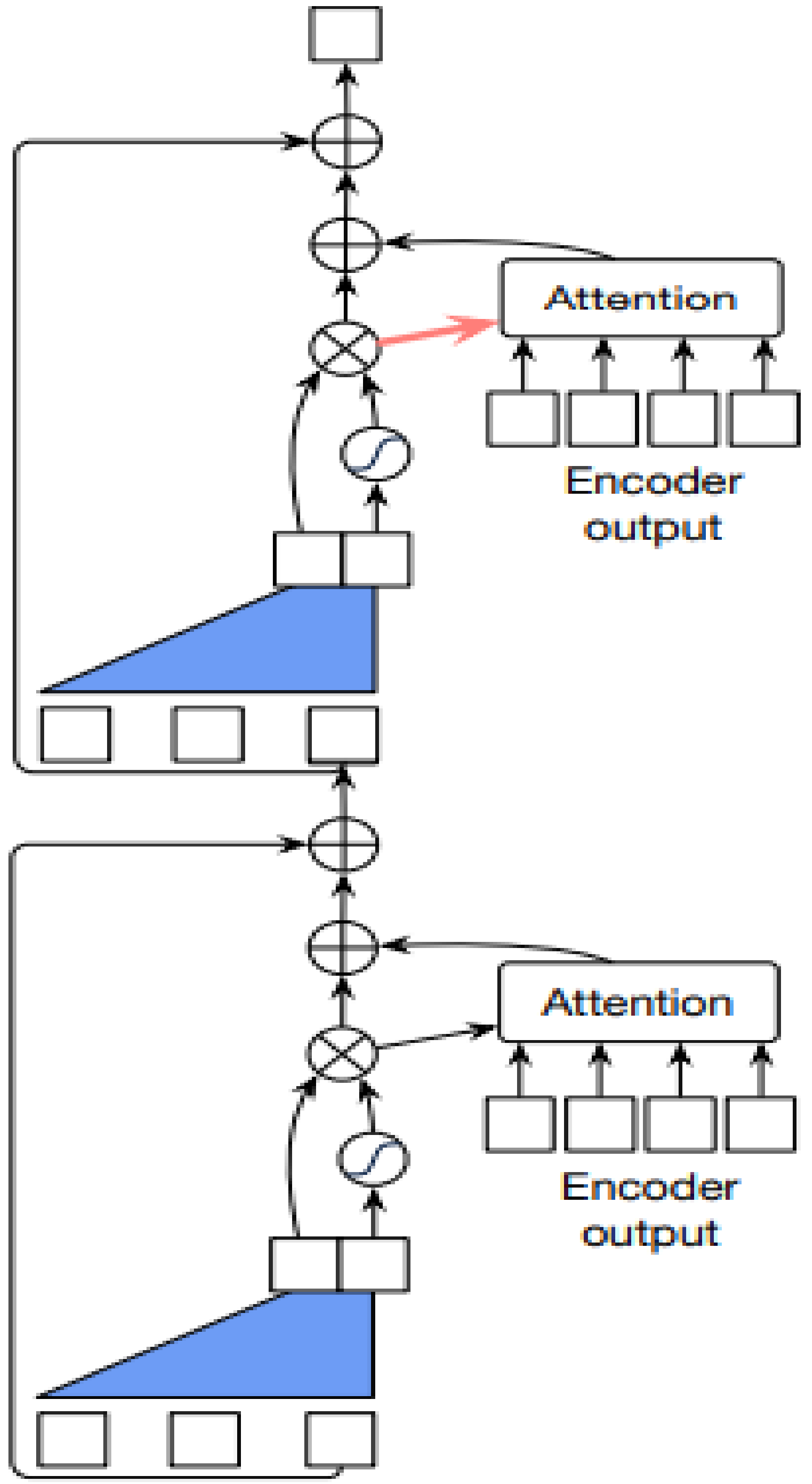


Conv S2S Architecture

Attention with multiple hops



<p> They agree </s> <p>
$e_j^{(d)} = w_j^{(d)} + p_j^{(d)}$
$W^{(2d\times kd)} \cdot e_{j,\dots,j+k-1}^{(kd)}$
$v([A\ B]) = A \otimes \sigma(B)$
$z_j^{[u](d)}$
$z_j^{[u]} + e_j^{[u]}$
Attention
j
$a_{ij}^{[l]} = \frac{\exp(d_i^{[l]} \cdot z_j^{[u]})}{\sum_{t=1}^{m} \exp(d_i^{[l]} \cdot z_t^{[u]})}$
$d_i^{[l]}$
$d_i^{[l]} = (W_d^{[l]} h_i^{[l]} + b_d^{[l]}) + g_i$
$g_i \rightarrow$ previous target elmt.
i
Dot products
a_{ij}
$c_i^{[l]} = \sum_{j=1}^{m} a_{ij}^{[l]}(z_j^{[u]} + e_j^{[u]})$
$h_i^{[l]}$
$h_i^{[l]} = v(W_{[l]}[h_{[i-k/2]}^{[l-1]}, \dots, h_{i+k/2}^{[l-1]}] + b_w^l) + h_i^{[l-1]}$
$v([A\ B]) = A \otimes \sigma(B)$
$p(y_{i+1}|y_1, \dots, y_i, \mathbf{x}) = softmax(W_o h_i^{[L]} + b_o) \in \mathbf{R}^T$
<p> <p> <s> Sie stimmen zu
Sie stimmen zu </s>



Experimental Setup

Experimental Setup

- Architecture

- 15 layers in both encoder and decoder
- Convolutional kernel size is 3
- Hidden size gradually increases from 512 to 4096
- Embedding size is 512

Experimental Setup

- Training

- Optimizer: Nesterov with momentum
- Learning rate 0.25, momentum 0.99
- Gradient clipping if norm exceeds 0.1
- Batch size 64
- Data parallel training: all-reduct gradients after each iteration
- Model is implemented in Torch (PyTorch implementation is coming)

Experimental Setup

- Tricks

- WeightNorm(Salimans et al. 2016)
- Dropout (Srivastava et. al. 2014)
- Scale outputs of each layer to normalize variance
- Careful weight initialization to ensure normally distributed variance

Result

Result

총 3가지 translation task로 평가 진행

- WMT’16 English-Romanian
- WMT’14 English-German
- WMT’14 English-French

ConvS2S: Results

Results on English-Romanian(WMT'16, newstest2016)

System	Vocabulary	BLEU
GRU (Sennrich et al., 2016b)	BPE 90 K	28.1
ConvS2S	**Word 40k**	**29.45**
ConvS2S	**BPE 40k**	**30.02**

ConvS2S: Results

Results on English-Romanian(WMT'16, newstest2016)

System	Vocabulary	BLEU
GRU (Sennrich et al., 2016b)	BPE 90 K	28.1
ConvS2S	**Word 40k**	**29.45**
ConvS2S	**BPE 40k**	**30.02**

- Attention-based sequence to sequence architecture of Bahdanau et al.,(2014)
- GRU cells both in the encoder and decoder

ConvS2S: Results

Results on English-Romanian(WMT'16, newstest2016)

System	Vocabulary	BLEU
GRU (Sennrich et al., 2016b)	BPE 90 K	28.1
ConvS2S	**Word 40k**	**29.45**
ConvS2S	**BPE 40k**	**30.02**

- 20 layers in the encoder, 20 layers in the decoder
- Kernels of width 3, hidden size 512
- 6 ~ 7.5 days on a single GPU

ConvS2S: Results

Results on English-German (WMT'14, newstest2014)

