

# Copyright Notice

These slides are distributed under the Creative Commons License.

[DeepLearning.AI](#) makes these slides available for educational purposes. You may not use or distribute these slides for commercial purposes. You may make copies of these slides and use or distribute them for educational purposes as long as you cite [DeepLearning.AI](#) as the source of the slides.

For the rest of the details of the license, see <https://creativecommons.org/licenses/by-sa/2.0/legalcode>



deeplearning.ai

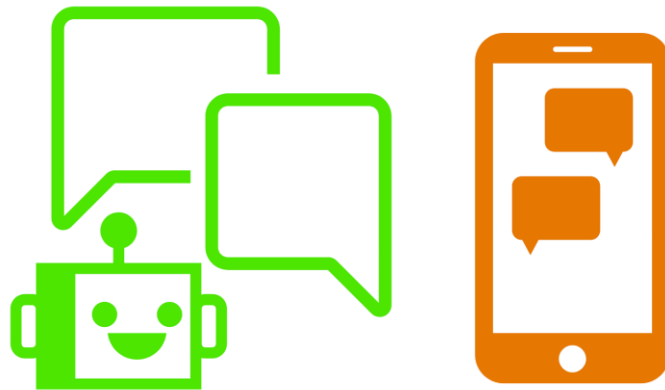
# Tasks with Long Sequences

# Long Text Sequences

Tasks In NLP:



Writing Books



Chatbots

# Chatbots

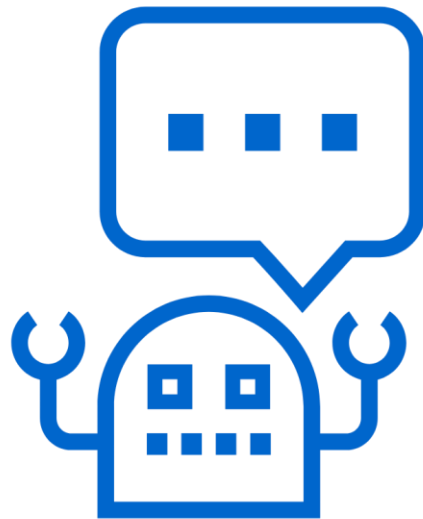
## Context Windows:

**User 1:** What's for dinner?

**Chatbot:** Who's cooking, you or me?

**User 1:** Hey now chatbot.

**Chatbot:** I hope it's not hay, that's  
what horses eat.





deeplearning.ai

# Transformer Complexity

# Transformer Issues

- Attention on sequence of length  $L$  takes  $L^2$  time and memory

$L=100$        $L^2 = 10K$       (0.001s at  
10M ops/s)

$L=1000$      $L^2 = 1M$     (0.1s at 10M  
ops/s)

- $L=10000$      $L^2 = 100M$     (10s at 10M  
ops/s)

GPT-10000096 layers and new models will have  
more

# Attention Complexity

- Attention:  $\text{softmax}(QK^T)V$
- $Q, K, V$  are all  $[L, d_{\text{model}}]$
- $QK^T$  is  $[L, L]$
- Save compute by using area of interest for large  $L$

# Memory with N Layers

- Activations need to be stored for backprop
- Big models are getting bigger
- Compute vs memory tradeoff





