TRANSFORMER MODELS

TODAY’S CLASS

GOALS

1. Review Contextual Embeddings
2. Go over transformer architecture
    1. Introduce components not yet covered
3. Discuss LLM and their use

CONTEXTUAL EMBEDDINGS

WHY WE NEED THEM

“I hope her dog doesn’t **bark** when I knock on the door.”

“The cedar’s tree **bark** has been used to make clothing, baskets, and many more items”

“Chocolate and mint are perfect partners in this delicious **bark** that makes a lovely gift”

The word embedding for bark contains information for all these cases

The rest of the network has to learn to filter out the information that is not necessary

CONTEXTUAL EMBEDDINGS

WHY WE NEED THEM

Transform the word embedding:

$$E_i' = E_i + \Delta_i$$

Compute Δ_i with Self-Attention:

$Q_i = E_i W_Q$ — query vector (bark looks for dog)

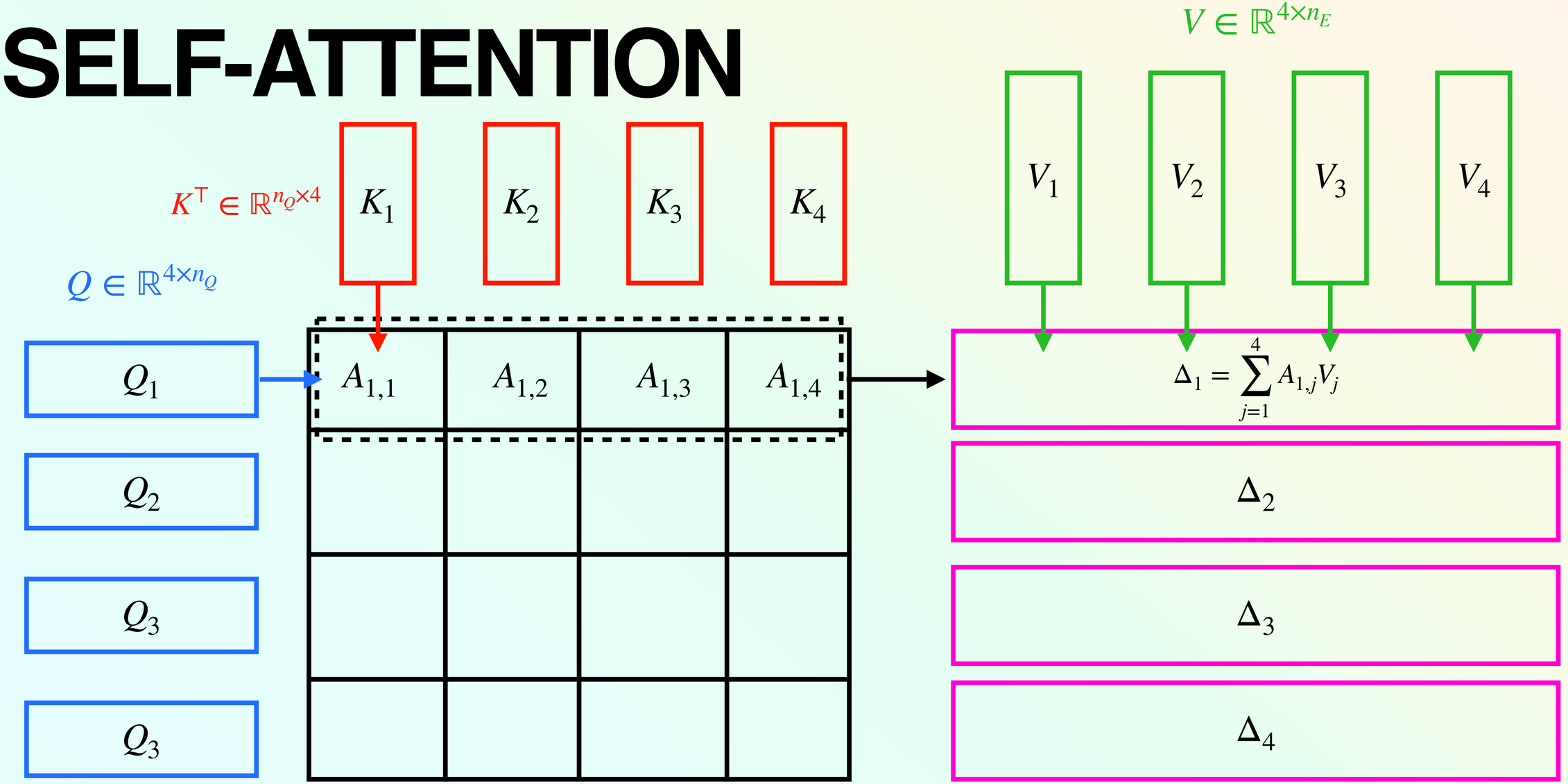
$K_i = E_i W_E$ — key vector (same size as Q_i)

$V_i = E_i W_V$ — Value vector (same size as E_i)

$$A_{i,j} = \frac{e^{Q_i K_j^\top}}{\sum_{j'=1}^{n} e^{Q_i K_{j'}^\top}}$$ — Attention that embedding i gives to embedding j

$$\Delta_i = \sum_{j=1}^{n} A_{i,j} V_j$$

SELF-ATTENTION



Q, K, V computed with matrix multiplication

The attention matrix computed from the dot product of Q_i and K_j. Softmax applied along rows.

