

# Derin Öğrenmeye Giriş

# Kursa Başlamadan Önce

1. Bu kurs tamamlandığı takdirde giriş düzeyi yapay zeka algoritmaları ve veri analizine temel oluşturabilecek genel bilgileri edinmiş olacaksınız
2. Spesifik konular anlaşılması zor ve kişide ders esnasında mantığın oturması kolay olmayacağından bol bol bireysel pratik gerekmektedir
3. Slaytlar yazılara boğulmadan görsellerle anlatılacaktır. Bu yüzden ders esnasında not tutulması **son derece** önemlidir
4. Konu başlıkları temel düzey algoritmalar için yeterli olduğundan başlıklar araştırılmalı, bol bol uygulama ve teorik bilgiler içeren sitelerde araştırma yapılmalıdır

# Bu eğitimde neler öğreneceksiniz

1. Natural Language Processing over time
2. Transformer Motivation: How an RNN works
3. Transformer
  - a. Encoder-Decoder
  - b. Attention
  - c. Word Embedding
  - d. Positional Encoding
  - e. Dropout
  - f. Residual Connection
  - g. Advantages
4. Improved Transformers
5. Courses on NLP

# Natural Language Processing

Common tasks include:

- text classification
- translation
- summarization
- named entity recognition
- dialogue (chatbots)
- question answering

For more info visit <https://paperswithcode.com/area/natural-language-processing>

# Machine Translation on WMT2014 English-German

Leaderboard

Dataset

View

BLEU score

▼

by

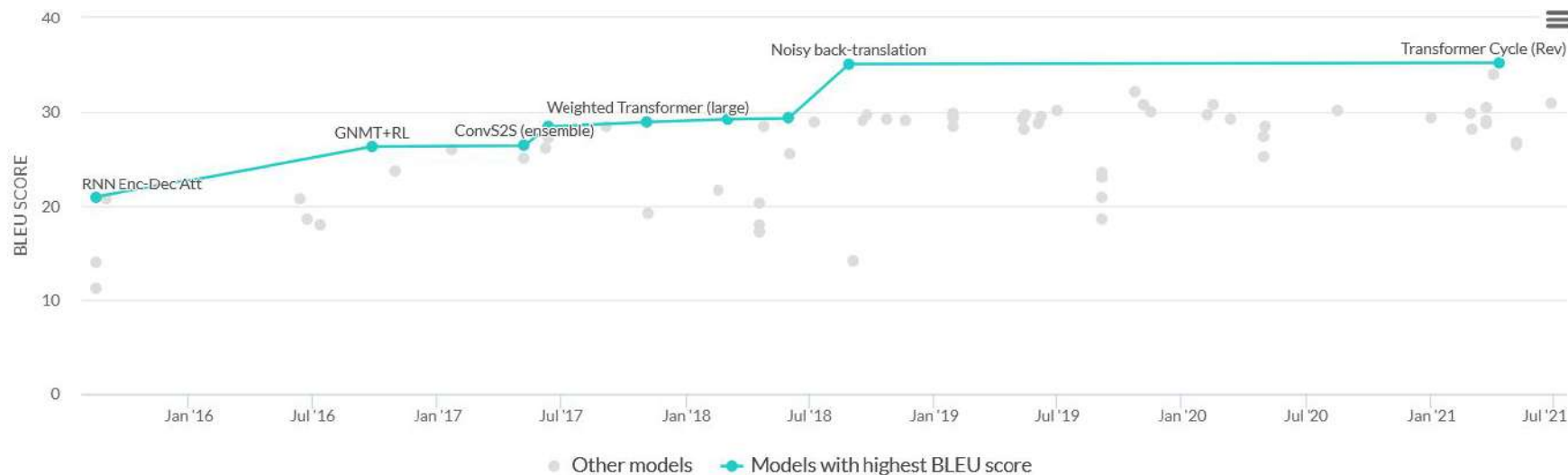
Date


















▼

for

All models

▼



1	<b>Transformer Cycle</b> (Rev)	35.14	33.54	✓	<a href="#">Lessons on Parameter Sharing across Layers in Transformers</a>			2021	<a href="#">Transformer</a>
2	<b>Noisy back-translation</b>	35.0	33.8	✓	<a href="#">Understanding Back-Translation at Scale</a>			2018	
3	<b>Transformer+Rep</b> (Uni)	33.89	32.35	✓	<a href="#">Rethinking Perturbations in Encoder-Decoders for Fast Training</a>			2021	<a href="#">Transformer</a>
4	<b>T5-11B</b>	32.1		✓	<a href="#">Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer</a>			2019	<a href="#">Transformer</a>
5	<b>Transformer + R-Drop</b>	30.91		×	<a href="#">R-Drop: Regularized Dropout for Neural Networks</a>			2021	<a href="#">Transformer</a>
6	<b>BERT-fused NMT</b>	30.75		×	<a href="#">Incorporating BERT into Neural Machine Translation</a>			2020	<a href="#">Transformer</a>
7	<b>Data Diversification - Transformer</b>	30.7		×	<a href="#">Data Diversification: A Simple Strategy For Neural Machine Translation</a>			2019	<a href="#">Transformer</a>
8	<b>Mask Attention Network</b> (big)	30.4	215M	×	<a href="#">Mask Attention Networks: Rethinking and Strengthen Transformer</a>			2021	
9	<b>Transformer</b> (ADMIN init)	30.1	29.5	×	<a href="#">Very Deep Transformers for Neural Machine Translation</a>			2020	<a href="#">Transformer</a>

---

## Attention Is All You Need

---

# Transformer

<b>Ashish Vaswani*</b> Google Brain avaswani@google.com	<b>Noam Shazeer*</b> Google Brain noam@google.com	<b>Niki Parmar*</b> Google Research nikip@google.com	<b>Jakob Uszkoreit*</b> Google Research usz@google.com
<b>Llion Jones*</b> Google Research llion@google.com	<b>Aidan N. Gomez* †</b> University of Toronto aidan@cs.toronto.edu	<b>Łukasz Kaiser*</b> Google Brain lukaszkaiser@google.com	
<b>Illia Polosukhin* †</b> illia.polosukhin@gmail.com			

### Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. 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.