System	Vocabulary	BLEU
LSTM (Luong et al., 2015)	Word 50k	20.9
ByteNet v2 (Kalchbrenner et al., 2016)	Characters	23.75
GNMT (Wu et al., 2016)	Word 80k	23.12
GNMT (Wu et al., 2016)	Word pieces	24.61
ConvS2S	**BPE 40k**	**25.16**

ConvS2S: Results

Results on English-German (WMT'14, newstest2014)

System	Vocabulary	BLEU
LSTM (Luong et al., 2015)	Word 50k	20.9
ByteNet v2 (Kalchbrenner et al., 2016)	Characters	23.75
GNMT (Wu et al., 2016)	Word 80k	23.12
GNMT (Wu et al., 2016)	Word pieces	24.61
ConvS2S	**BPE 40k**	**25.16**

- 4 layer LSTM attention model

ConvS2S: Results

Results on English-German (WMT'14, newstest2014)

System	Vocabulary	BLEU
LSTM (Luong et al., 2015)	Word 50k	20.9
ByteNet v2 (Kalchbrenner et al., 2016)	Characters	23.75
GNMT (Wu et al., 2016)	Word 80k	23.12
GNMT (Wu et al., 2016)	Word pieces	24.61
ConvS2S	**BPE 40k**	**25.16**

- Convolutional model based on character without attention
- 30 layers in the encoder & 30 layers in the decoder

ConvS2S: Results

Results on English-German (WMT'14, newstest2014)

System	Vocabulary	BLEU
LSTM (Luong et al., 2015)	Word 50k	20.9
ByteNet v2 (Kalchbrenner et al., 2016)	Characters	23.75
GNMT (Wu et al., 2016)	Word 80k	23.12
GNMT (Wu et al., 2016)	Word pieces	24.61
ConvS2S	**BPE 40k**	**25.16**

- 8 encoder LSTM & 8 decoder LSTM
- Word based model

ConvS2S: Results

Results on English-German (WMT'14, newstest2014)

System	Vocabulary	BLEU
LSTM (Luong et al., 2015)	Word 50k	20.9
ByteNet v2 (Kalchbrenner et al., 2016)	Characters	23.75
GNMT (Wu et al., 2016)	Word 80k	23.12
GNMT (Wu et al., 2016)	Word pieces	24.61
ConvS2S	**BPE 40k**	**25.16**

- 15 encoder & 15 decoder, kernel width 3
- 512 hidden unit in the first ten layers
- 768 hidden unit in the subsequent 3 layers

ConvS2S: Results

Results on English-German (WMT'14, newstest2014)

System	Vocabulary	BLEU
LSTM (Luong et al., 2015)	Word 50k	20.9
ByteNet v2 (Kalchbrenner et al., 2016)	Characters	23.75
GNMT (Wu et al., 2016)	Word 80k	23.12
GNMT (Wu et al., 2016)	Word pieces	24.61
ConvS2S	**BPE 40k**	**25.16**

- 2048 unit in the final two layers
- 18.5 days on a single GPU
- Batch size 48

ConvS2S: Results

Results on English-French (WMT'14, newstest2014)

System	Vocabulary	BLEU
GNMT (Wu et al., 2016)	Word 80k	37.90
GNMT (Wu et al., 2016)	Word pieces	38.95
GNMT + RL (Wu et al., 2016)	Word pieces	39.92
ConvS2S	**BPE 40k**	**40.46**

ConvS2S: Results

Results on English-French (WMT'14, newstest2014)

System	Vocabulary	BLEU
GNMT (Wu et al., 2016)	Word 80k	37.90
GNMT (Wu et al., 2016)	Word pieces	38.95
GNMT + RL (Wu et al., 2016)	Word pieces	39.92
ConvS2S	**BPE 40k**	**40.46**

- 15 layers in the encoder & 15 layers in the decoder
- 512 hidden units in the first 5 layers
- 768 hidden units in the subsequent 4 layers
- 1026 hidden units in the next 3 layers
- kernel width 3

ConvS2S: Results

Results on English-French (WMT'14, newstest2014)

System	Vocabulary	BLEU
GNMT (Wu et al., 2016)	Word 80k	37.90
GNMT (Wu et al., 2016)	Word pieces	38.95
GNMT + RL (Wu et al., 2016)	Word pieces	39.92
ConvS2S	**BPE 40k**	**40.46**

- 2048 hidden units in the 2 layers
- 4096 hidden units in the 1 layers
- Kernel width 1
- Batch size 32
- Training with 8 GPUs for about 37 days

ConvS2S: Ensemble Results

Results on English-German (WMT'14, newstest2014)

System	BLEU
GNMT (Wu et al., 2016)	26.20
GNMT + RL (Wu et al., 2016)	26.30
ConvS2S	**26.43**

Results on English-French (WMT'14, newstest2014)

System	BLEU
GNMT (Wu et al., 2016)	40.35
GNMT + RL (Wu et al., 2016)	41.16
ConvS2S (10 models)	**26.43**

ConvS2S: Speed

Translation speed on English-French (WMT'14, dev set)

System	Hardware	BLEU	Time (s)
GNMT (Wu et al., 2016)	CPU (88 cores)	31.20	1322
GNMT (Wu et al., 2016)	GPU (K80)	31.20	3028
GNMT + RL (Wu et al., 2016)	TPU	31.21	384
ConvS2S, beam=5	**CPU (48 cores)**	**34.10**	**482**
ConvS2S, beam=5	**GPU (K40)**	**34.10**	**587**
ConvS2S, beam=5	**GPU (GTX-1080ti)**	**34.10**	**406**
ConvS2S, beam=1	**CPU (48 cores)**	**33.45**	**142**

ConvS2S: Position Embedding

Results on English-German (WMT'14, newstest2014)

System	PPL	BLEU
ConvS2S	6.64	**21.7**
Source position	6.69	**21.3**
Target position	6.63	**21.5**
Source & Target position	6.68	**21.2**

