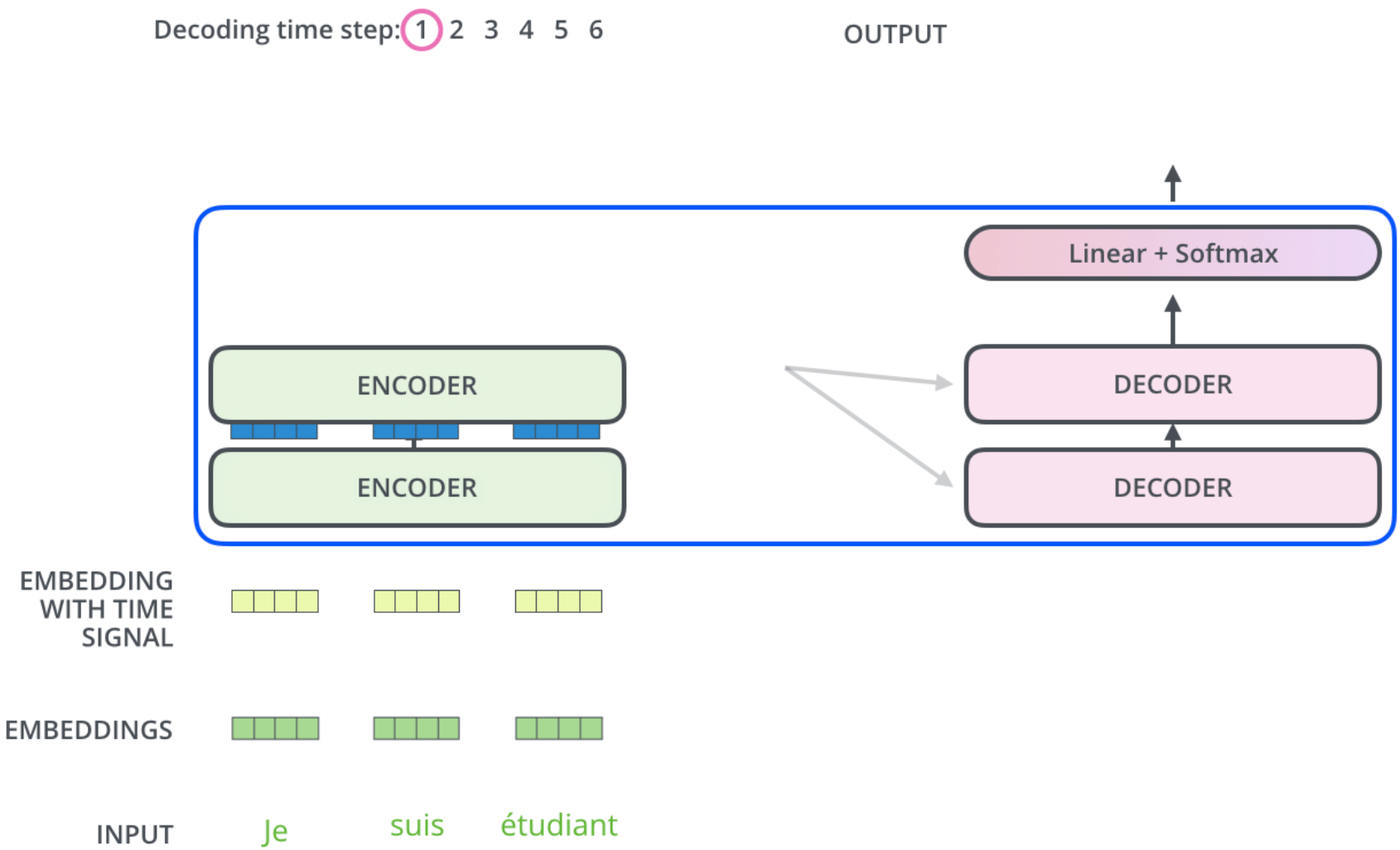
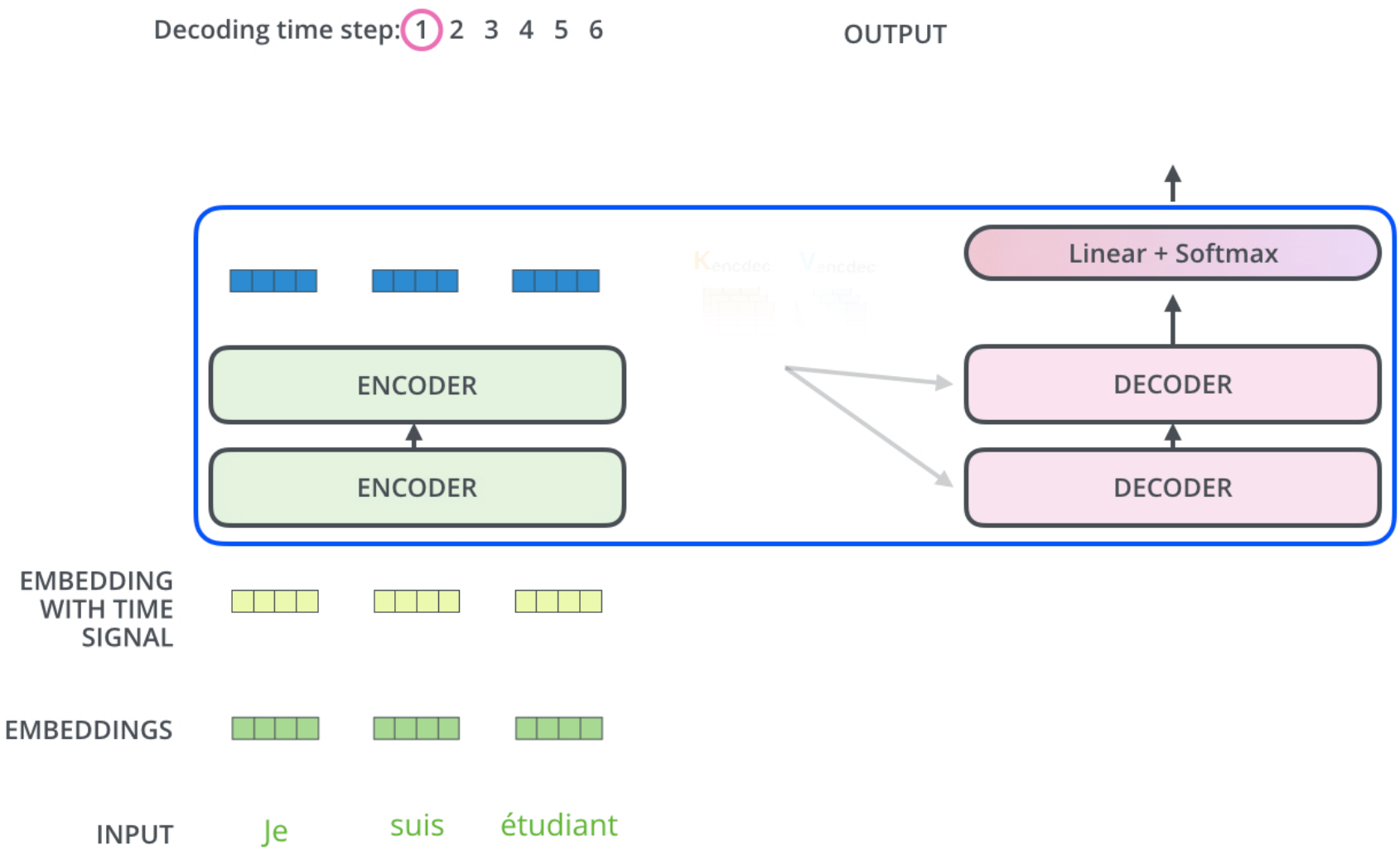
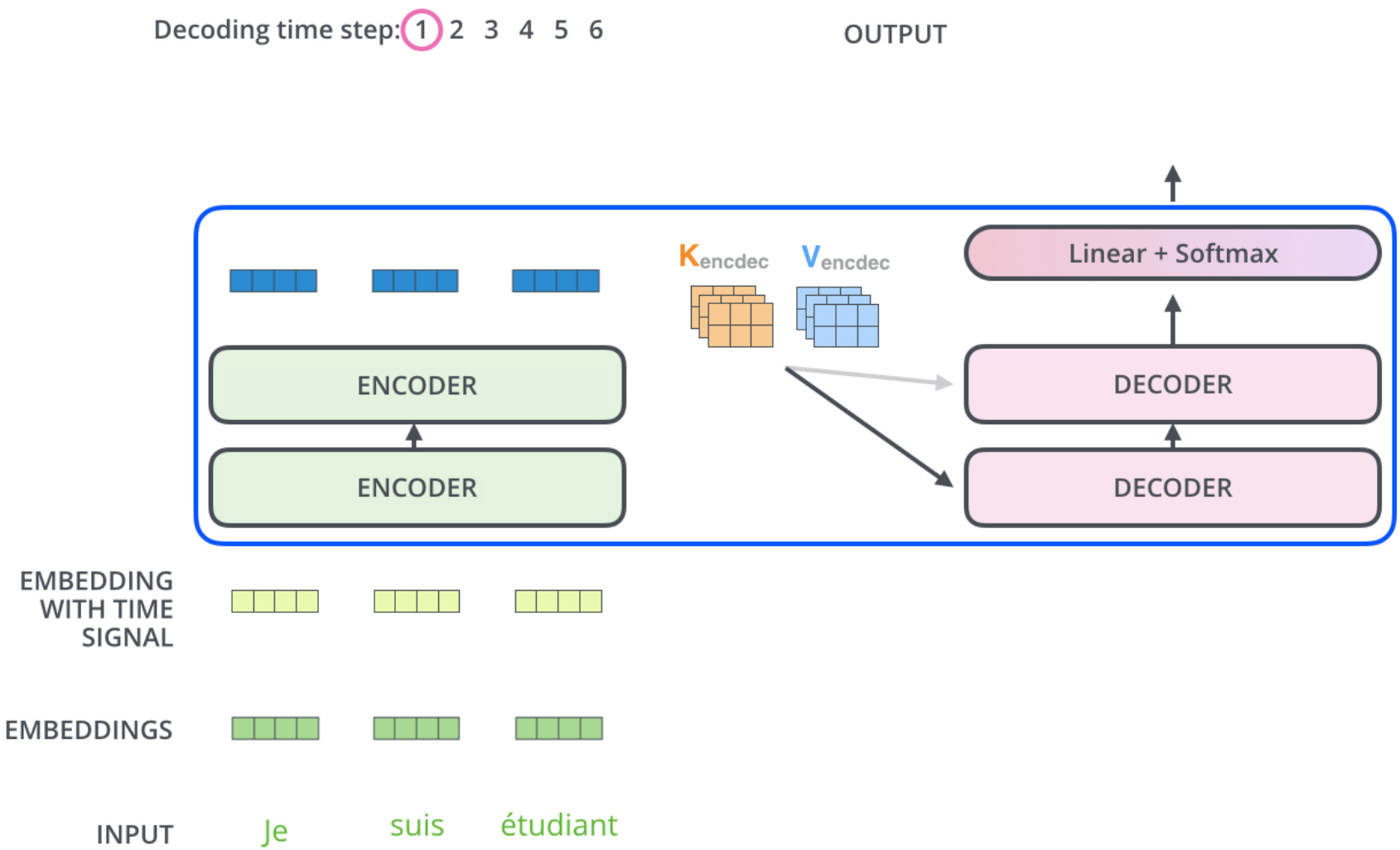
Decoding


Decoding time step: 1 2 3 4 5 6
OUTPUT
Linear + Softmax
ENCODER
ENCODER
DECODER
DECODER
EMBEDDING WITH TIME SIGNAL
EMBEDDINGS
INPUT
Je
suis
étudiant