deeplearning.ai

# LSH Attention

# What does Attention do?

Select Nearest Neighbors (K,Q) and return corresponding V

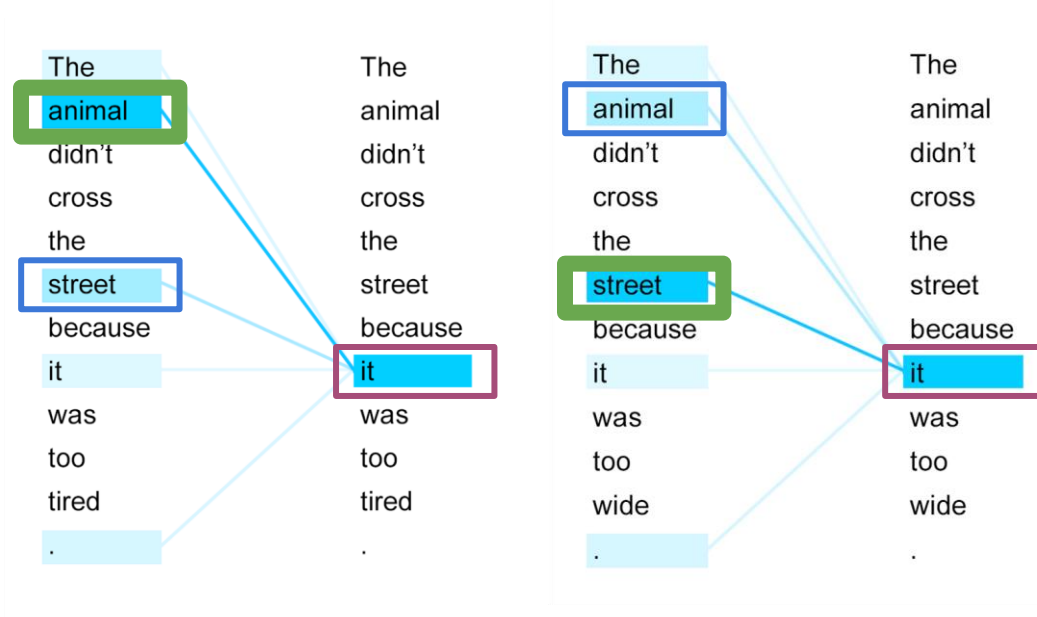


image ©  
([Transformer: A Novel Neural Network Architecture for Language Understanding.](#))

# Nearest Neighbors

Course:

Natural Language Processing with Classification and Vector Spaces

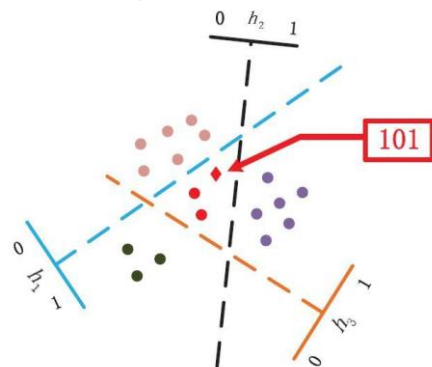
Lessons:

- KNN
- Hash Tables and Hash Functions
- Locality Sensitive Hashing
- Multiple Planes

# Nearest Neighbors

Compute the nearest neighbor to  $q$  among vectors  $\{k_1, \dots, k_n\}$

- Attention computes  $d(q, k_i)$  for  $i$  from 1 to  $n$  which can be slow
- Faster *approximate* uses locality sensitive hashing (LSH)
- Locality sensitive: if  $q$  is close to  $k_i$ :  
 $\text{hash}(q) == \text{hash}(k_i)$
- Achieve by randomly cutting space  
 $\text{hash}(x) = \text{sign}(xR)$        $R: [d, n\_hash\_$



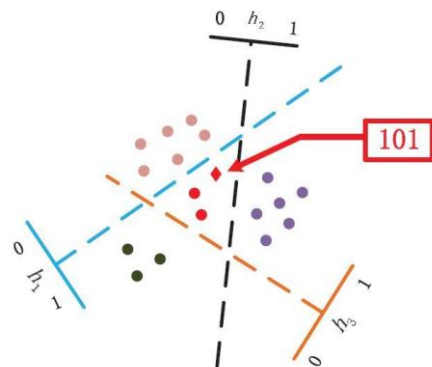
# LSH Attention

Standard Attention:

$$A(Q, K, V) = \text{softmax}(QK^T)V$$

LSH Attention:

- Hash Q and K
- Standard attention within same-hash bins
- Repeat a few times to increase probability of key in the same bin