# Recurrent Neural Network (RNN)

## The Vanilla RNN Model

First time-step ( $t = 1$ ):

$$\mathbf{h}_1 = \tanh(W^{xh} \cdot \mathbf{x}_1 + W^{hh} \cdot \mathbf{h}_0)$$

$$\hat{\mathbf{y}}_1 = \text{softmax}(W^{hy} \cdot \mathbf{h}_1)$$

$$L_1 = CE(\hat{\mathbf{y}}_1, \mathbf{y}_1)$$

In general:

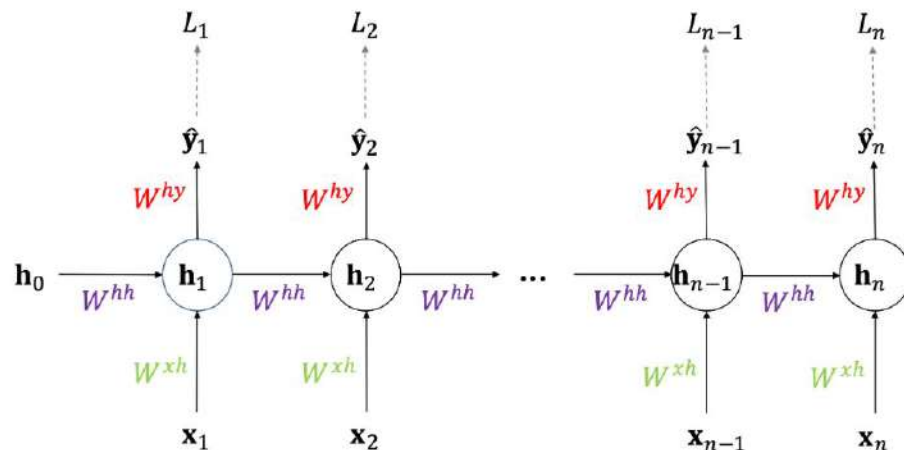
$$\mathbf{h}_t = \tanh(W^{xh} \cdot \mathbf{x}_t + W^{hh} \cdot \mathbf{h}_{t-1})$$

$$\hat{\mathbf{y}}_t = \text{softmax}(W^{hy} \cdot \mathbf{h}_t)$$

$$L_t = CE(\hat{\mathbf{y}}_t, \mathbf{y}_t)$$

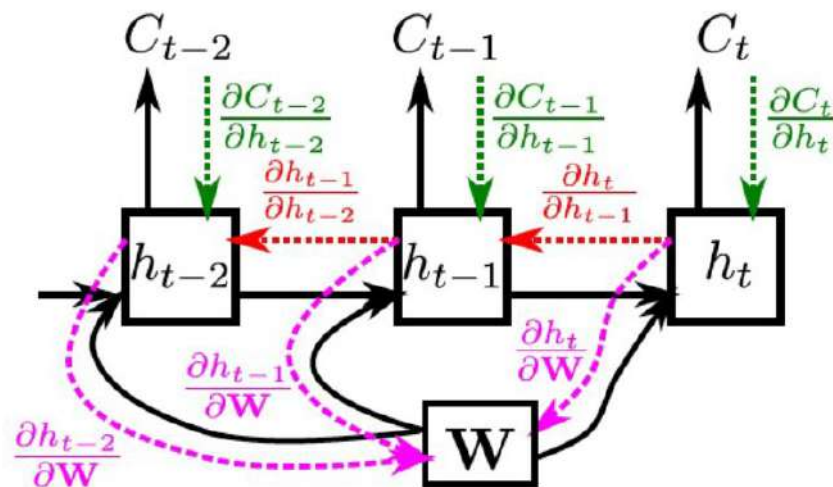
In total:

$$L = \sum_t L_t$$



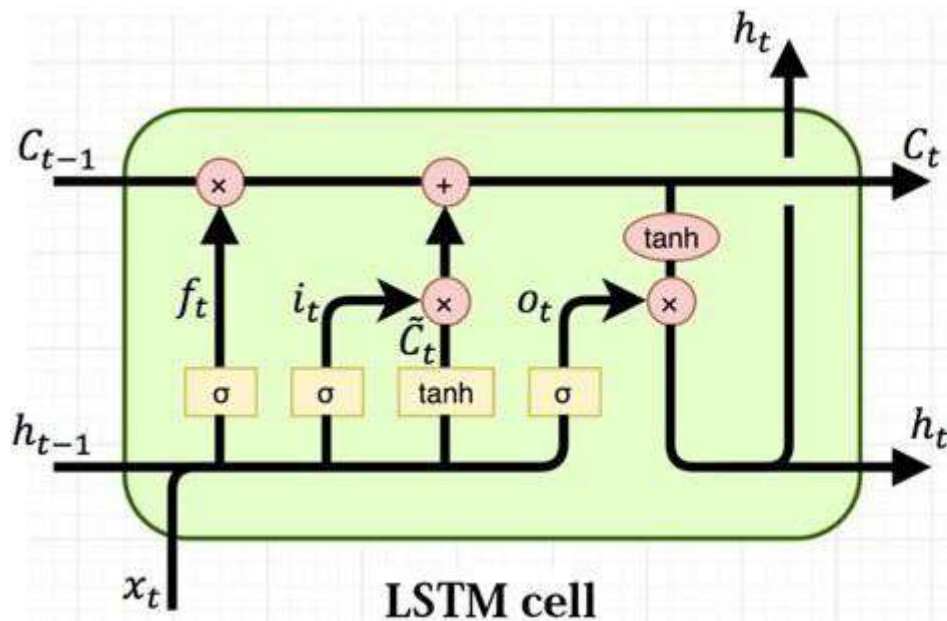


# Backpropagation through time



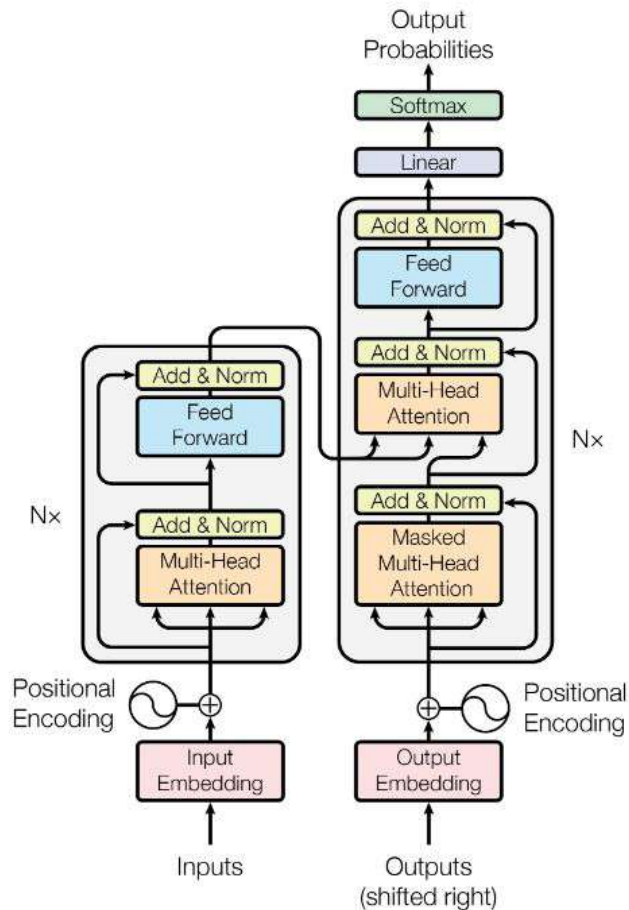
$$\frac{\partial C_t}{\partial W} = \sum_{t'=1}^t \frac{\partial C_t}{\partial h_t} \frac{\partial h_t}{\partial h_{t'}} \frac{\partial h_{t'}}{\partial W}, \text{ where } \frac{\partial h_t}{\partial h_{t'}} = \prod_{k=t'+1}^t \frac{\partial h_k}{\partial h_{k-1}}$$