Change to embedding vector

SELF-ATTENTION

BREAK DOWN

Quiz

SELF-ATTENTION

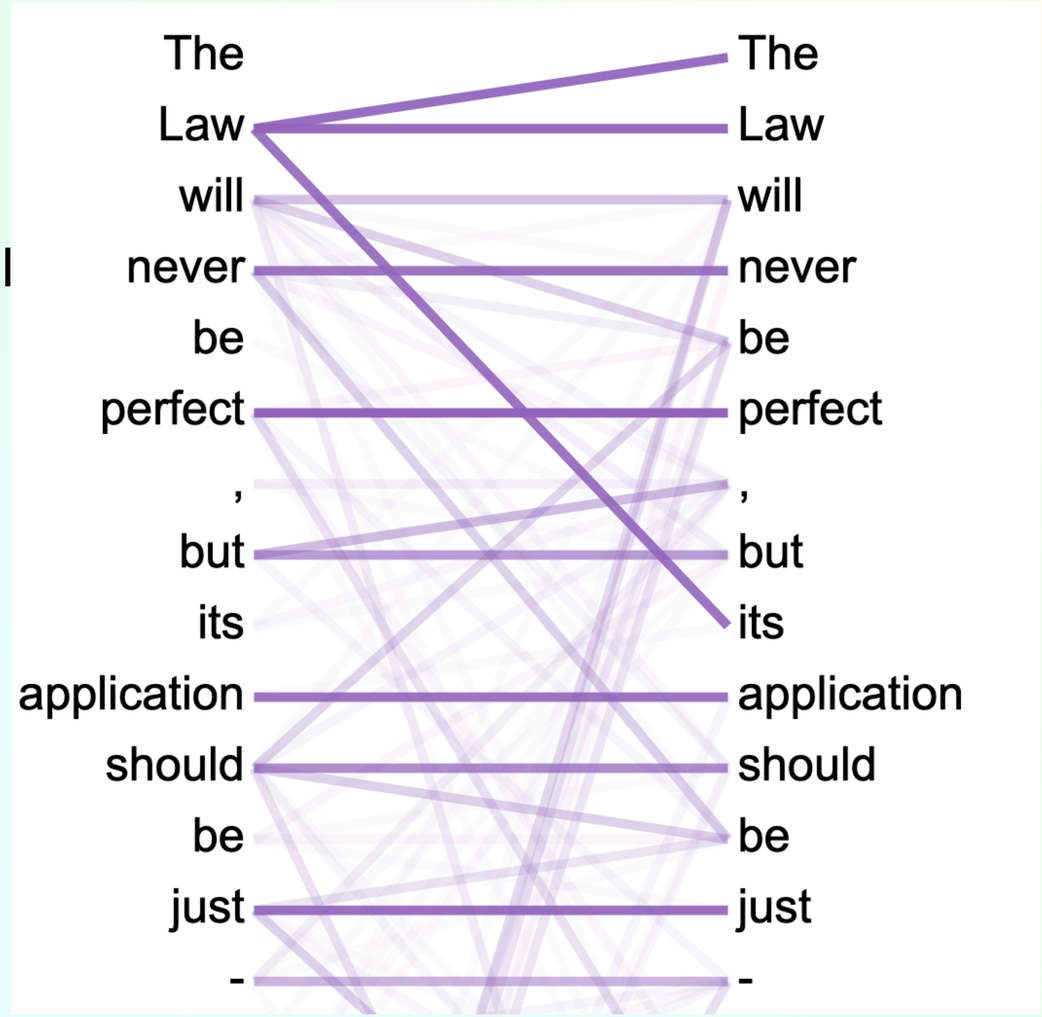
BREAK DOWN

Q_i — each word needs to look for different contexts

SELF-ATTENTION

BREAK DOWN

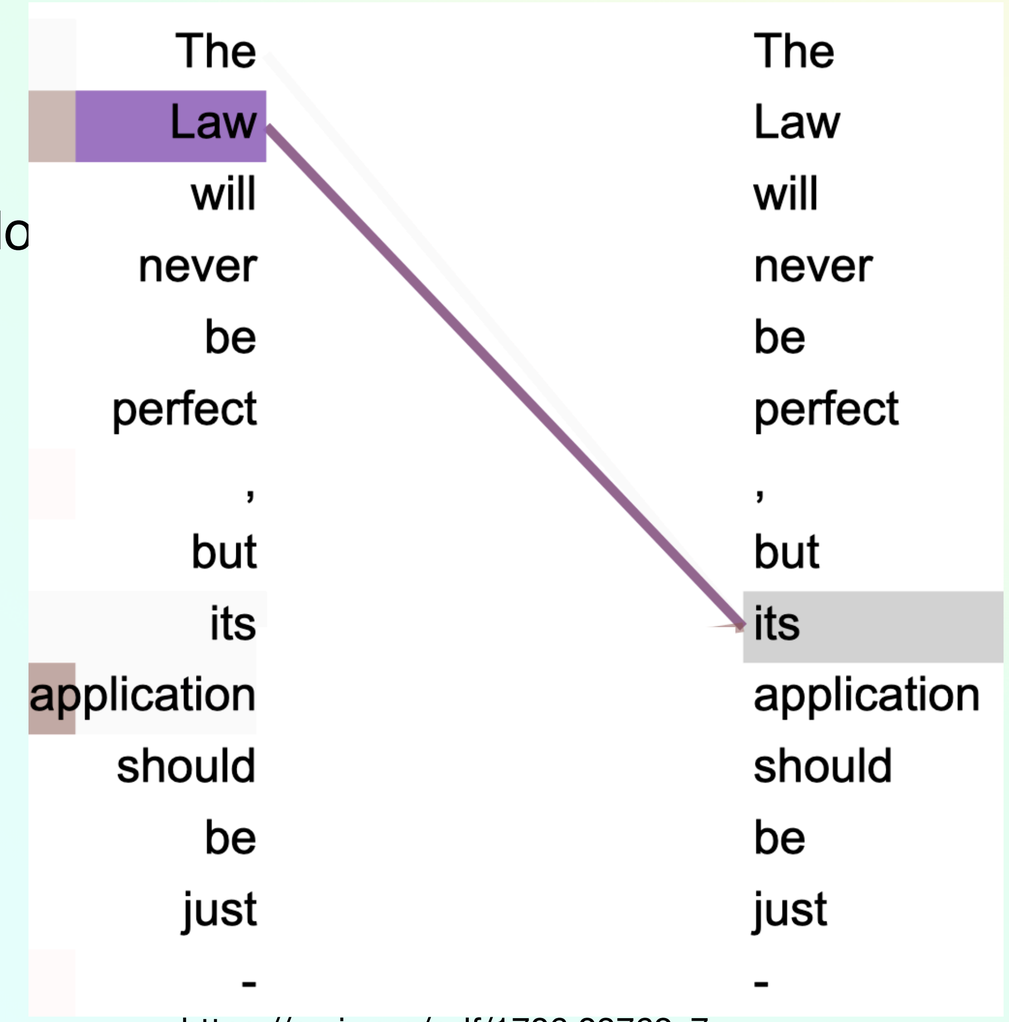
Q_i — each word needs to l



SELF-ATTENTION

BREAK DOWN

Q_i — each word needs to lo



https://arxiv.org/pdf/1706.03762v7

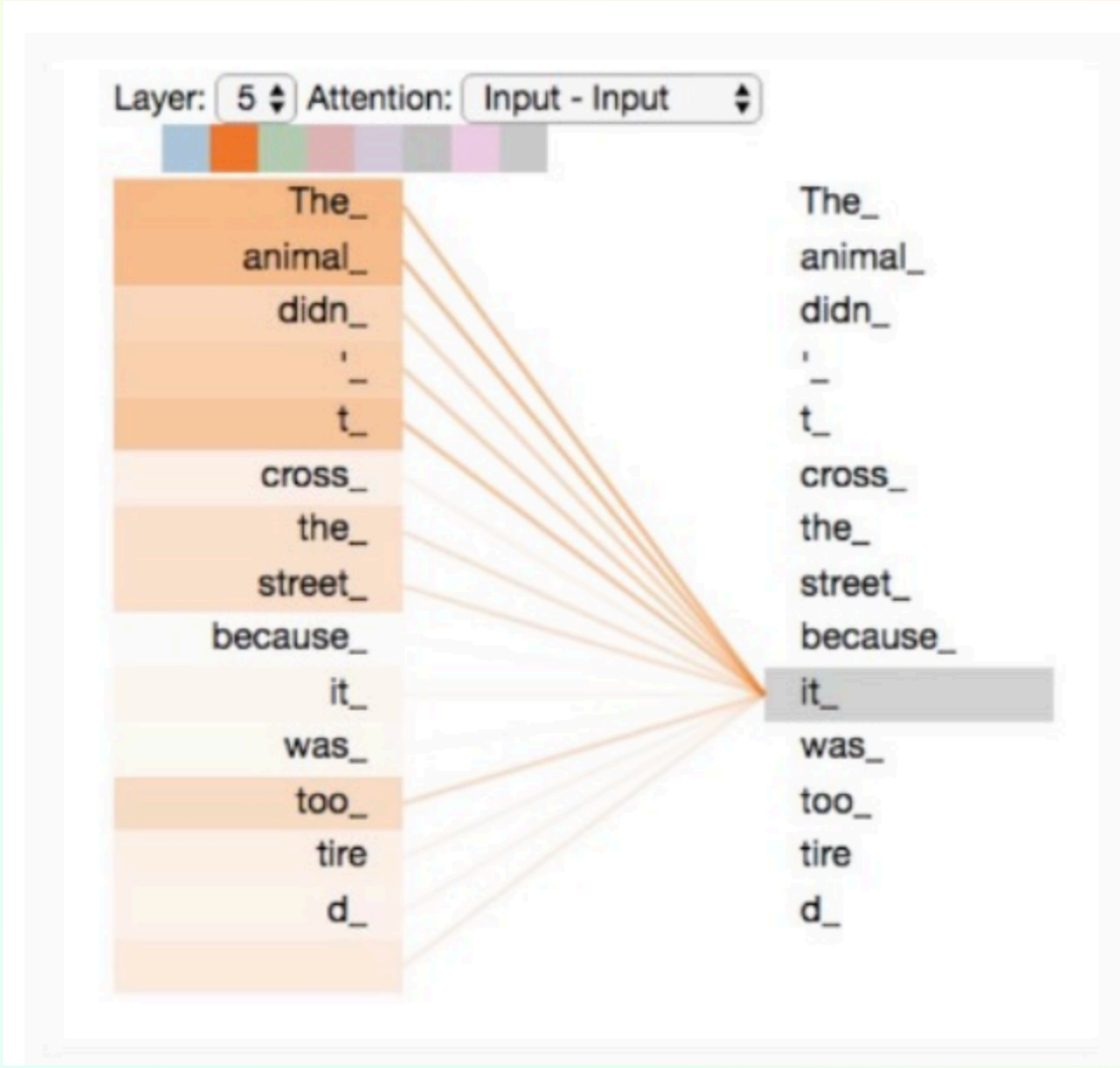
SELF-ATTENTION

BREAK DOWN

The animal didn’t cross the street because it was too tired.

‘It’ focuses largely on the corresponding subject, “the animal”

Attention is “soft”. It does not focus just on a few parts, but there is influence from many areas.



Source David Mortensen

MULTI-HEADED SELF-ATTENTION

QUIZ

Why

MULTI-HEADED SELF-ATTENTION

BREAK DOWN

One attention head —> one query.