# LSH Attention

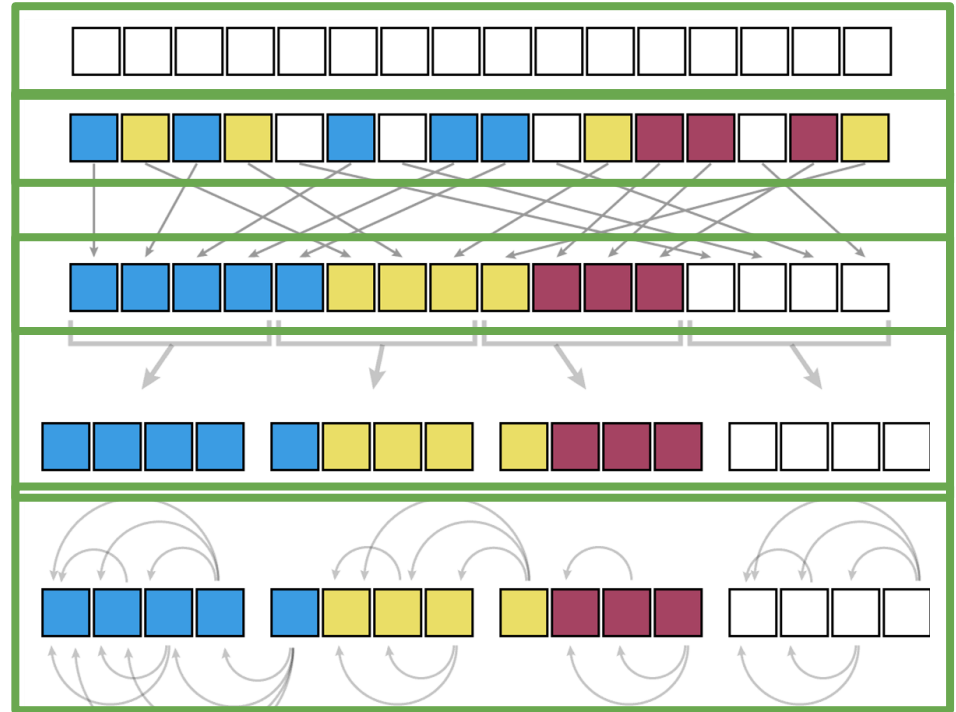
Sequence of Queries = Keys

LSH bucketing

Sort by LSH bucket

Chunk sorted sequence  
to parallelize

Attend within same bucket of  
own chunk and previous chunk





deeplearning.ai

# Motivation for Reversible Layers: Memory!

---

# Memory Efficiency



$d_{\text{model}} = 512$

~ 2 GB

input

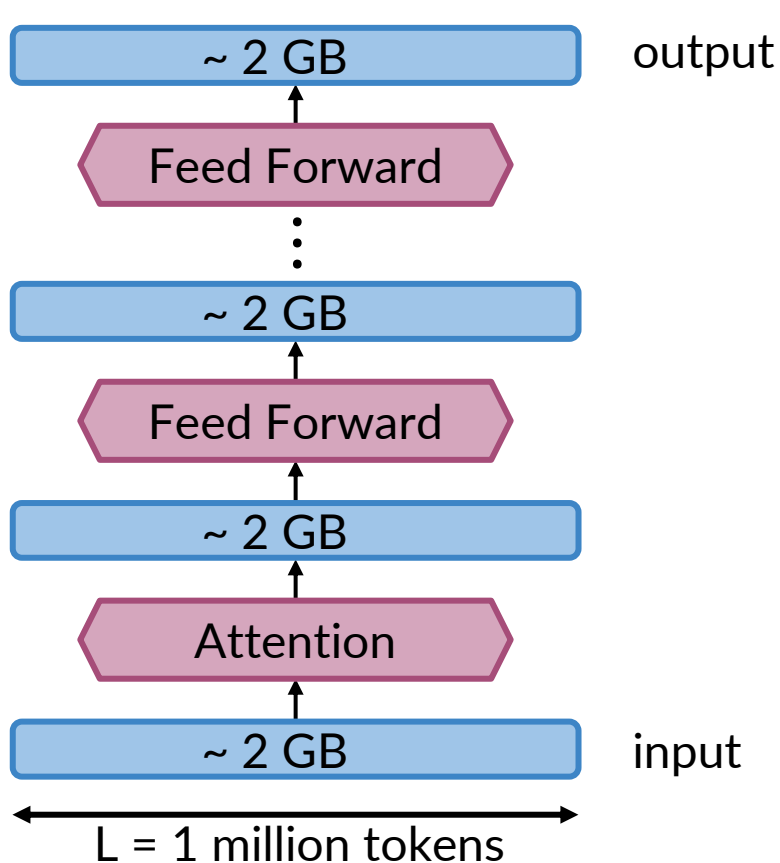
$L = 1$  million tokens