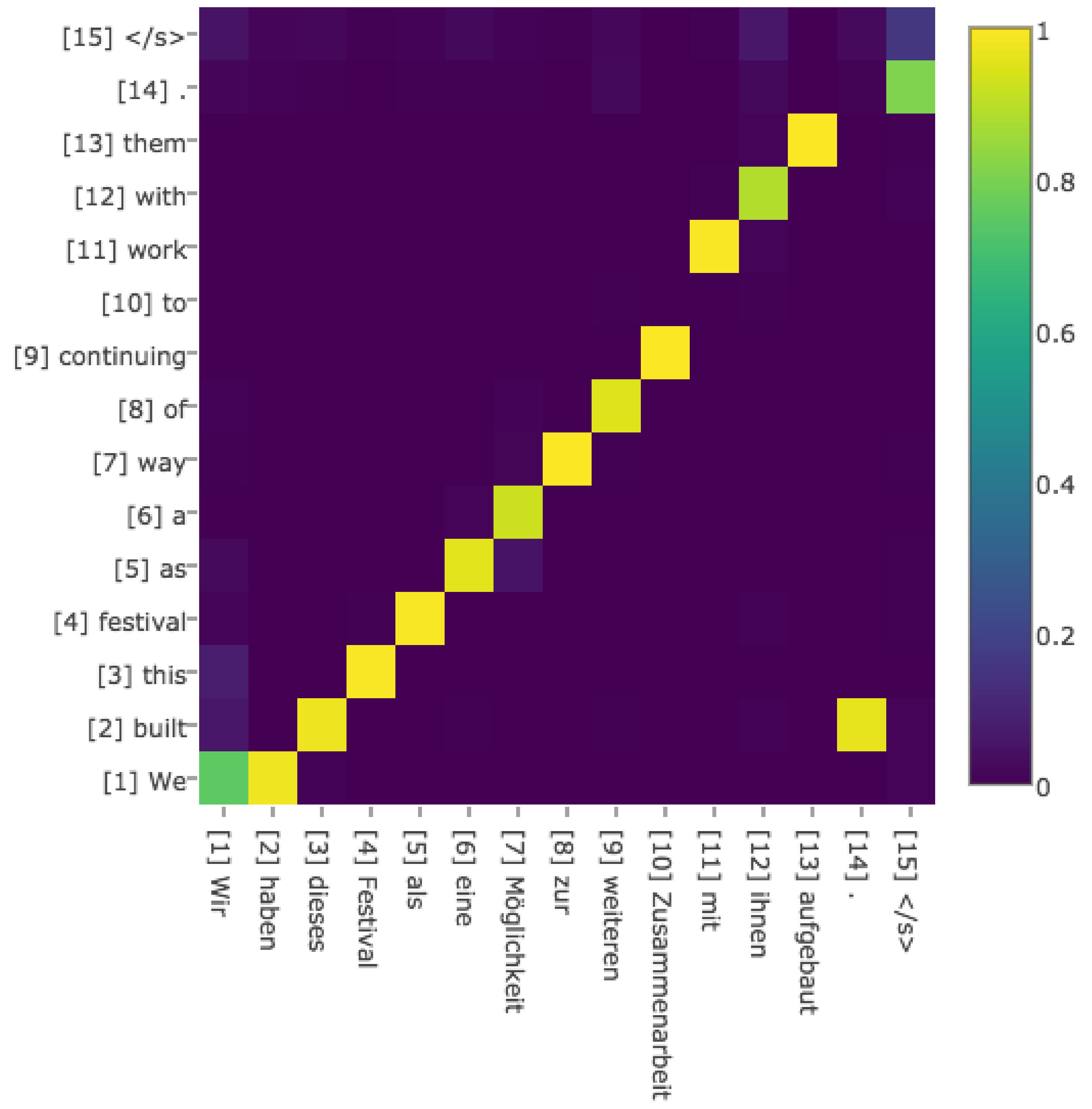
- 13 encoder layers at kernel size 3
- 5 decoder layers at kernel size 5
- Target vocabulary of 160K words

ConvS2S: Multi step Attention

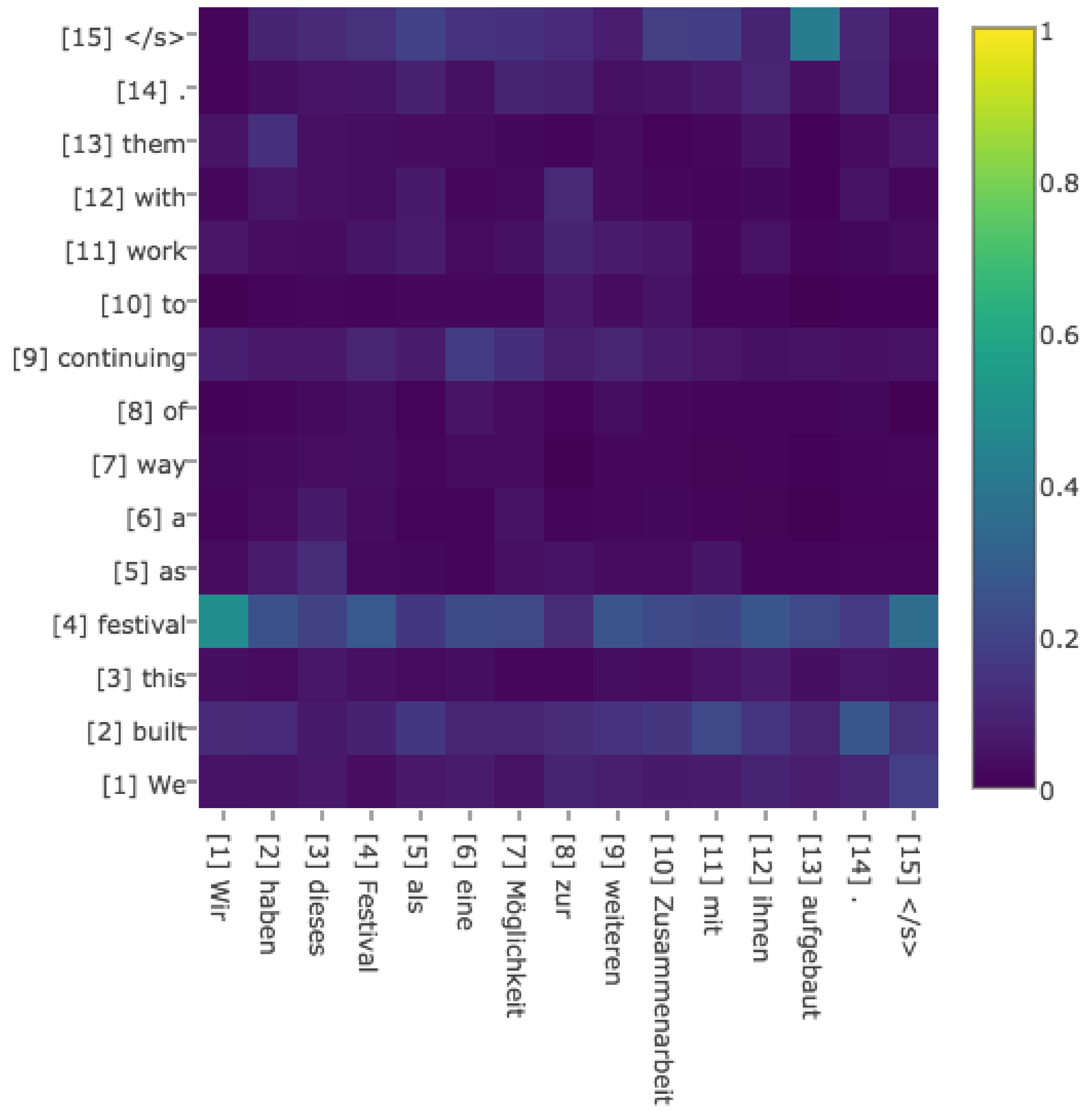
Attn Layers	PPL	BLEU
1,2,3,4,5	**6.65**	**21.63**
1,2,3,4	6.70	21.54
1,2,3	6.95	21.36
1,2	6.92	21.47
1,3,5	6.97	21.10
1	7.15	21.26
2	7.09	21.30
3	7.11	21.19
4	7.19	21.31
5	7.66	21.24