# Long Short Term Memory (LSTM)

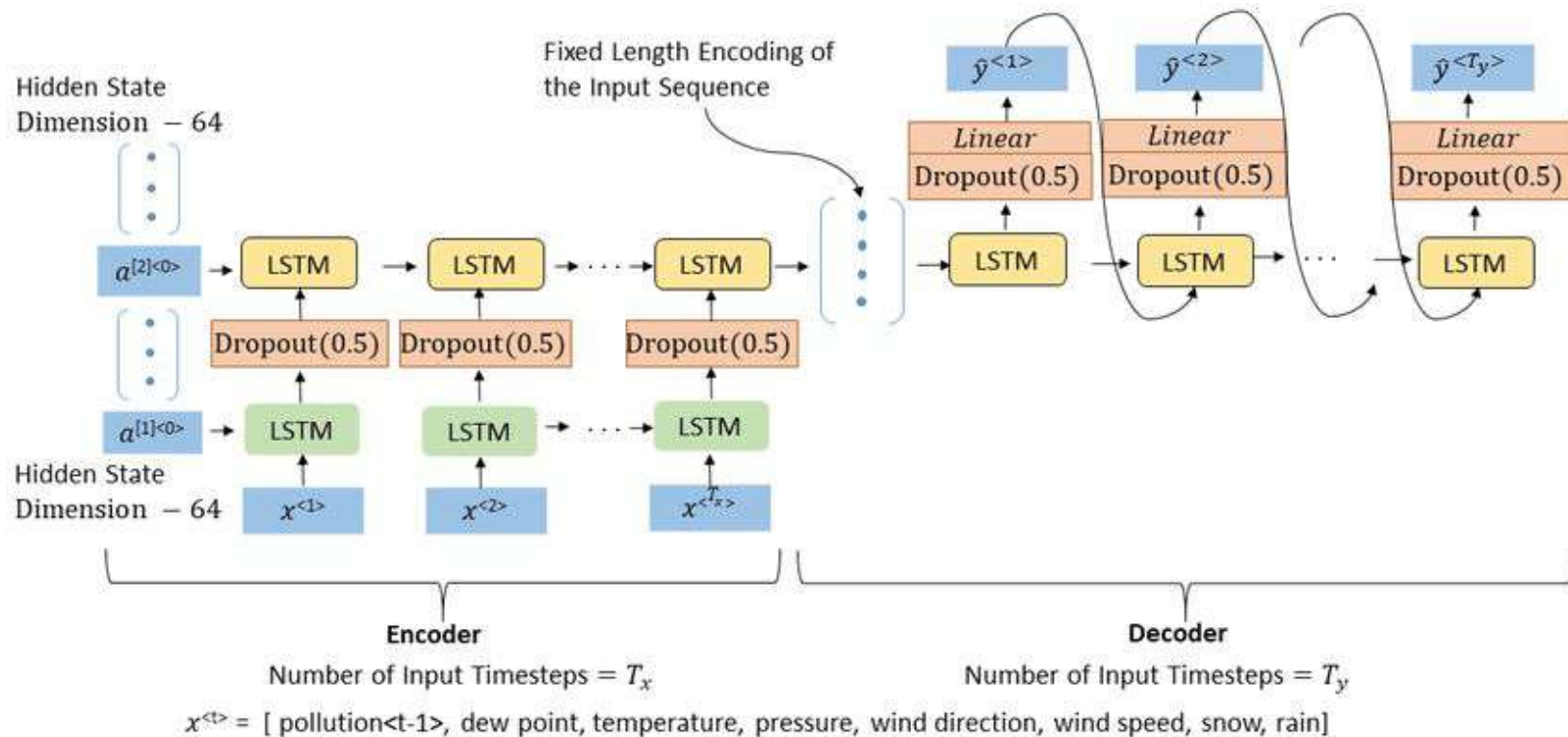


$$\begin{aligned}
 i_t &= \sigma(x_t U^i + h_{t-1} W^i) \\
 f_t &= \sigma(x_t U^f + h_{t-1} W^f) \\
 o_t &= \sigma(x_t U^o + h_{t-1} W^o) \\
 \tilde{C}_t &= \tanh(x_t U^g + h_{t-1} W^g) \\
 C_t &= \sigma(f_t * C_{t-1} + i_t * \tilde{C}_t) \\
 h_t &= \tanh(C_t) * o_t
 \end{aligned}$$