# Memory Efficiency

12 x Attention  
12 x Feed-Forward

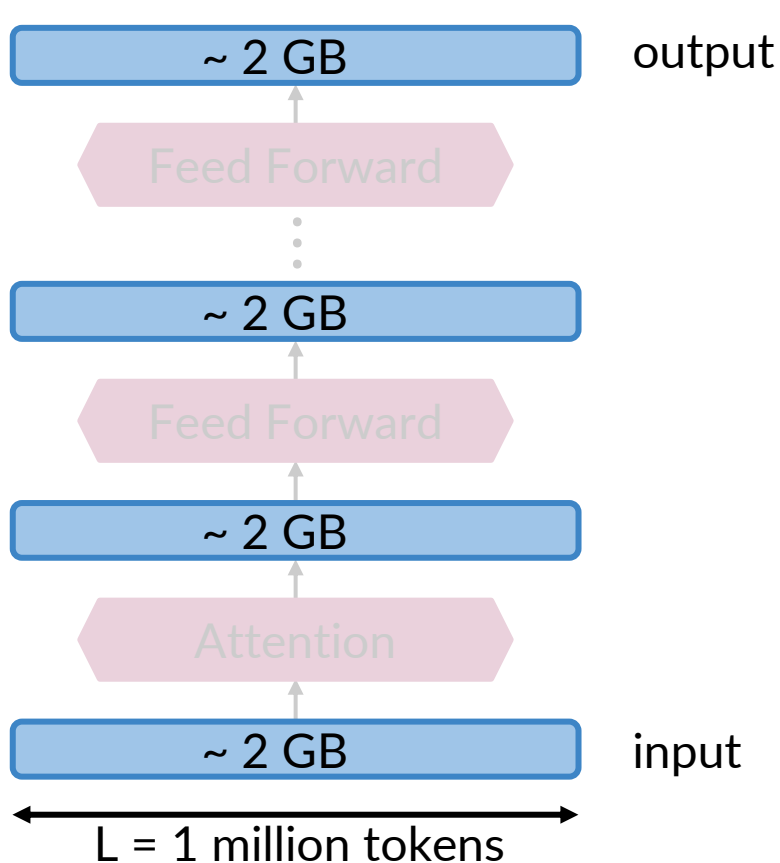
$d_{\text{model}} = 512$



# Memory Efficiency

12 x Attention  
12 x Feed-Forward

**50 GB total**

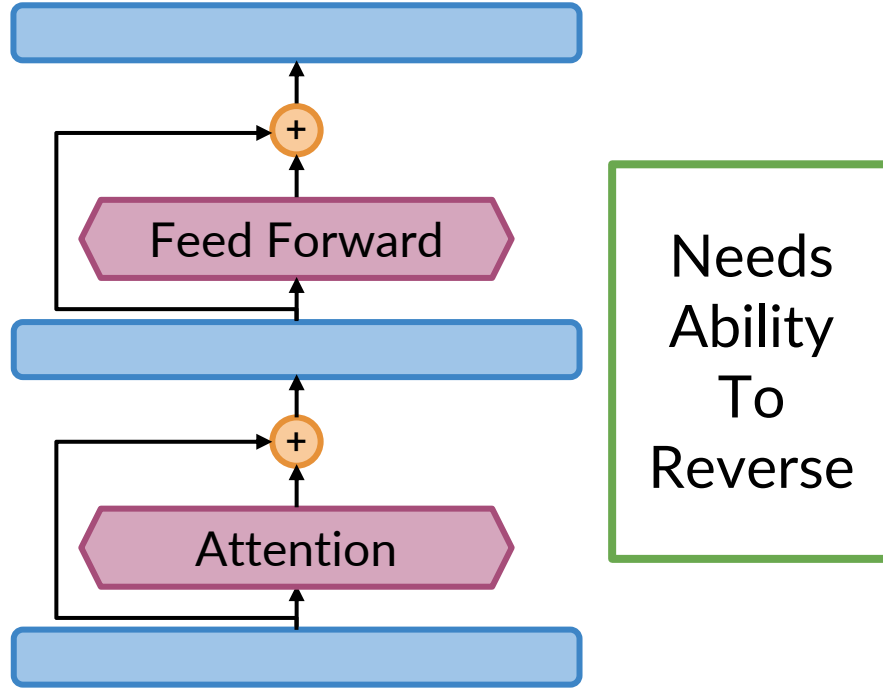




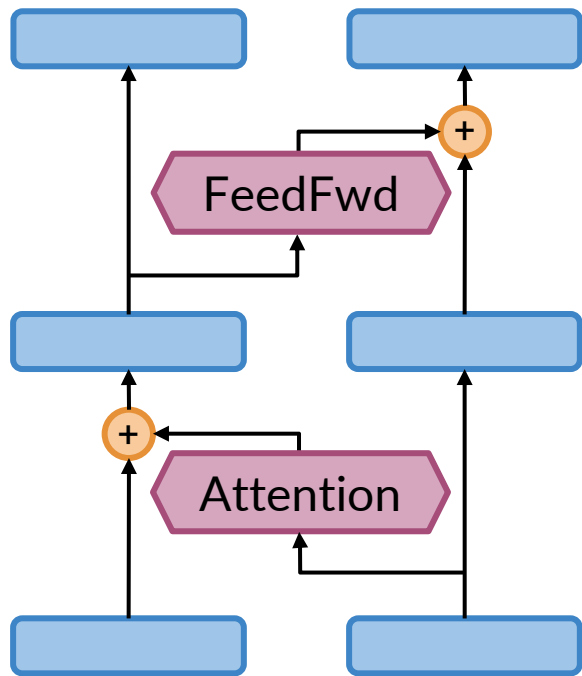
deeplearning.ai

# Reversible Residual Layers

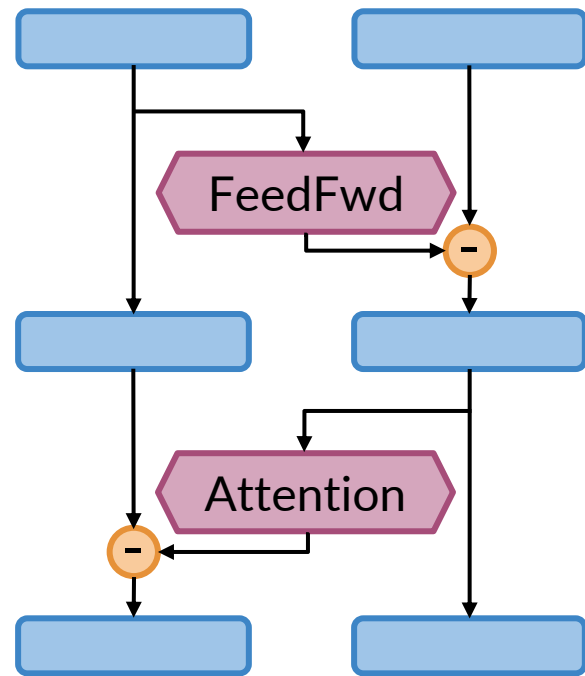
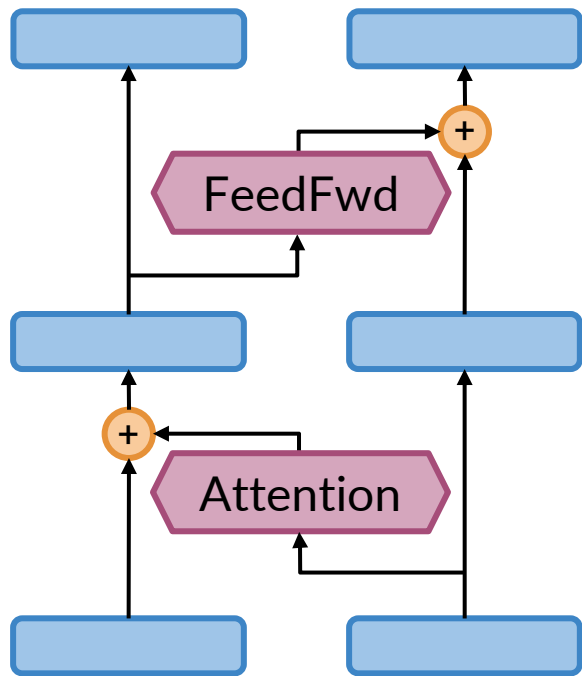
