Deep Learning

Lecture 7: Sequential Models

Ruochen Li

ruochen.li@durham.ac.uk

Durham University

An example application

- **Train station coordination system**

Tom would like to arrive **Durham** on **November 21**.

Coordination system

The system should automatically identify

Destination:

Date:

Others:

An example application

- **Train station coordination system**

Tom would like to arrive **Durham** on **November 21**.

↓

Coordination system

↓ The system should automatically identify

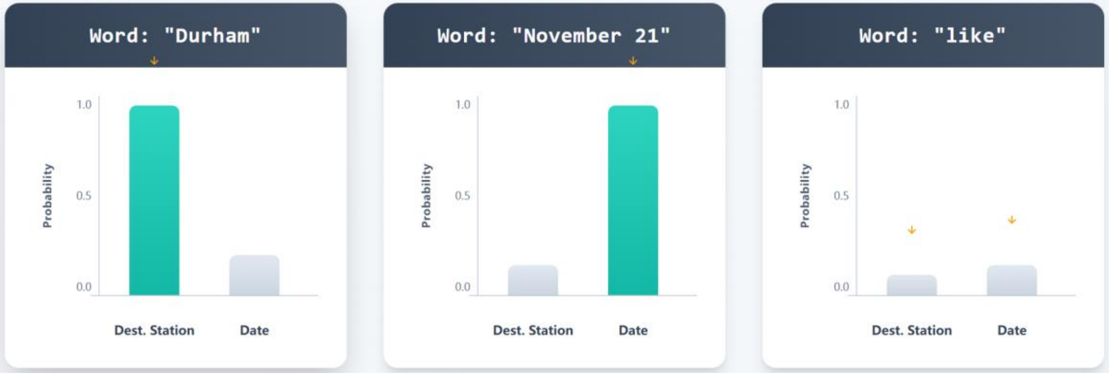
Destination:	**Durham**
Date:	**November 21**
Others:	Tom, would, like, to, arrive, on

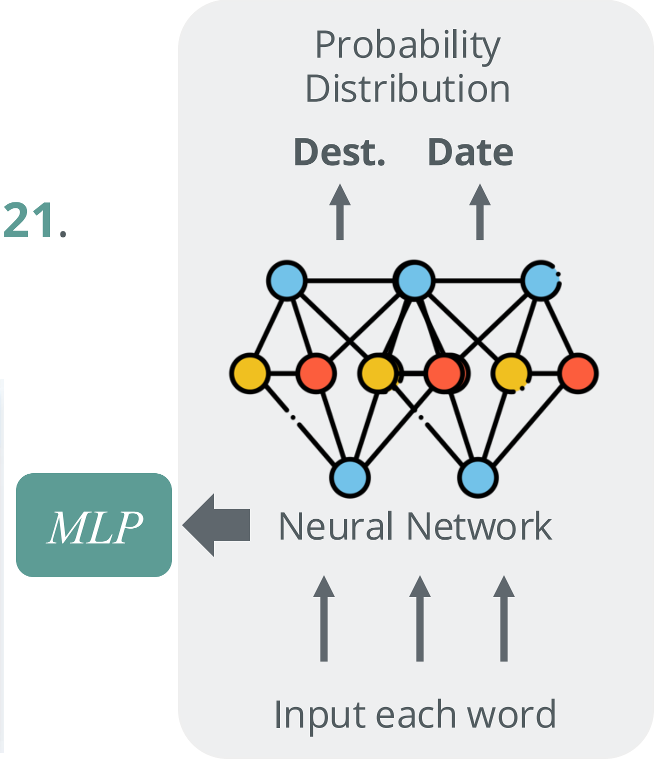
An example application

- **Train station coordination system**

Tom would like to arrive **Durham** on **November 21**.

Destination station





Slot Filling

An example application

- **Train station coordination system**

Tom would like to $\boxed{\text{arrive}}$ **Durham** on **November 21**.

↑

Destination station

Tom would like to $\boxed{\text{leave}}$ **Durham** on **November 21**.

↑

Departure station

MLP is unable to identify:

Arrive + city → Destination

Leave + city → Departure

No context modelling

No sequence structure

Lecture Overview

❶ Recurrent neural networks

- Definition and implementation
- Backpropagation through time
- vanishing/exploding gradients

❷ Long short-term memory

- Definition
- Properties
- Seq2Seq
- Attention

❸ Transformers

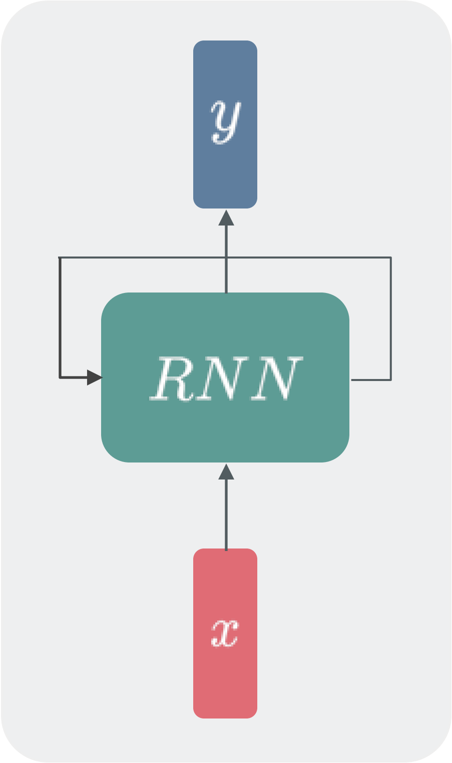
- Self-Attention
- End-to-end object detection
- GPT
- DALL-E / SORA

Recurrent Neural Networks Definition

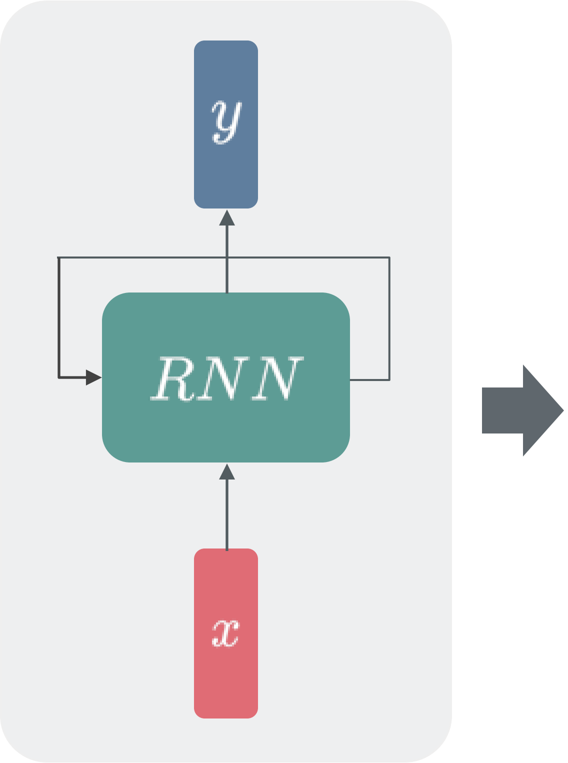
Definition: recurrent neural networks

A function applied to nodes on a directed graph. *Unlike* one-way directed graphs (e.g. text, audio), sequential data is modelled using a cyclic connection that allows information to be stored [Rumelhart et al., 1986]. The same function f is applied to inputs at each time step, updating a hidden state vector h which acts as the network's memory:

$$h_{t+1} = f_\theta(h_t, x_t)$$

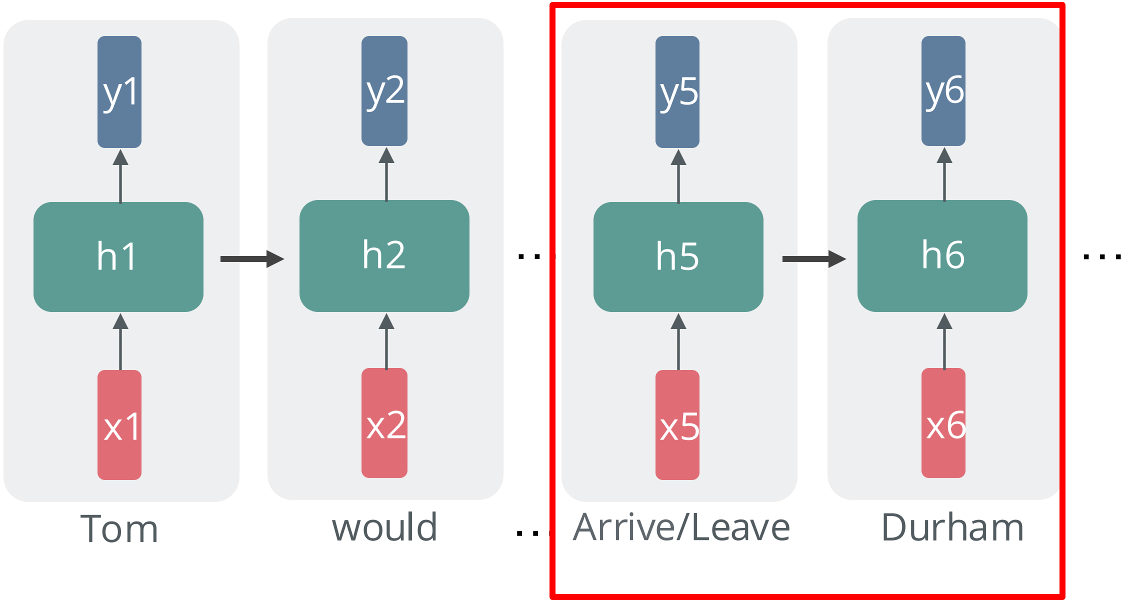


Recurrent Neural Networks Definition

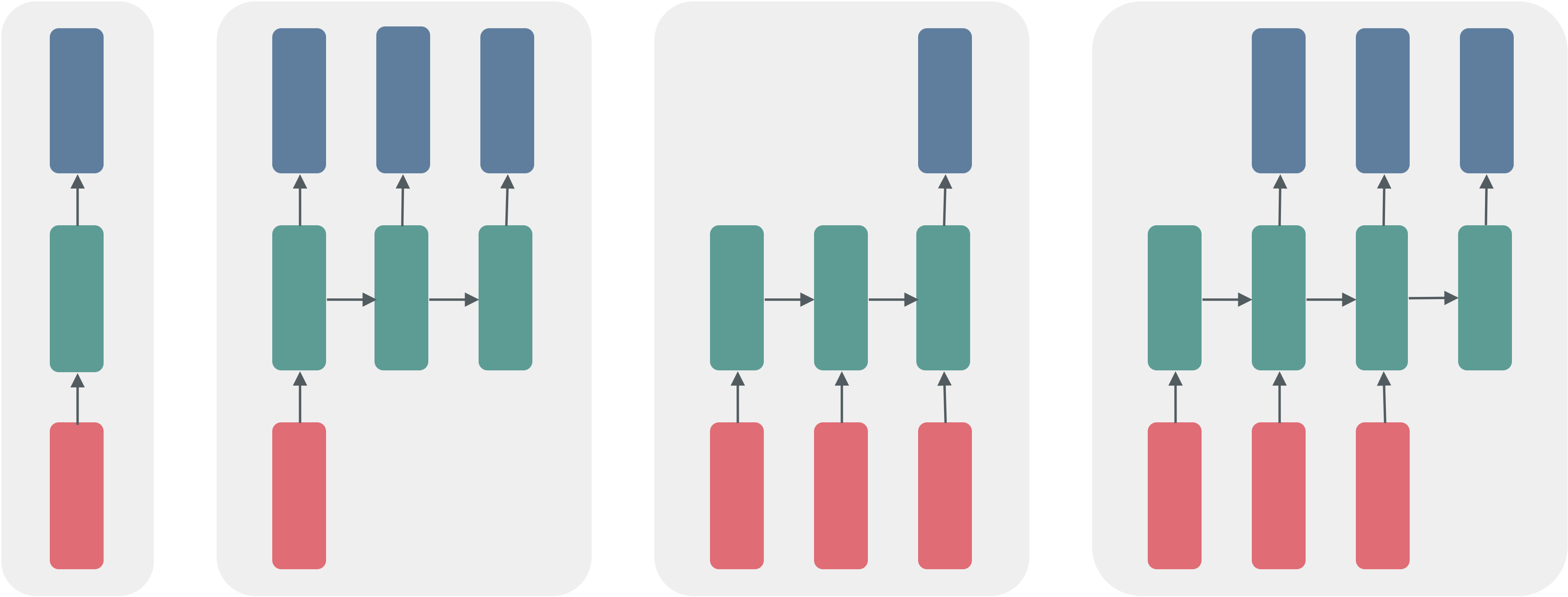


Tom would like to **arrive** Durham on November 21.

Tom would like to **leave** Durham on November 21.



Computational Graphs



One-to-One | One-to-Many | Many-to-One | Many-to-Many

Feedforward network

e.g., image captioning

e.g., sentence classification

ChatGPT, Gemini, DeepSeek

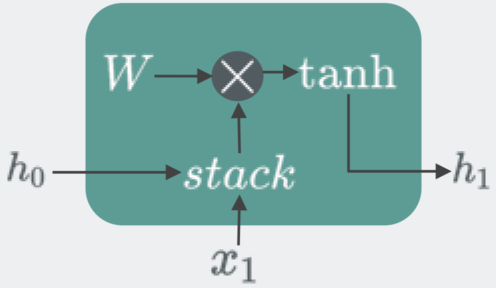
Recurrent Neural Networks Implementation

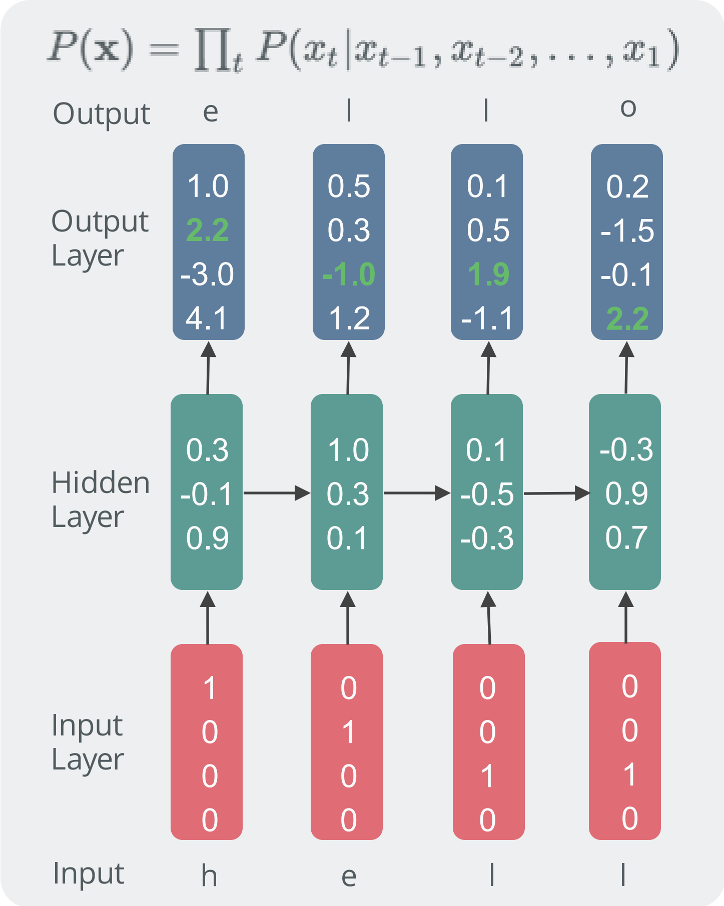
Example: RNN

A simple implementation is:

$$h_t = \tanh(W_{hh} h_{t-1} + W_{xh} x_t)$$

which is visually interpreted as a “cell”:





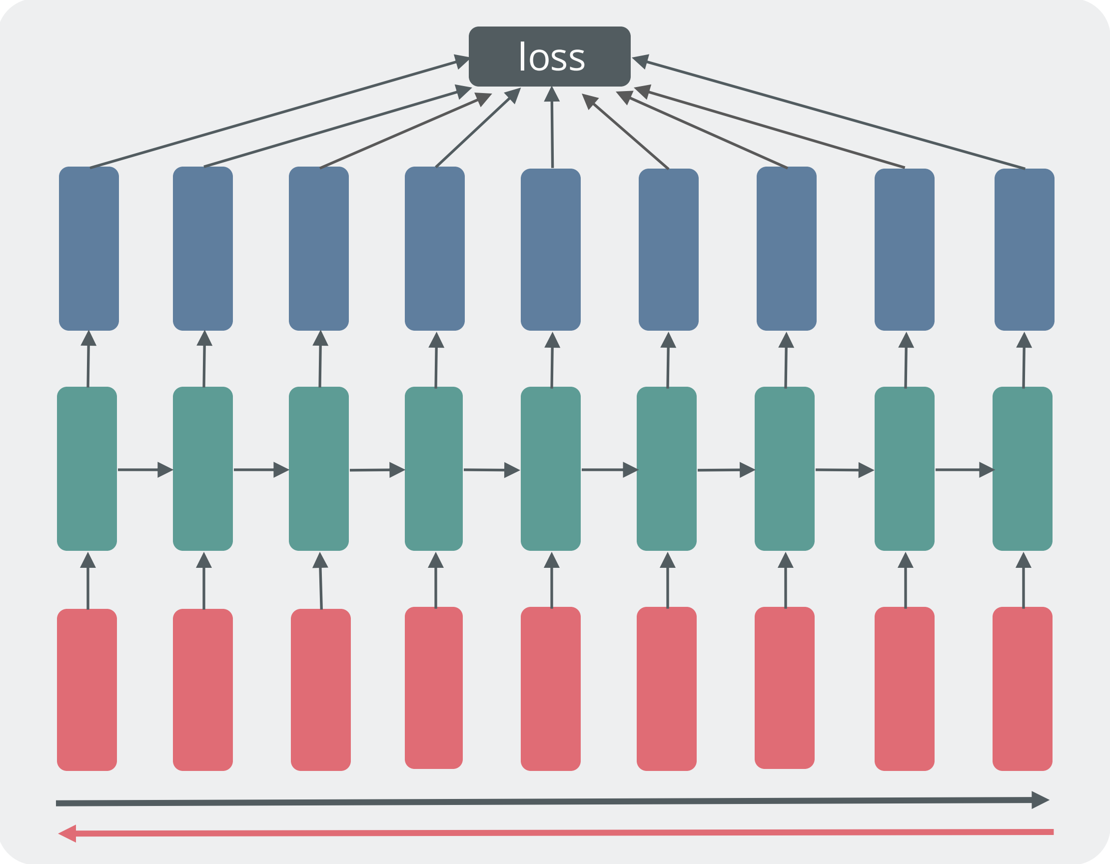
Recurrent Neural Networks Backpropagation

Definition:

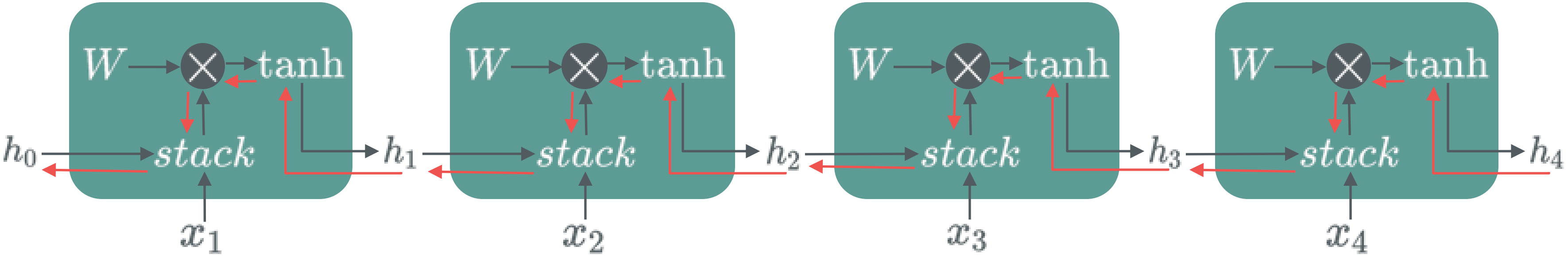
Backpropagation applied to an **unrolled** RNN is called backprop through time (BPTT). Gradients accumulate in W additively:

$$\frac{\partial \mathcal{L}_T}{\partial W} = \sum_{t \leq T} \frac{\partial \mathcal{L}_T}{\partial h_t} \frac{\partial h_t}{\partial W}$$

Long sequences use truncated BPTT where sequences are split into batches but hidden connections remain.