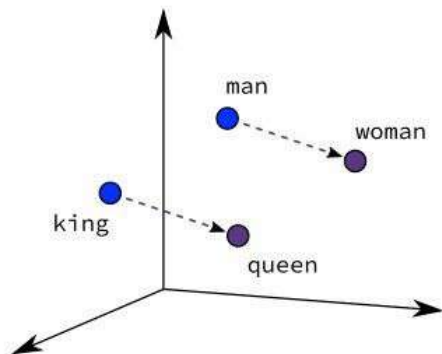
# Transformer



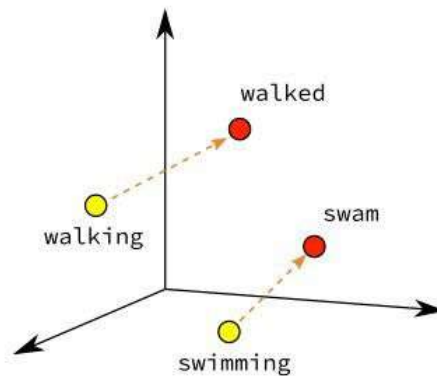
# Encoder Decoder



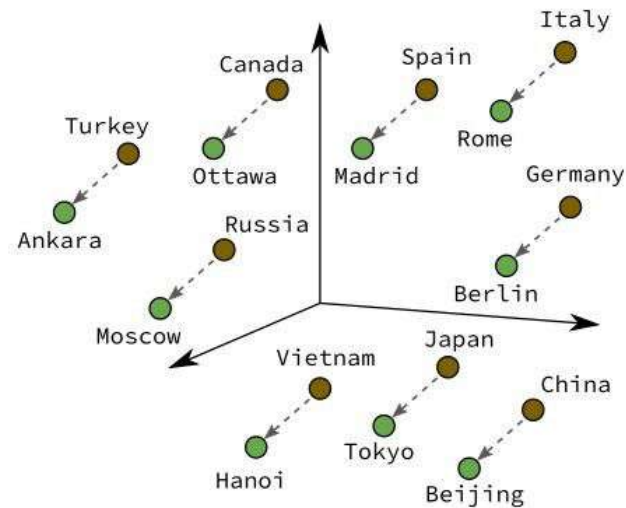
# Word Embedding



Male-Female



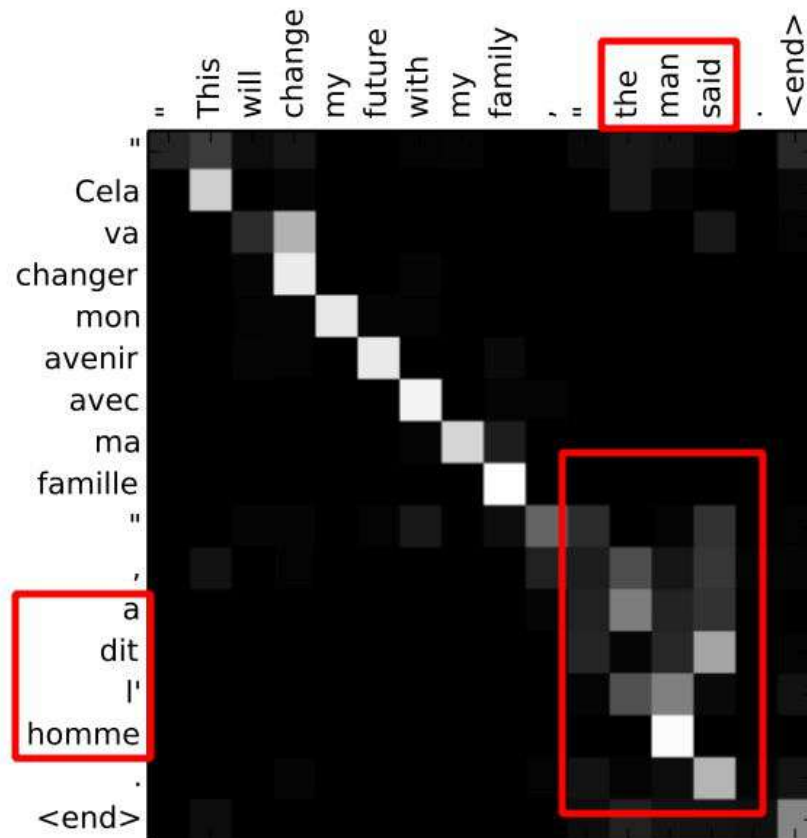
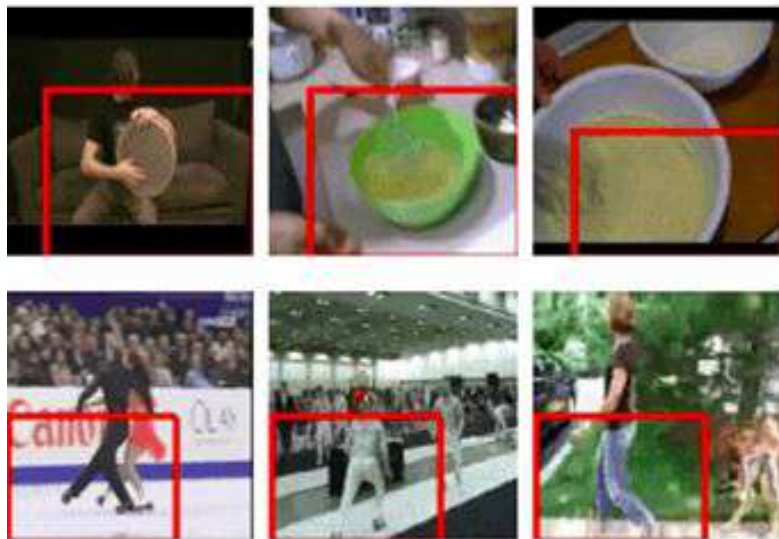
Verb Tense



Country-Capital

<https://projector.tensorflow.org/>

# Attention



# Scaled Dot Product Attention

$$attention(Q,K,V) = softmax\left(\frac{QK^T}{\sqrt{d}}\right) V \quad (4)$$

- Q -> query
- K -> key
- V -> value

yemek	koşarken	yemek	yemek							
<table border="1" style="background-color: #4a7ebb; color: white; width: 100px; height: 40px;"> <tr> <td style="padding: 5px;">2</td> <td style="padding: 5px;">-1</td> </tr> </table>	2	-1	<div style="display: flex; align-items: center; justify-content: center;"> <div style="width: 10px; height: 10px; background-color: black; border-radius: 50%; margin: 0 10px;"></div> <table border="1" style="background-color: #4a7ebb; color: white; width: 40px; height: 100px;"> <tr> <td style="padding: 5px;">-3</td> </tr> <tr> <td style="padding: 5px;">1</td> </tr> </table> </div>	-3	1	<div style="display: flex; align-items: center; justify-content: center;"> <div style="width: 10px; height: 10px; background-color: black; border-radius: 50%; margin: 0 10px;"></div> <table border="1" style="background-color: #4a7ebb; color: white; width: 40px; height: 100px;"> <tr> <td style="padding: 5px;">2</td> </tr> <tr> <td style="padding: 5px;">-1</td> </tr> </table> </div>	2	-1	$= 2 * (-3) + (-1) * 1$ $= -7$	$= 5$
2	-1									
-3										
1										
2										
-1										