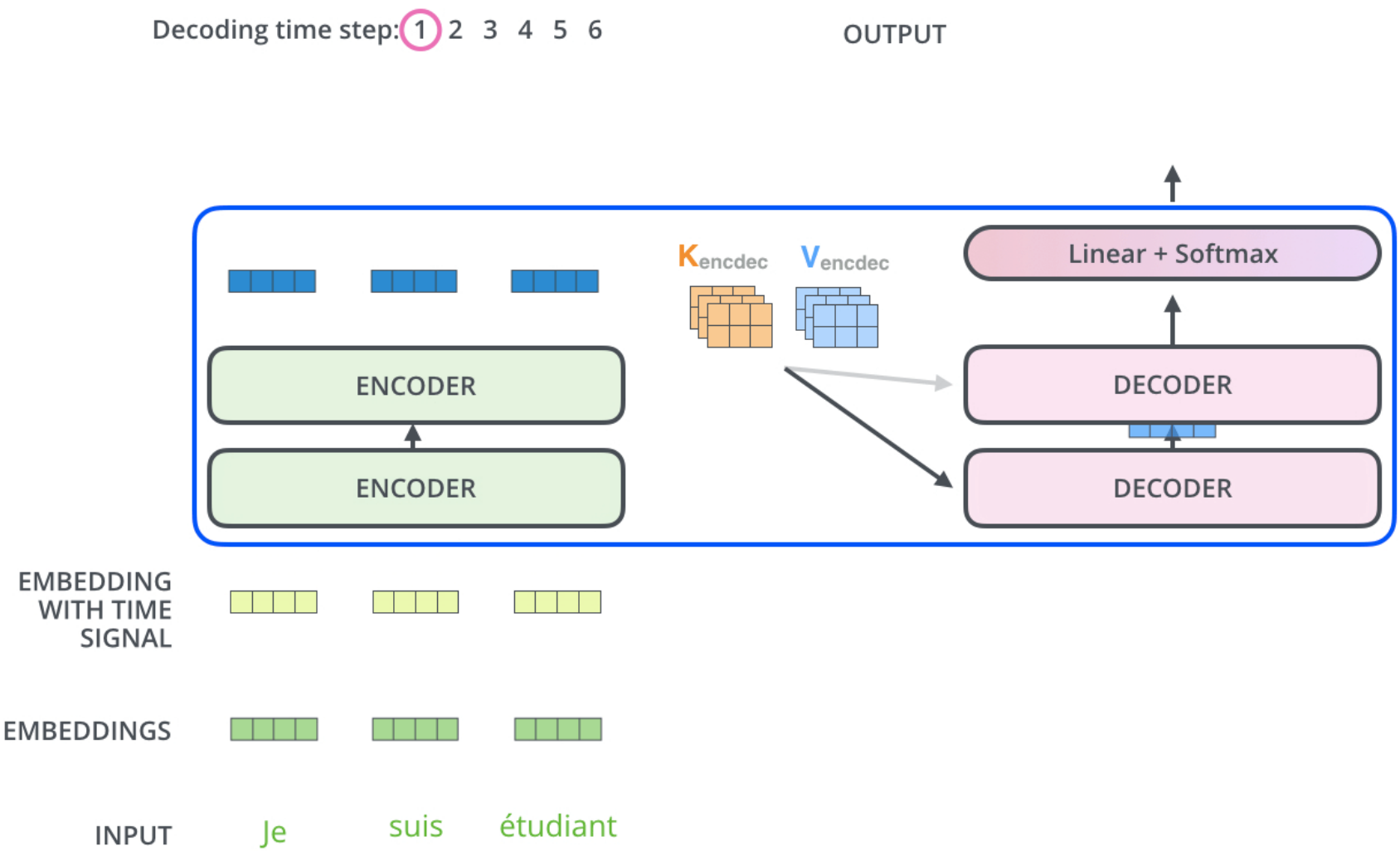
Decoding


Decoding time step: 1 2 3 4 5 6
OUTPUT
Linear + Softmax
ENCODER
ENCODER
DECODER
DECODER
EMBEDDING WITH TIME SIGNAL
EMBEDDINGS
INPUT
Je
suis
étudiant

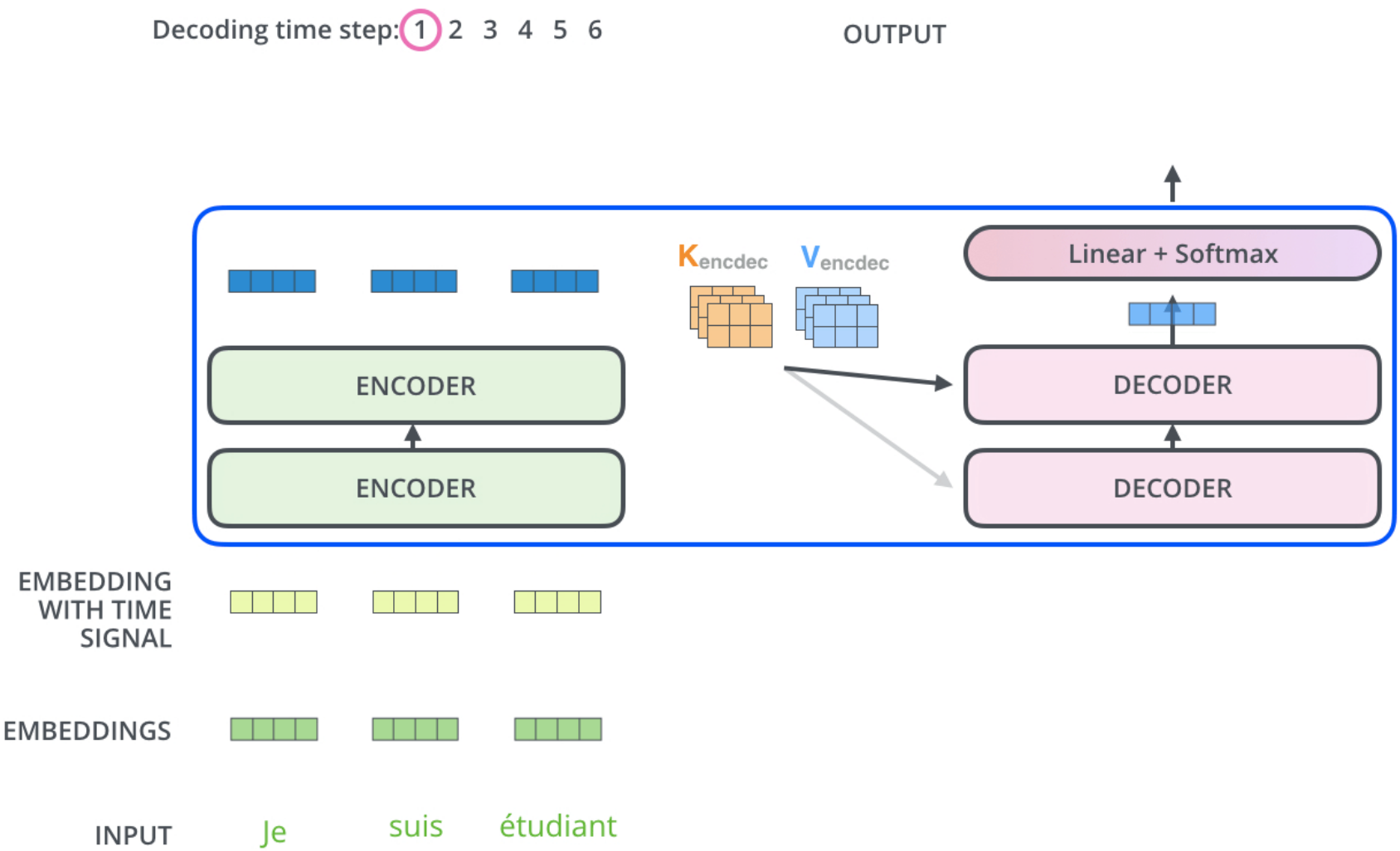
Decoding


Decoding time step: 1 2 3 4 5 6
OUTPUT
Linear + Softmax
K_{encdec}
V_{encdec}
ENCODER
ENCODER
DECODER
DECODER
EMBEDDING WITH TIME SIGNAL
EMBEDDINGS
INPUT
Je
suis
étudiant

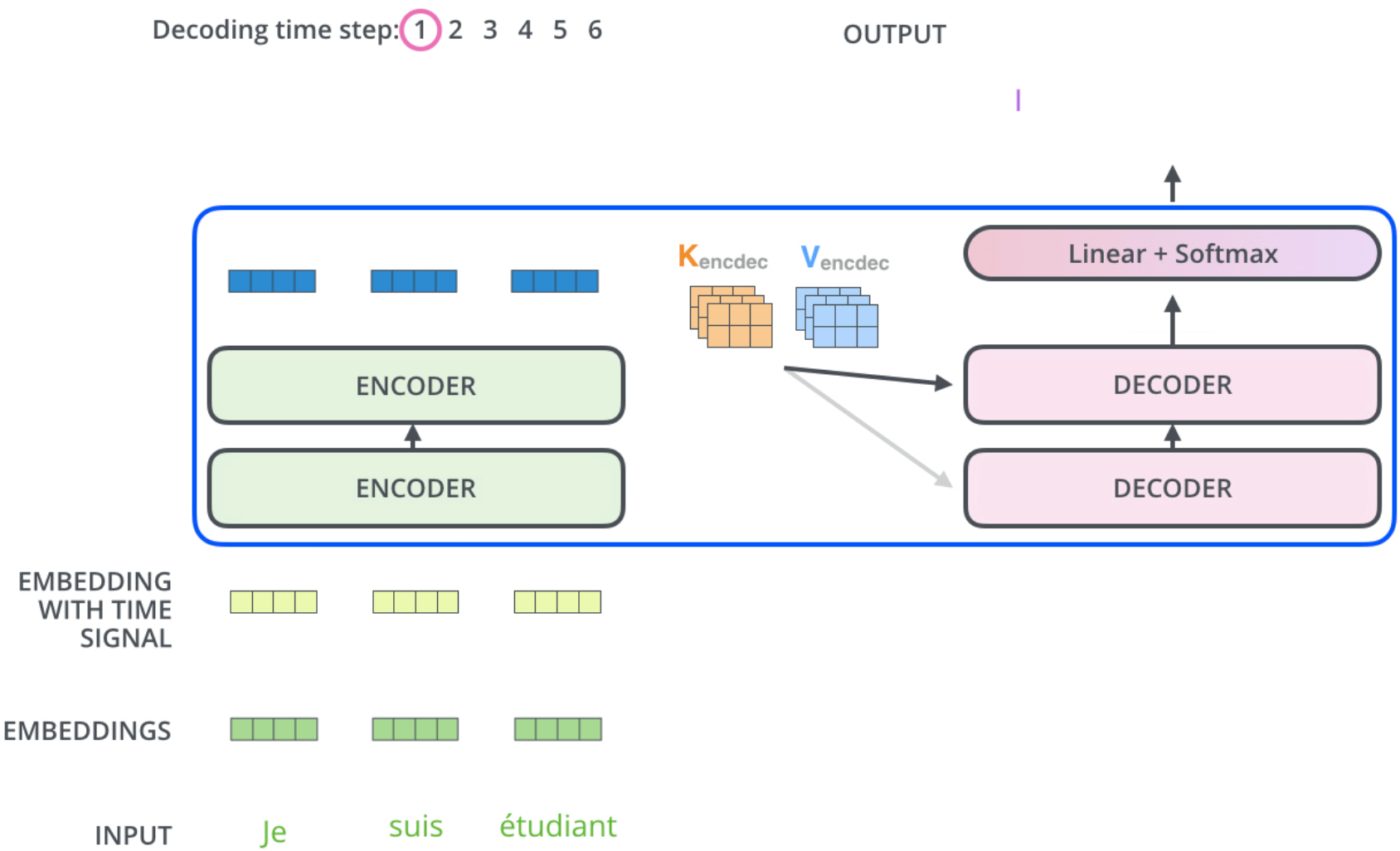
Decoding


Decoding time step: 1 2 3 4 5 6
OUTPUT
Linear + Softmax
K_{encdec}
V_{encdec}
ENCODER
ENCODER
DECODER
DECODER
EMBEDDING WITH TIME SIGNAL
EMBEDDINGS
INPUT
Je
suis
étudiant

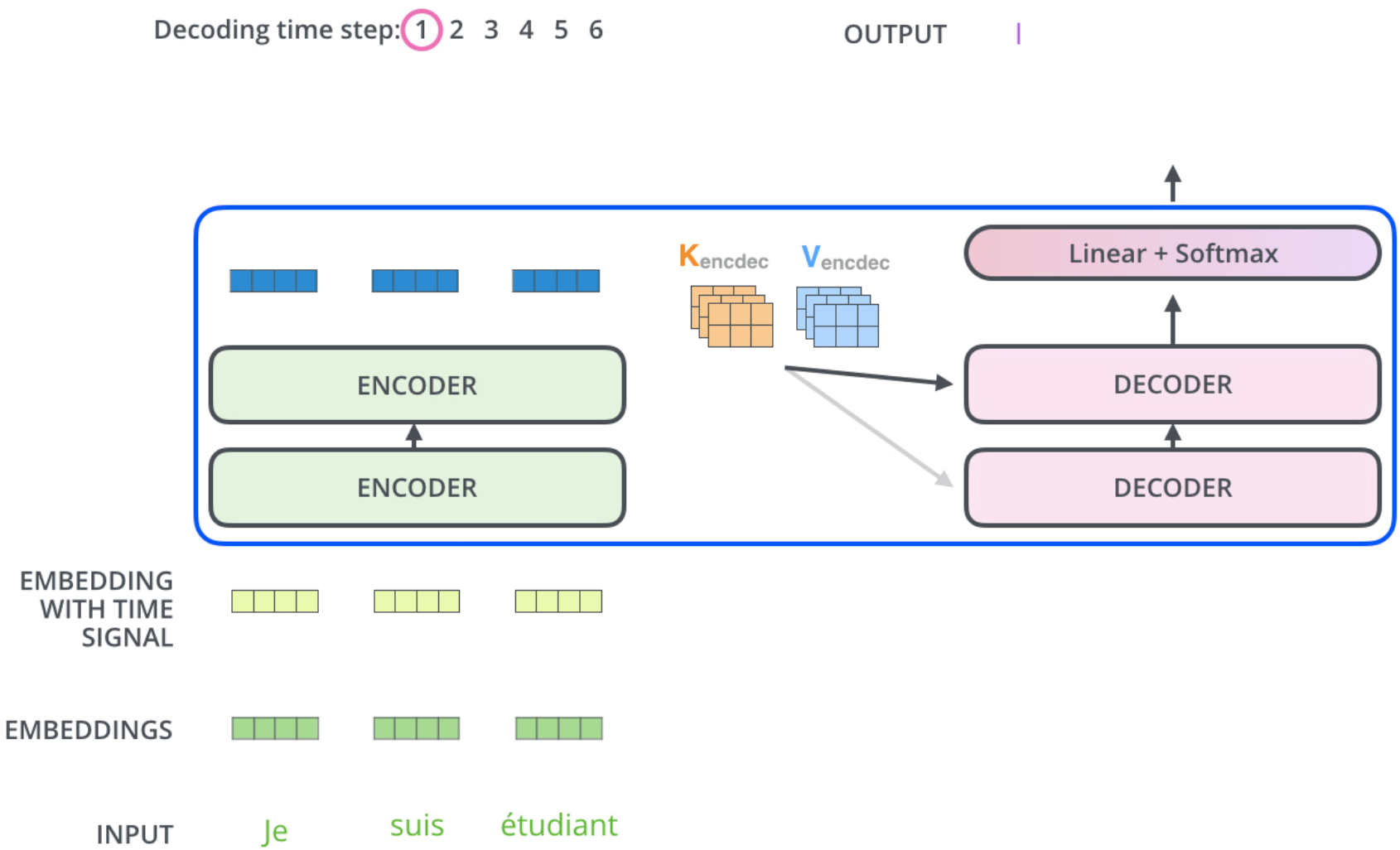
Decoding


Decoding time step: 1 2 3 4 5 6
OUTPUT
Linear + Softmax
K_{encdec}
V_{encdec}
ENCODER
ENCODER
DECODER
DECODER
EMBEDDING WITH TIME SIGNAL
EMBEDDINGS
INPUT
Je
suis
étudiant