Recurrent Neural Networks



Why do gradients vanish/explode?

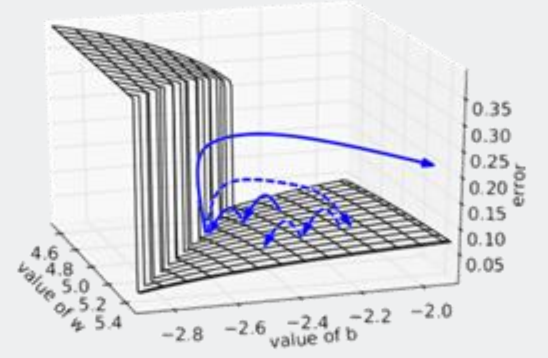
The gradient of involves many factors of W (and tanh).

The product of T matrices converges to 0 (or grows to infinity) at an exponential rate in T.

The gradient of involves many factors of W (and tanh).
The product of T matrices converges to 0 (or grows to infinity) at an exponential rate in T.

$$= \sum_{t \leq T} \frac{\partial \mathcal{L}_T}{\partial h_T} \boxed{\frac{\partial h_T}{\partial h_t}} \frac{\partial h_t}{\partial W}$$

Solution to exploding gradients: **clip gradients**



The gradient of the final loss with respect to the recurrent weight W must be backpropagated through all preceding time steps.

Let’s Look at Some Code!

Simple RNN Example for Text Classification

Code on GitHub : https://github.com/atapour/dl-pytorch/blob/main/RNN_Sentiment_Analysis/RNN_Sentiment_Analysis.ipynb

Code on Colab: https://colab.research.google.com/github/atapour/dl-pytorch/blob/main/RNN_Sentiment_Analysis/RNN_Sentiment_Analysis.ipynb

Long Short Term Memory Preventing Vanishing Gradient

Where should we put the hyphen “-”

(A) Long-Short Term Memory

(B) Long Short-Term Memory

(C) Long Short Term-Memory

Long Short-Term Memory

Preventing Vanishing Gradient

Definition: long short-term memory

LSTMs [Hochreiter et al., 1997] learn longer sequences than vanilla RNNs using better gradient flow. Backpropagation from c_t to c_{t-1} has no direct matrix multiplication by W.

Gates determine how much information passes through.

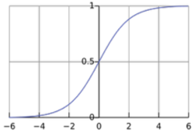
Always adding new information to the hidden state can be overwhelming.

Sometimes we want to forget things!

Example: LSTM cell

Gates control how much information is passed through using a sigmoid function and a dot product.

Sigmoid()



$$\sigma(x) = \frac{1}{1+e^{-x}}$$

Long Short-Term Memory Properties

LSTM Properties

Main Strengths

- Allows for variable length sequences
- Efficient parameter usage
- Theoretically able to store arbitrarily old information

Main Limitations

- Practically unable to store very long term dependencies
- Limited by fixed size of hidden state
- Slow training and synthesis

Further Reading:

https://karpathy.github.io/2015/05/21/rnn-effectiveness/

http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Let's Look at Some Code!

Simple LSTM Example for Text Classification

Code on GitHub : https://github.com/atapour/dl-pytorch/blob/main/LSTM_Sentiment_Analysis/LSTM_Sentiment_Analysis.ipynb

Code on Colab: https://colab.research.google.com/github/atapour/dl-pytorch/blob/main/LSTM_Sentiment_Analysis/LSTM_Sentiment_Analysis.ipynb

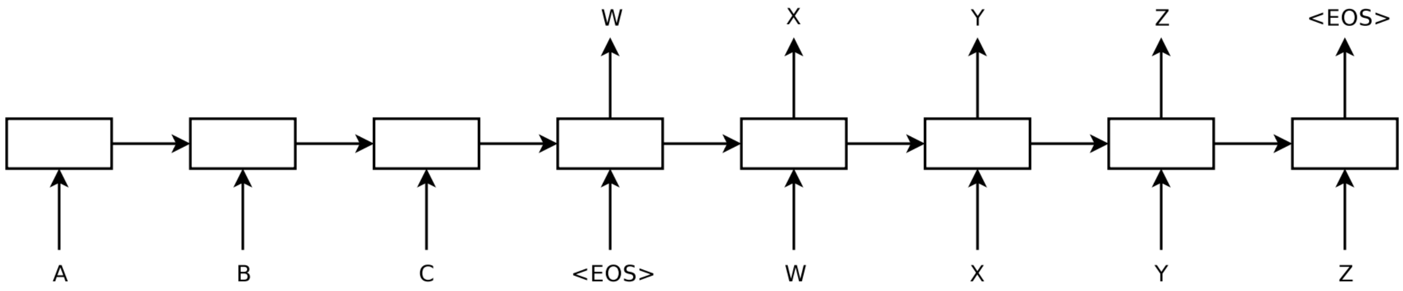
Seq2Seq Architecture

- Seq2Seq (sequence to sequence) receives a sentence as the input and produces a sentence as the output.
- Seq2Seq will include two networks, one encoder and one decoder.
- An example of this would be "**Machine Translation**":

Source Sentence	(English)	*granny liked my dishes.*
Target Sentence	(French)	*mamie a aimé mes mets.*

Seq2Seq Architecture

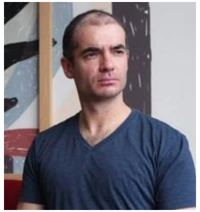
- Seq2Seq (sequence to sequence) receives a sentence as the input and produces a sentence as the output.
- Seq2Seq will include two networks, one encoder and one decoder.
- Sequence to Sequence Learning with Neural Networks



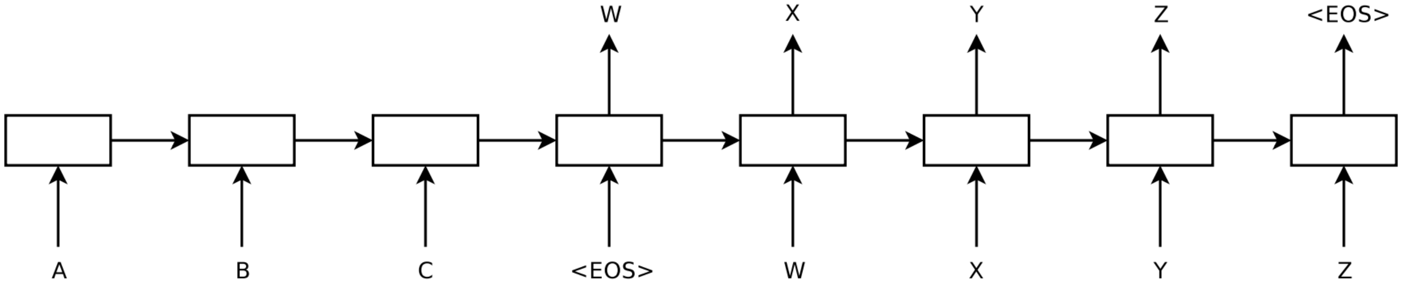
https://arxiv.org/pdf/1409.3215 (2014)

Seq2Seq Architecture

- Seq2Seq (sequence to sequence) receives a sentence as the input and produces a sentence as the output.
- Seq2Seq will include two networks, one encoder and one decoder.
- Sequence to Sequence Learning with Neural Networks