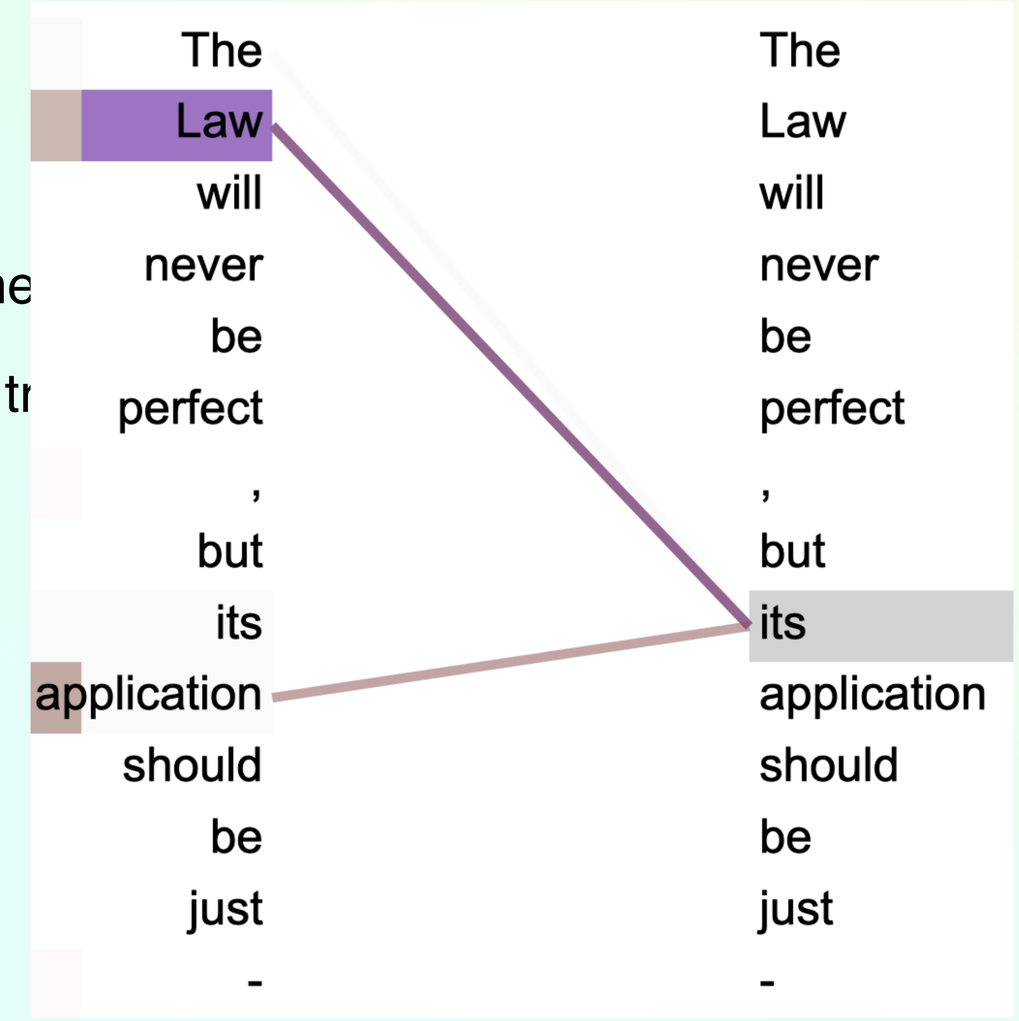
Bark —> Looks for dog, tree, or candy

MULTI-HEADED SELF-ATTENTION

BREAK DOWN

One attention head —> one

Bark —> Looks for dog, tr

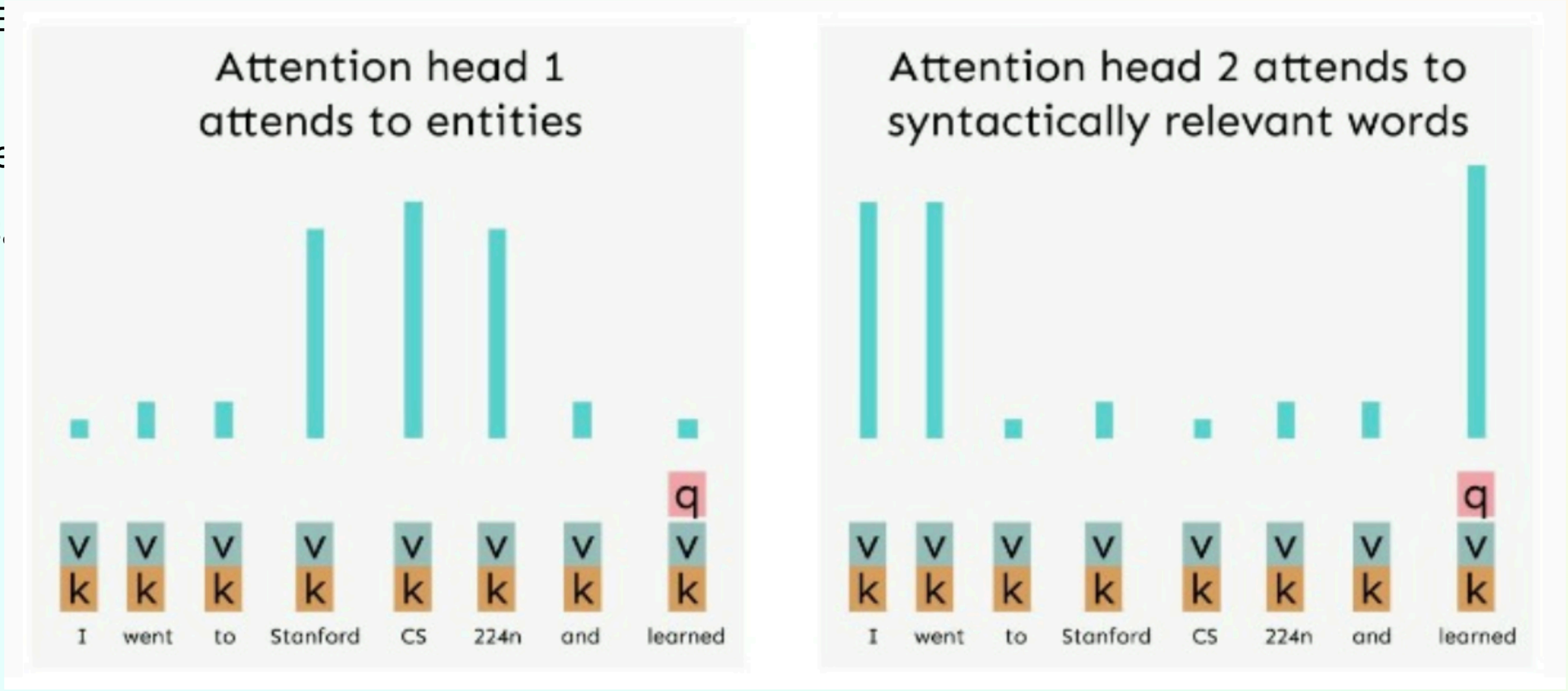


MULTI-HEADED SELF-ATTENTION

BRE

One

B



SELF-ATTENTION

COMMENT

These examples are from deep layers in transformers.

A single self-attention layer may not learn these relationships.

What we think the network will do versus what it does is different.

TRANSFORMERS

OVERVIEW

Many varieties:

- Encoder-decoder: used for translations and other sequence-to-sequence tasks
- Decoder: used for GPT style models

TRANSFORMERS

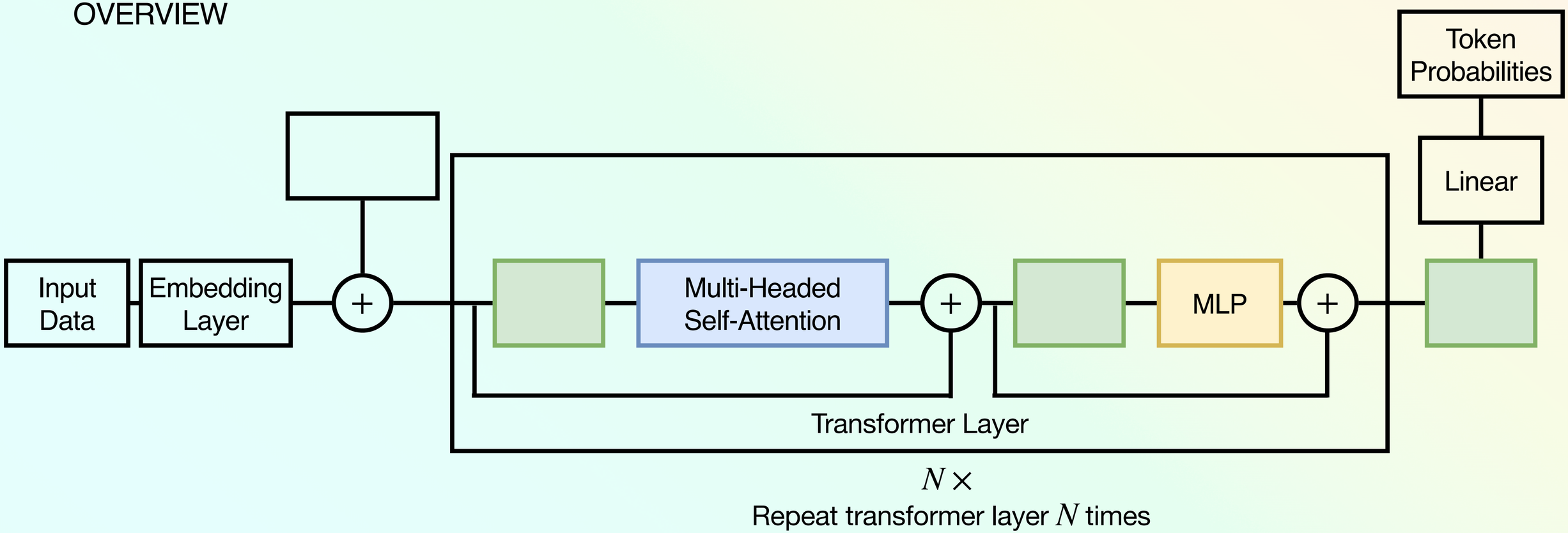
OVERVIEW

Many varieties:

- Encoder-decoder: used for translations and other sequence-to-sequence tasks
- **Decoder: used for GPT style models**