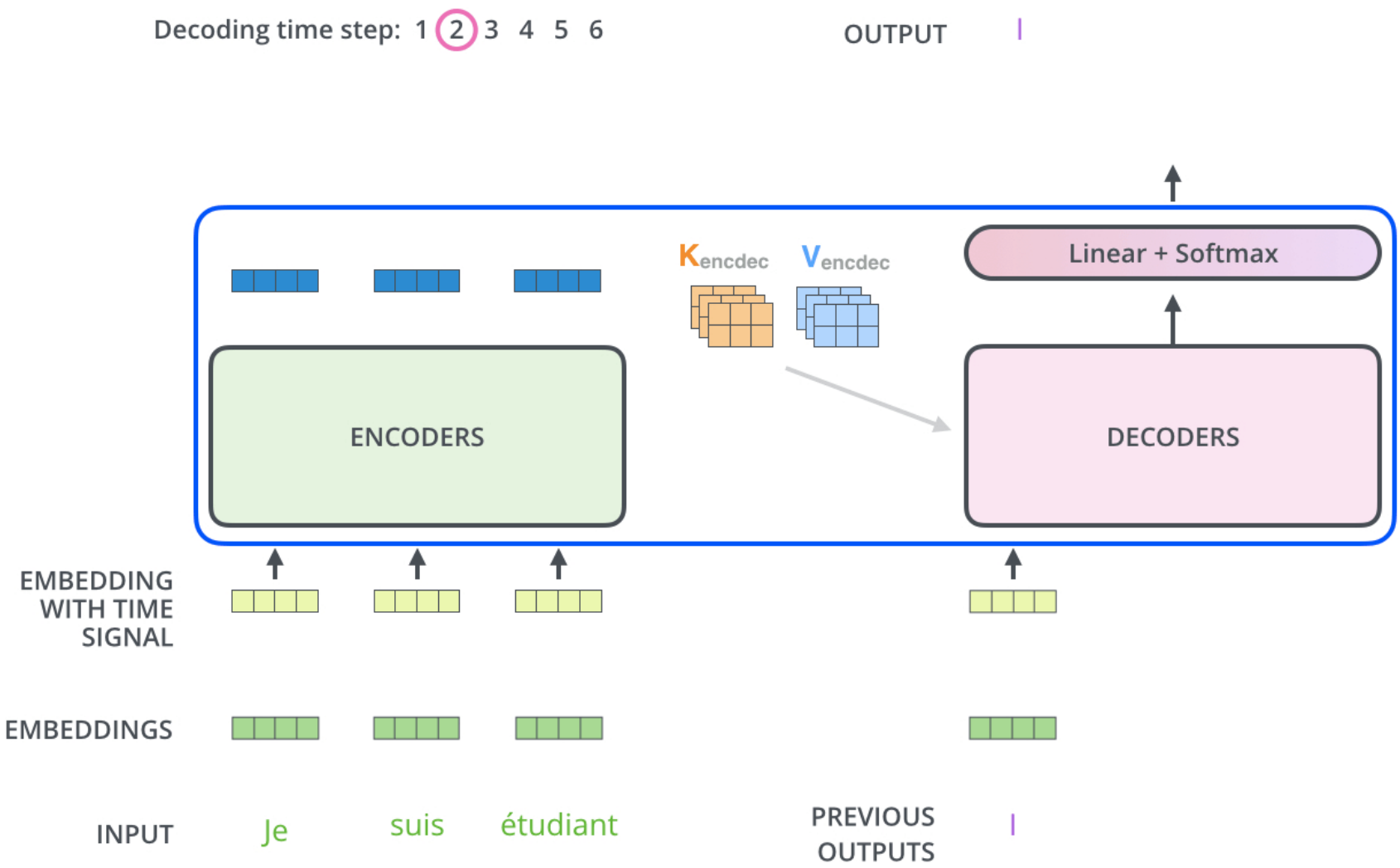
Decoding


Decoding time step: 1 2 3 4 5 6
OUTPUT
I
Linear + Softmax
DECODER
DECODER
ENCODER
ENCODER
Kencdec
Vencdec
EMBEDDING WITH TIME SIGNAL
EMBEDDINGS
INPUT
Je
suis
étudiant

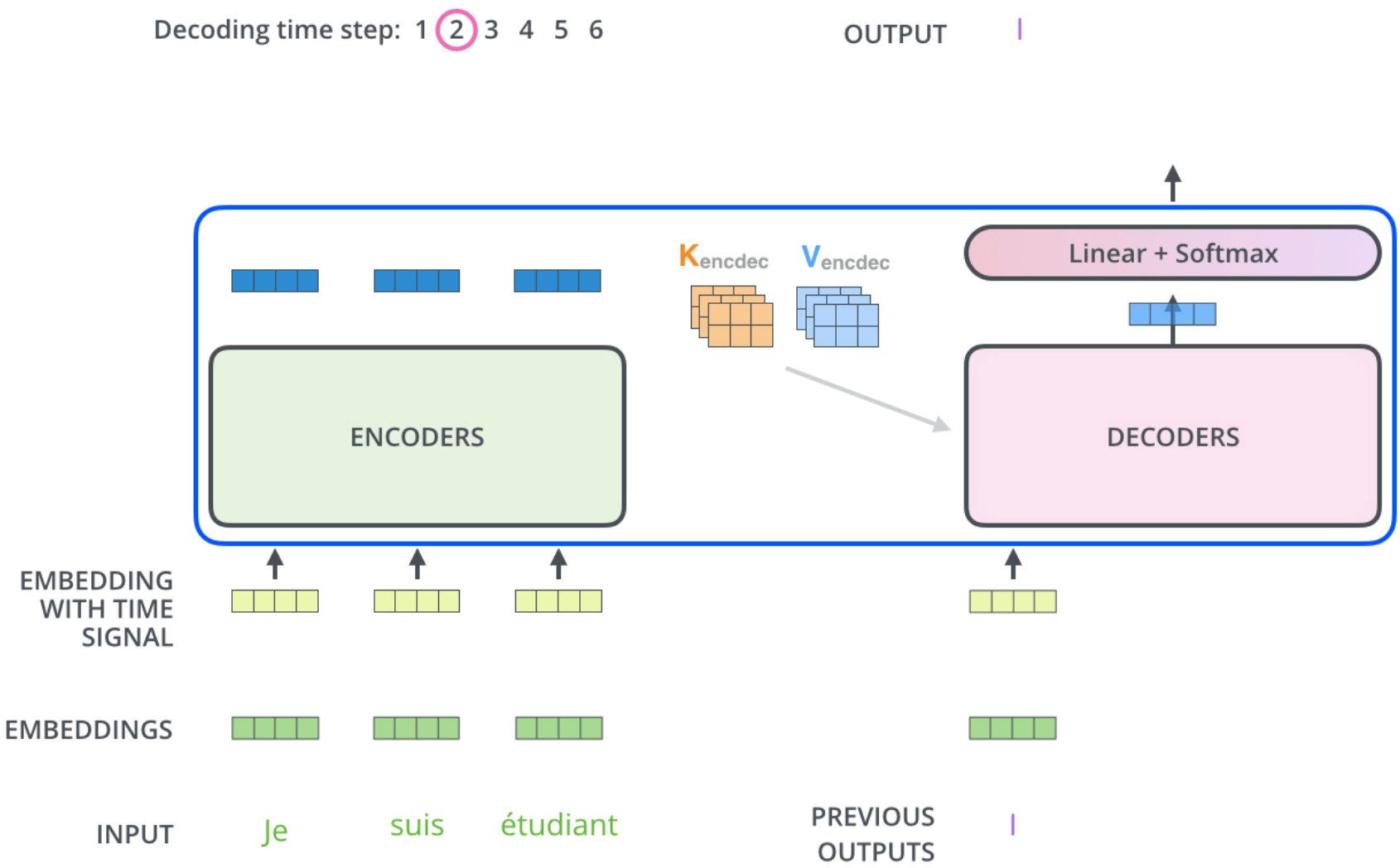
Decoding


Decoding time step: 1 2 3 4 5 6
OUTPUT
I
Linear + Softmax
K_{encdec}
V_{encdec}
ENCODER
ENCODER
DECODER
DECODER
EMBEDDING WITH TIME SIGNAL
EMBEDDINGS
INPUT
Je
suis
étudiant

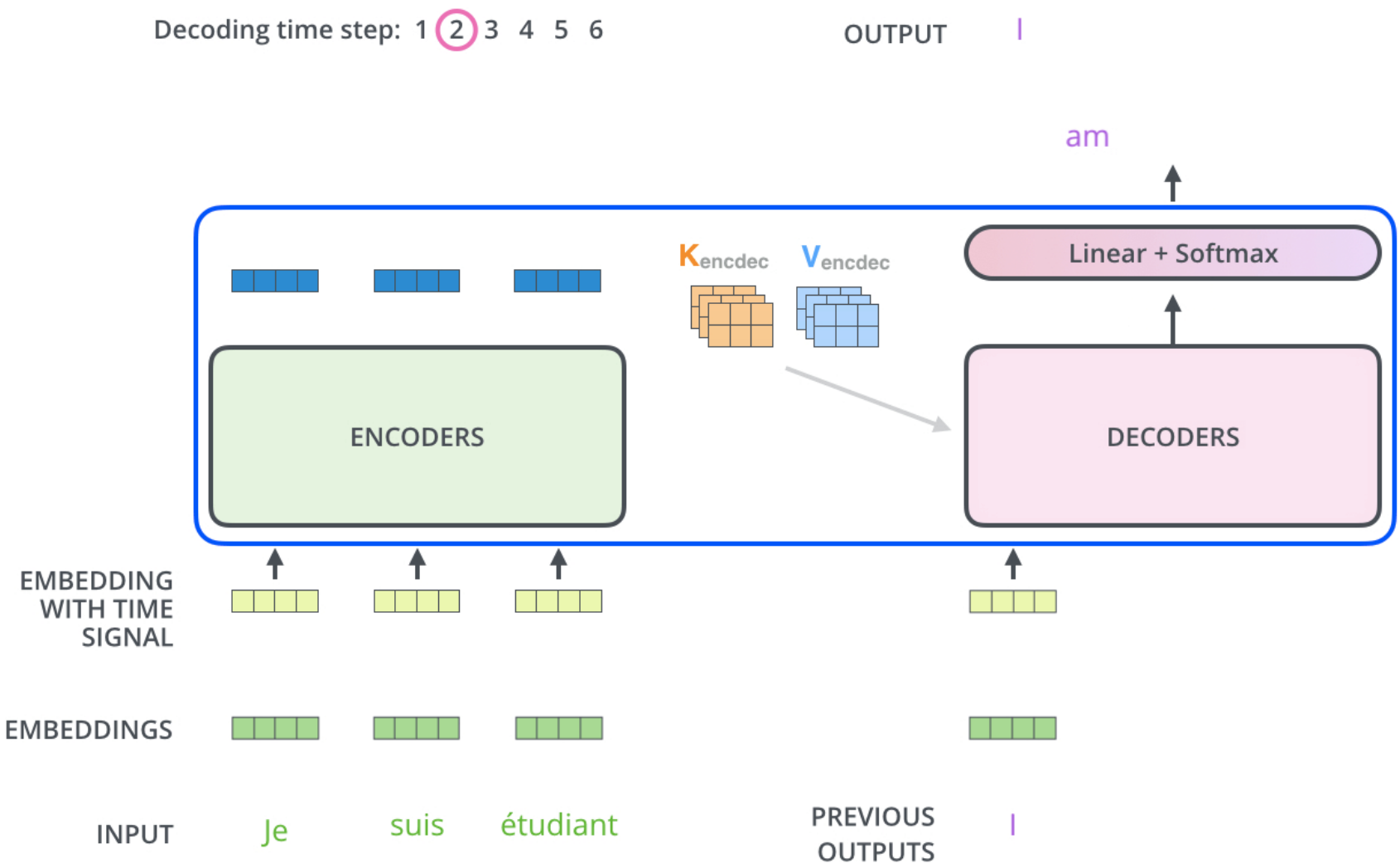
Decoding


Decoding time step: 1 2 3 4 5 6
OUTPUT
I
Linear + Softmax
K_{encdec}
V_{encdec}
ENCODERS
DECODERS
EMBEDDING WITH TIME SIGNAL
EMBEDDINGS
INPUT
Je
suis
étudiant
PREVIOUS OUTPUTS
I