# Residual Blocks in Transformer



# Reversible Residual Blocks



# Reversible layers



# Reversible layers equations

Standard Transformer:

$$y_a = x + \text{Attention}(x)$$

$$y_b = y_a + \text{FeedFwd}(y_a)$$

Reversible:

$$y_1 = x_1 + \text{Attention}(x_2)$$



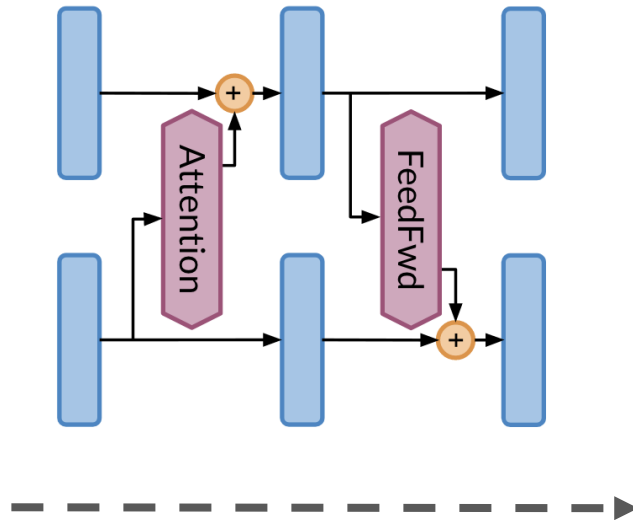
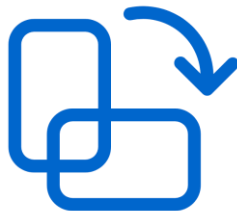
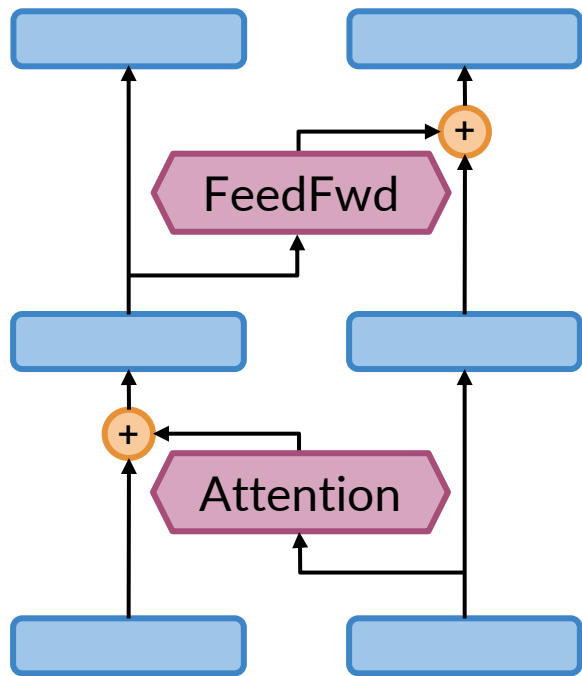
$$y_2 = x_2 + \text{FeedFwd}(y_1)$$