Attention Visualization

Attention Visualization

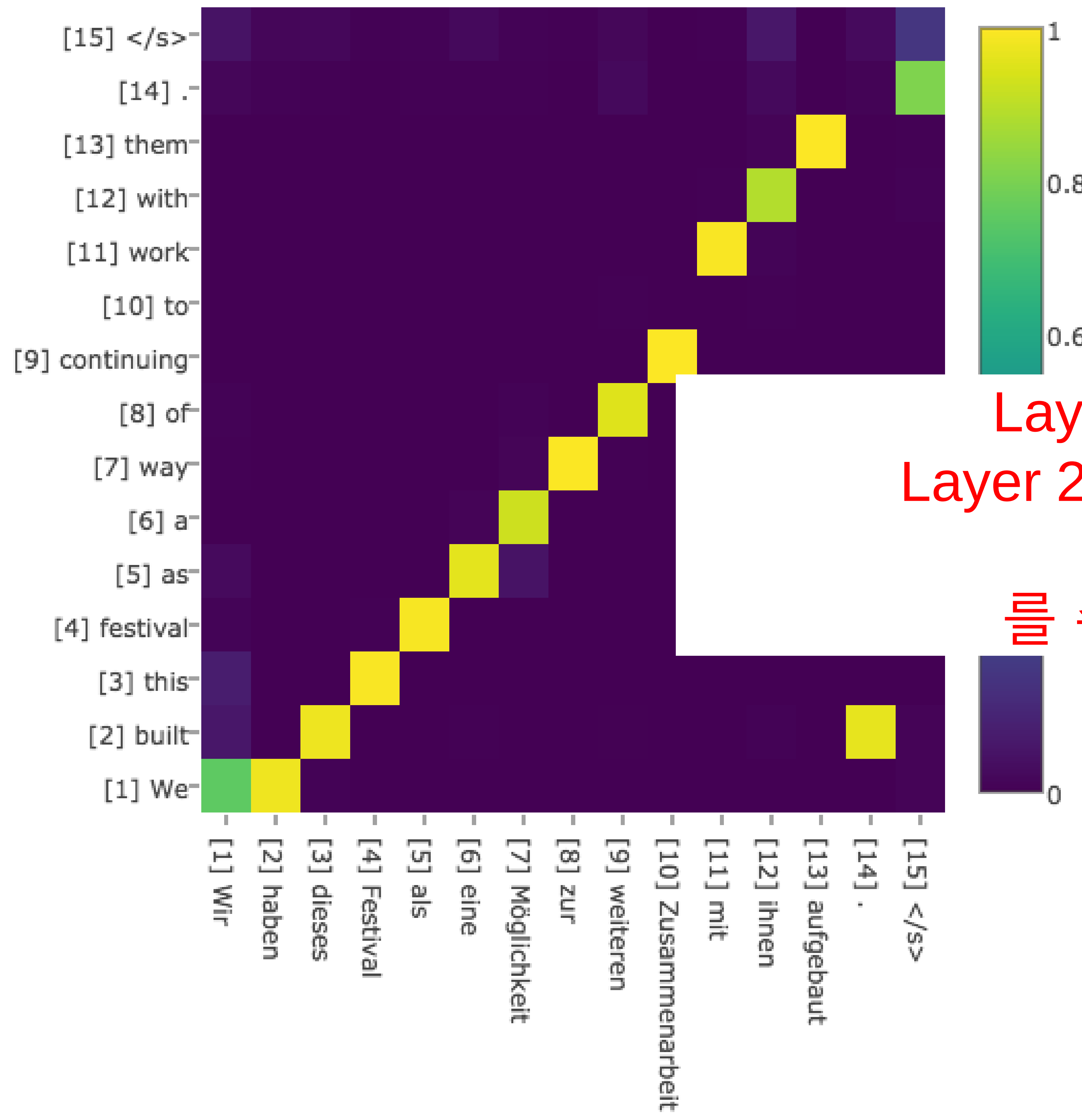


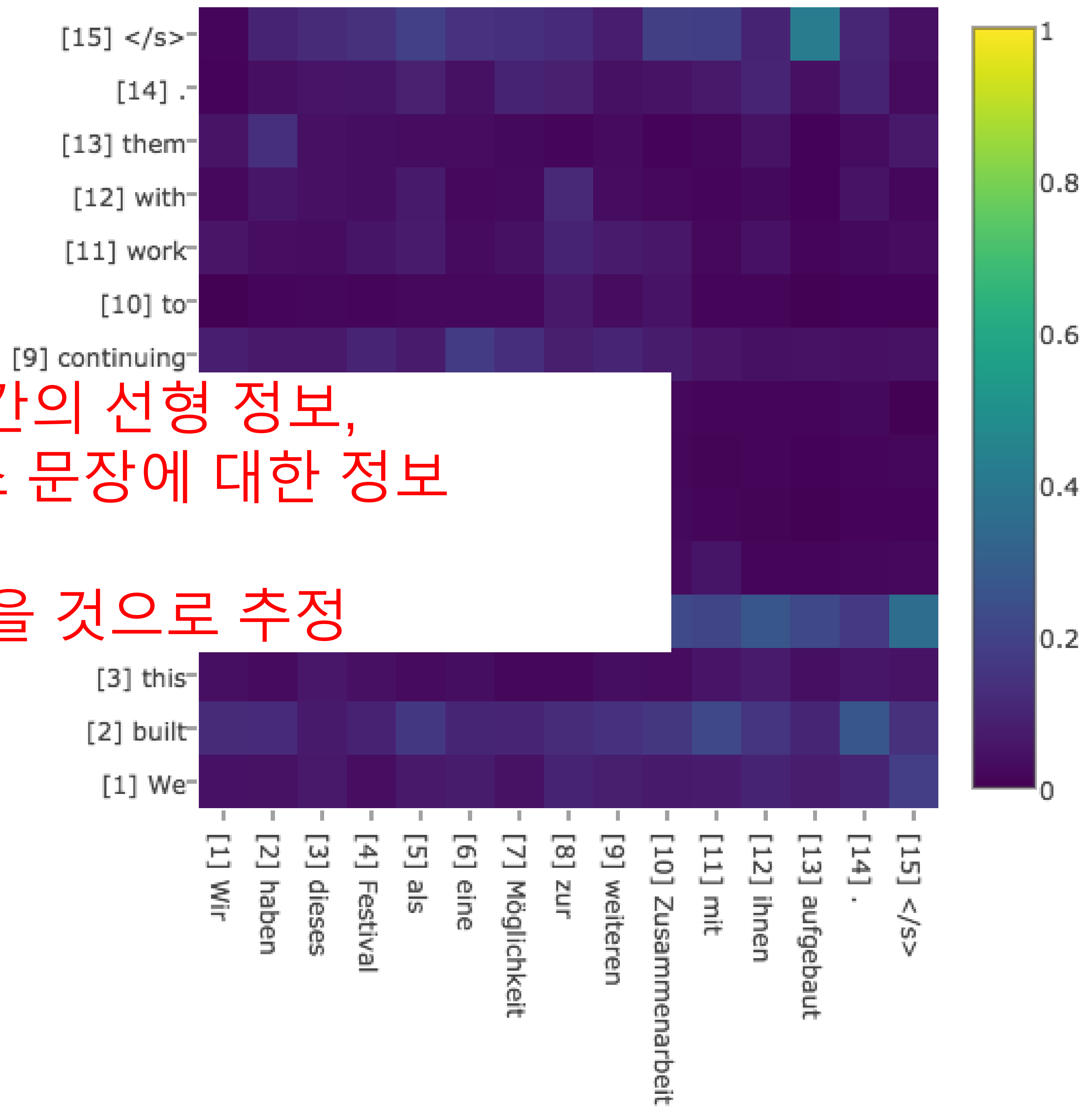
1st Layer



2nd Layer

Attention Visualization





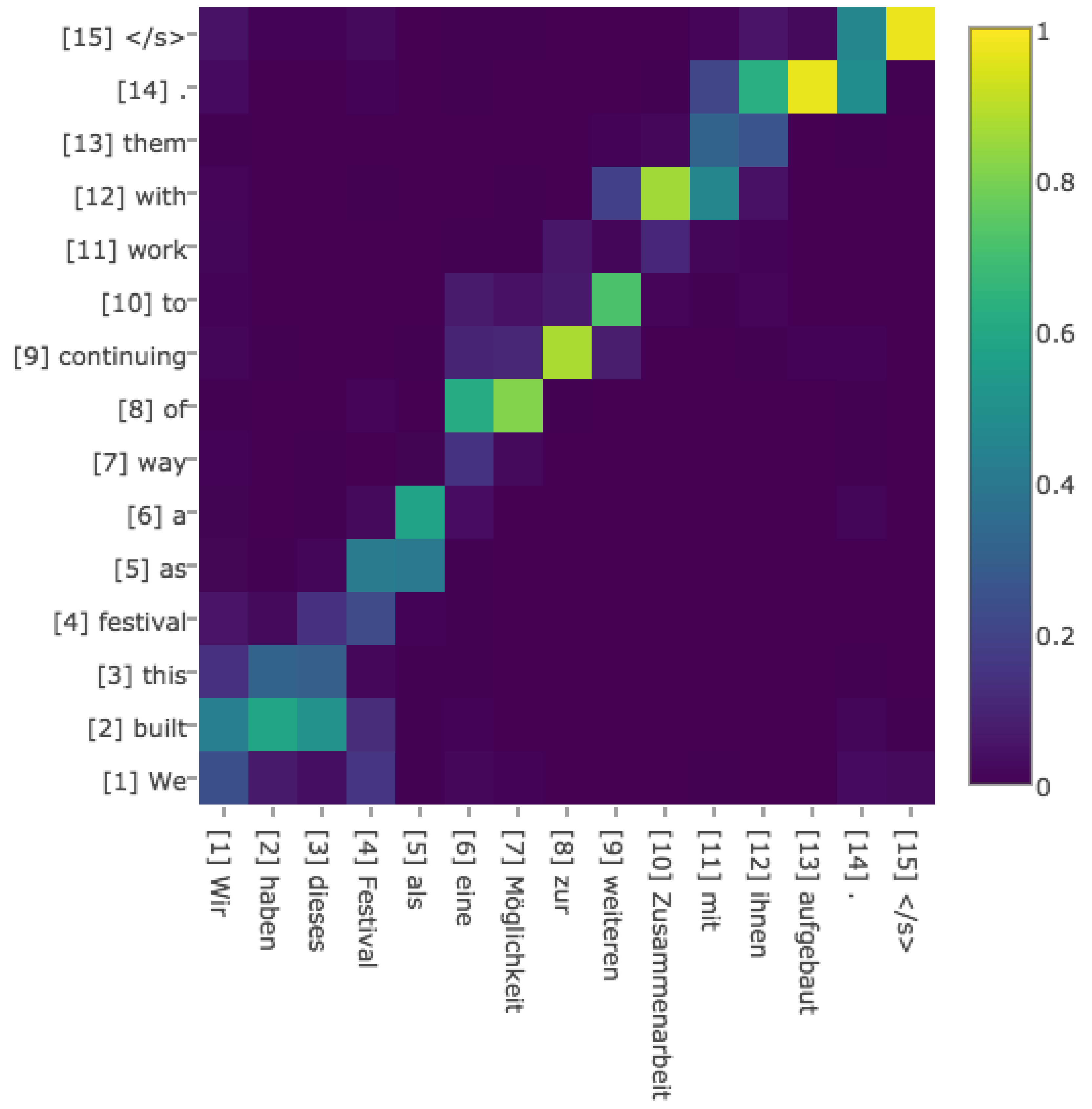
Layer 1은 단어간의 선형 정보,
Layer 2는 전체 소스 문장에 대한 정보