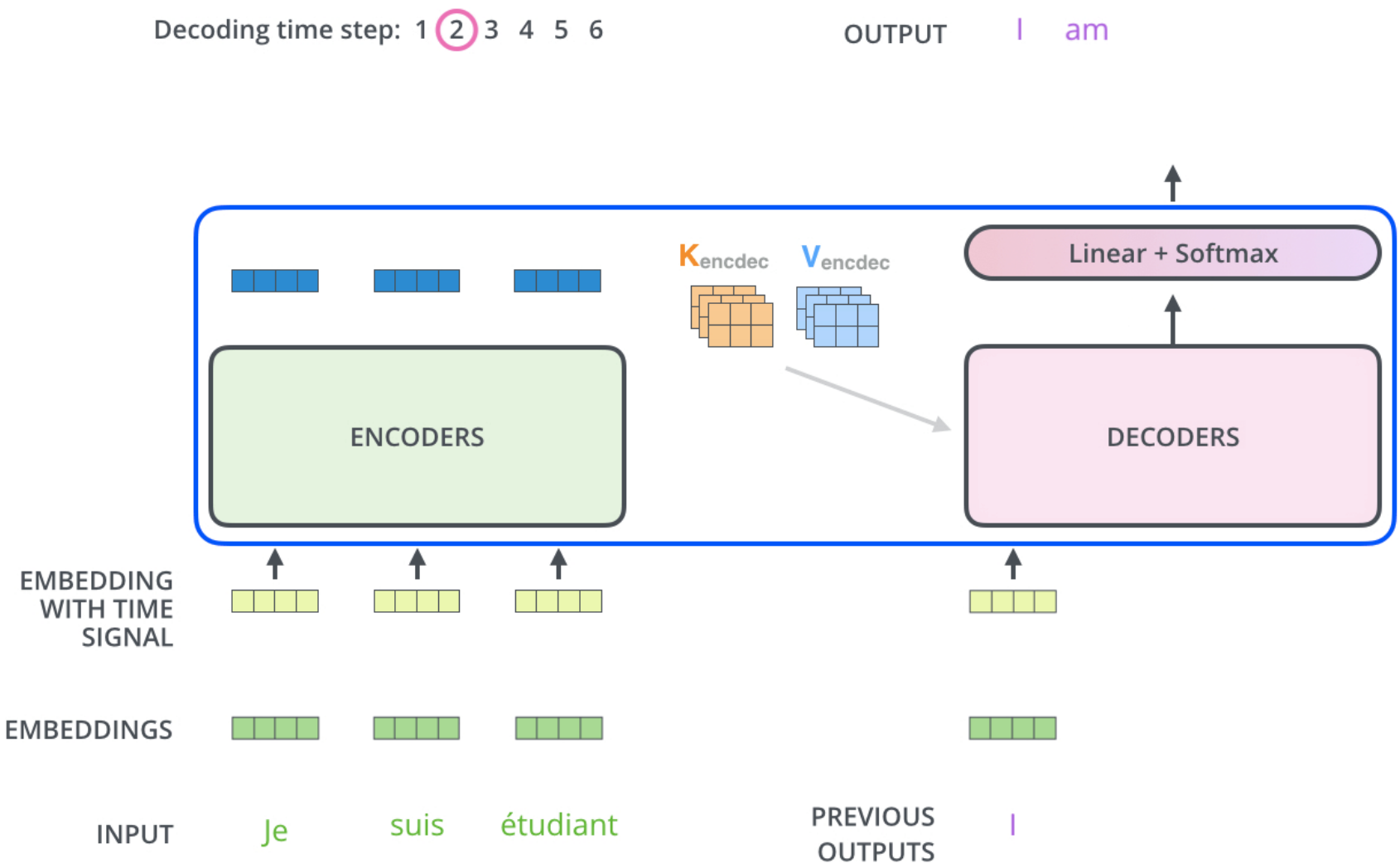
Decoding


Decoding time step: 1 2 3 4 5 6
OUTPUT
I
Linear + Softmax
K_{encdec}
V_{encdec}
ENCODERS
DECODERS
EMBEDDING WITH TIME SIGNAL
EMBEDDINGS
INPUT
Je
suis
étudiant
PREVIOUS OUTPUTS
I

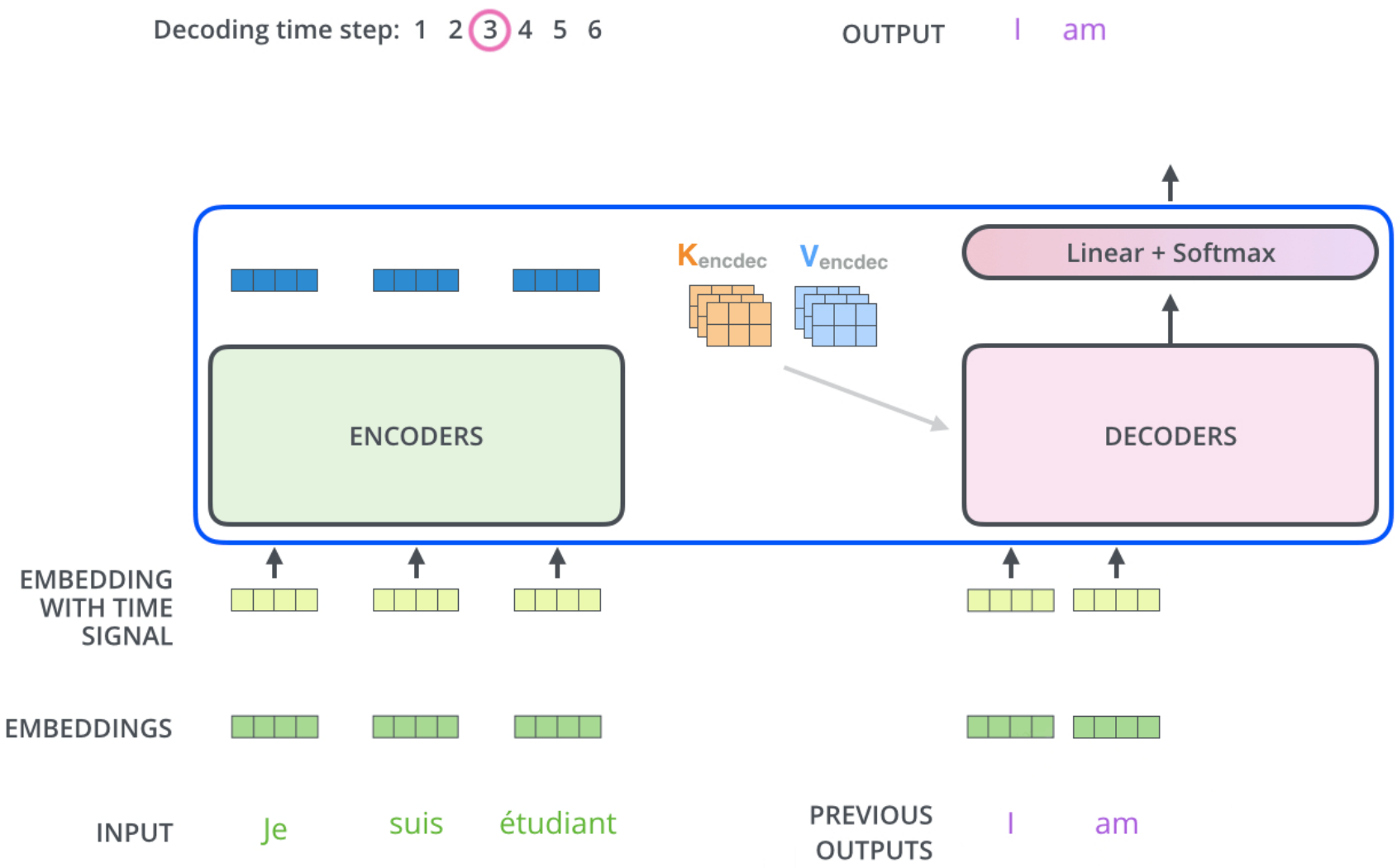
Decoding


Decoding time step: 1 2 3 4 5 6
OUTPUT
I
am
Linear + Softmax
K_{encdec}
V_{encdec}
ENCODERS
DECODERS
EMBEDDING WITH TIME SIGNAL
EMBEDDINGS
INPUT
Je
suis
étudiant
PREVIOUS OUTPUTS
I

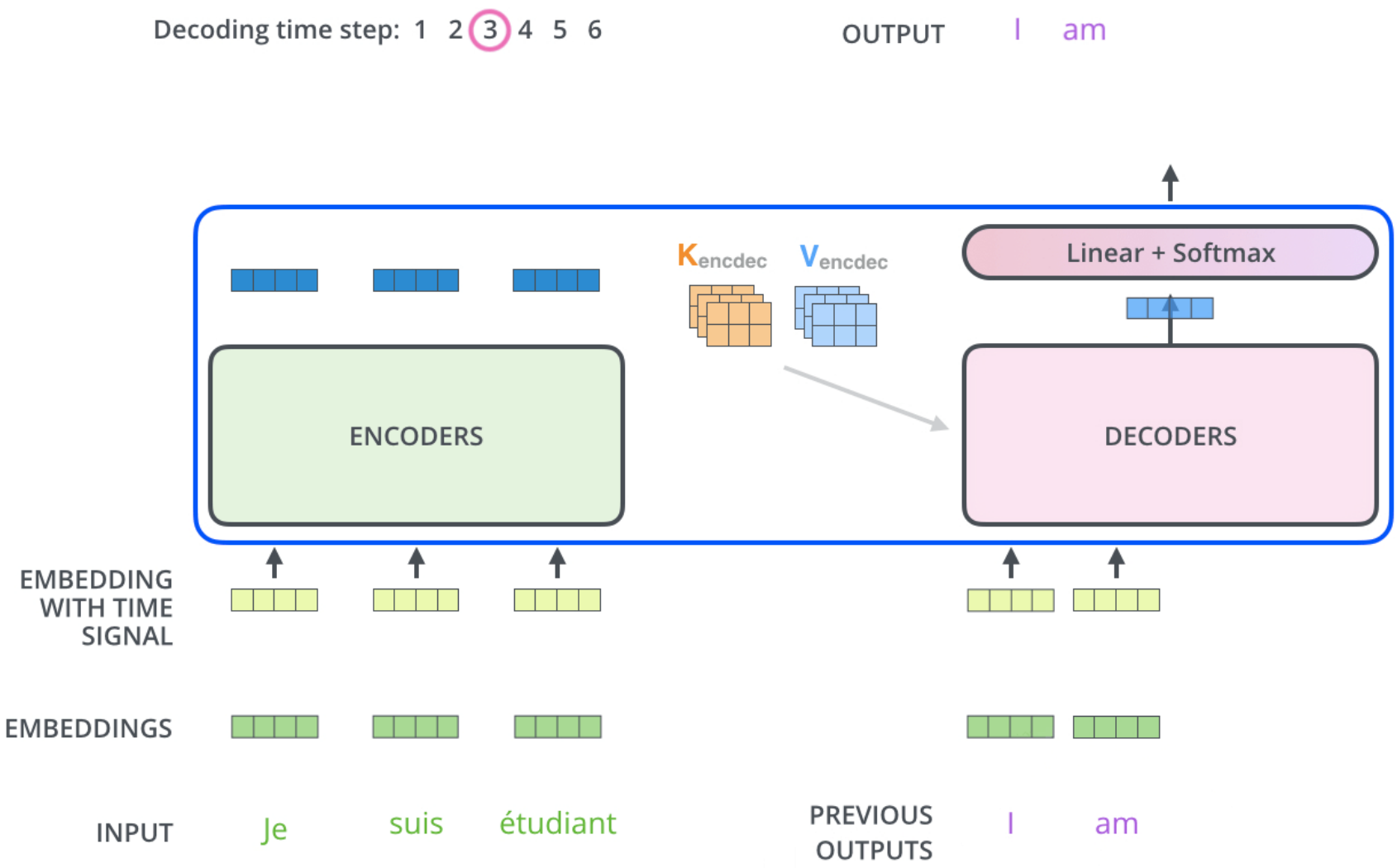
Decoding


Decoding time step: 1 2 3 4 5 6
OUTPUT I am
Linear + Softmax
K_{encdec}
V_{encdec}
ENCODERS
DECODERS
EMBEDDING WITH TIME SIGNAL
EMBEDDINGS
INPUT Je suis étudiant
PREVIOUS OUTPUTS I