Recompute  $x_1, x_2$  from  $y_1, y_2$ :

$$x_1 = y_1 - \text{Attention}(x_2)$$

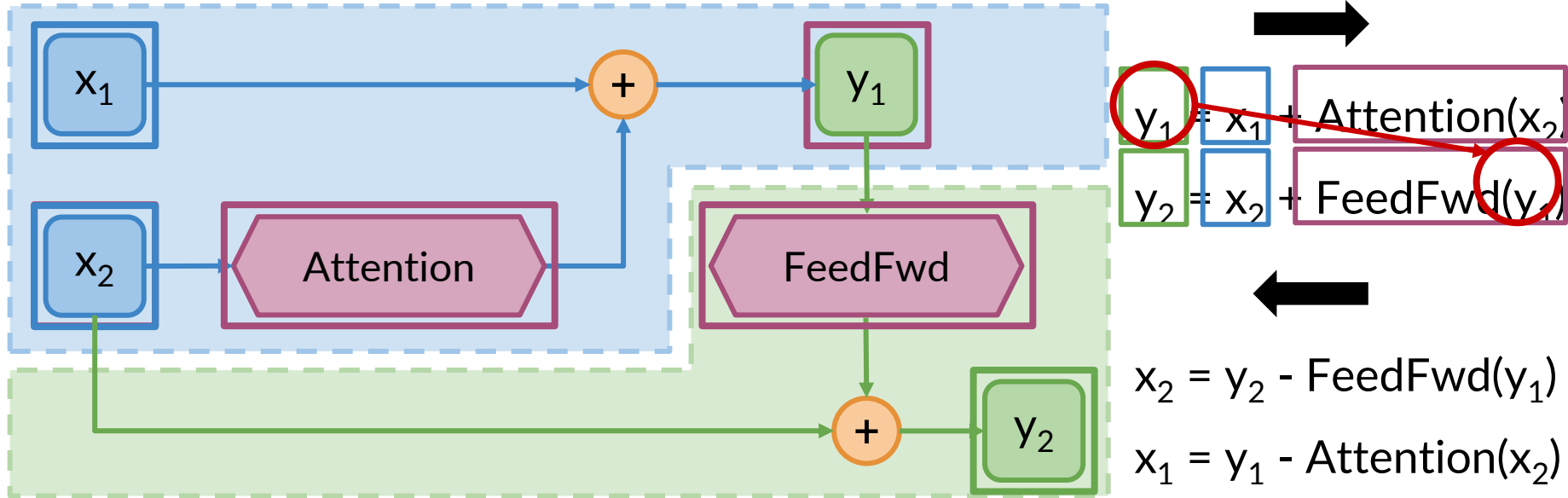
$$x_2 = y_2 - \text{FeedFwd}(y_1)$$

# Reversible layers equations

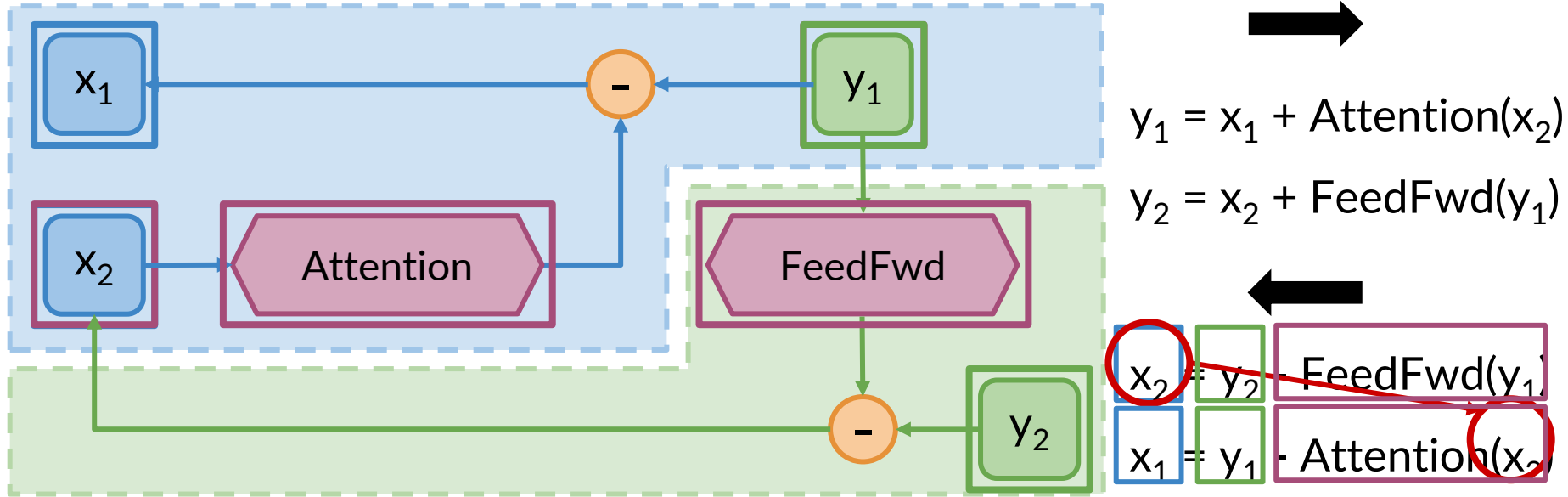