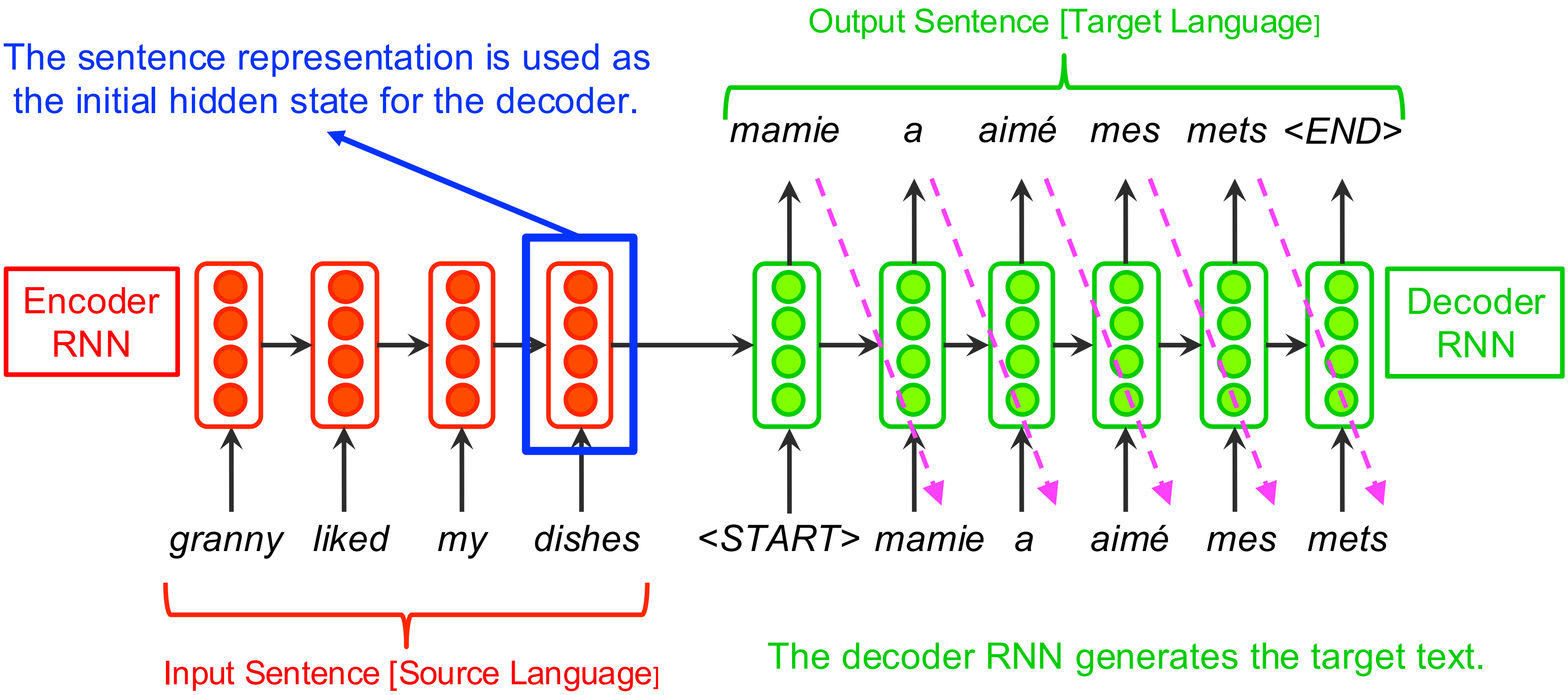
Ilya Sutskever
Co-founder of OpenAI



https://arxiv.org/pdf/1409.3215 (2014)

Seq2Seq Architecture
Machine Translation

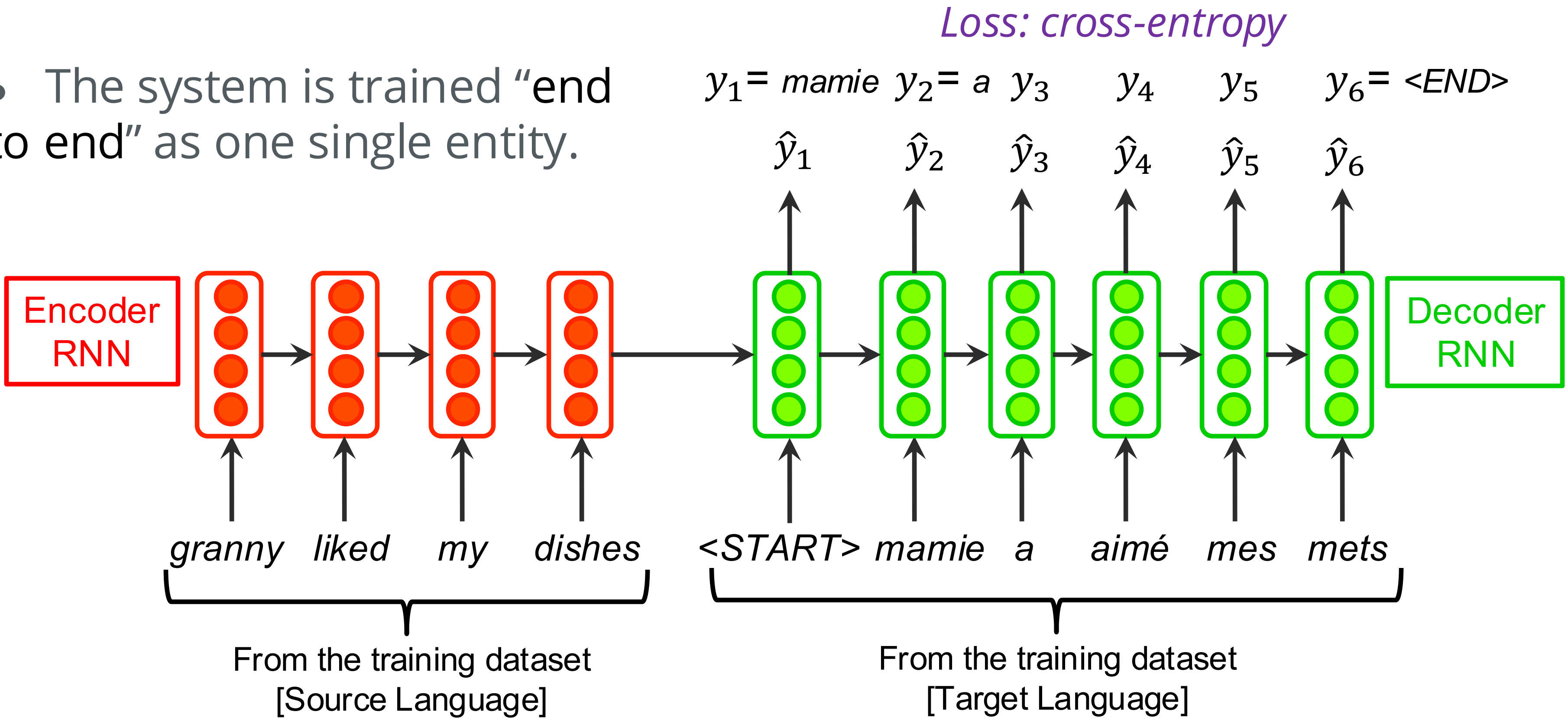
- Let's see how inference works. We will talk about the training later.



Seq2Seq Architecture Machine Translation

- Now, let's see how we would train this.

- The system is trained "end to end" as one single entity.