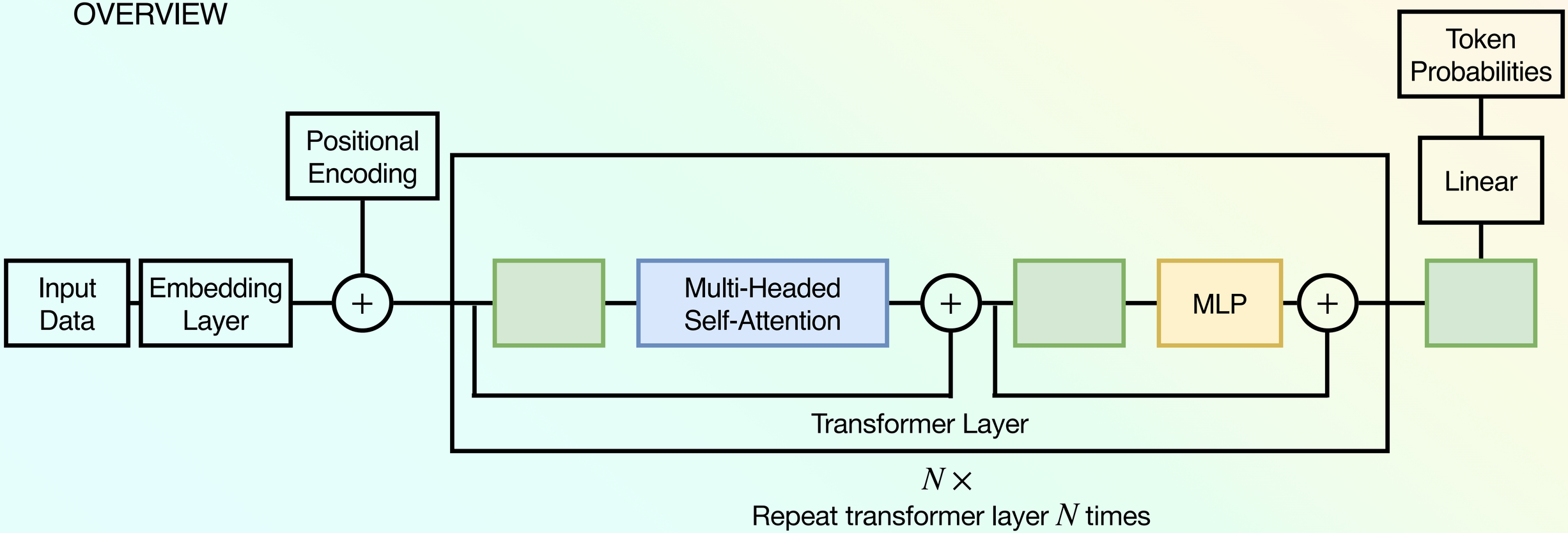
TRANSFORMERS

OVERVIEW



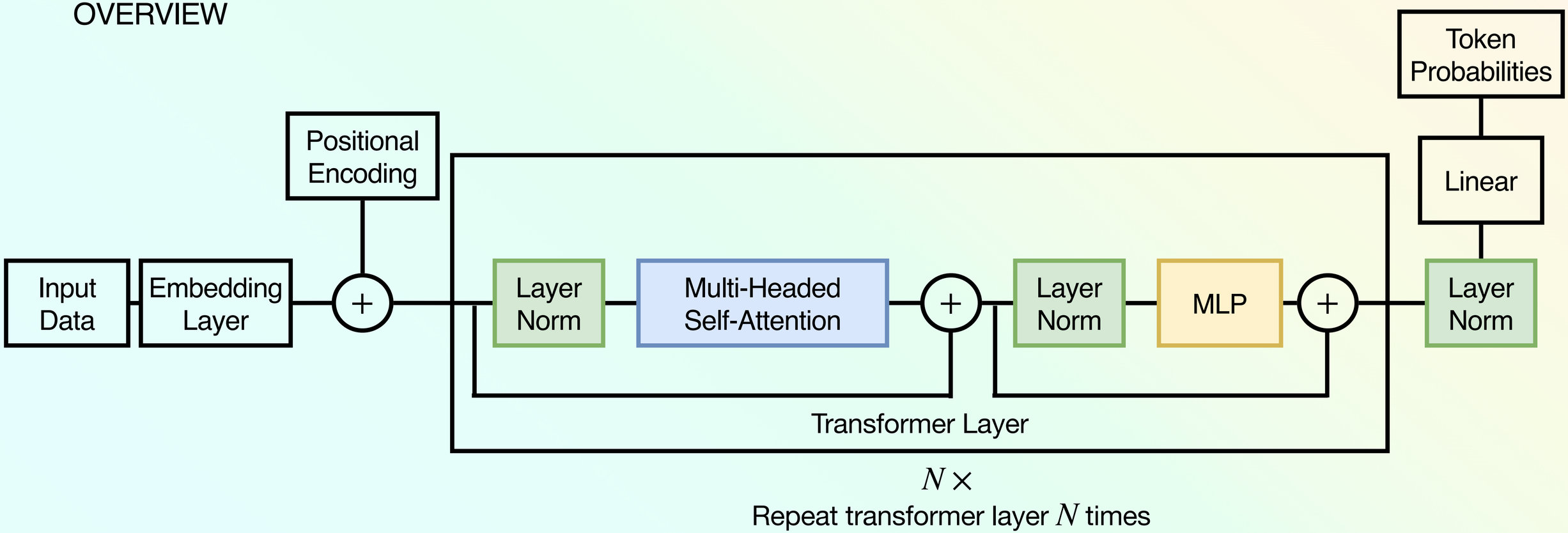
TRANSFORMERS

OVERVIEW



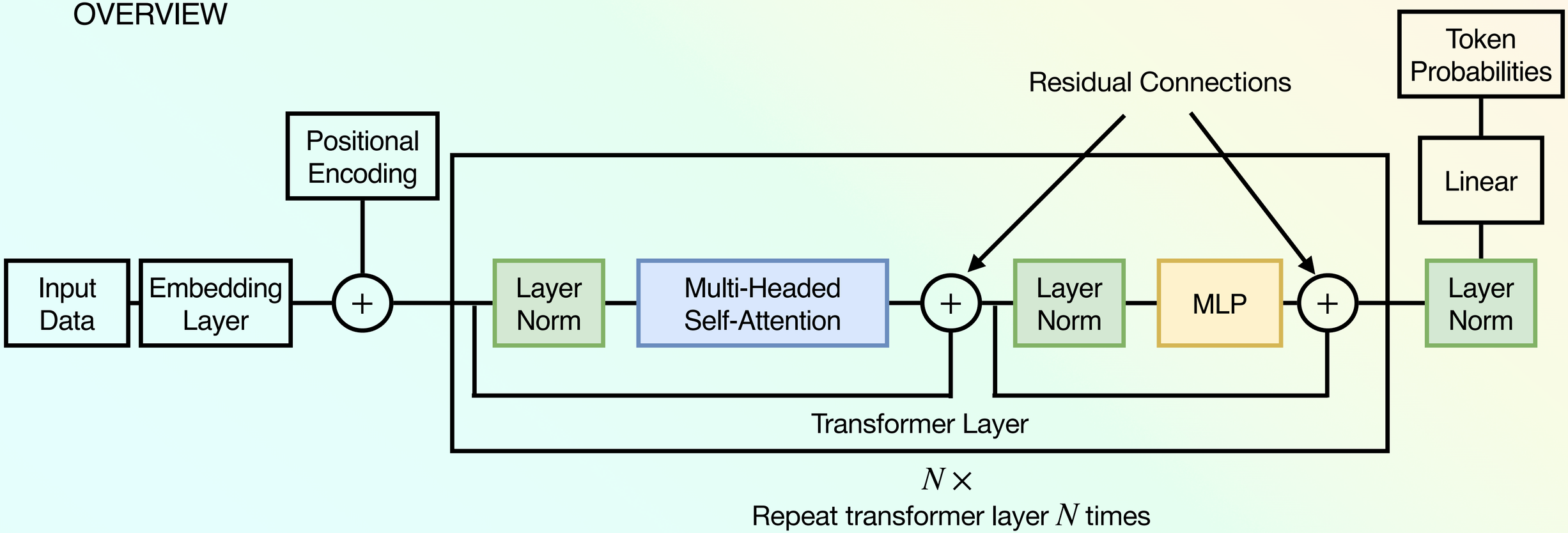
TRANSFORMERS

OVERVIEW



TRANSFORMERS

OVERVIEW



TRANSFORMERS

OVERVIEW

Core Components:

- Token embedding
- Positional Encoding
- Multi-Headed Self-Attention (with masking)
- MLP (or other feedforward networks)

Helpful Components:

- Layer norm
- Residual Connections

TRANSFORMERS

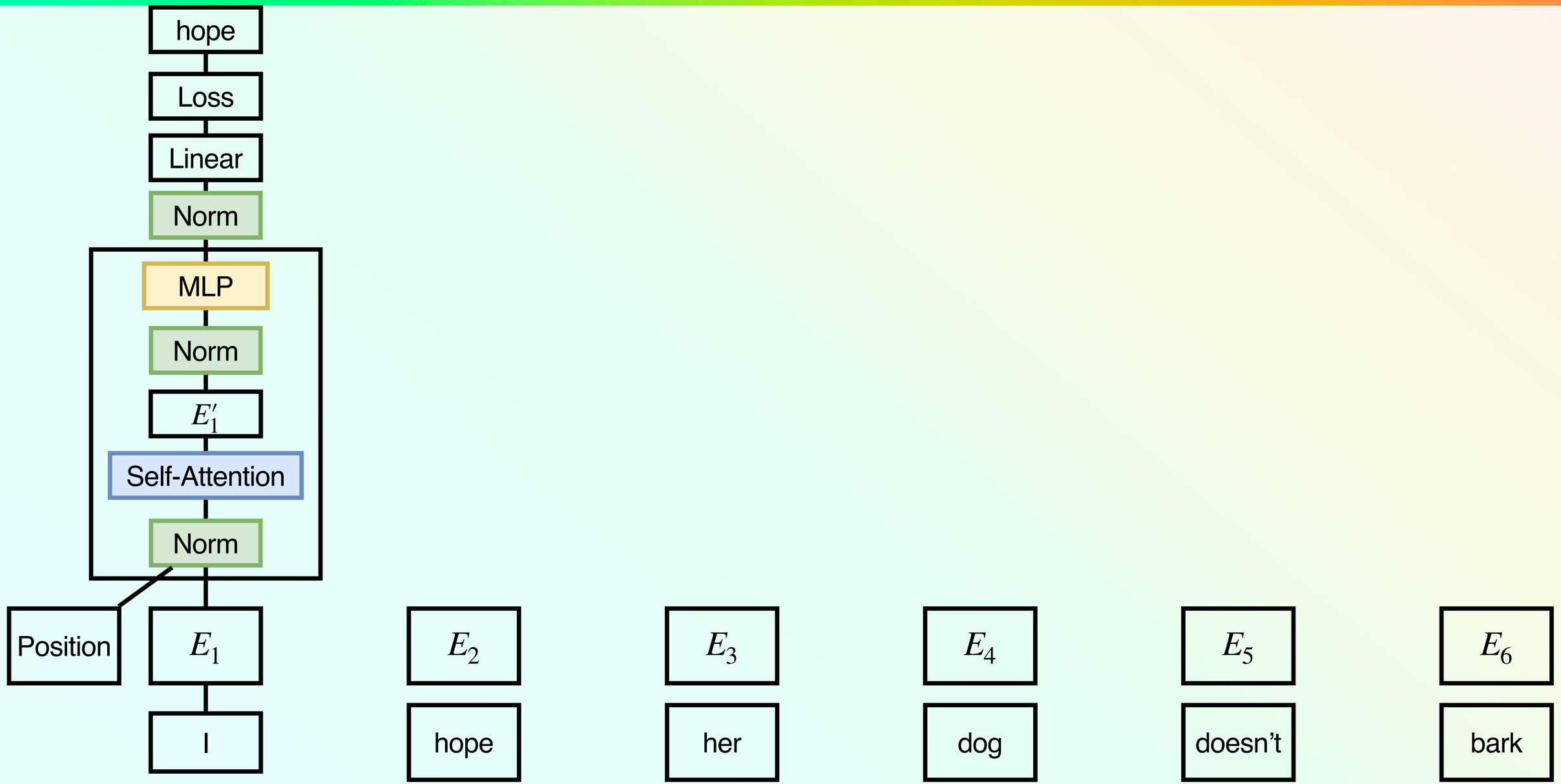
OVERVIEW

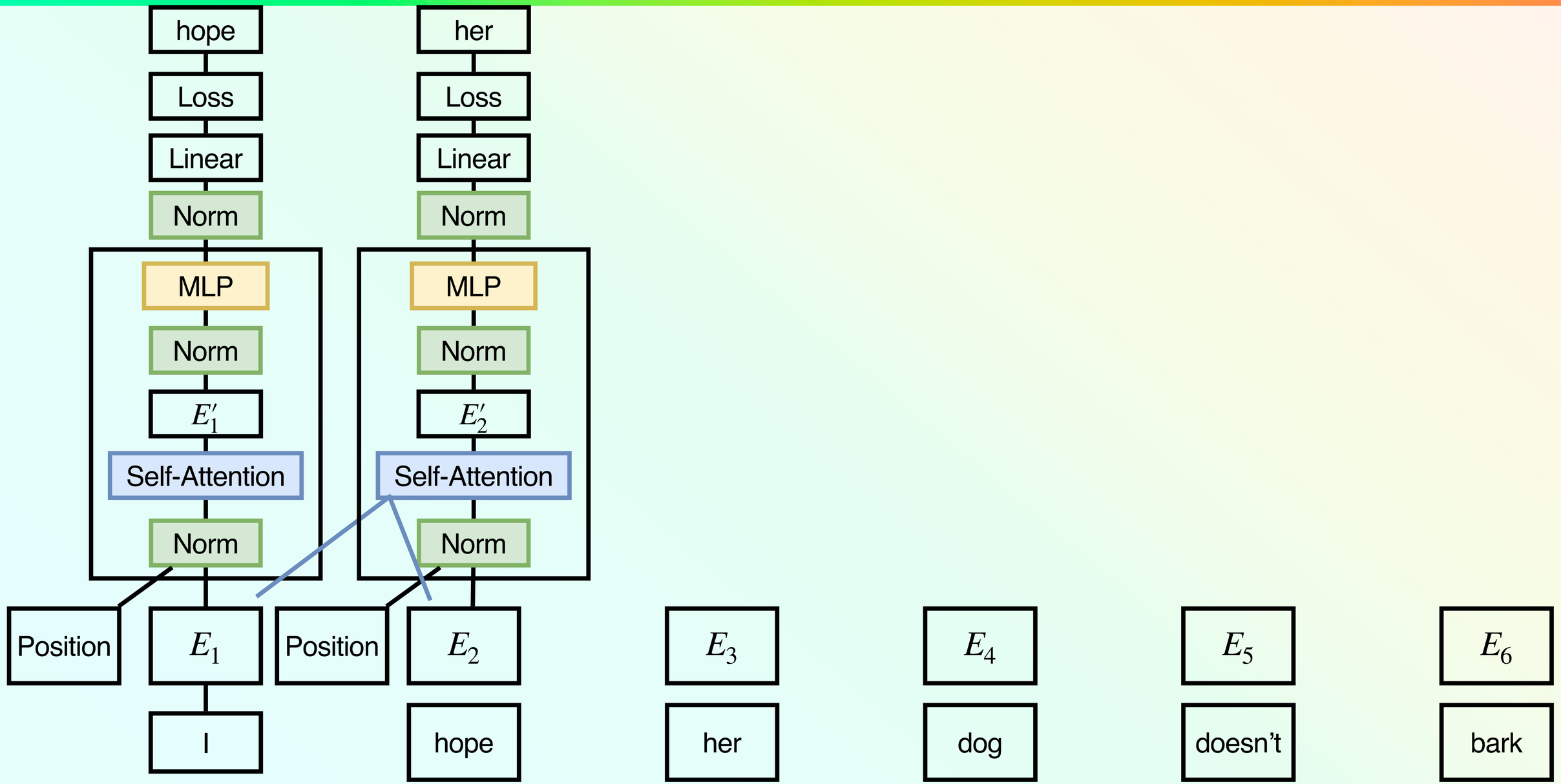
Core Components:

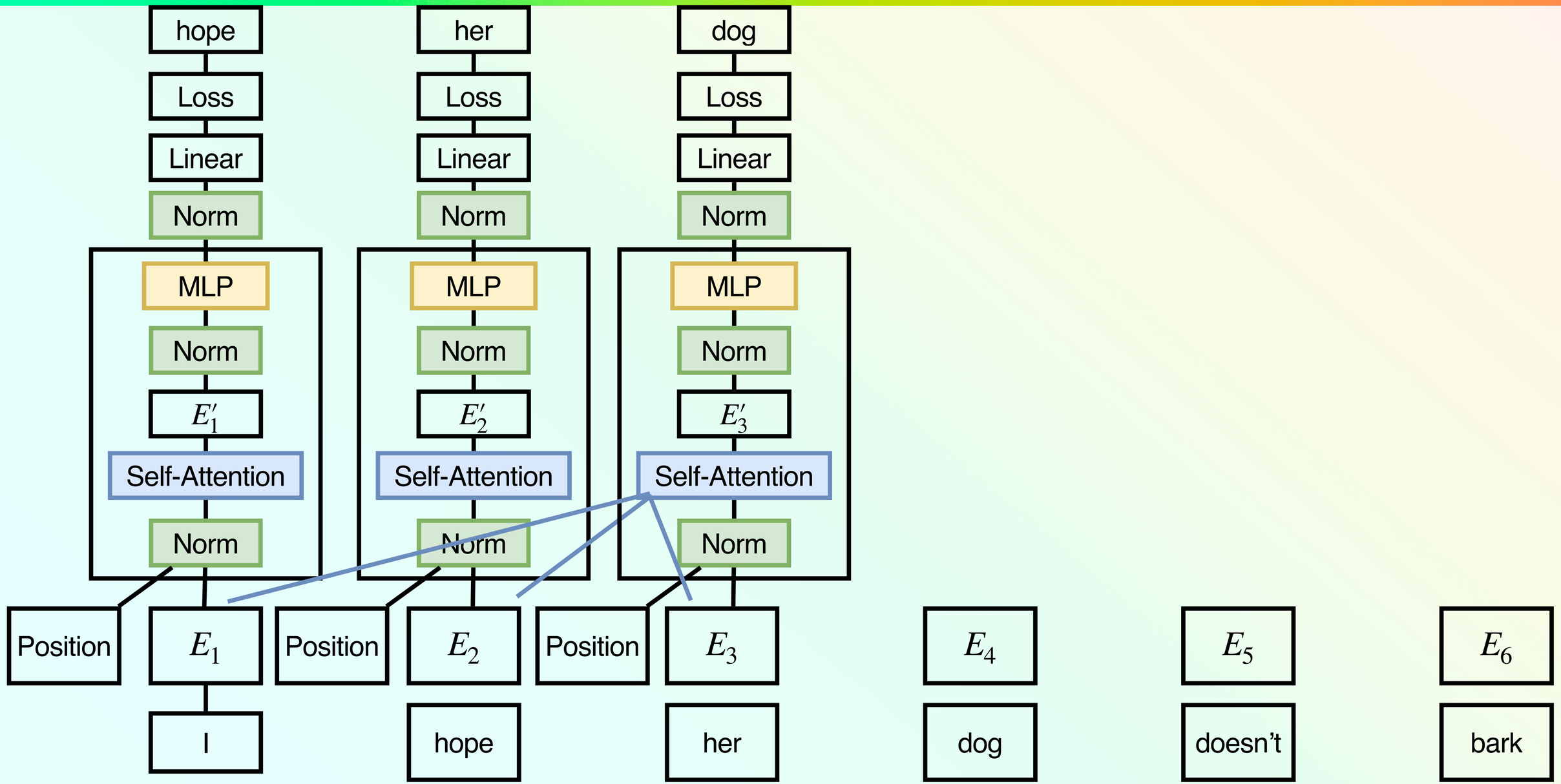
- Token embedding
- **Positional Encoding**
- Multi-Headed Self-Attention (with masking)
- MLP (or other feedforward networks)

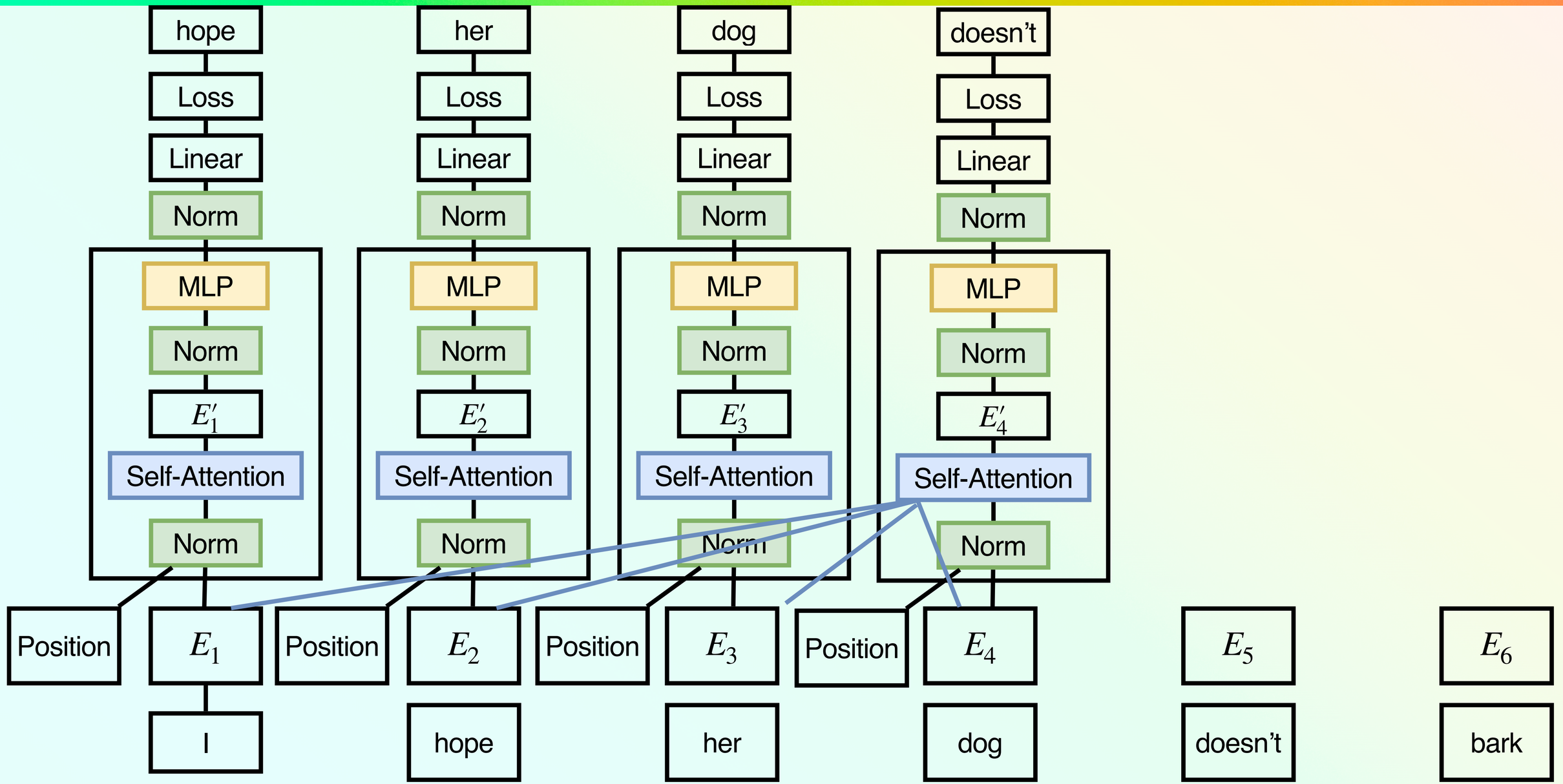
Helpful Components:

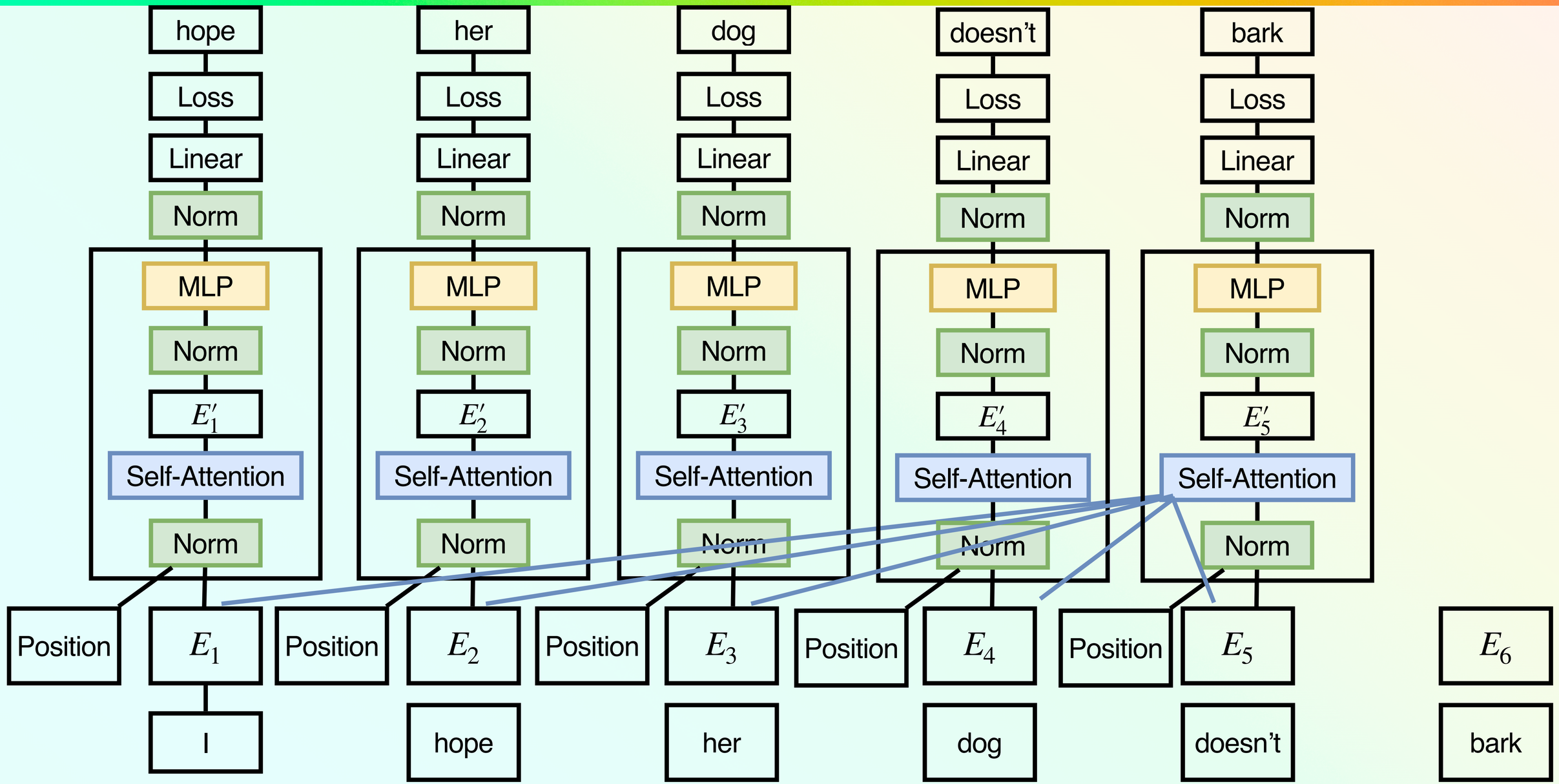
- **Layer norm**
- **Residual Connections**