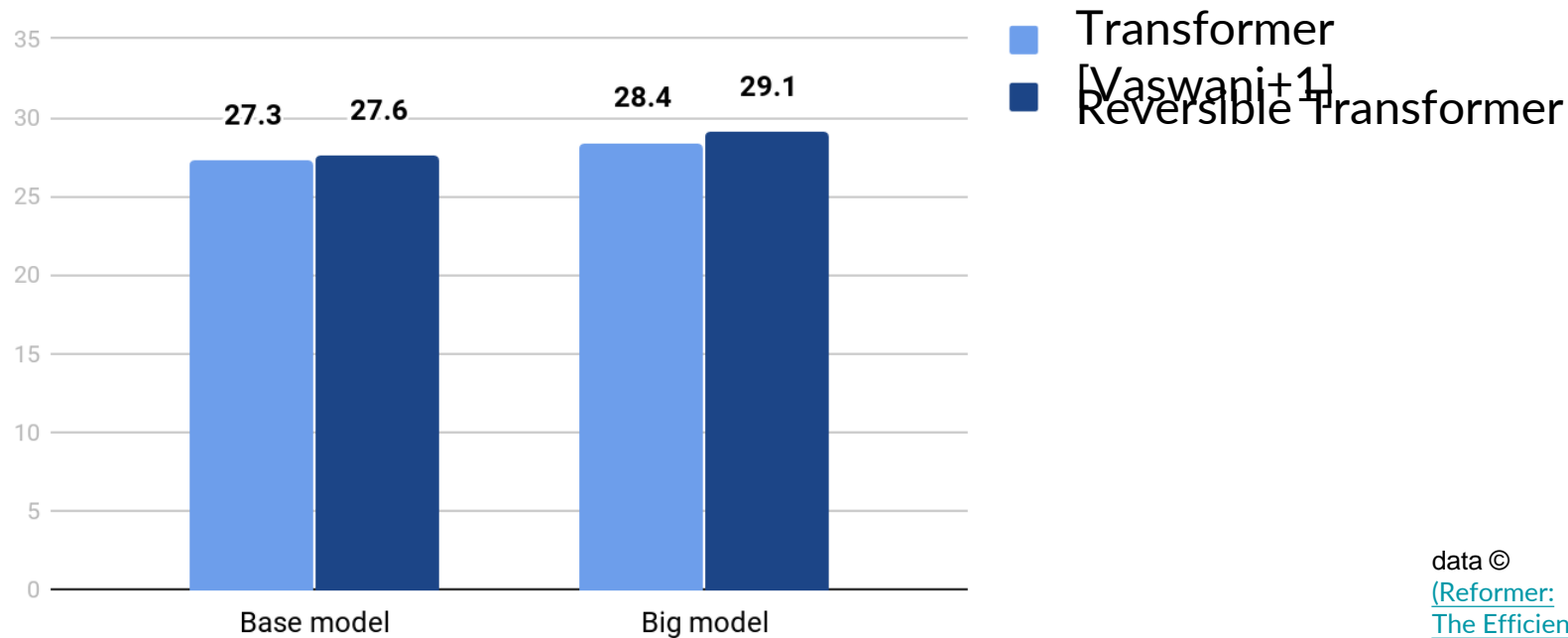
# Reversible layers equations



# Reversible layers equations



# Reversible Transformer: BLEU Scores



data ©  
[\(Reformer:  
The Efficient  
Transformer\)](#)