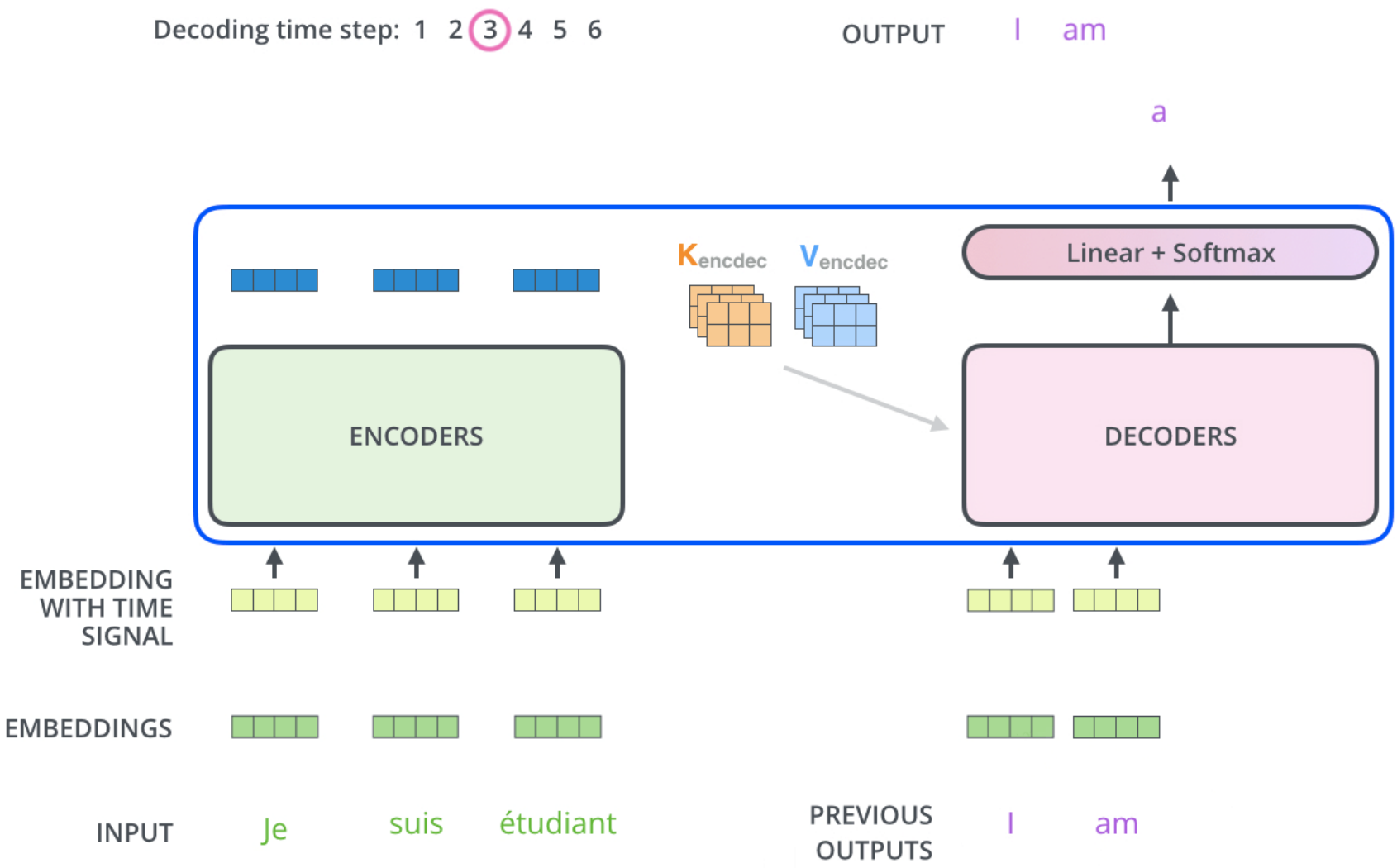
Decoding


Decoding time step: 1 2 3 4 5 6
OUTPUT
I
am
Linear + Softmax
K_{encdec}
V_{encdec}
ENCODERS
DECODERS
EMBEDDING WITH TIME SIGNAL
EMBEDDINGS
INPUT
Je
suis
étudiant
PREVIOUS OUTPUTS
I
am

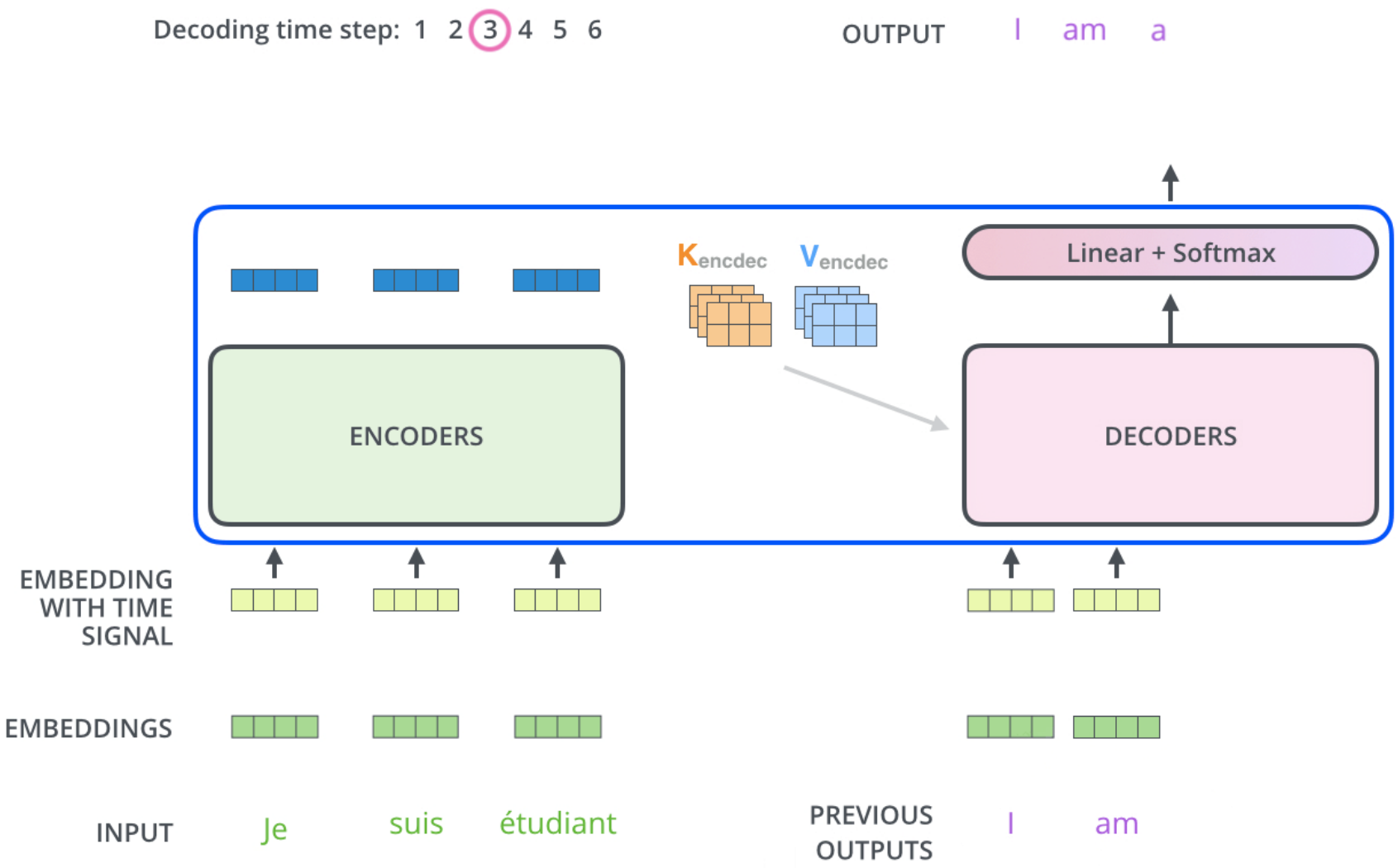
Decoding


Decoding time step: 1 2 3 4 5 6
OUTPUT I am
Linear + Softmax
K_{encdec}
V_{encdec}
ENCODERS
DECODERS
EMBEDDING WITH TIME SIGNAL
EMBEDDINGS
INPUT Je suis étudiant
PREVIOUS OUTPUTS I am

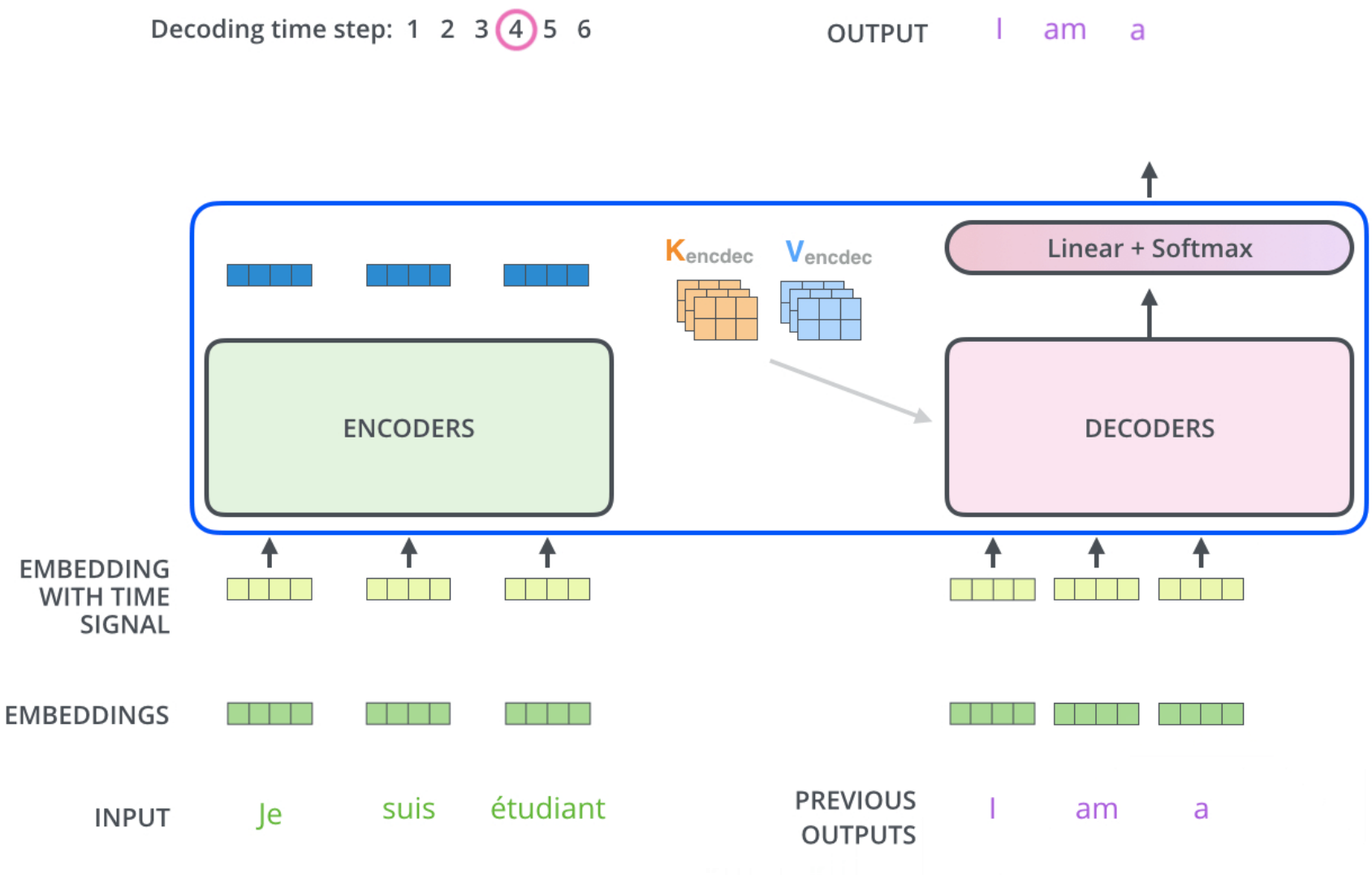
Decoding


Decoding time step: 1 2 3 4 5 6
OUTPUT
I am
a
Linear + Softmax
K_{encdec}
V_{encdec}
ENCODERS
DECODERS
EMBEDDING WITH TIME SIGNAL
EMBEDDINGS
INPUT
Je
suis
étudiant
PREVIOUS OUTPUTS
I
am