(1)

yemek	yedik				
<table border="1" style="background-color: #4a7ebb; color: white; width: 100px; height: 40px;"> <tr> <td style="padding: 5px;">2</td> <td style="padding: 5px;">-1</td> </tr> </table>	2	-1	<div style="display: flex; align-items: center; justify-content: center;"> <div style="width: 10px; height: 10px; background-color: black; border-radius: 50%; margin: 0 10px;"></div> <table border="1" style="background-color: #4a7ebb; color: white; width: 40px; height: 100px;"> <tr> <td style="padding: 5px;">2</td> </tr> <tr> <td style="padding: 5px;">1</td> </tr> </table> </div>	2	1
2	-1				
2					
1					
	$= 3$				



$$\textit{softmax}(-7) = 10^{-6}$$

$$\textit{softmax}(5) = 0.88 \quad (2)$$

$$\textit{softmax}(3) = 0.12$$

$$10^{-6} \begin{array}{|c|c|} \hline \text{koşarken} \\ \hline -3 & 1 \\ \hline \end{array} + 0.88 \begin{array}{|c|c|} \hline \text{yemek} \\ \hline 2 & -1 \\ \hline \end{array} + 0.12 \begin{array}{|c|c|} \hline \text{yedik} \\ \hline 2 & 1 \\ \hline \end{array}$$

(3)

$$= \begin{array}{|c|c|} \hline \text{yemek'} \\ \hline 2 & -0.76 \\ \hline \end{array}$$

$$\begin{array}{l} \text{koşarken} \\ \text{yemek} \\ \text{yedik} \end{array} \begin{array}{|c|c|} \hline \text{Query} \\ \hline \begin{array}{cc} -3 & 1 \\ 2 & -1 \\ 2 & 1 \end{array} \\ \hline \end{array} \times \begin{array}{|c|c|c|} \hline \text{Key}^T \\ \hline \begin{array}{ccc} -3 & 2 & 2 \\ 1 & -1 & 1 \end{array} \\ \hline \begin{array}{c} \text{koşarken} \\ \text{yemek} \\ \text{yedik} \end{array} \end{array} = \begin{array}{l} \text{koşarken} \\ \text{yemek} \\ \text{yedik} \end{array} \begin{array}{|c|c|c|} \hline \begin{array}{ccc} 10 & -7 & -5 \\ -7 & 5 & 3 \\ -5 & 3 & 5 \end{array} \\ \hline \begin{array}{c} \text{koşarken} \\ \text{yemek} \\ \text{yedik} \end{array} \end{array} \quad (5)$$

$$\text{softmax}\left(\frac{\begin{array}{|c|c|c|} \hline 10 & -7 & -5 \\ \hline -7 & 5 & 3 \\ \hline -5 & 3 & 5 \\ \hline \end{array}}{\sqrt{2}}\right) = \begin{array}{|c|c|c|} \hline 0.99 & 10^{-6} & 10^{-5} \\ \hline 10^{-4} & 0.80 & 0.19 \\ \hline 10^{-4} & 0.19 & 0.80 \\ \hline \end{array} \quad (6)$$

0.99	$10^{-6}$	$10^{-5}$
$10^{-4}$	0.80	0.19
$10^{-4}$	0.19	0.80

×

Value	
-3	1
2	-1
2	1

=

Y	
-2.97	0.99
1.98	-0.61
1.98	0.61

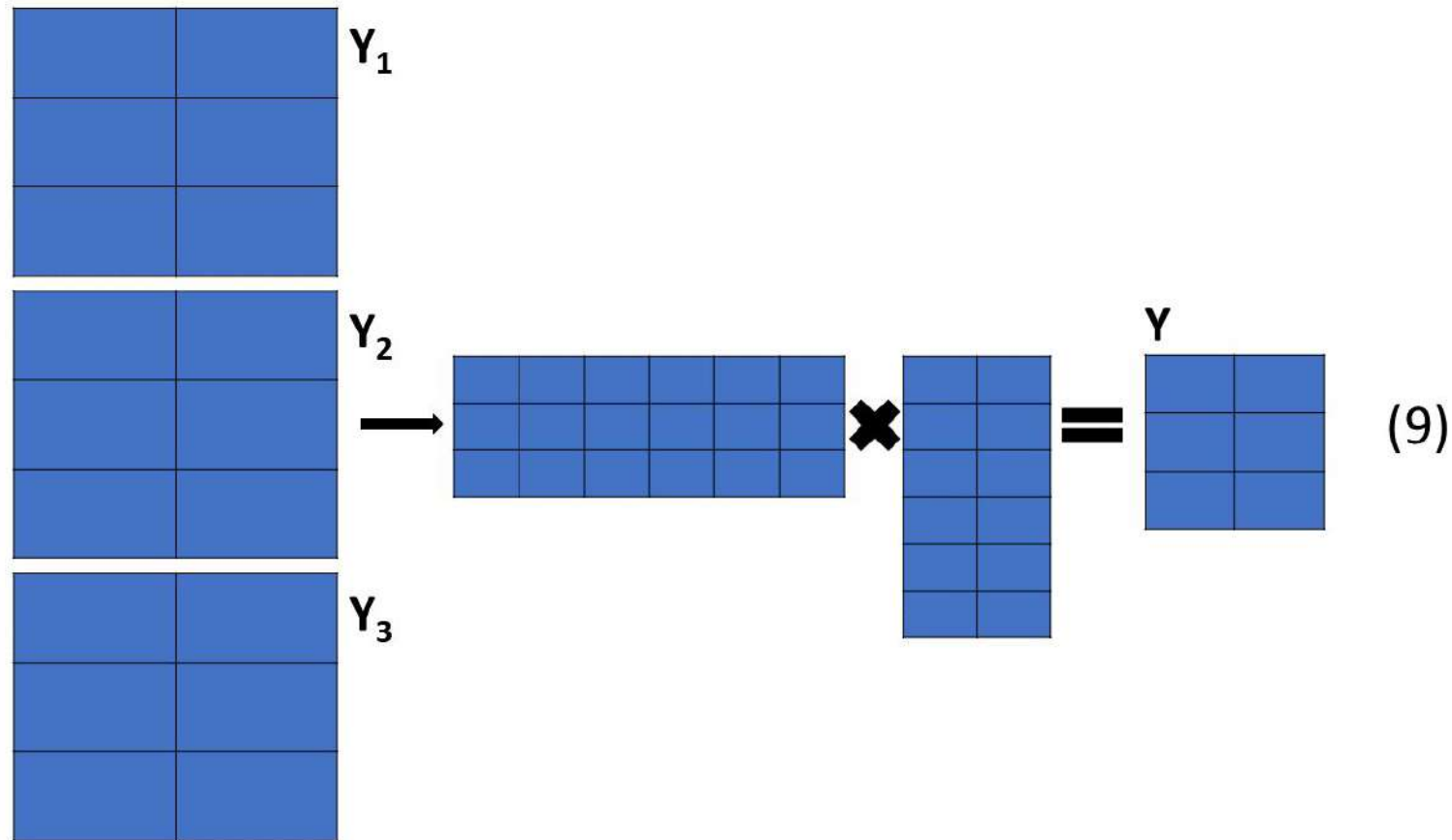
koşarken

yemek (7)