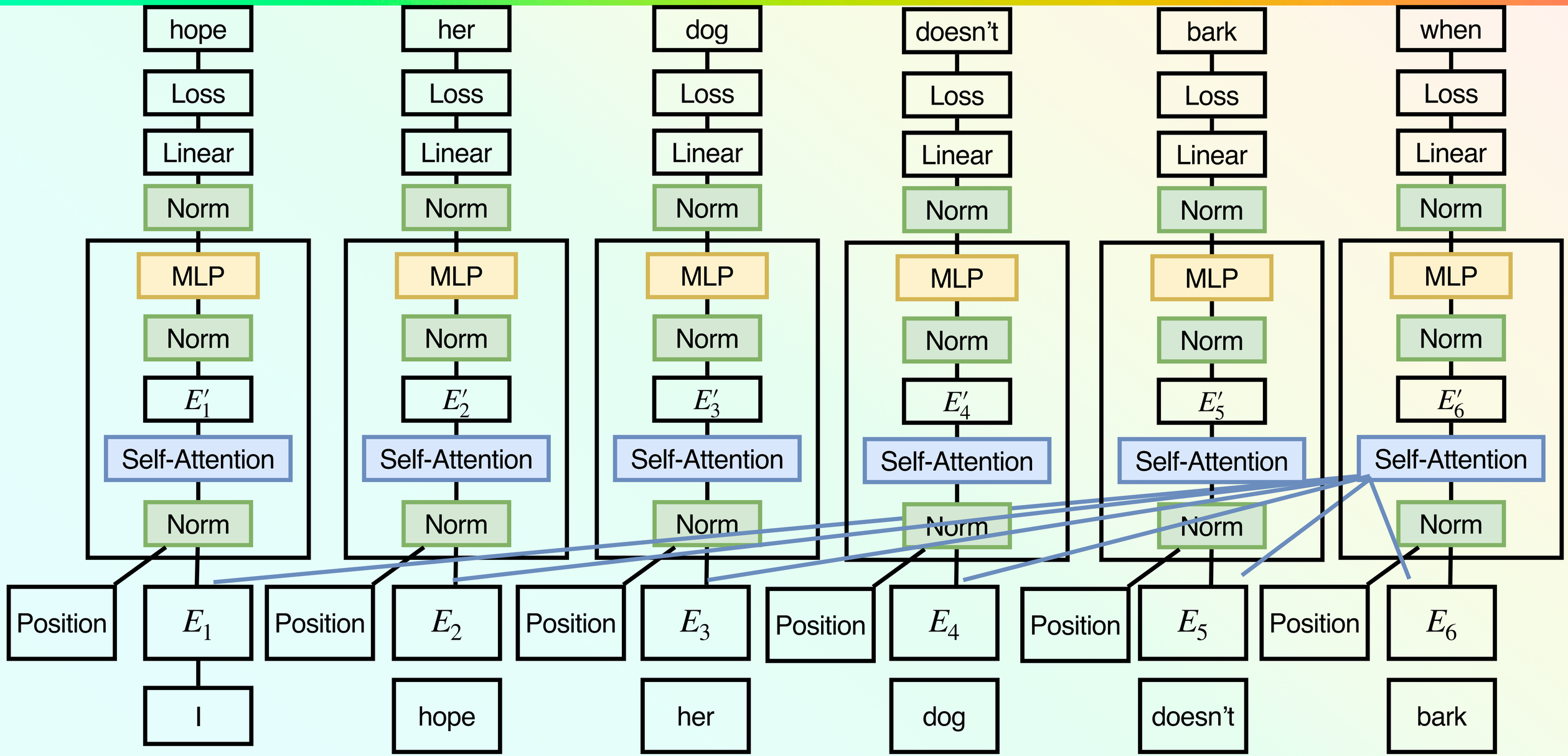

hope
Loss
Linear
Norm
MLP
Norm
E'_1
Self-Attention
Norm
Position
E_1
I
E_2
hope
E_3
her
E_4
dog
E_5
doesn't
E_6
bark


hope
Loss
Linear
Norm
MLP
Norm
E'_1
Self-Attention
Norm
her
Loss
Linear
Norm
MLP
Norm
E'_2
Self-Attention
Norm
Position
E_1
I
Position
E_2
hope
E_3
her
E_4
dog
E_5
doesn't
E_6
bark


hope
Loss
Linear
Norm
MLP
Norm
E'_1
Self-Attention
Norm
her
Loss
Linear
Norm
MLP
Norm
E'_2
Self-Attention
Norm
dog
Loss
Linear
Norm
MLP
Norm
E'_3
Self-Attention
Norm
Position
E_1
I
Position
E_2
hope
Position
E_3
her
E_4
dog
E_5
doesn't
E_6
bark

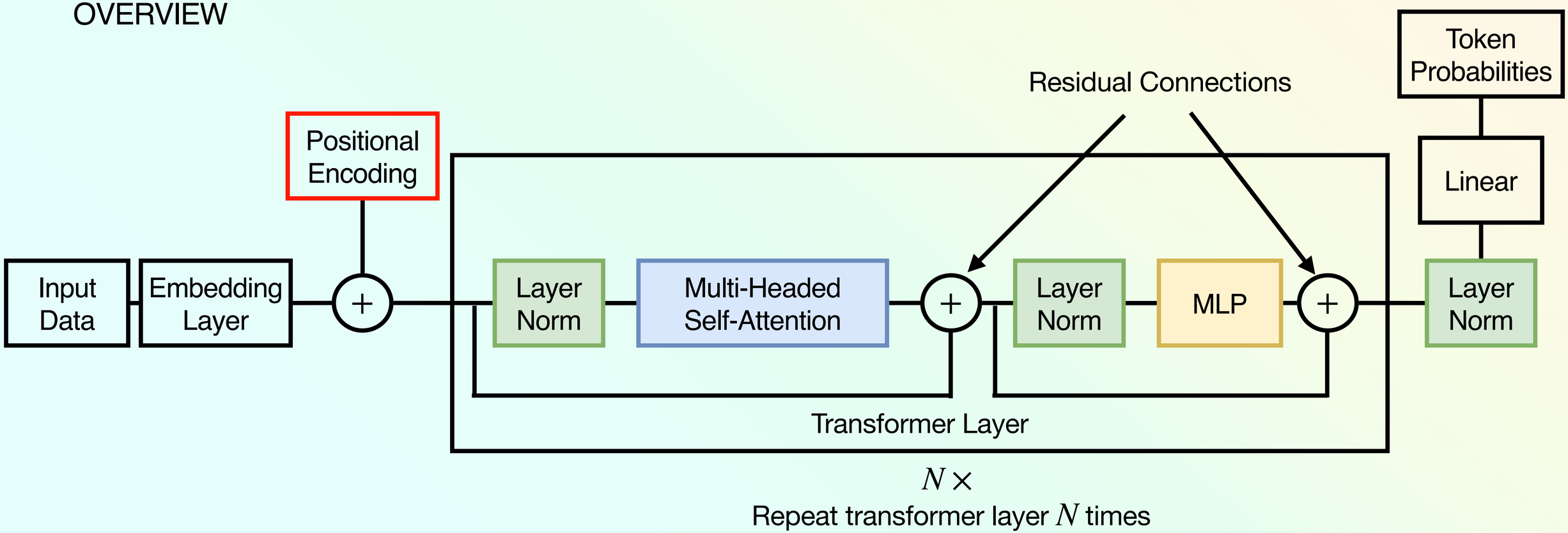

hope
Loss
Linear
Norm
MLP
Norm
E'_1
Self-Attention
Norm
Position
E_1
I
her
Loss
Linear
Norm
MLP
Norm
E'_2
Self-Attention
Norm
Position
E_2
hope
dog
Loss
Linear
Norm
MLP
Norm
E'_3
Self-Attention
Norm
Position
E_3
her
doesn’t
Loss
Linear
Norm
MLP
Norm
E'_4
Self-Attention
Norm
Position
E_4
dog
E_5
doesn’t
E_6
bark


hope
Loss
Linear
Norm
MLP
Norm
E'_1
Self-Attention
Norm
Position
E_1
I
her
Loss
Linear
Norm
MLP
Norm
E'_2
Self-Attention
Norm
Position
E_2
hope
dog
Loss
Linear
Norm
MLP
Norm
E'_3
Self-Attention
Norm
Position
E_3
her
doesn't
Loss
Linear
Norm
MLP
Norm
E'_4
Self-Attention
Norm
Position
E_4
dog
bark
Loss
Linear
Norm
MLP
Norm
E'_5
Self-Attention
Norm
Position
E_5
doesn't
E_6
bark


hope
Loss
Linear
Norm
MLP
Norm
E'_1
Self-Attention
Norm
Position
E_1
I
her
Loss
Linear
Norm
MLP
Norm
E'_2
Self-Attention
Norm
Position
E_2
hope
dog
Loss
Linear
Norm
MLP
Norm
E'_3
Self-Attention
Norm
Position
E_3
her
doesn't
Loss
Linear
Norm
MLP
Norm
E'_4
Self-Attention
Norm
Position
E_4
dog
bark
Loss
Linear
Norm
MLP
Norm
E'_5
Self-Attention
Norm
Position
E_5
doesn't
when
Loss
Linear
Norm
MLP
Norm
E'_6
Self-Attention
Norm
Position
E_6
bark

TRANSFORMERS

OVERVIEW



POSITION ENCODING

OVERVIEW

Why: attention does not care about relative position.