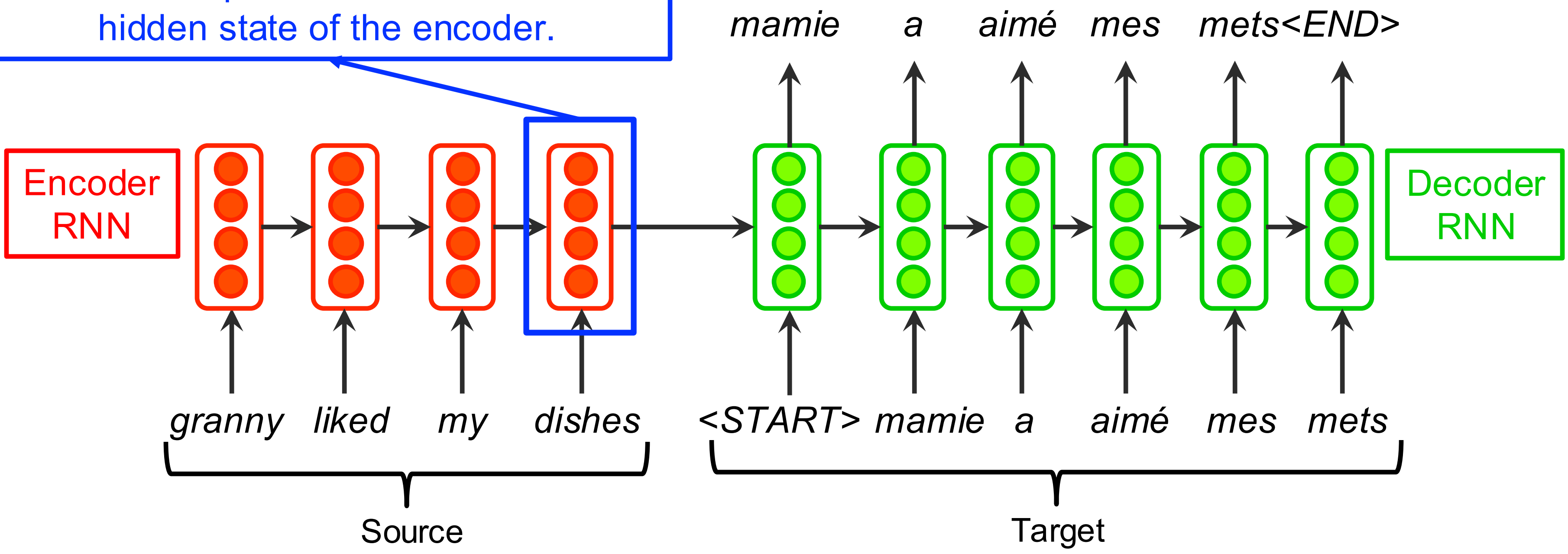


Seq2Seq Architecture Issue

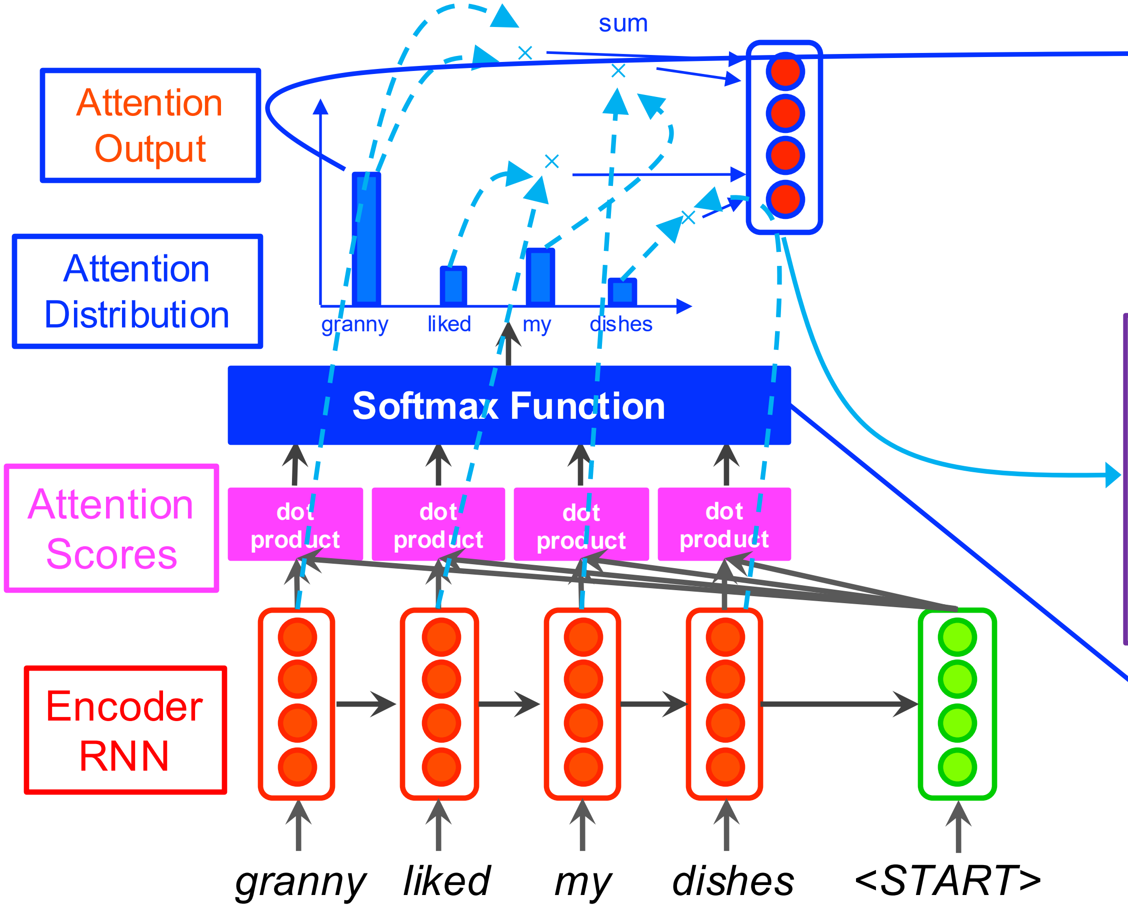
Solution: **Attention**

Informational Bottleneck:
too much pressure on a single vector

The entirety of the source sentence should be represented within the final hidden state of the encoder.



Seq2Seq with Attention

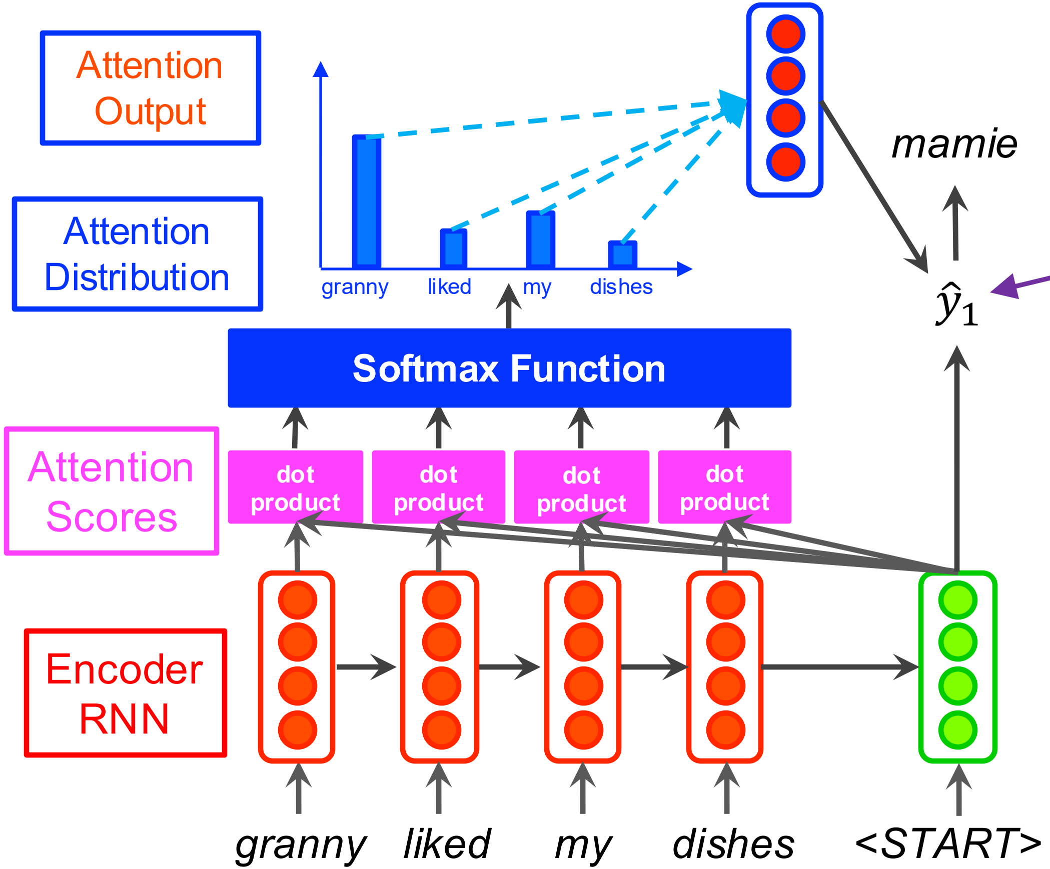


At this timestep, the decoder is **focusing** more on "granny" but could rely on information from the other words as well.

Use the attention distribution to take a **weighted sum** of the encoder hidden states.
Most of the information comes from hidden states that received higher attention levels.

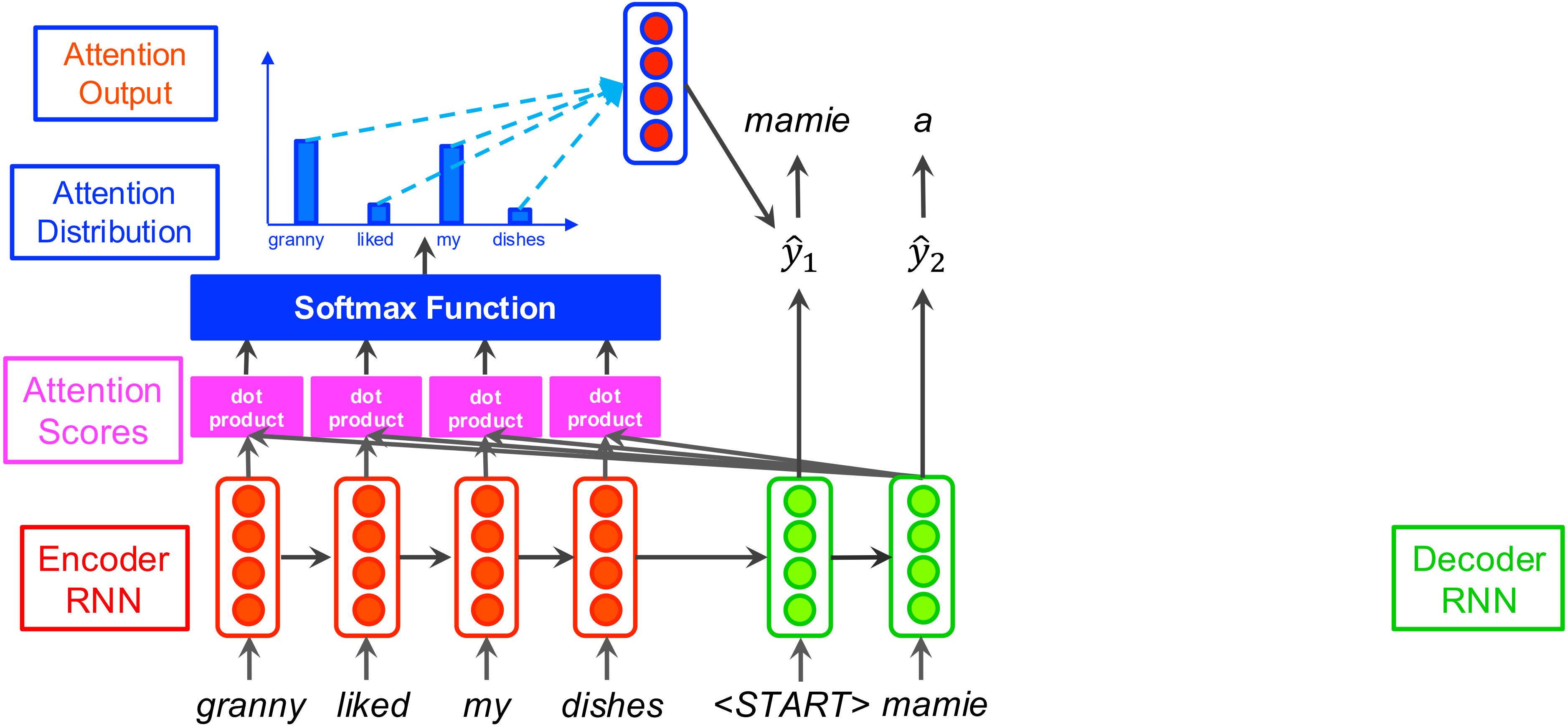
The scores are turned into a probability distribution.