yedik

$$\begin{array}{c}
 \text{Word embedding} \\
 \begin{array}{c|c}
 \text{koşarken} & -3 & 1 \\
 \text{yemek} & 2 & -1 \\
 \text{yedik} & 2 & 1
 \end{array}
 \end{array}
 \begin{array}{c}
 \times \\
 \times \\
 \times
 \end{array}
 \begin{array}{c}
 \text{Weight}_Q \\
 \text{Weight}_K \\
 \text{Weight}_V
 \end{array}
 \begin{array}{c}
 = \\
 = \\
 =
 \end{array}
 \begin{array}{c}
 Q \\
 K \\
 V
 \end{array}
 \quad (8)$$

The diagram illustrates the matrix multiplication of word embeddings with weight matrices to produce query (Q), key (K), and value (V) matrices. The word embedding matrix is a 3x2 matrix with values:  $\begin{bmatrix} -3 & 1 \\ 2 & -1 \\ 2 & 1 \end{bmatrix}$ . Each of the three weight matrices ( $\text{Weight}_Q$ ,  $\text{Weight}_K$ , and  $\text{Weight}_V$ ) is a 2x4 matrix, and the resulting Q, K, and V matrices are 3x4 matrices.



# Positional Encoding

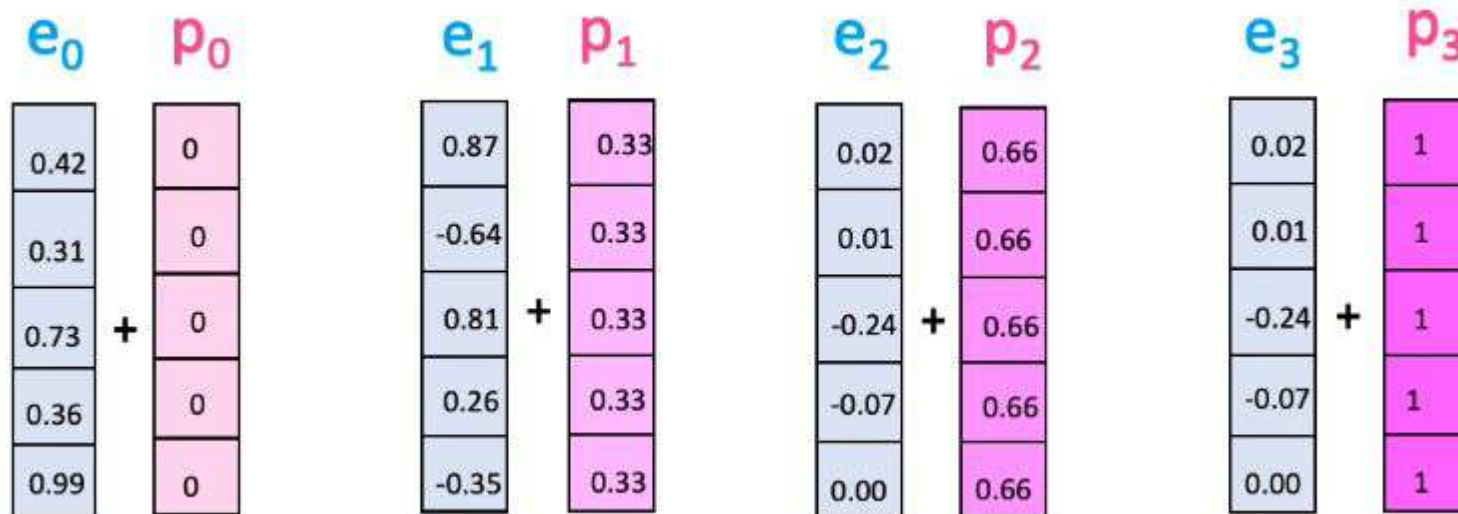
Solution #1

$e_0$	$p_0$		$e_1$	$p_1$		$e_2$	$p_2$		$e_3$	$p_3$
0.42	0		0.87	1		0.02	2		0.02	3
0.31	0		-0.64	1		0.01	2		0.01	3
0.73	0	+	0.81	1	+	-0.24	2	+	-0.24	3
0.36	0		0.26	1		-0.07	2		-0.07	3
0.99	0		-0.35	1		0.00	2		0.00	3