> “I hope her dog does bark when I knock on the door. My dog never barks at the door.”
>
> The relative position of words matters in lots of cases

POSITION ENCODING

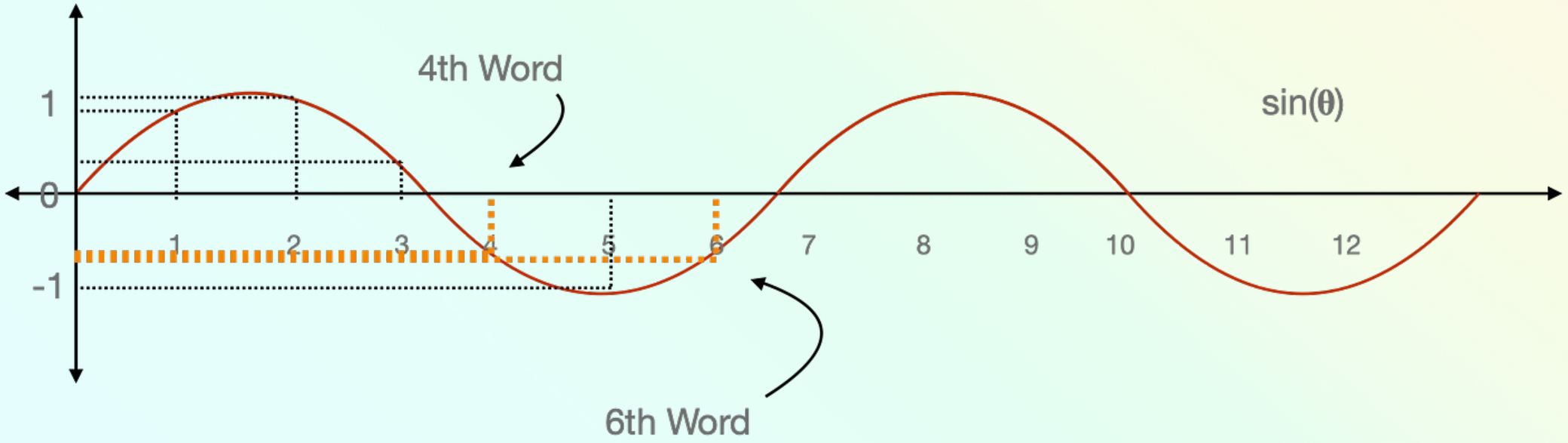
OVERVIEW

Desirable Properties:

- unique encoding for each time step
- Distance between any two time-steps should be consistent across strings with different lengths
- The model should be able to generalize to longer sentences without additional effort.
- Values should be bounded, e.g., [0,N] is not bounded.
- It needs to be deterministic

POSITION ENCODING

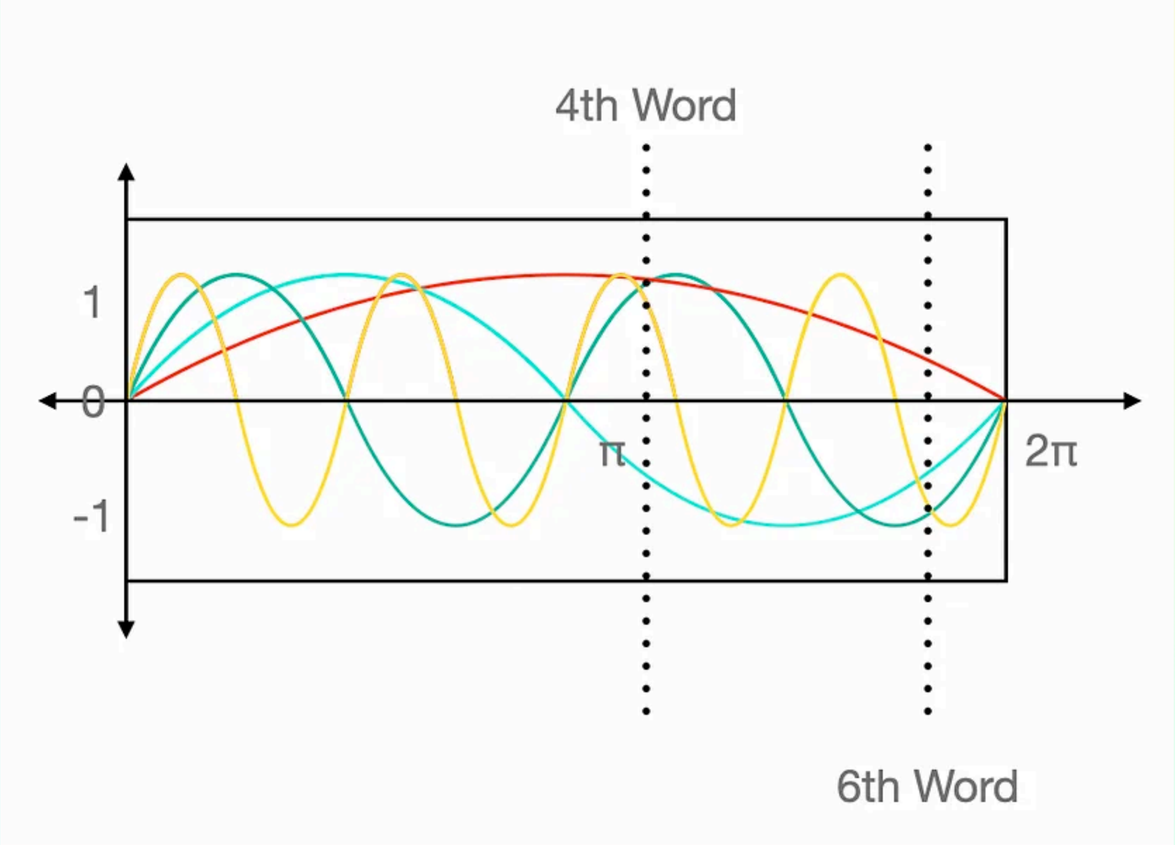
MOTIVATION



https://lih-verma.medium.com/positional-embeddings-in-transformer-eab35e5cb40d

POSITION ENCODING

MOTIVATION



https://lih-verma.medium.com/positional-embeddings-in-transformer-eab35e5cb40d

POSITION ENCODING

DEFINITION

t be the position in an input string

Let $p_t \in \mathbb{R}^d$ be the position encoding for t

$$p_{t,i} = f(t)_i \doteq \begin{cases} \sin(w_k t) & \text{if } i = 2k \\ \cos(w_k t) & \text{if } i = 2k+1 \end{cases}$$

$$w_k = \frac{1}{N^{2k/d}}, N = 10{,}000$$

$$p_t = \begin{bmatrix} \sin(w_1 t) \\ \cos(w_1 t) \\ \sin(w_2 t) \\ \cos(w_2 t) \\ \vdots \\ \sin(w_{d/2} t) \\ \cos(w_{d/2} t) \end{bmatrix}$$

See https://kazemnejad.com/blog/transformer_architecture_positional_encoding/

POSITION ENCODING

DEFINITION

Model can learn to attend to relative positions.

Change in position is a linear operation:

$$p_{t+j} = A_j p_t$$

$$\begin{bmatrix} \sin(w_k(t+j)) \\ \cos(w_k(t+j)) \end{bmatrix} = \underbrace{\begin{bmatrix} \cos(w_k j) & \sin(w_k j) \\ -\sin(w_k j) & \cos(w_k j) \end{bmatrix}}_{A_j} \begin{bmatrix} \sin(w_k t) \\ \cos(w_k t) \end{bmatrix}$$

Through W_Q, and W_K the network can learn to attend to relative positions

COMBING EMBEDDINGS

WORD EMBEDDING WITH POSITIONAL ENCODING

Word embedding E_i

Positional encoding p_i

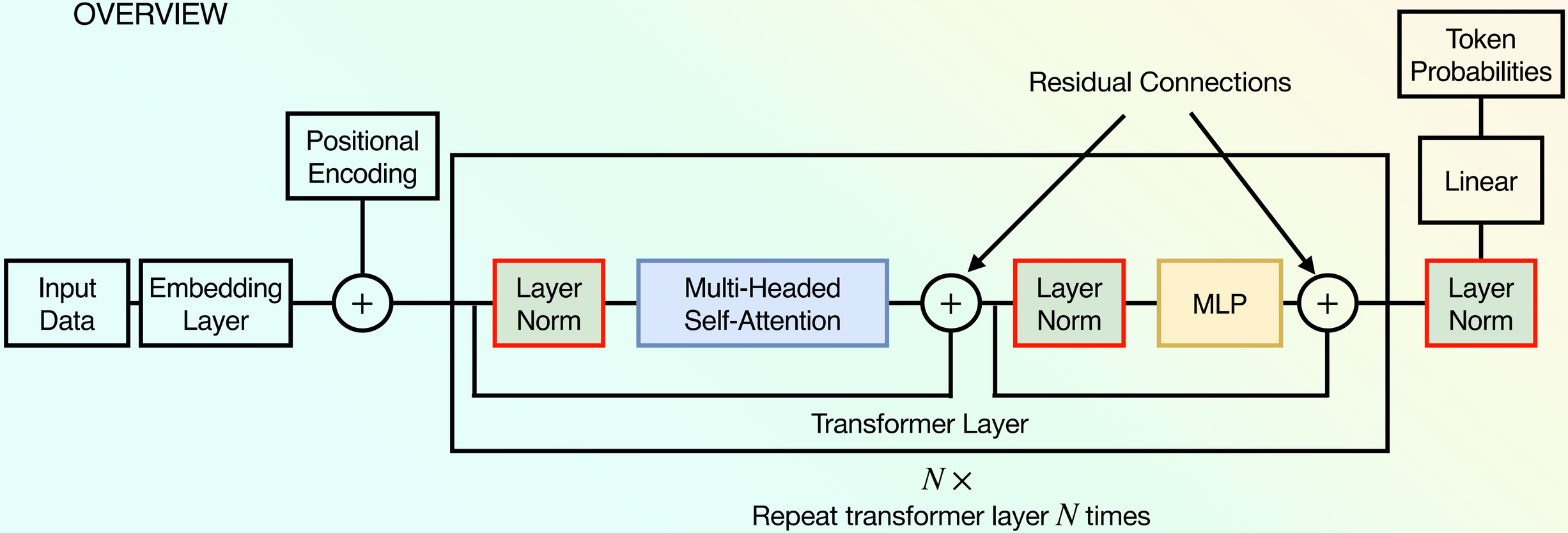
Methods:

Concat(E_i, p_i)

$E_i + p_i$ —most common

TRANSFORMERS

OVERVIEW



LAYER NORM

OVERVIEW

Idea: normalize the hidden units in a layer to have a mean zero and standard deviation of 1

$$h_{1,i}^{k'} = \frac{h_{1,i}^{k} - \mu_1}{\sqrt{\sigma_1^2 + \epsilon}}$$

$$\mu_1 = \frac{1}{n_k} \sum_{j=1}^{n_k} h_{1,j}^{k} \qquad \sigma_1^2 = \frac{1}{n_k} \sum_{j=1}^{n_k} (h_{1,j}^{k} - \mu_1)^2$$

LAYER NORM

OVERVIEW

$$h_{1,i}^{k'} = \frac{h_{1,i}^{k} - \mu_1}{\sqrt{\sigma_1^2 + \epsilon}}$$

Centers all activations

Make activations smaller (larger) is there is a large (small) variation in activations

LAYER NORM

OVERVIEW

$$h_{1,i}^{k'} = \gamma_i \frac{h_{1,i}^{k} - \mu_1}{\sqrt{\sigma_1^2 + \epsilon}} + \beta_i$$

Centers all activations

Make activations smaller (larger) is there is a large (small) variation in activations

Learn scalars γ_i, β_i to restore functional capacity lost by normalization

LAYER NORM

BENEFITS

$$h_{1,i}^{k'} = \frac{h_{1,i}^{k} - \mu_1}{\sqrt{\sigma_1^2 + \epsilon}}$$

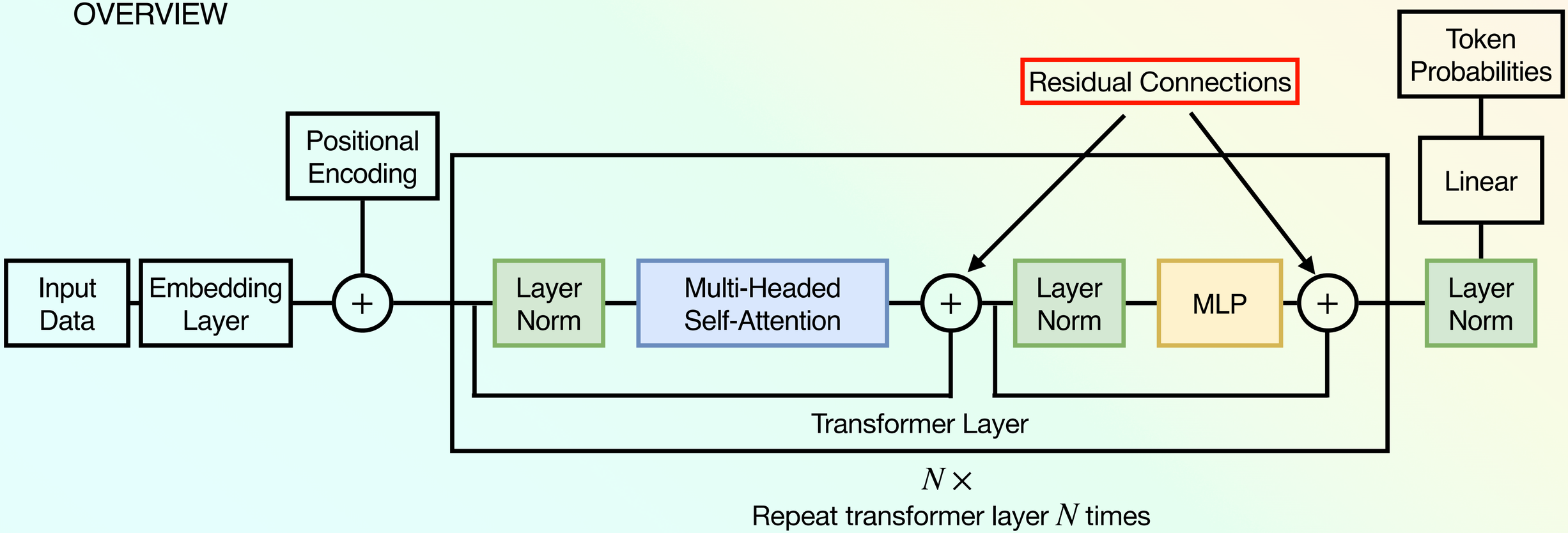
Stabilizes and accelerates training:

- Makes layer inputs have a constant variance and zero mean
- Normalization impacts the gradients

Improves generalization to unseen examples

TRANSFORMERS

OVERVIEW



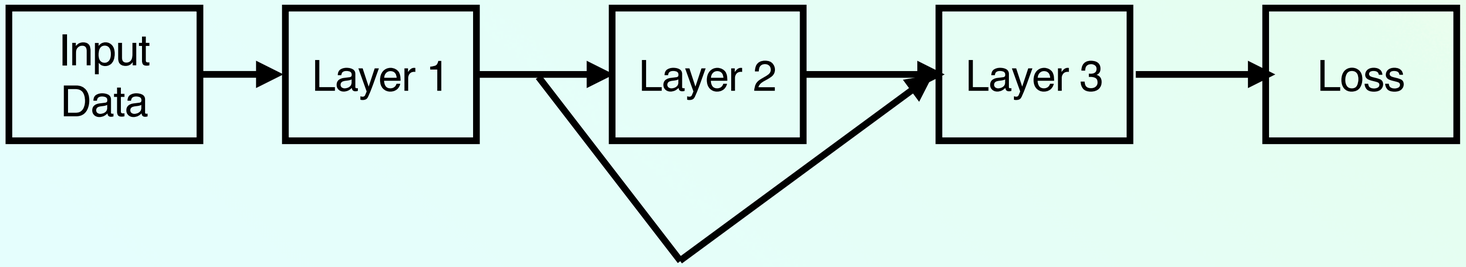
RESIDUAL CONNECTIONS

OVERVIEW

Problem: vanishing gradients

Idea: provide shortcut connections from earlier layers to later layers

Reason: gradient does not have to go through all layers



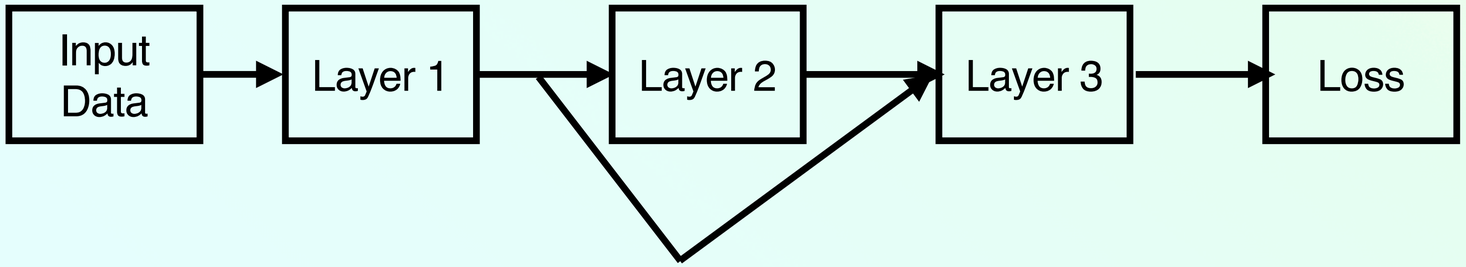
RESIDUAL CONNECTIONS

DEFINITION

Assume layers have the same width

$$h^{2'} = h^1 + h^2$$

$$h^3 = f^3(h^{2'}, W^3)$$



RESIDUAL CONNECTIONS

DEFINITION

Assume layers have the same width

$$h^{2'} = h^1 + h^2$$

$$h^3 = f^3(h^{2'}, W^3)$$

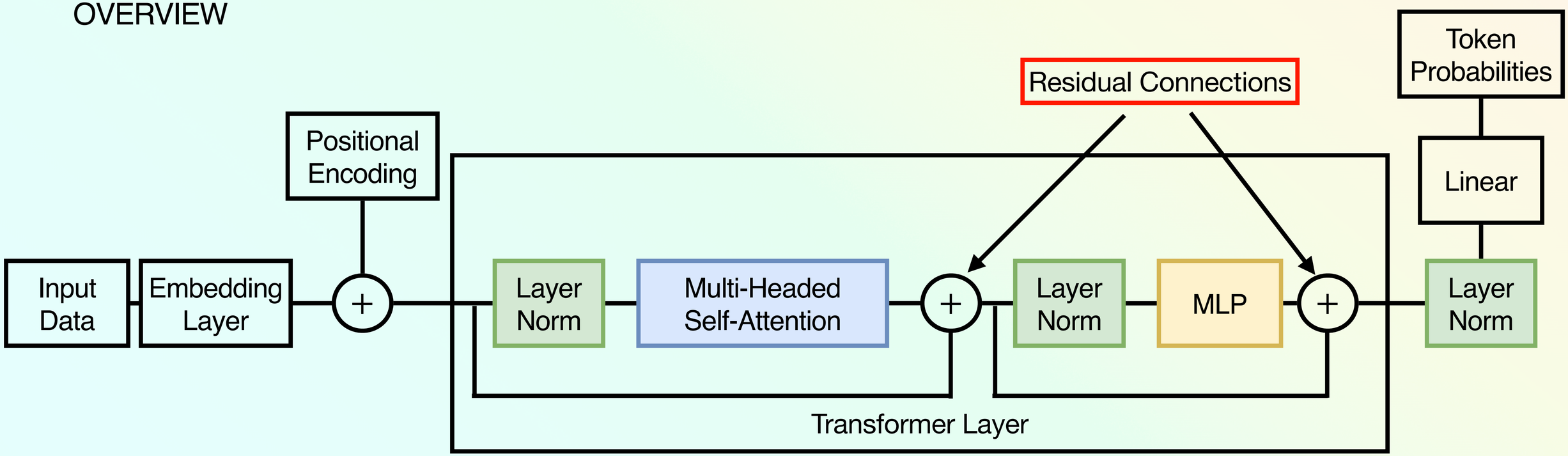
Residual connection lets layers learn changes to the previous layer

layer does not need to remember everything about the input, only the change

Gradient has a shorter path from loss function to each layer.

TRANSFORMERS

OVERVIEW



Residual after several layers in MLP

Derivatives through attention (softmax) can be small

$N \times$

Repeat transformer layer N times

GPT

DEFINITION

Generative Pretrained Transformer

Train on a MASSIVE dataset

Model predictions are “generally” “good”

- Fine-tune on new tasks
- Pretraining is a smart weight initialization for the new task

Chatbots have extra steps

NEXT CLASS

Transformers