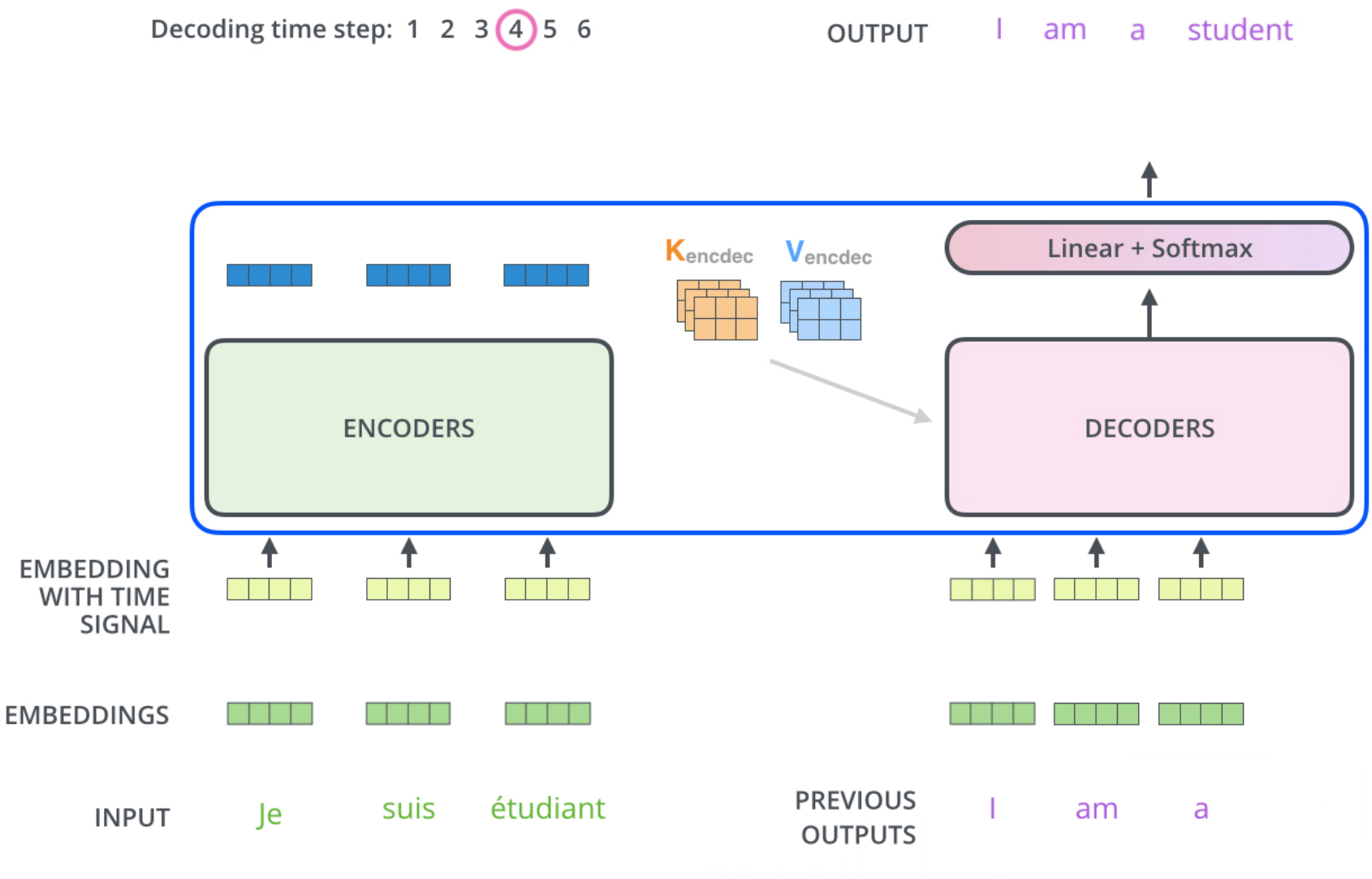
Decoding


Decoding time step: 1 2 3 4 5 6
OUTPUT
I am a
Linear + Softmax
K_{encdec}
V_{encdec}
ENCODERS
DECODERS
EMBEDDING WITH TIME SIGNAL
EMBEDDINGS
INPUT
Je suis étudiant
PREVIOUS OUTPUTS
I am

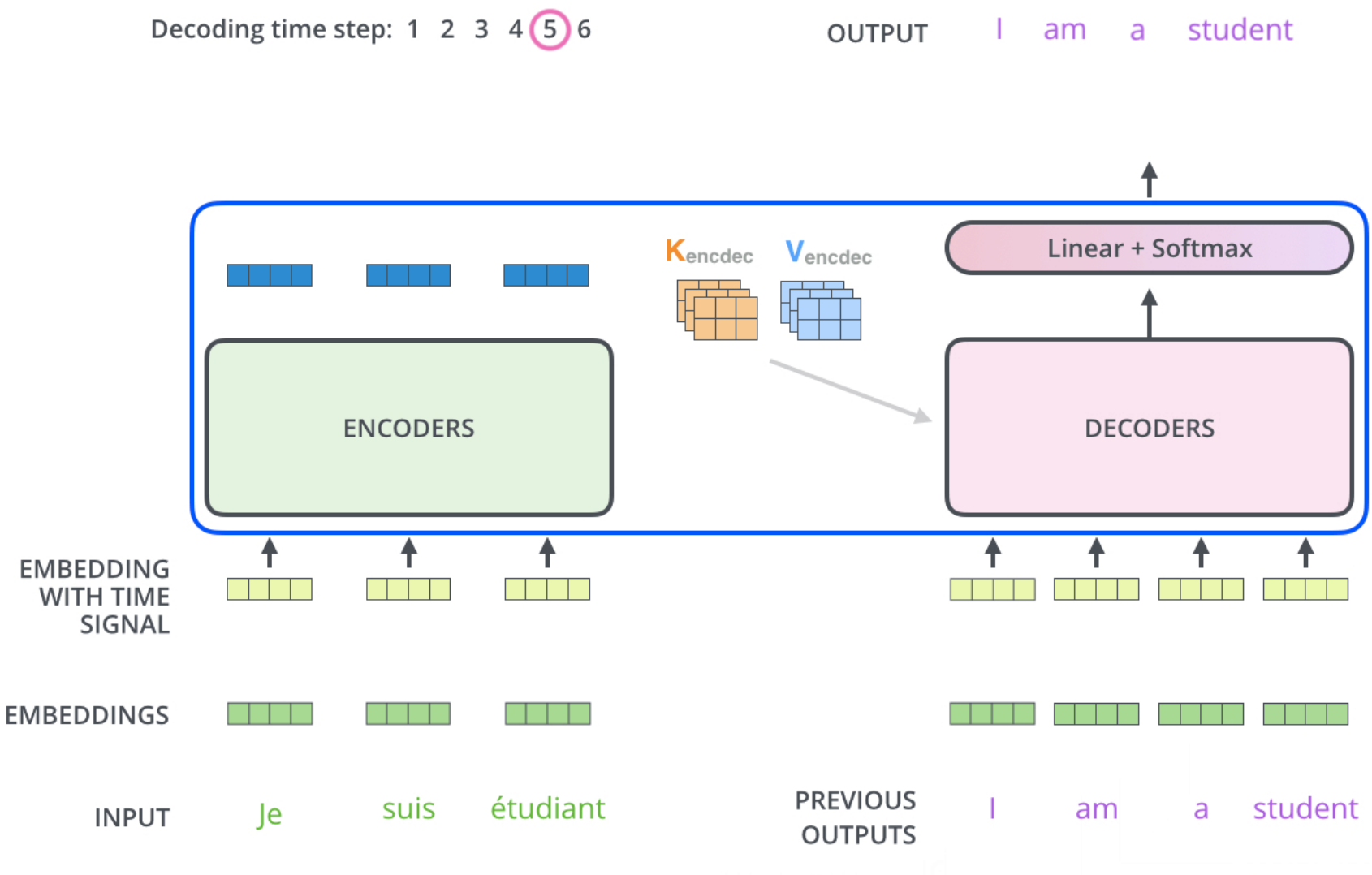
Decoding


Decoding time step: 1 2 3 4 5 6
OUTPUT
I am a
Linear + Softmax
K_{encdec}
V_{encdec}
ENCODERS
DECODERS
EMBEDDING WITH TIME SIGNAL
EMBEDDINGS
INPUT
Je suis étudiant
PREVIOUS OUTPUTS
I am a

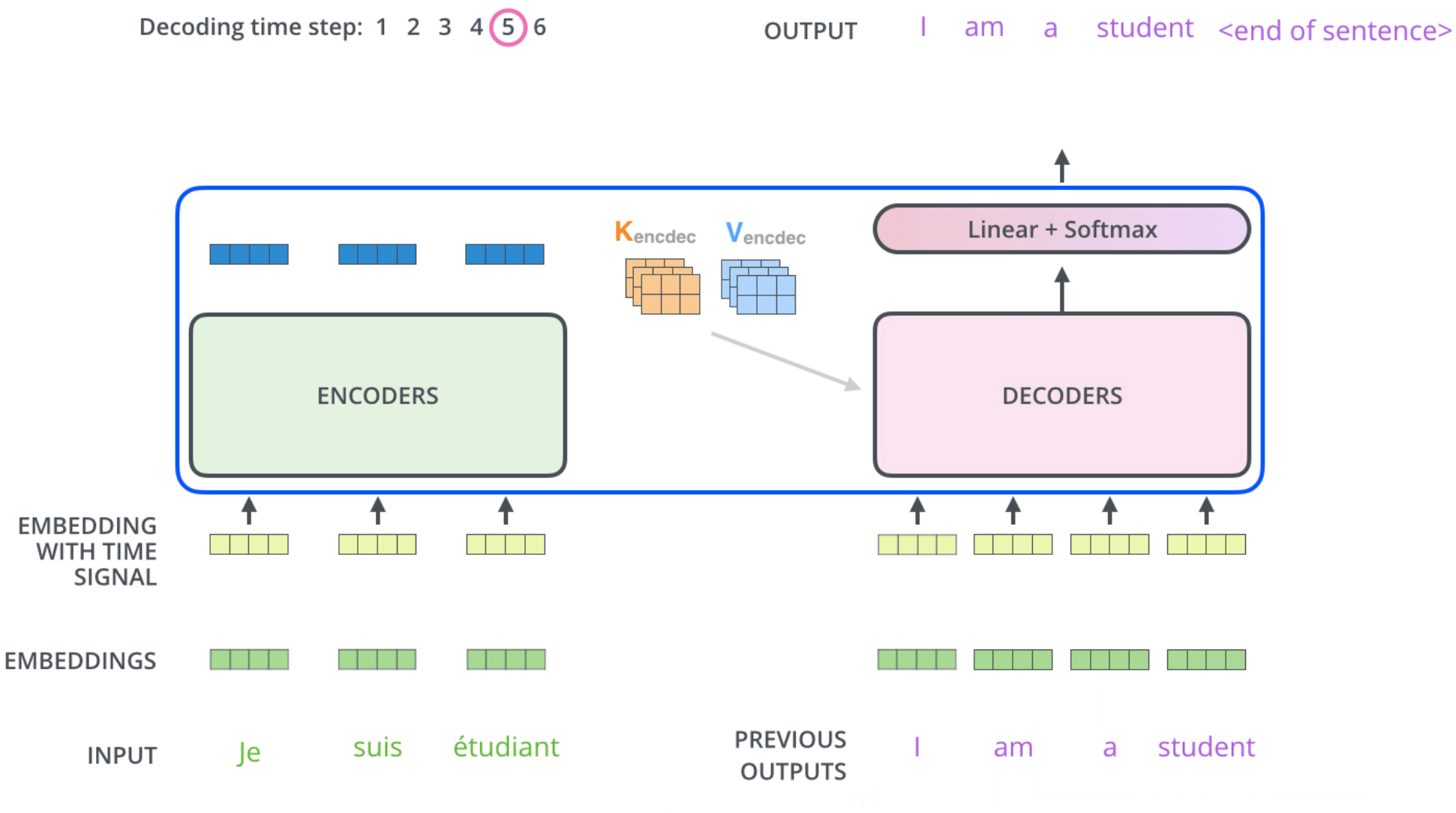
Decoding


Decoding time step: 1 2 3 4 5 6
OUTPUT
I am a student
Linear + Softmax
K_{encdec}
V_{encdec}
ENCODERS
DECODERS
EMBEDDING WITH TIME SIGNAL
EMBEDDINGS
INPUT
Je suis étudiant
PREVIOUS OUTPUTS
I am a

Decoding


Decoding time step: 1 2 3 4 5 6
OUTPUT
I am a student
Linear + Softmax
K_{encdec}
V_{encdec}
ENCODERS
DECODERS
EMBEDDING WITH TIME SIGNAL
EMBEDDINGS
INPUT
Je suis étudiant
PREVIOUS OUTPUTS
I am a student

Decoding


Decoding time step: 1 2 3 4 5 6
OUTPUT
I am a student <end of sentence>
Linear + Softmax
K_{encdec}
V_{encdec}
ENCODERS
DECODERS
EMBEDDING WITH TIME SIGNAL
EMBEDDINGS
INPUT
Je suis étudiant
PREVIOUS OUTPUTS
I am a student