# Positional Encoding

Solution #2

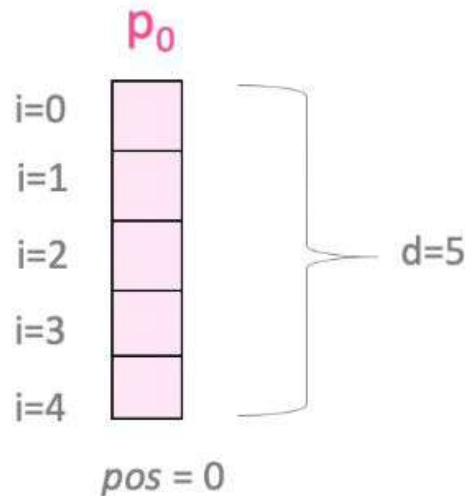


# Positional Encoding

Solution #3

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

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



<https://www.youtube.com/watch?v=dichIcUZfOw>

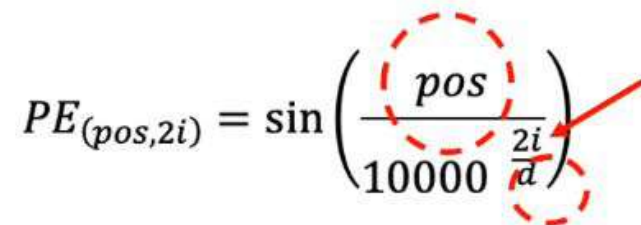
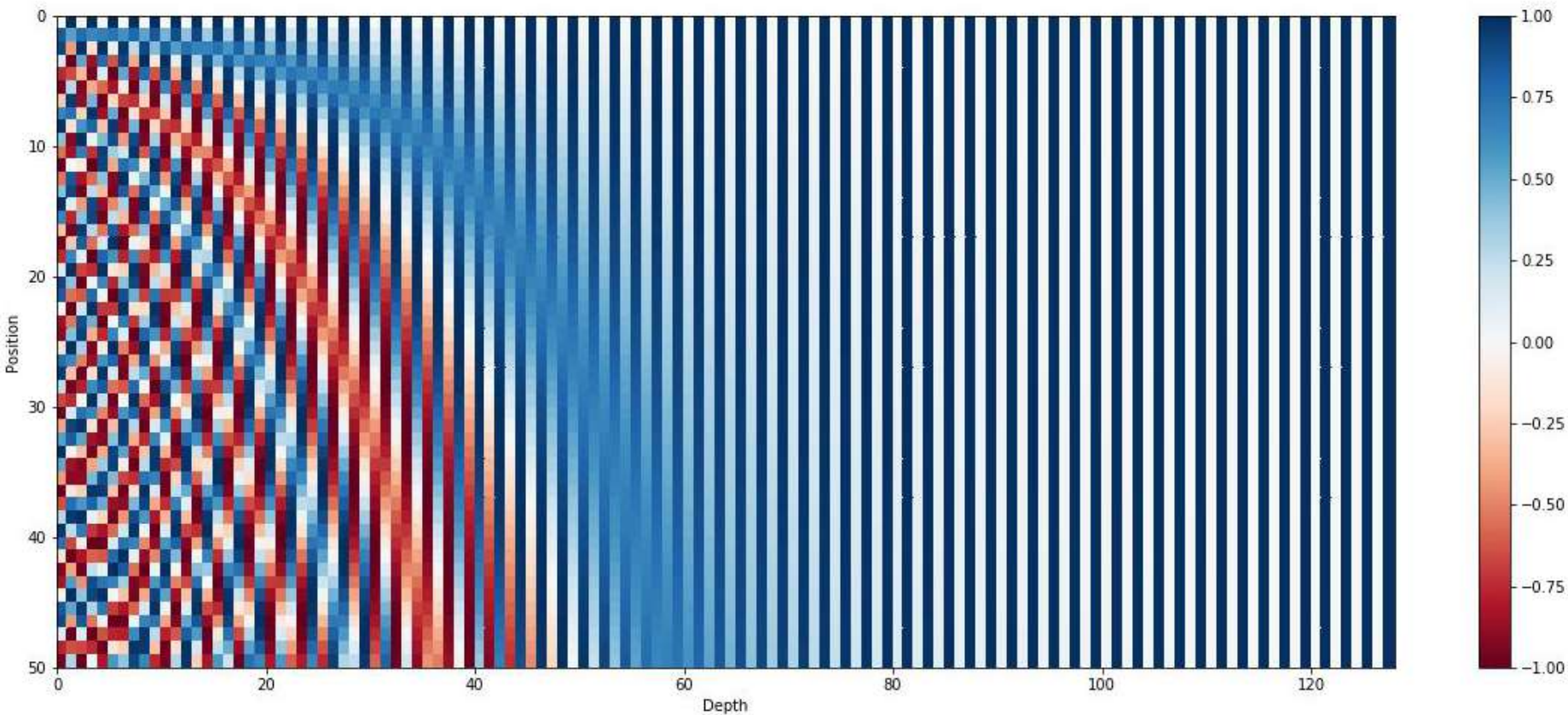
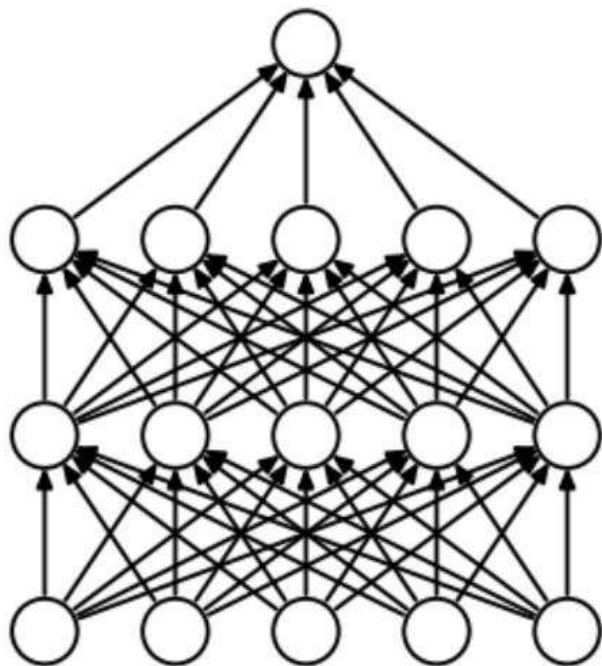
$$PE_{(pos, 2i)} = \sin\left(\frac{pos}{10000^{\frac{2i}{d}}}\right)$$


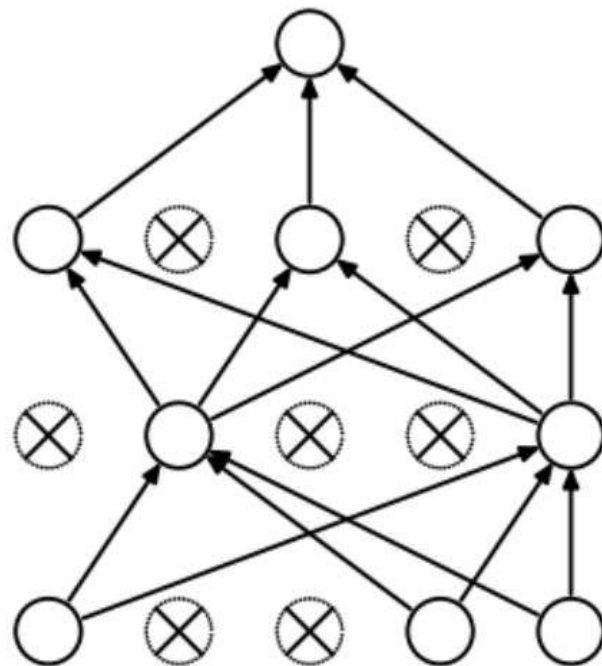
Diagram illustrating the positional encoding formula with annotations. The formula is  $PE_{(pos, 2i)} = \sin\left(\frac{pos}{10000^{\frac{2i}{d}}}\right)$ . Red dashed circles highlight the variables  $pos$  and  $\frac{2i}{d}$ . A red arrow points to the denominator  $10000^{\frac{2i}{d}}$ .