를 수집하고 있을 것으로 추정

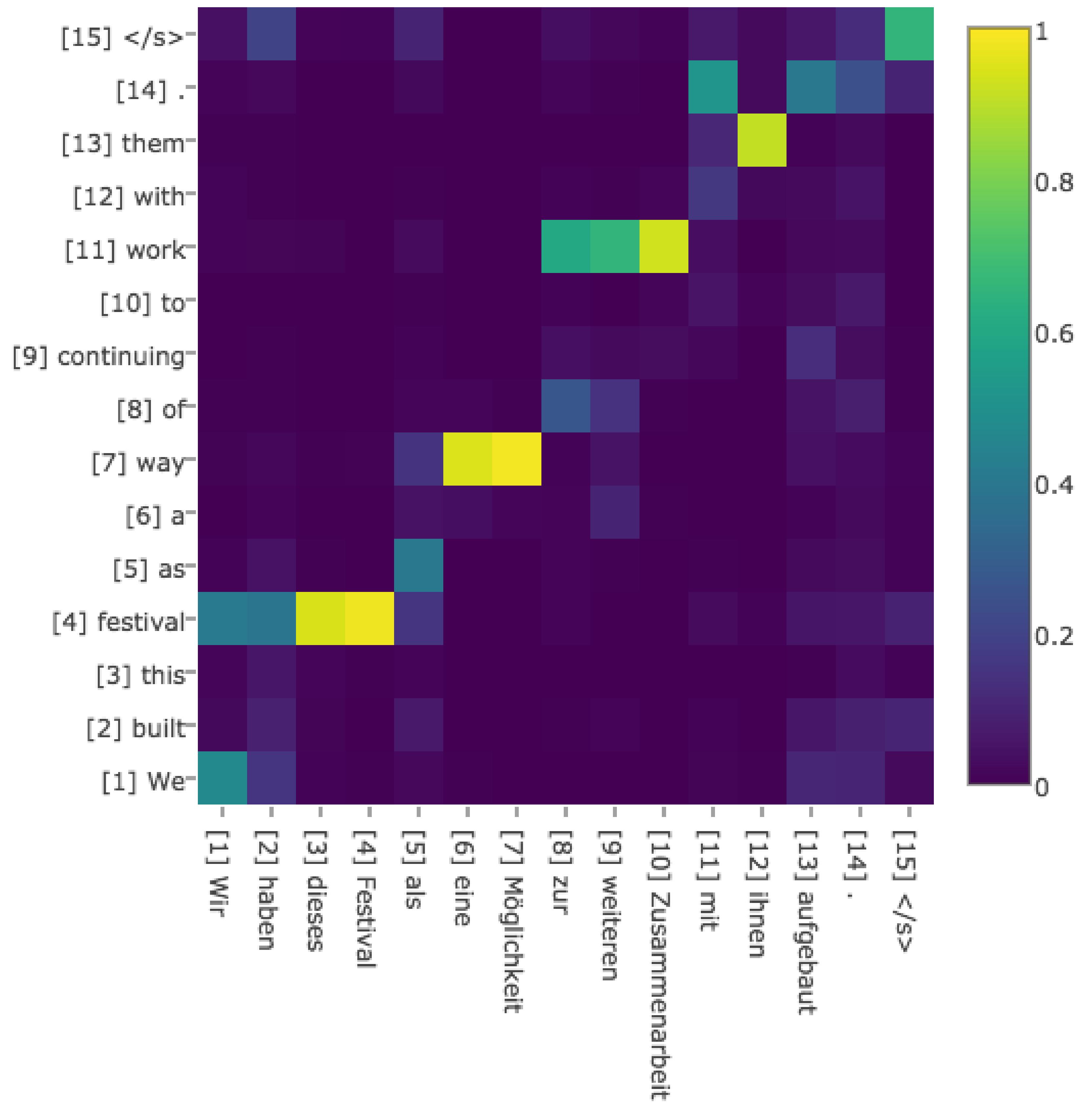
1st Layer

2nd Layer

Attention Visualization

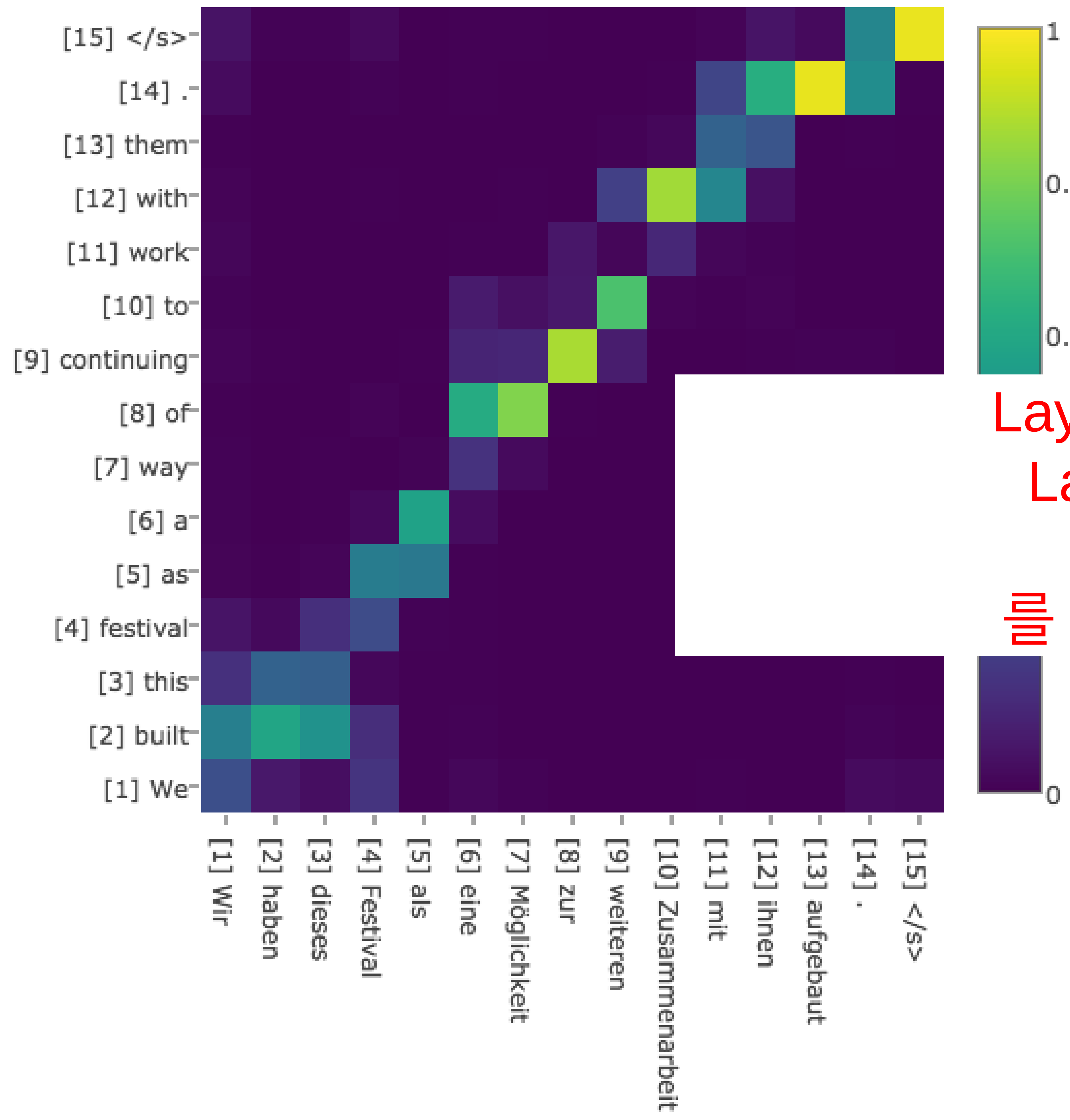


3rd Layer



4th Layer