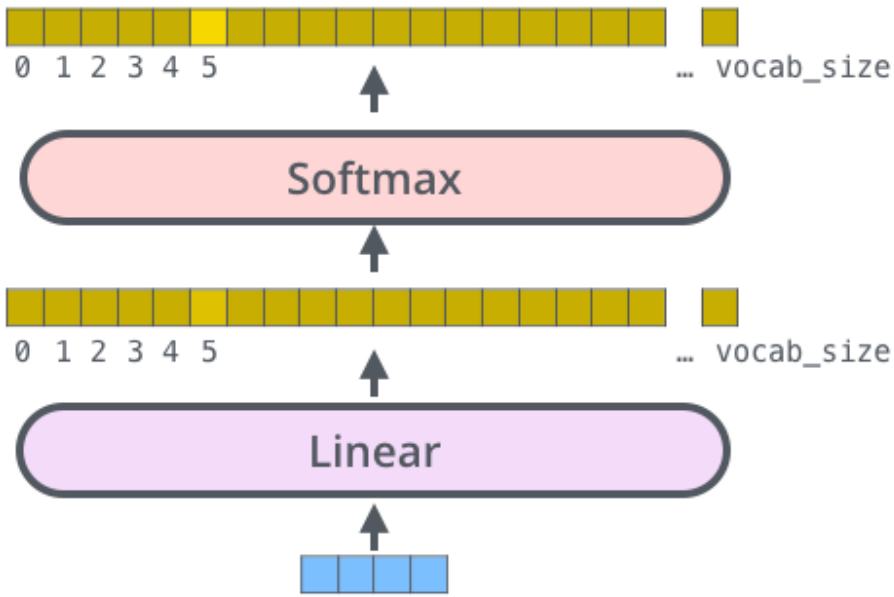
The Final Linear and Softmax Layer

Which word in our vocabulary is associated with this index?

am

Get the index of the cell with the highest value (**argmax**)

5



Results

- Machine Translation: WMT-2014 BLEU

	EN-DE	EN-FR
GNMT (orig)	24.6	39.9
ConvSeq2Seq	25.2	40.5
Transformer*	**28.4**	**41.8**

Transformer models trained >3x faster than the others

Content

Content

How can we define a language? (Recap)

— The probabilistic approach

- A language can also be defined as a probabilistic distribution over all the possible sentences.
- A statistical language model is a probability distribution over sequences of words (sentences) in a given language L:

$$\sum_{s \in V^+} P_{LM}(s) = 1$$

- Or:

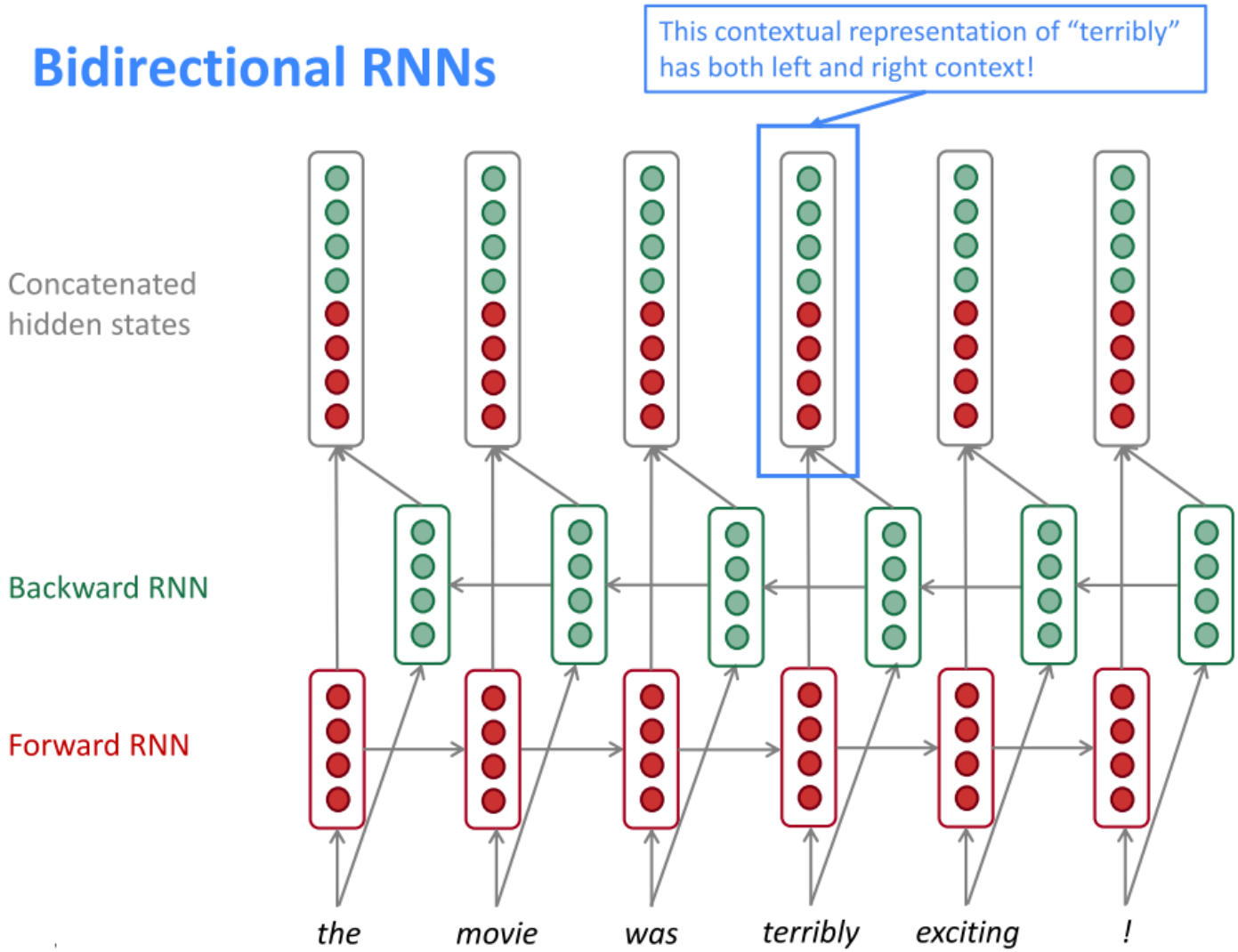
$$\sum_{\substack{s = w_1 w_2 \ldots w_n \\ w_i \in V, n > 0}} P_{LM}(s) = 1$$

NLP tasks as Conditional LMs (Recap)

x "input"	w "**text** output"
An author	A document written by that author
A topic label	An article about that topic
{SPAM, NOT_SPAM}	An email
A sentence in French	Its English translation
A sentence in English	Its French translation
A sentence in English	Its Chinese translation
An image	A text description of the image
A document	Its summary
A document	Its translation
Meterological measurements	A weather report
Acoustic signal	Transcription of speech
Conversational history + database	Dialogue system response
A question + a document	Its answer
A question + an image	Its answer

Chris Dyer, Conditional LMs (slides)

Bidirectional RNN Language Models (Recap)



Christopher Manning, Natural Language Processing with Deep Learning, Standford U. CS224n

Language models as encoders

- The motivation of language modeling is to estimate to what extent a word sequence is fluent in a certain language.
- A neural language model not only gives a scoring for a sentence, but also provides a contextualized word embedding for each word in the sentence.
- As contextualized word embeddings, NLM could be used as features for other NLP tasks, which may be better than context free word embeddings.

Language models as encoders

- The success of NMT has proved that language models are good encoders which keep the meaning of the source text for generating its translation.
- Further, the training data for a NLM could be any unannotated text, which means we can use nearly unlimited raw text to train an NLM and obtain contextualized word embeddings to help other NLP tasks.

Pre-trained language models (PLMs): introduction

- A Pre-trained Language Model (PLM) is a neural language model which is pre-trained on a large amount of unannotated text, and fune-tuned on task specific annotated data to solve downstream NLP tasks.
- The use of PLMs in NLP is similar to the use of image classification models pre-trained on ImageNet dataset in computer vision.

Pre-trained language models (PLMs): introduction

- PLMs provide contextualized word embeddings, rather than context free word embeddings like word2vec.
- PLMs adopt an unsupervised (or self-supervised) leaning method.
- Applying PLMs in other NLP tasks could be viewed as an inductive transfer learning method.

Inductive Transfer Learning

- Transfer Learning: Transfer learning is a machine learning method where a model developed for a task is reused as the starting point for a model on a second task.
- Inductive Transfer Learning vs Transductive Transfer Learning:
  - Inductive Transfer Learning: downstream task is unknown.
  - Transductive Transfer Learning: downstream task is known.
- PLMs adopt a inductive transfer learning method, which means it can be used in any downstream tasks.

Content

BERT: Bidirectional Encoder Representations from Transformers

- Main ideas
  - Propose a new pre-training objective so that a deep bidirectional Transformer can be trained
    - **The "masked language model" (MLM)**: the objective is to predict the original word of a masked word based only on its context
    - "**Next sentence prediction**"
- Merits of BERT
  - Just fine-tune BERT model for specific tasks to achieve state-of-the-art performance
  - BERT advances the state-of-the-art for eleven NLP tasks

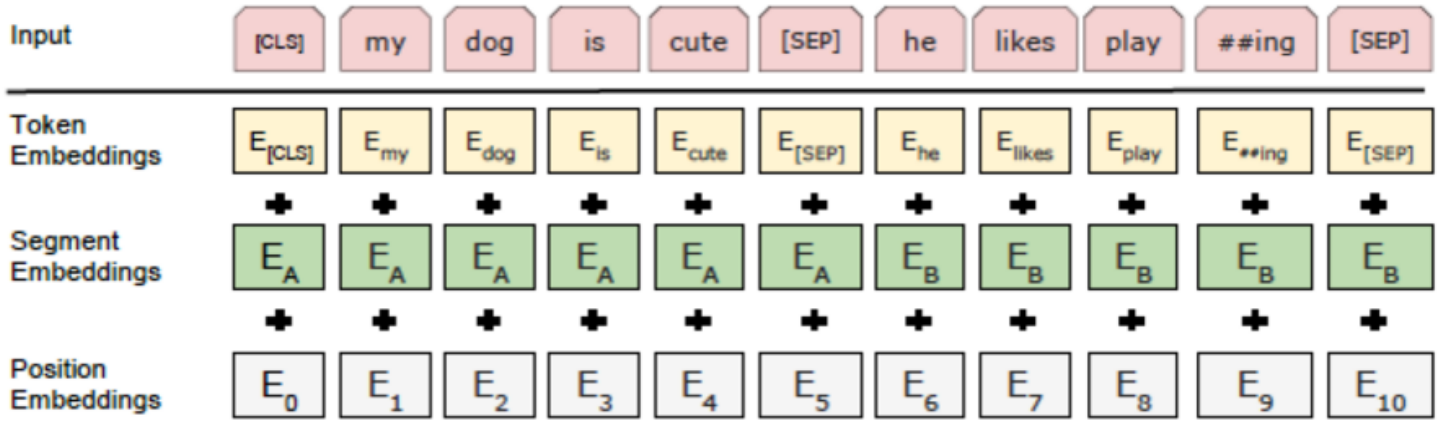
Devlin et al., 2018, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (Slides))

BERT - model architecture

- BERT's model architecture is a multi-layer **bidirectional Transformer encoder**
  - (Vaswani et al., 2017) "Attention is all you need"
- Two models with different sizes were investigated
  - $\text{BERT}_{\text{BASE}}$: L=12, H=768, A=12, Total Parameters=110M
    - (*L*: number of layers (Transformer blocks), *H* is the hidden size, *A*: the number of self-attention heads)
  - $\text{BERT}_{\text{LARGE}}$: L=24, H=1024, A=16, Total Parameters=340M

Devlin et al., 2018, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (Slides))

BERT: Input Representation



- Token Embeddings: Use pretrained WordPiece embeddings
- Position Embeddings: Use learned Position Embeddings
  - Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. Convolutionalsequence to sequence learning. arXiv preprint arXiv:1705.03122v2, 2017.
- Added sentence embedding to every tokens of each sentence
- Use [CLS] for the classification tasks
- Separate sentences by using a special token [SEP]

Devlin et al., 2018, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (Slides))

BERT: pre-training tasks

Task#1: Masked LM

- 15% of the words are masked at random
  - and the task is to predict the masked words based on its left and right context
- Not all tokens were masked in the same way (example sentence “My dog is hairy”)
  - 80% were replaced by the <MASK> token: “My dog is <MASK>”
  - 10% were replaced by a random token: “My dog is apple”
  - 10% were left intact: “My dog is hairy”

Devlin et al., 2018, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (Slides))

BERT: pre-training tasks

Task#2: Next Sentence Prediction

- Motivation
  - Many downstream tasks are based on understanding the relationship between two text sentences
    - Question Answering (QA) and Natural Language Inference (NLI)
  - Language modeling does not directly capture that relationship
- The task is pre-training binarized next sentence prediction task

Input = [CLS] the man went to [MASK] store [SEP] he bought a gallon [MASK] milk [SEP]
Label = isNext

Input = [CLS] the man [MASK] to the store [SEP]penguin [MASK] are flight ##less birds [SEP]
Label = NotNext