# Dropout

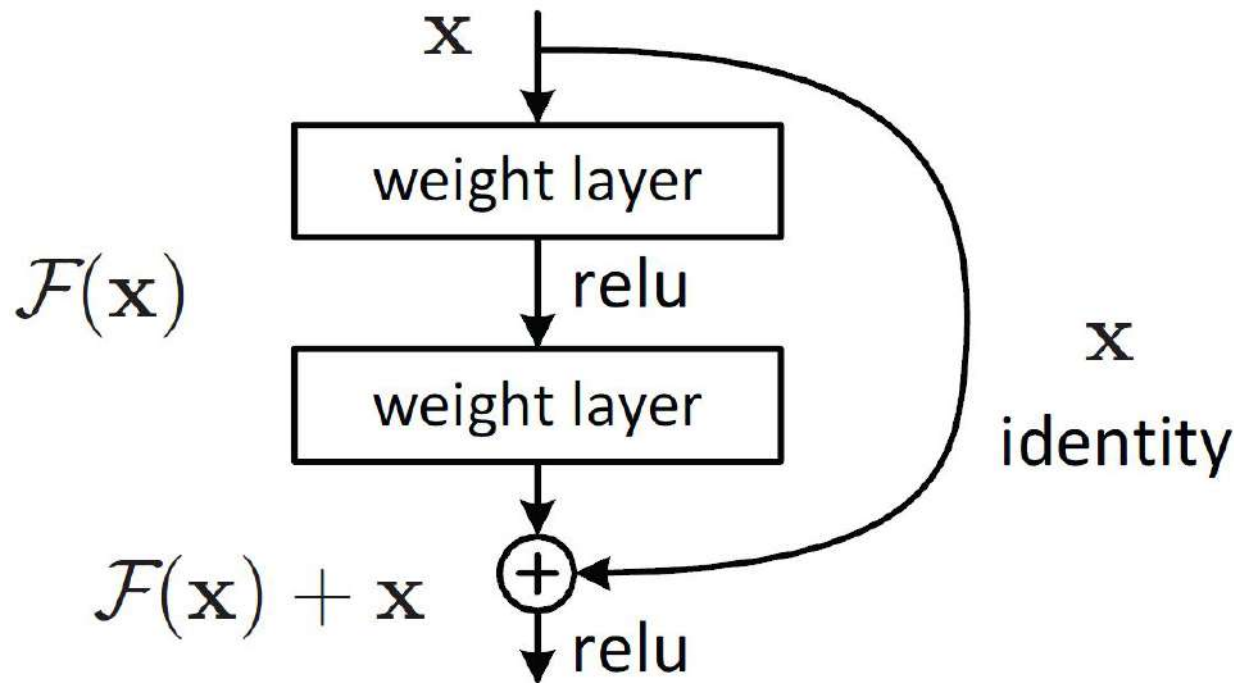


(a) Standard Neural Net



(b) After applying dropout.

# Residual Connection



# Complexity

Table 1: Maximum path lengths, per-layer complexity and minimum number of sequential operations for different layer types.  $n$  is the sequence length,  $d$  is the representation dimension,  $k$  is the kernel size of convolutions and  $r$  the size of the neighborhood in restricted self-attention.

Layer Type	Complexity per Layer	Sequential Operations	Maximum Path Length
Self-Attention	$O(n^2 \cdot d)$	$O(1)$	$O(1)$
Recurrent	$O(n \cdot d^2)$	$O(n)$	$O(n)$
Convolutional	$O(k \cdot n \cdot d^2)$	$O(1)$	$O(\log_k(n))$
Self-Attention (restricted)	$O(r \cdot n \cdot d)$	$O(1)$	$O(n/r)$

# Advantages

parallelizable (thus faster)

computationally less complex (most of the time)

better capture longer dependencies

more interpretable

# Improved Transformers

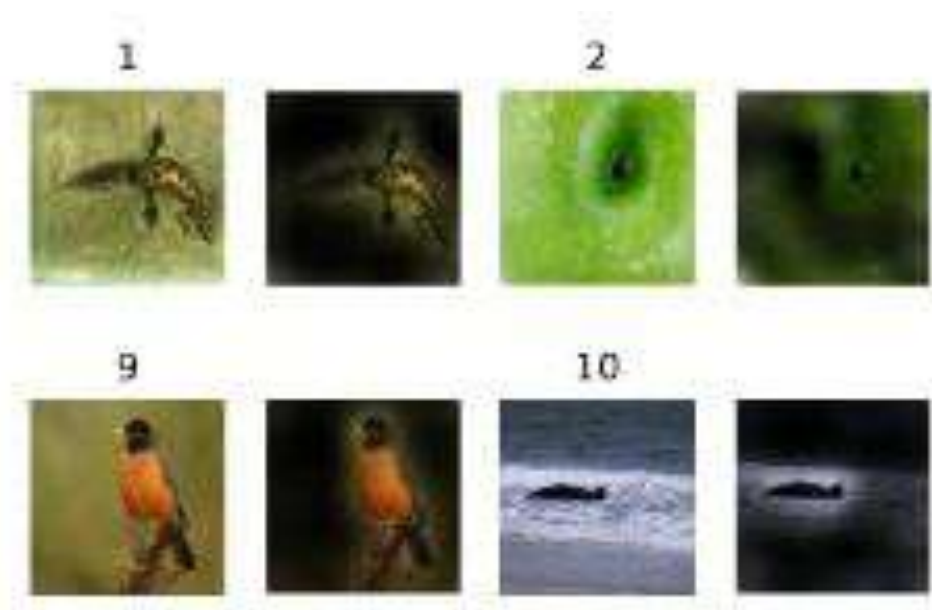
- **BERT**
- GPT
- T5
- BART
- Pegasus
- XLM
- Reformer
- Longformer
- ELECTRA
- RoBERTa
- ...



# Vision Transformers

AN IMAGE IS WORTH 16X16 WORDS: TRANSFORMERS FOR IMAGE RECOGNITION AT SCALE

<https://arxiv.org/pdf/2010.11929.pdf>



# Courses & Resources

- Stanford University NLP w/ DL - <http://web.stanford.edu/class/cs224n/>
- Huggingface - <https://huggingface.co/course/chapter1>
- Deeplearning.ai NLP - <https://www.deeplearning.ai/program/natural-language-processing-specialization/>

# Bibliography

- [https://user.ceng.metu.edu.tr/~skalkan/DL/week\\_13.pdf](https://user.ceng.metu.edu.tr/~skalkan/DL/week_13.pdf)
- [https://kazemnejad.com/blog/transformer\\_architecture\\_positional\\_encoding/](https://kazemnejad.com/blog/transformer_architecture_positional_encoding/)
- <https://www.youtube.com/watch?v=dichIcUZfOw>
- <https://arxiv.org/pdf/1706.03762.pdf>