deeplearning.ai

# Reformer

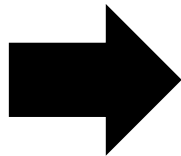
---

# Reformer

The Reversible Transformer



$L = 1$  million tokens



**1 GPU**  
(16 GB)

# Reformer

- LSH Attention
- Reversible Layers

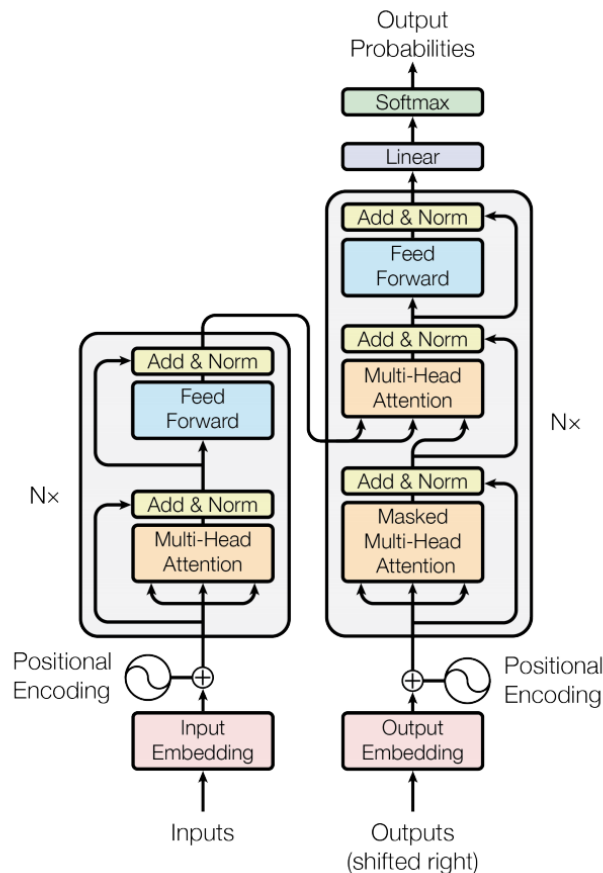


image ©  
[\(Attention Is All You Need\)](#)

# Reformer

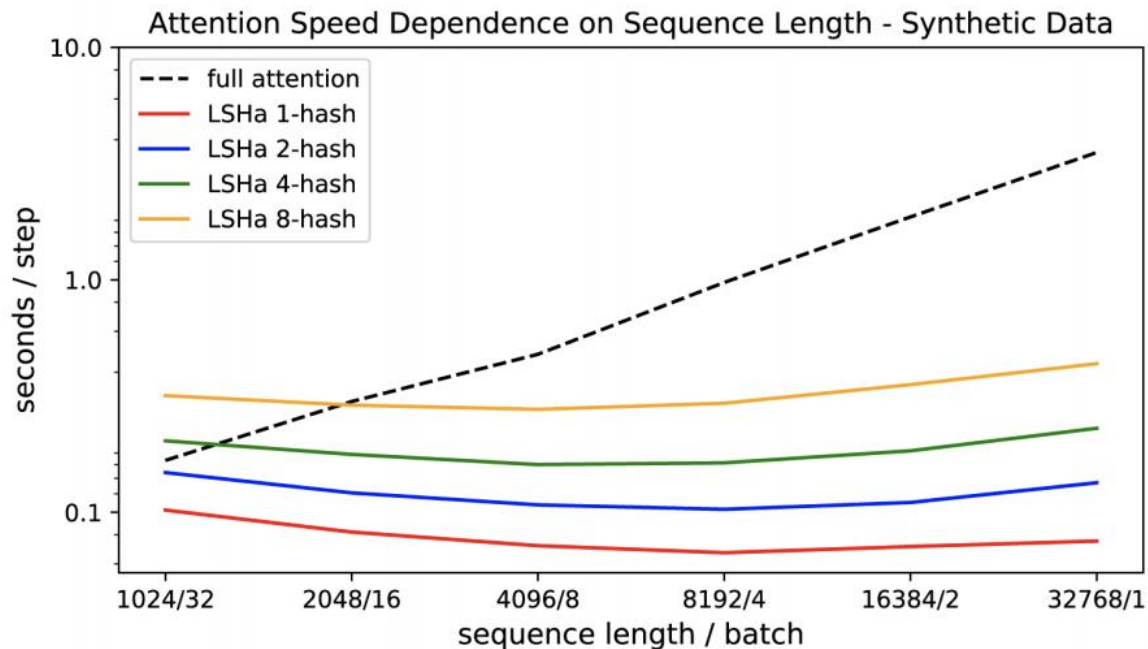
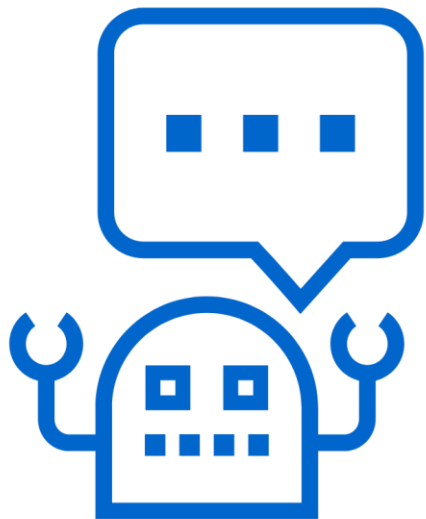


image ©  
[\(Reformer:  
The Efficient  
Transformer\)](#)

# Chatbot



- Reformer
- MultiWOZ dataset
- Trax