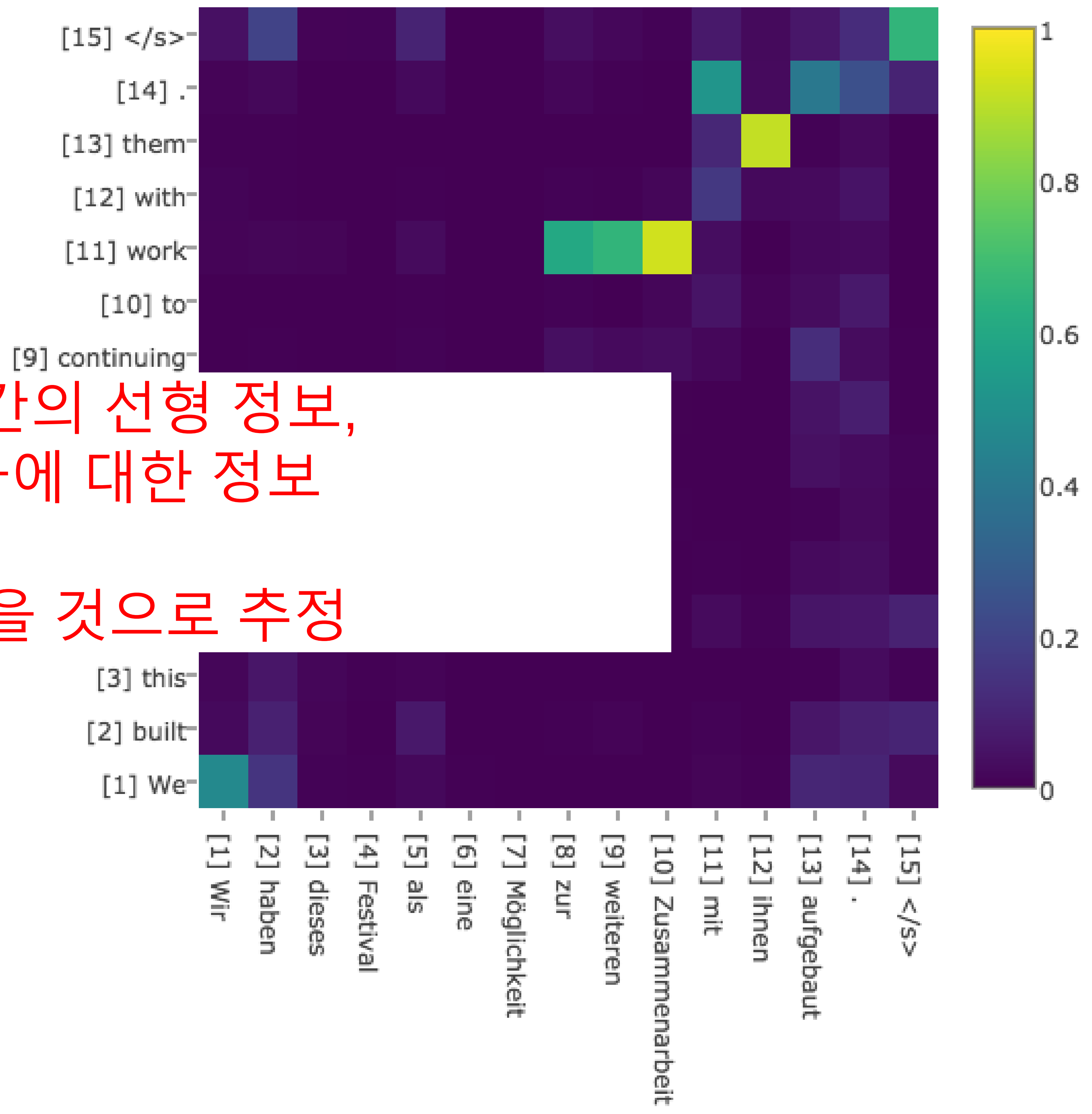
Attention Visualization



3rd Layer

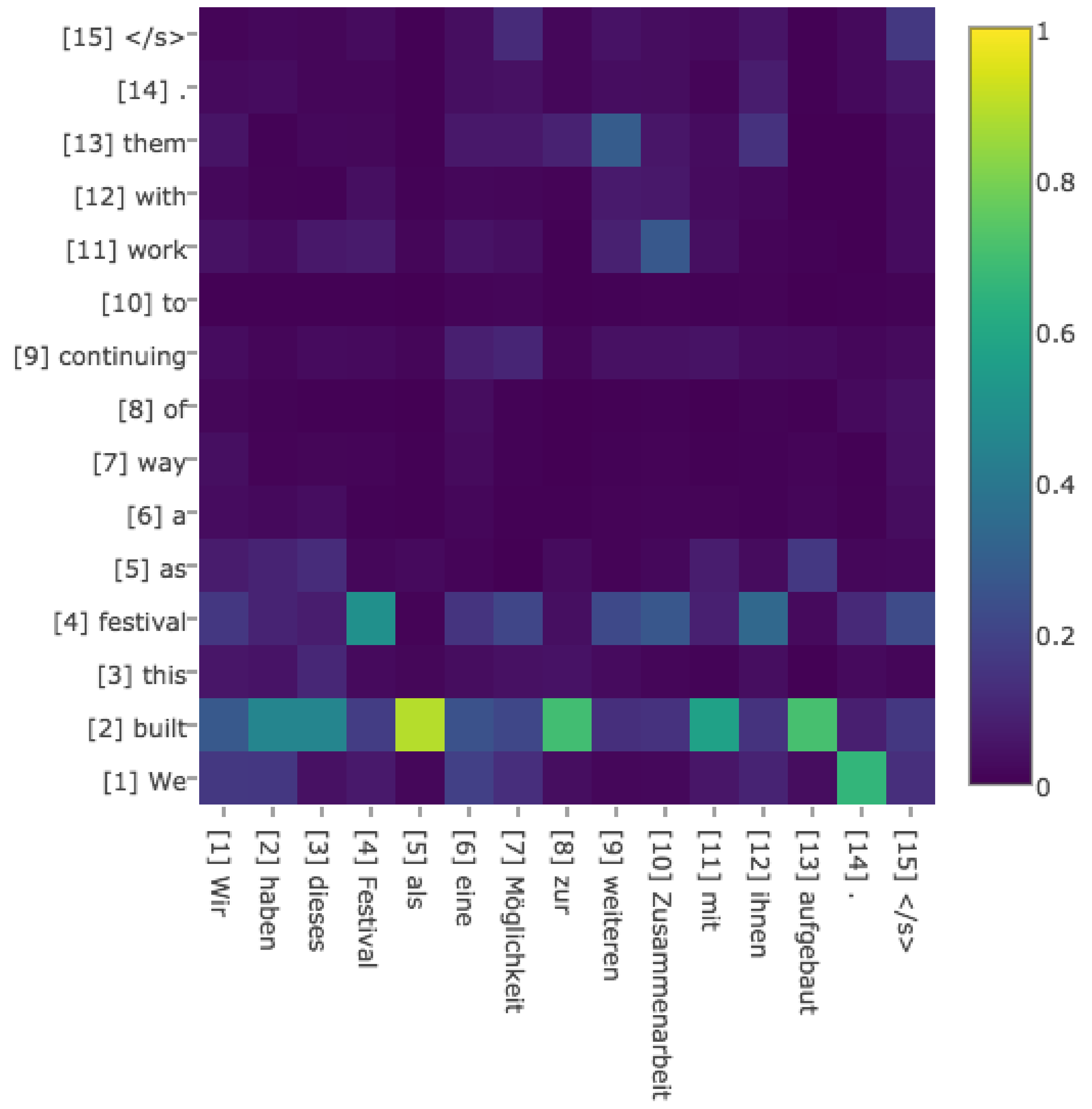
Layer 3은 단어간의 선형 정보,
Layer 4는 명사에 대한 정보

를 수집하고 있을 것으로 추정