Seq2Seq with Attention

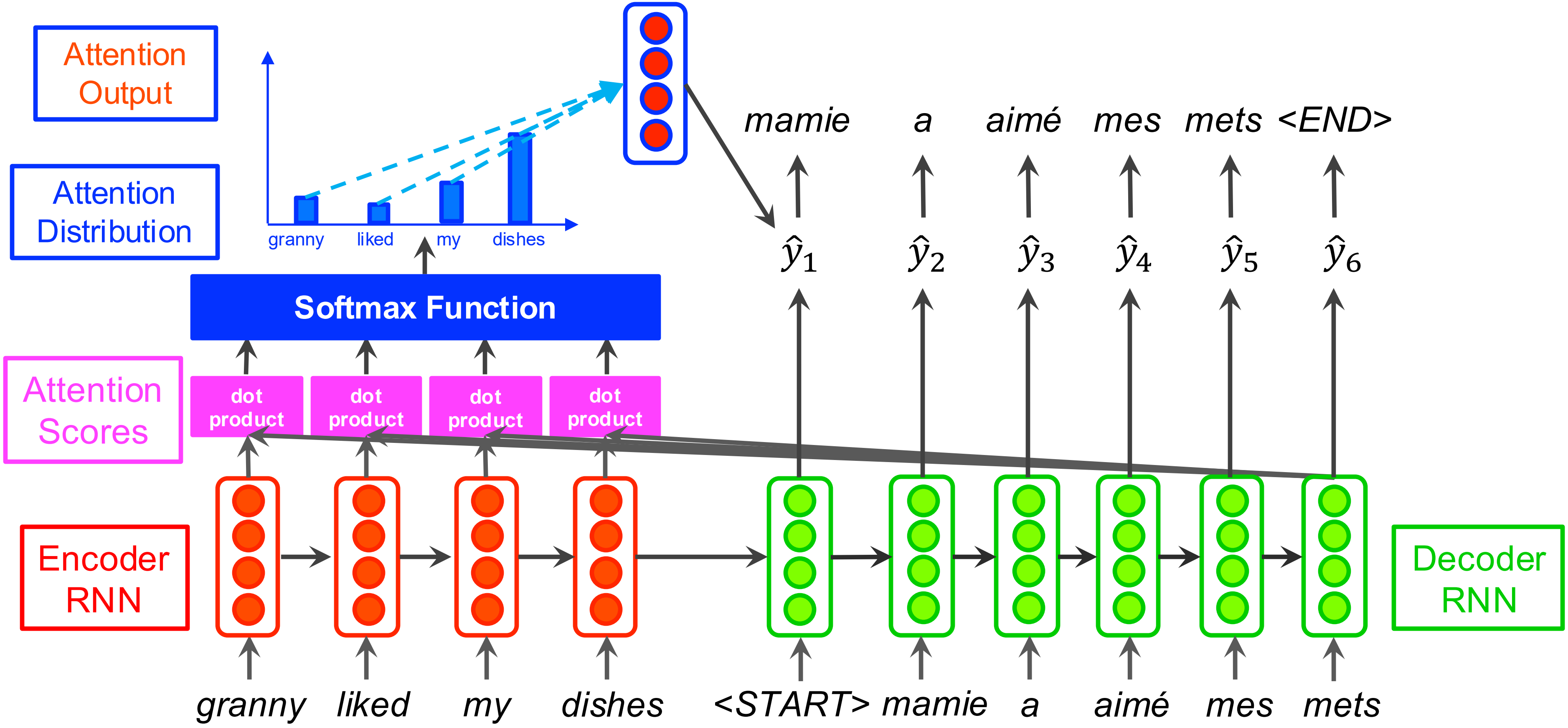


We concatenate the attention output with the decoder hidden state and use the result to obtain $\hat{y}_1$ as we did before.

Seq2Seq with Attention



Seq2Seq with Attention



Attention

Definition and Calculation

1. Encoder hidden states: $h_1, \dots, h_N \in \mathbb{R}^h$
2. At timestep t, decoder hidden state: $s_t \in \mathbb{R}^h$
3. Attention scores at timestep t: $e^t = [s_t^T h_1, \dots, s_t^T h_N] \in \mathbb{R}^N$
4. Softmax to get attention distribution: $\alpha^t = \mathrm{softmax}(e^t) \in \mathbb{R}^N$
5. Use α^t to take a weighted sum of the encoder hidden states to get the attention output: $a_t = \sum_{i=1}^{N} \alpha_i^t h_i \in \mathbb{R}^h$
6. Concatenate attention output a_t with the decoder state s_t and continue with the rest of model training: $[a_t; s_t] \in \mathbb{R}^{2h}$

Encoder hidden states: $h_1, \dots, h_N$ Decoder hidden state: s_t

Definition: Given a set of vectors *values* and a vector *query*, attention is a weighted sum of the values, dependant on the query.

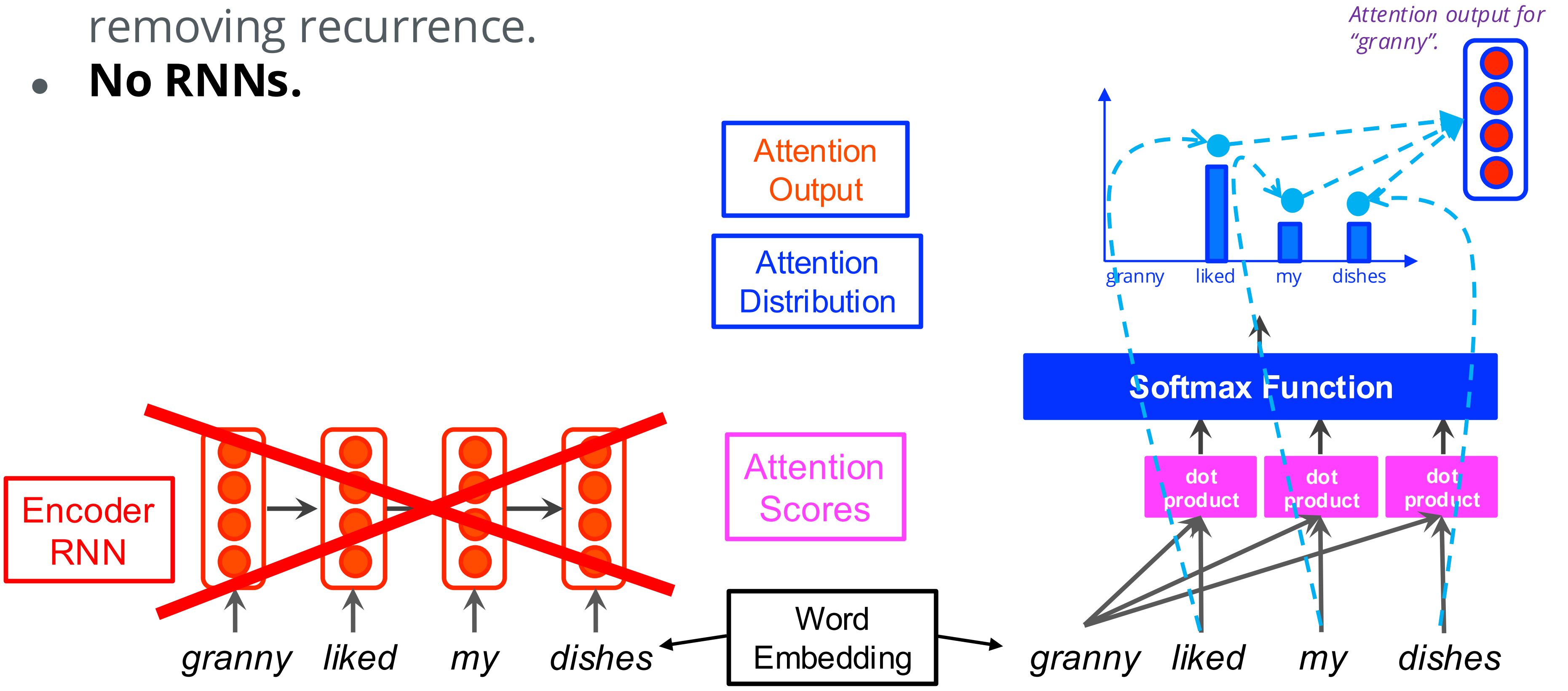
Attention types:
- Dot product
- Additive
- Multiplicative

Further reading: https://arxiv.org/pdf/1703.03906.pdf

Attention Is All You Need - Transformers

[Vaswani et al., 2017]

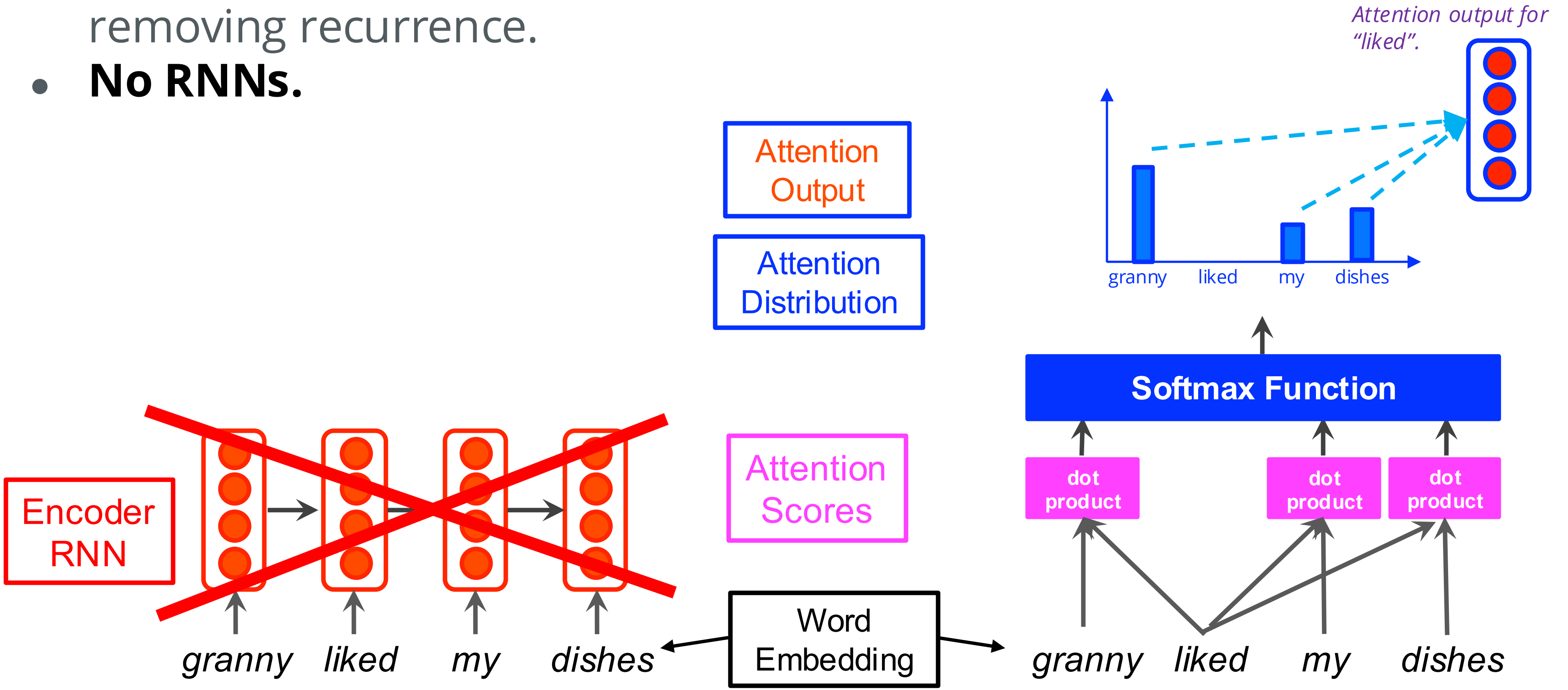
- **Self-Attention** provided a route to removing recurrence.
- **No RNNs.**