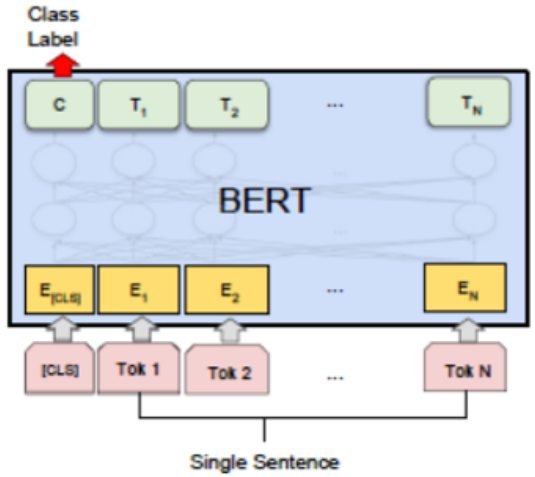
Devlin et al., 2018, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (Slides))

BERT: fine-tuning procedure

Fine-tuning procedure

- For sequence-level classification task
  - Obtain the representation of the input sequence by using the final hidden state (hidden state at the position of the special token [CLS]) $C \in R^H$
  - Just add a classification layer and use softmax to calculate label probabilities. Parameters W $\in R^{K \times H}$

$$P = \text{softmax}(CW^T)$$



Devlin et al., 2018, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (Slides))

BERT: fine-tuning procedure

Fine-tuning procedure

- For sequence-level classification task
  - All of the parameters of BERT and W are fine-tuned jointly
- Most model hyperparameters are the same as in pre-training
  - except the batch size, learning rate, and number of training epochs

Devlin et al., 2018, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (Slides))

BERT: fine-tuning procedure

Fine-tuning procedure

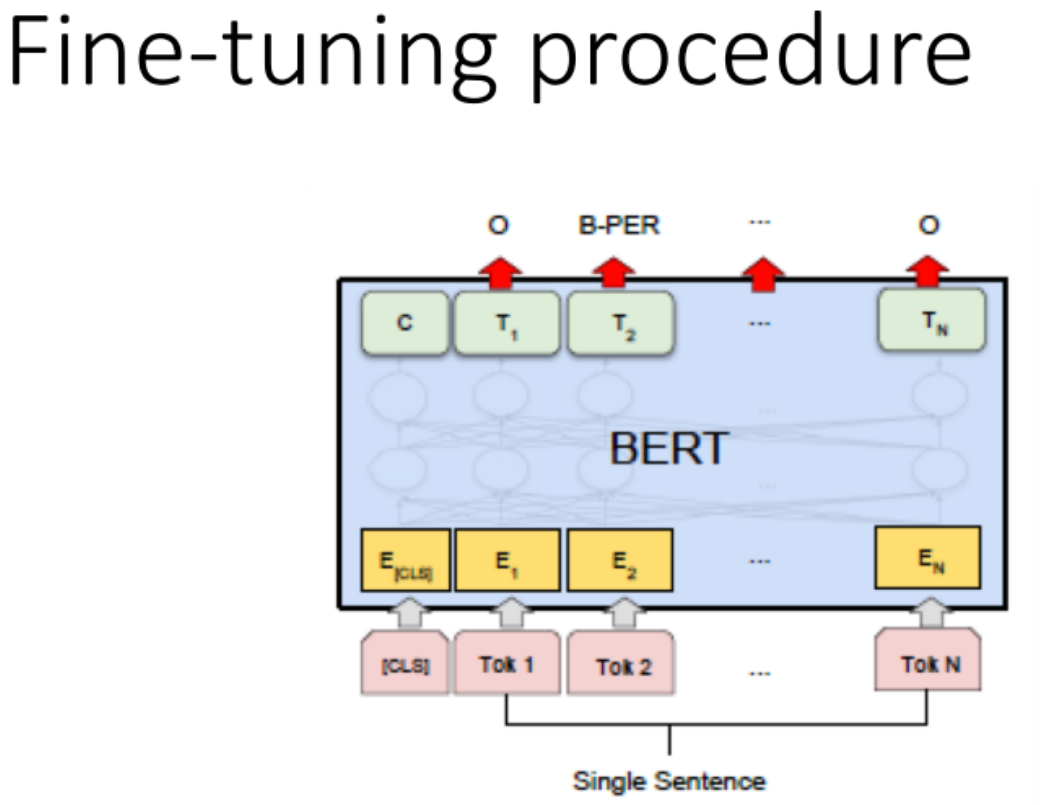
- Token tagging task (e.g., Named Entity Recognition)
  - Feed the final hidden representation $T_i \in R^H$ for each token i into a classification layer for the tagset (NER label set)
  - To make the task compatible with WordPiece tokenization

```
Jim   Hen   ##son was a puppet ##eer
I-PER I-PER X     O   O O      X
```

  - Predict the tag for the first sub-token of a word
  - No prediction is made for X

Devlin et al., 2018, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (Slides))

BERT: fine-tuning procedure



Single Sentence Tagging Tasks: CoNLL-2003 NER

Devlin et al., 2018, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (Slides))

BERT: fine-tuning procedure

Fine-tuning procedure

- Span-level task: SQuAD v1.1
  - Input Question:
    `Where do water droplets collide with ice crystals to form precipitation?`
  - Input Paragraph:
    `.... Precipitation forms as smaller dropletscoalesce via collision with other rain dropsor ice crystals within a cloud. ...`
  - Output Answer:
    `within a cloud`

Devlin et al., 2018, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (Slides))

BERT: fine-tuning procedure

Fine-tuning procedure

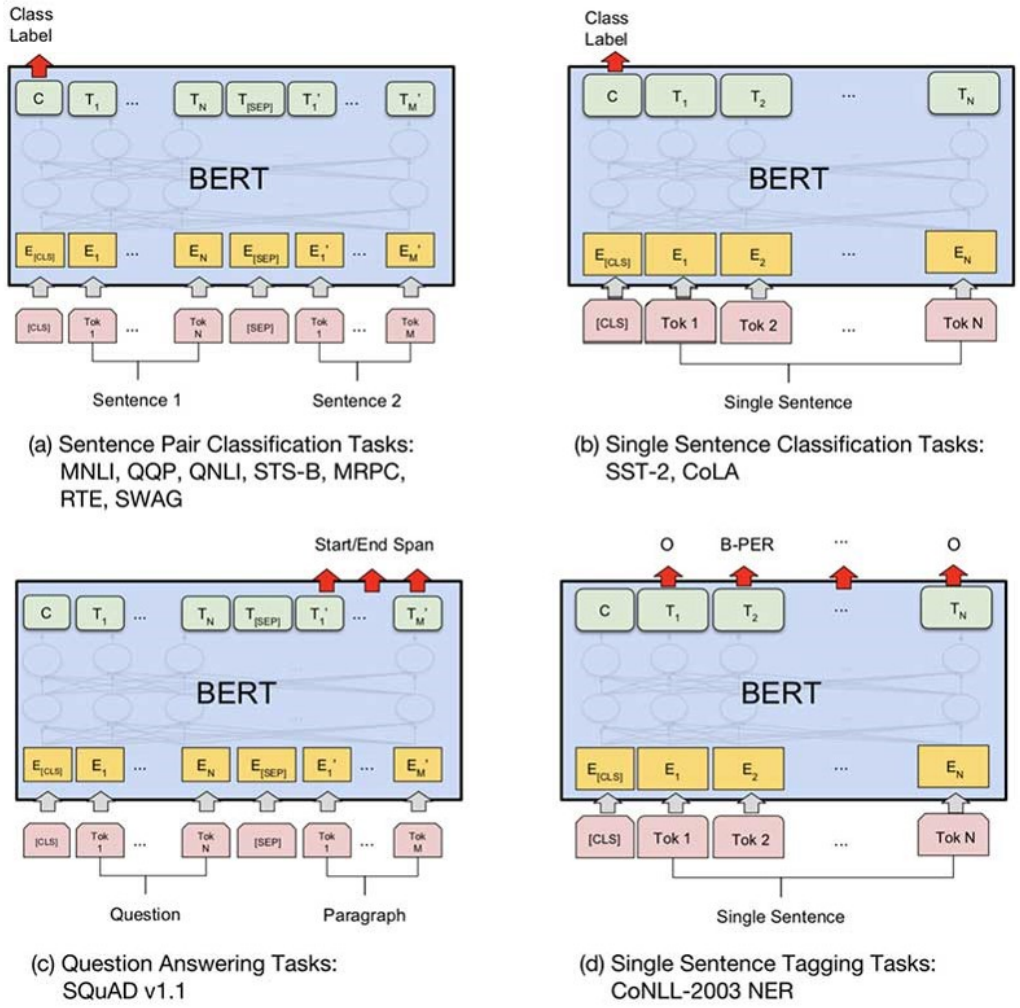
- Span-level task: SQuAD v1.1
  - Represent the input question and paragraph as a single packed sequence
    - The question uses the A embedding and the paragraph uses the B embedding
  - New parameters to be learned in fine-tuning are start vector $S \in \mathbb{R}^H$ and end vector $E \in \mathbb{R}^H$
  - Calculate the probability of word i being the start of the answer span

$$P_i = \frac{e^{S \cdot T_i}}{\sum_j e^{S \cdot T_j}}$$

  - The training objective is the log-likelihood the correct and end positions

Devlin et al., 2018, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (Slides))

BERT: fine-tuning summary



Devlin et al., 2018, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (Slides))

BERT: experiments

Experiments

- GLUE (General Language Understanding Evaluation) benchmark
  - Distribute canonical Train, Dev and Test splits
  - Labels for Test set are not provided
- Datasets in GLUE:
  - MNLI: Multi-Genre Natural Language Inference
  - QQP: Quora Question Pairs
  - QNLI: Question Natural Language Inference
  - SST-2: Stanford Sentiment Treebank
  - CoLA: The corpus of Linguistic Acceptability
  - STS-B: The Semantic Textual Similarity Benchmark
  - MRPC: Microsoft Research Paraphrase Corpus
  - RTE: Recognizing Textual Entailment
  - WNLI: Winograd NLI

Devlin et al., 2018, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (Slides))

BERT: GLUE results

GLUE Results

System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	Average -
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.9	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	88.1	91.3	45.4	80.0	82.3	56.0	75.2
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.1	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**91.1**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**81.9**

Table 1: GLUE Test results, scored by the GLUE evaluation server. The number below each task denotes the number of training examples. The "Average" column is slightly different than the official GLUE score, since we exclude the problematic WNLI set. OpenAI GPT = (L=12, H=768, A=12); $\text{BERT}_{\text{BASE}}$ = (L=12, H=768, A=12); $\text{BERT}_{\text{LARGE}}$ = (L=24, H=1024, A=16). BERT and OpenAI GPT are single-model, single task. All results obtained from https://gluebenchmark.com/leaderboard and https://blog.openai.com/language-unsupervised/.

Devlin et al., 2018, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (Slides))

BERT: SQuAD v1.1 results

SQuAD v1.1

System	Dev EM	Dev F1	Test EM	Test F1
Leaderboard (Oct 8th, 2018)				
Human	-	-	82.3	91.2
#1 Ensemble - nlnet	-	-	86.0	91.7
#2 Ensemble - QANet	-	-	84.5	90.5
#1 Single - nlnet	-	-	83.5	90.1
#2 Single - QANet	-	-	82.5	89.3
Published				
BiDAF+ELMo (Single)	-	85.8	-	-
R.M. Reader (Single)	78.9	86.3	79.5	86.6
R.M. Reader (Ensemble)	81.2	87.9	82.3	88.5
Ours				
$\text{BERT}_{\text{BASE}}$ (Single)	80.8	88.5	-	-
$\text{BERT}_{\text{LARGE}}$ (Single)	84.1	90.9	-	-
$\text{BERT}_{\text{LARGE}}$ (Ensemble)	85.8	91.8	-	-
$\text{BERT}_{\text{LARGE}}$ (Sgl.+TriviaQA)	**84.2**	**91.1**	**85.1**	**91.8**
$\text{BERT}_{\text{LARGE}}$ (Ens.+TriviaQA)	**86.2**	**92.2**	**87.4**	**93.2**

Table 2: SQuAD results. The BERT ensemble is 7x systems which use different pre-training checkpoints and fine-tuning seeds.

Reference: https://rajpurkar.github.io/SQuAD-explorer

Devlin et al., 2018, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (Slides))

Content

GPT: introduction

- Following the similar idea of ELMo, OpenAI GPT, short for Generative Pre-training Transformer (Radford et al., 2018), expands the unsupervised language model to a much larger scale by training on a giant collection of free text corpora.
- Despite of the similarity, GPT has two major differences from ELMo:
  - The model architectures are different: ELMo uses a shallow concatenation of independently trained left-to-right and right-to-left multi-layer LSTMs, while GPT is a multi-layer transformer decoder.
  - The use of contextualized embeddings in downstream tasks are different: ELMo feeds embeddings into models customized for specific tasks as additional features, while GPT fine-tunes the same base model for all end tasks.

Liliang Wen, Generalized Language Models: ULMFiT & OpenAI GPT (blog)

GPT: Transformer Decoder as Language Model

- Compared to the original transformer architecture, the transformer decoder model discards the encoder part, so there is only one single input sentence rather than two separate source and target sequences.
- This model applies multiple transformer blocks over the embeddings of input sequences. Each block contains a masked multi-headed self-attention layer and a pointwise feed-forward layer. The final output produces a distribution over target tokens after softmax normalization.

Liliang Wen, Generalized Language Models: Ulmfit & OpenAI GPT (blog)