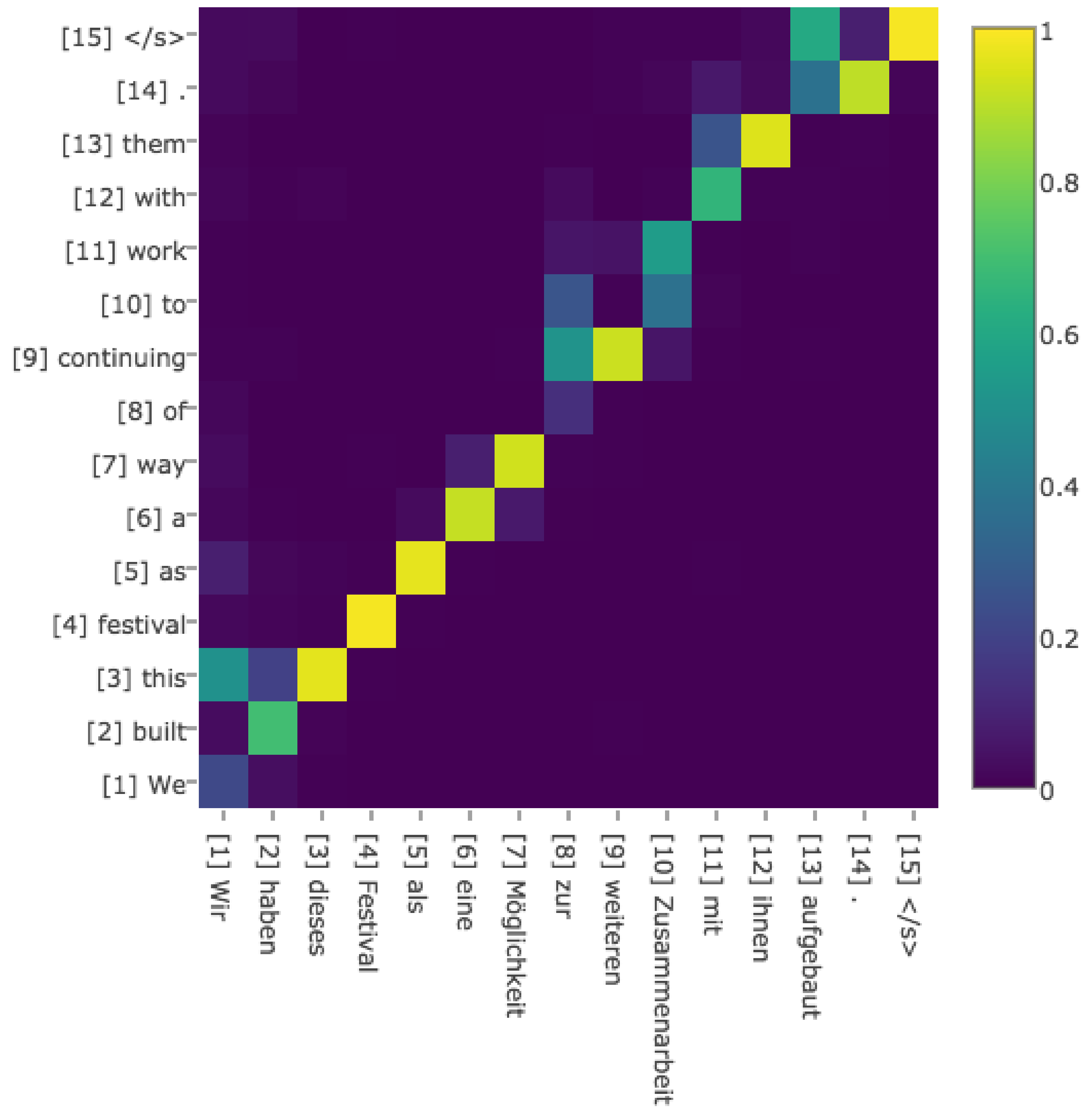


4th Layer

Attention Visualization

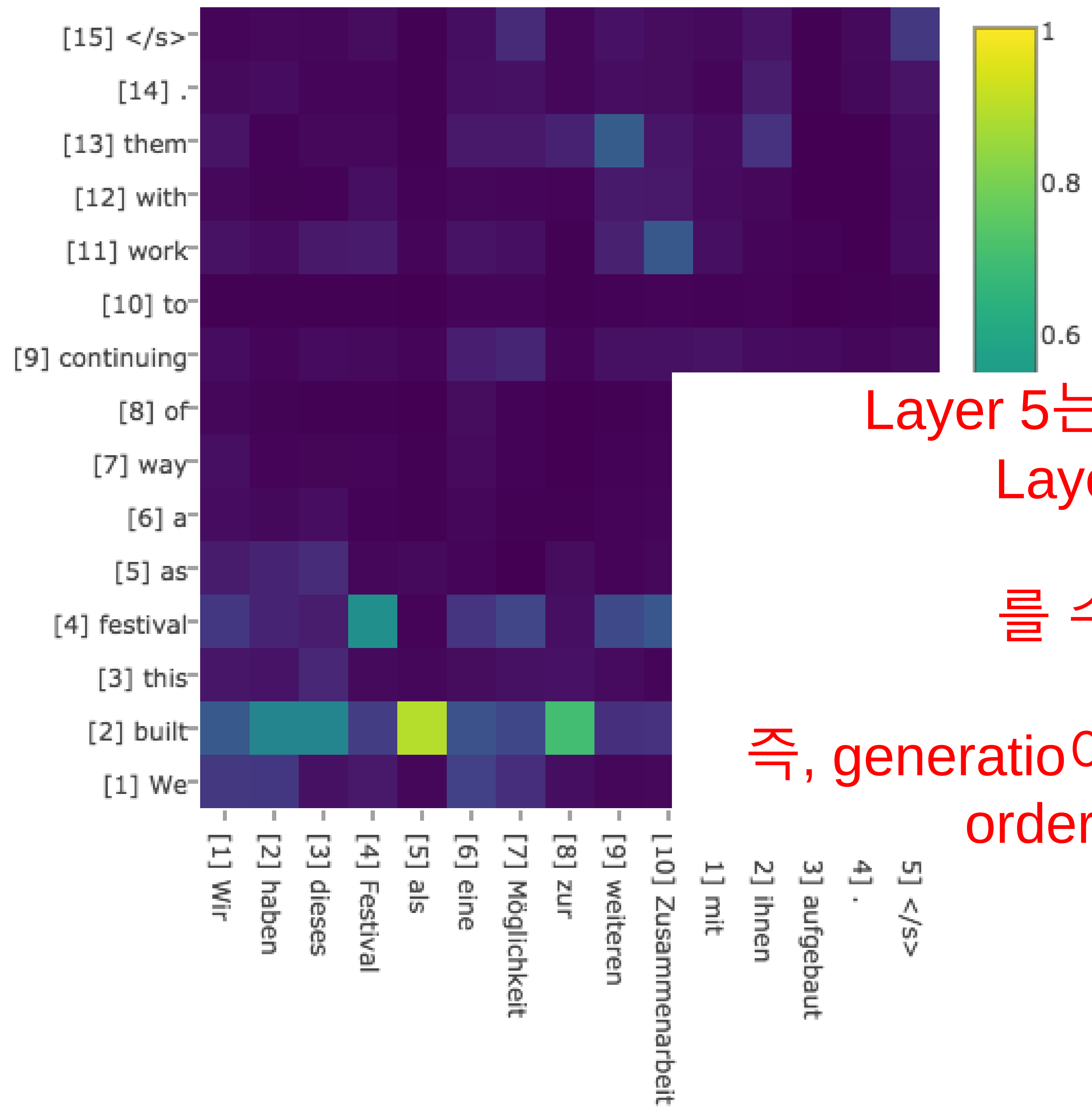


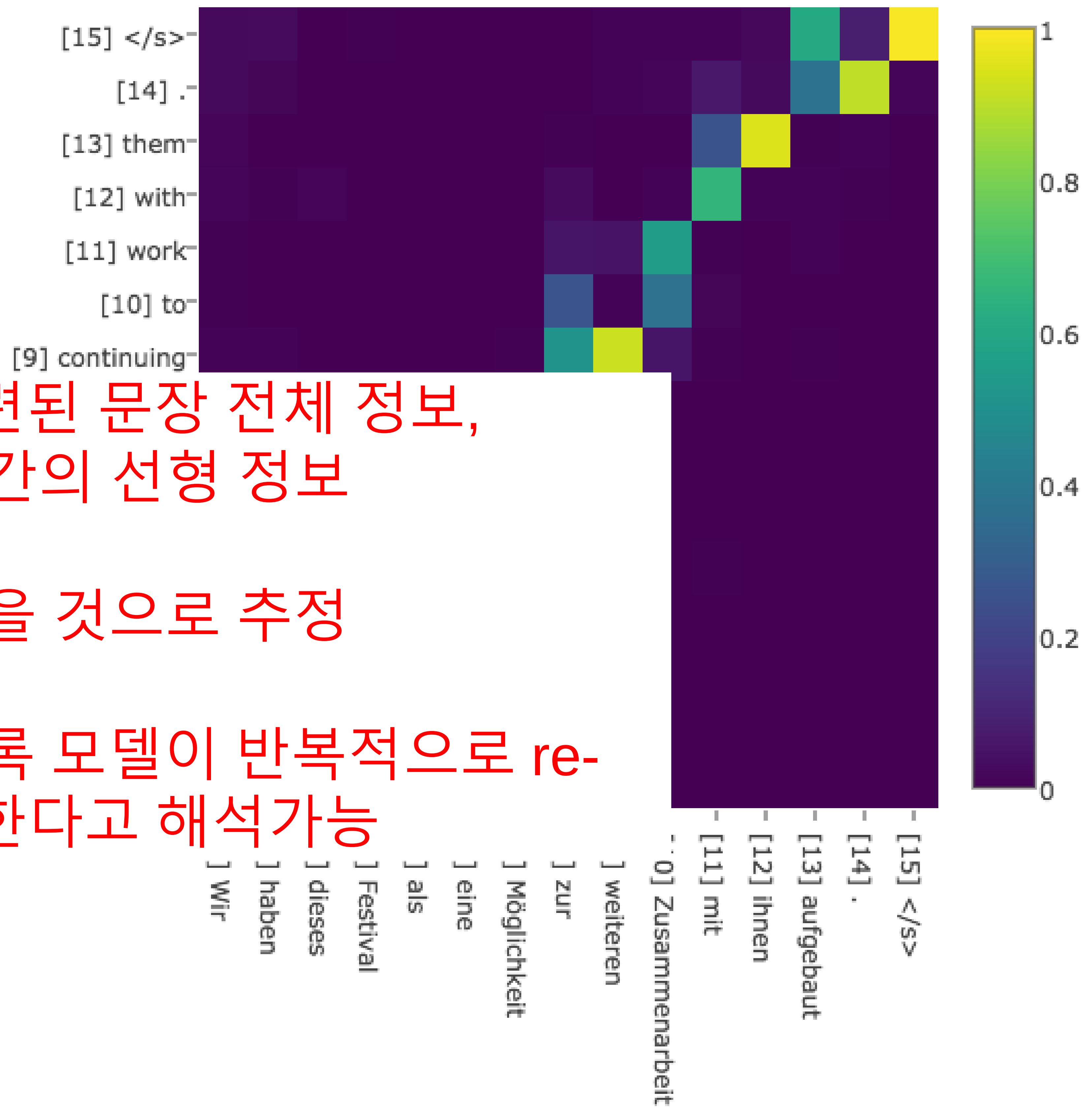
5th Layer



6th Layer

Attention Visualization





Layer 5는 built와 관련된 문장 전체 정보,
Layer 6은 단어간의 선형 정보

를 수집하고 있을 것으로 추정

즉, generatio이 진행될수록 모델이 반복적으로 re-ordering을 수행한다고 해석가능

5th Layer

6th Layer

Appendix