Attention Is All You Need - Transformers

[Vaswani et al., 2017]

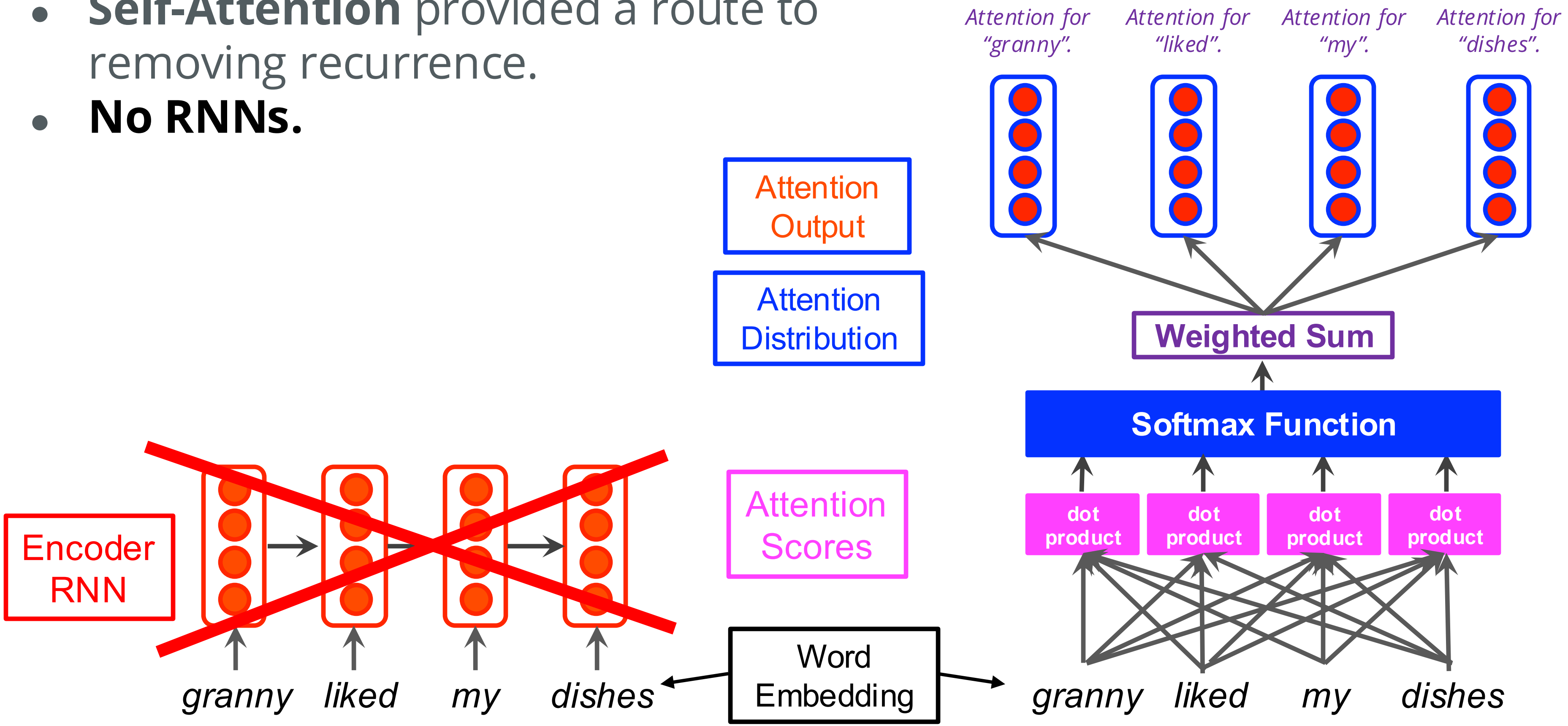
- **Self-Attention** provided a route to removing recurrence.
- **No RNNs.**



Attention Is All You Need - Transformers

[Vaswani et al., 2017]

- **Self-Attention** provided a route to removing recurrence.
- **No RNNs.**

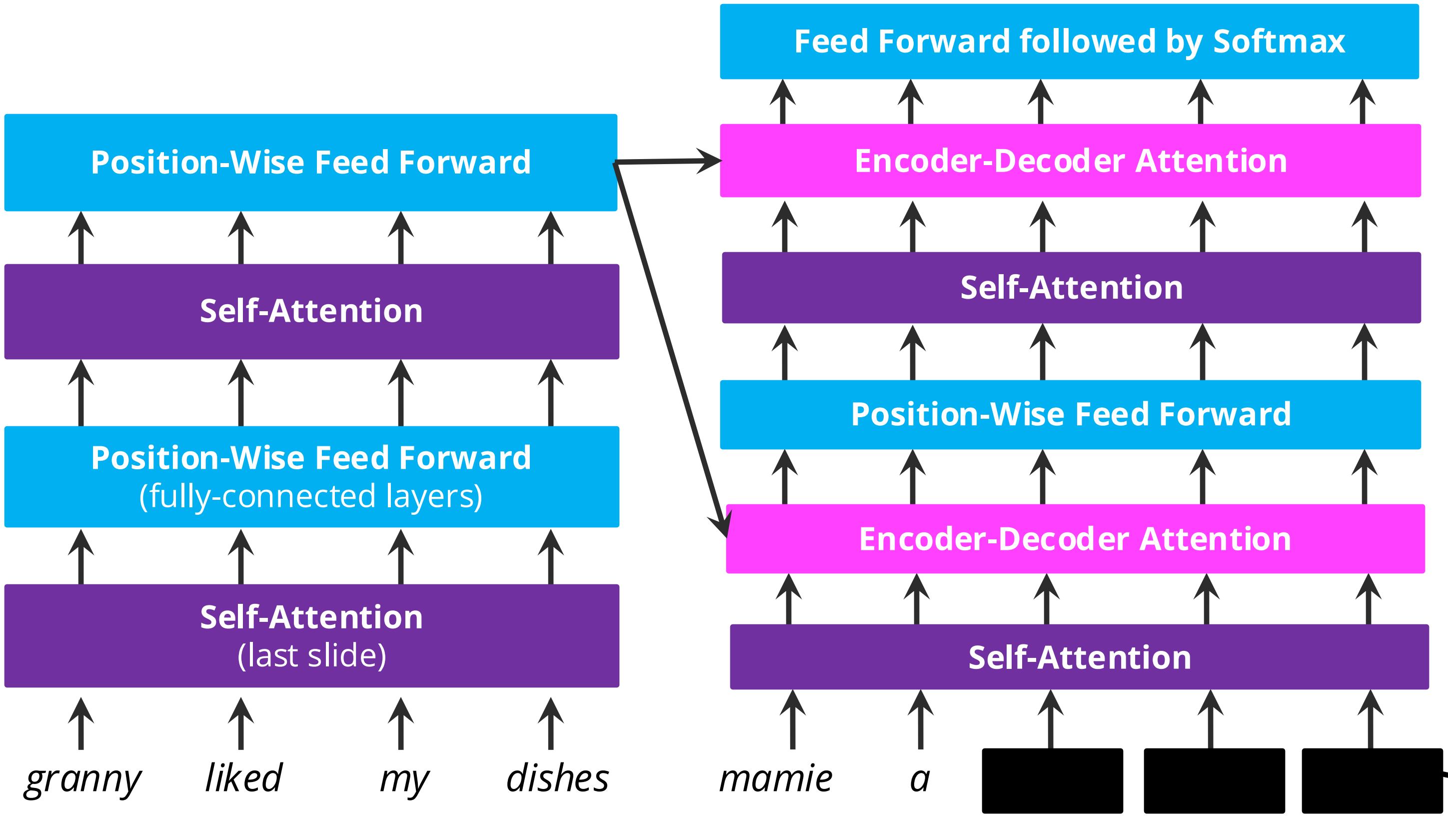


Attention Is All You Need - Transformers

[Vaswani et al., 2017]

Parallelisable

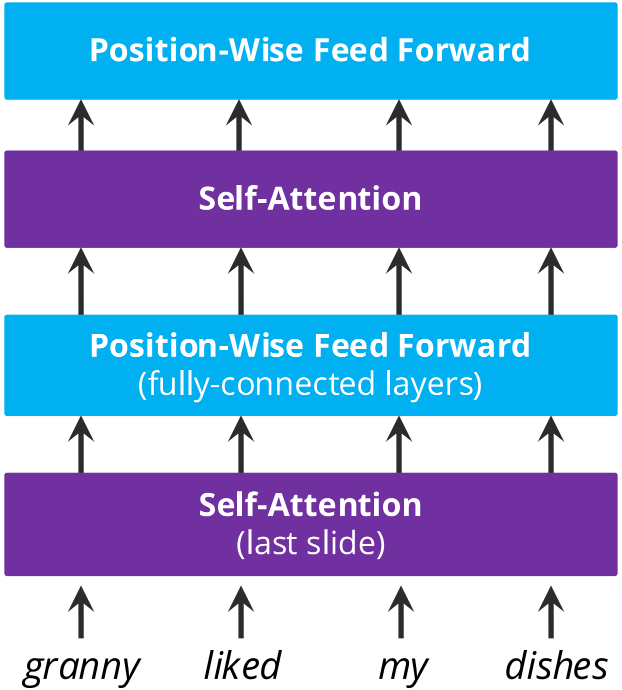
Transformers offer improvements in performance and efficiency.



In the decoder, we **mask out** words that the model is not supposed to have seen yet.

Attention Is All You Need - Transformers

[Vaswani et al., 2017]



One potential issue of self-attention is:

There is no positional information

granny *liked* *my* *dishes*

One approach:

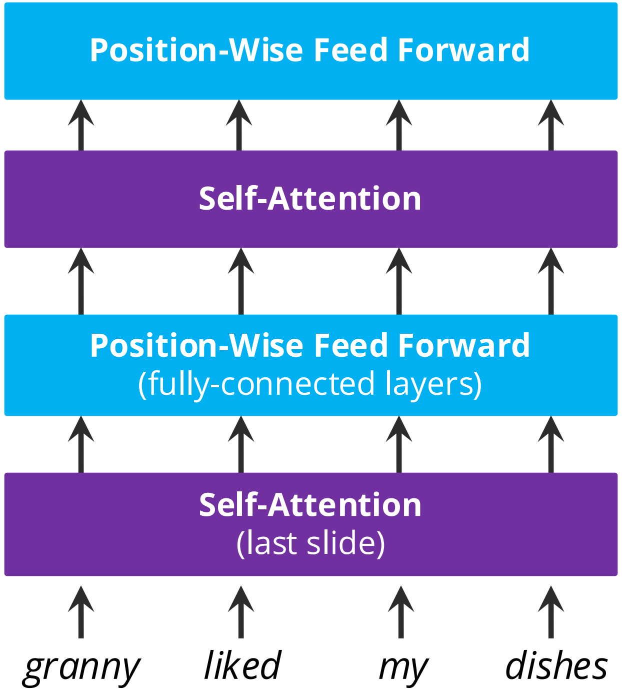
granny $_{0}$ *liked* $_{1}$ *my* $_{2}$ *dishes* $_{3}$

What if: *granny* *liked* *my* *dishes* *and* *… 1000 words*

How can we add positional information?

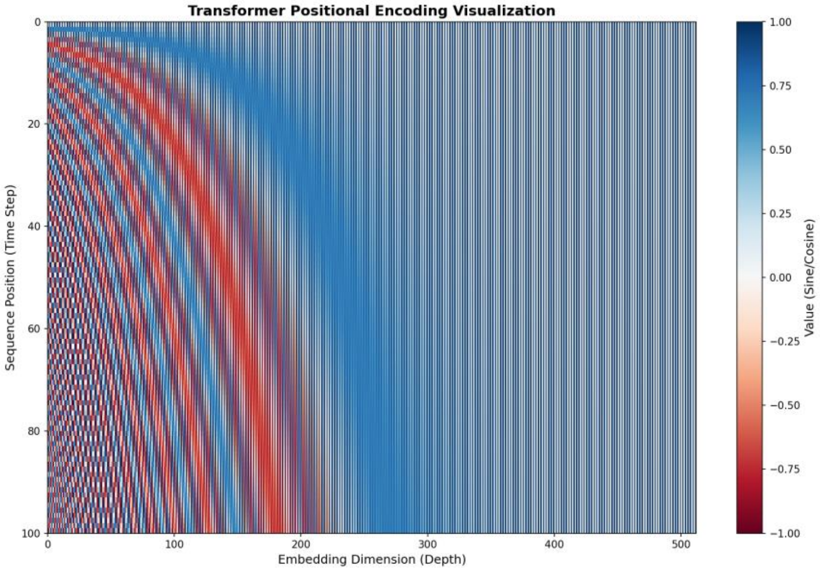
Attention Is All You Need - Transformers

[Vaswani et al., 2017]



Sine and Cosine Positional Encoding

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$



Let’s Look at Some Code!

Simple Transformer Example for Text Classification

Code on GitHub : https://github.com/atapour/dl-pytorch/blob/main/Transformer_Sentiment_Analysis/Transformer_Sentiment_Analysis.ipynb

Code on Colab: https://colab.research.google.com/github/atapour/dl-pytorch/blob/main/Transformer_Sentiment_Analysis/Transformer_Sentiment_Analysis.ipynb

Transformers

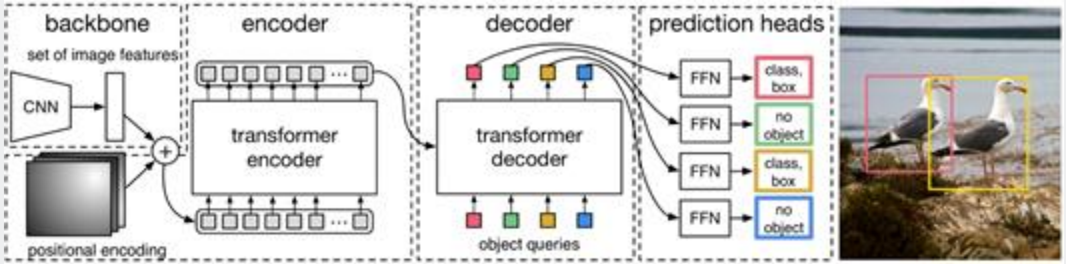
End-to-End Object Detection

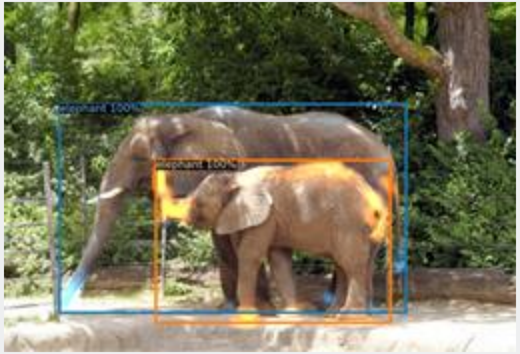
DEtection TRansformer

Fast object detection is crucial for many tasks including self-driving cars. Training end to end is difficult due to the discrete nature of objects.

DETR [Carion et al., 2020] enables global search and 'query' of the image for information. Attention matrices can also be used to make segmentation maps.

Example: architecture and examples





Transformers GPT

GPT Training and Evaluation

GPT-3:

- 175B parameters (96 layers with 96 heads each with 12,228 neurons)
- Modalities: **Text** only

GPT-4:

- **ChatGPT**: much larger (probably)
- Modalities: Text + Image

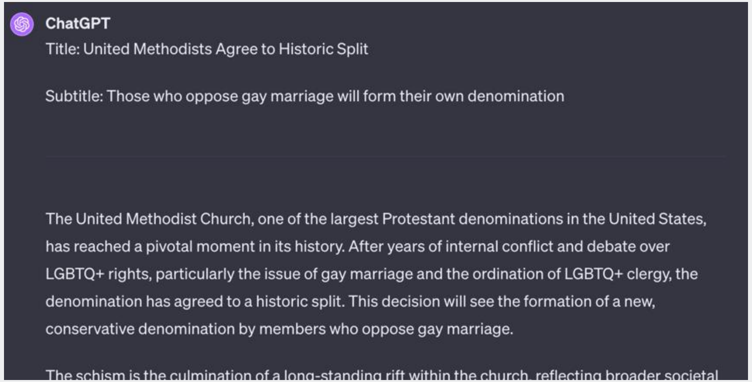
GPT-5:

- Modalities: Text + Image + Audio + Video + Tools

Evaluation Tasks

- Few shot translation
- Reading comprehension (Q&A)
- Closed book question & answering
- Natural language inference

Example: ChatGPT article


ChatGPT

Title: United Methodists Agree to Historic Split

Subtitle: Those who oppose gay marriage will form their own denomination

The United Methodist Church, one of the largest Protestant denominations in the United States, has reached a pivotal moment in its history. After years of internal conflict and debate over LGBTQ+ rights, particularly the issue of gay marriage and the ordination of LGBTQ+ clergy, the denomination has agreed to a historic split. This decision will see the formation of a new, conservative denomination by members who oppose gay marriage.

The schism is the culmination of a long-standing rift within the church, reflecting broader societal

Alternative LLMs

- Google AI's Gemini
- Meta AI's LlaMA
- Anthropic's Claude
- Falcon
- ...

Transformers DALL-E

DALL-E Training

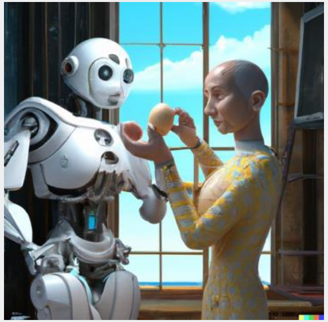
DALL-E [Ramesh et al., 2021]**:**

- 12-billion parameter version of GPT-3
- Generate images from detailed text descriptions
- **DALL·E 2** and now **DALL·E 3**

Capable of anthropomorphised versions of animals and objects, combining unrelated concepts in plausible ways, rendering text, and applying transformations to existing images

Example: DALL-E Image

A robot teaching his grandmother to suck eggs.



Good-looking man with golden hair and big bushy beard.

Transformers

Example: Sora (2024)

Sora is a next-generation text-to-video model trained on large-scale multimodal datasets, enabling high-fidelity, long-range video generation.

- Large-scale video generation model
- Equipped with world modeling capabilities
- Can generate 60-second high-resolution videos
- Produces physically consistent, interactive 3D scenes

Input

Prompt: A stylish woman walks down a Tokyo street filled with warm glowing neon and animated city signage. She wears a black leather jacket, a long red dress, and black boots, and carries a black purse. She wears sunglasses and red lipstick. She walks confidently and casually. The street is damp and reflective, creating a mirror effect of the colorful lights. Many pedestrians walk about.

Output

What we learned today!

❶ Recurrent neural networks

- Definition and implementation
- Backpropagation through time
- vanishing/exploding gradients

❷ Long short-term memory

- Definition
- Properties
- Seq2Seq
- Attention

❸ Transformers

- Self-Attention
- End-to-end object detection
- GPT-3
- DALL-E