

Advanced Deep Learning

Large Language Models

K. Breininger, V. Christlein

Artificial Intelligence in Medical Imaging + Pattern Recognition Lab, Friedrich-Alexander-Universität

Erlangen-Nürnberg SoSe 2023

1. Introduction to Large Language Models

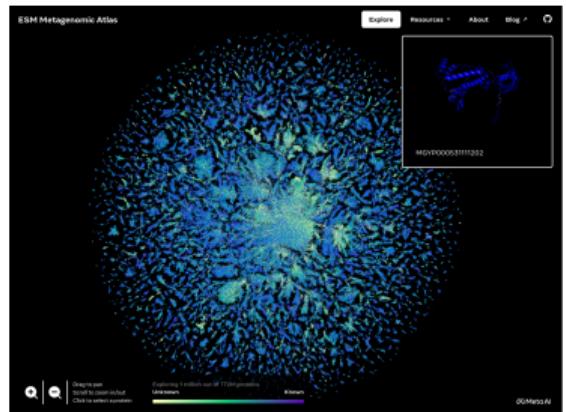
2. Transformers

3. Large Language Models

3.1 Training LLMs

3.2 From Language to Vision

- Text generation (ChatGPT, etc.)
- Language translation, e.g., Translating between 200 languages [1]
- Protein folding, e.g., AlphaFold [2]
- Question answering



ESM Metagenomic Atlas

Source: <https://esmatlas.com/explore?at=1%2C1%2C21.999999344348925>

1. Introduction to Large Language Models

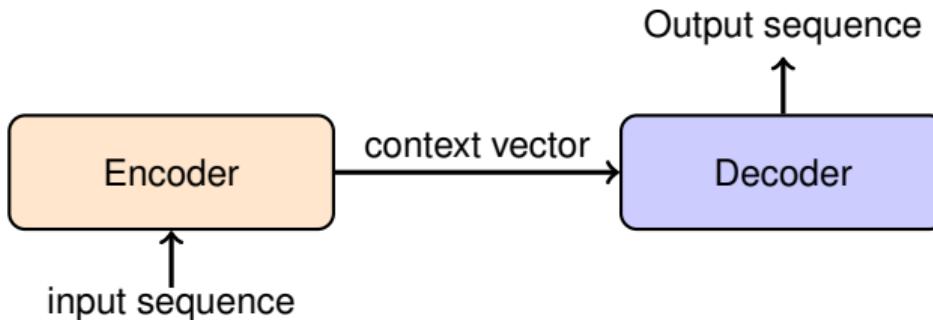
2. Transformers

3. Large Language Models

3.1 Training LLMs

3.2 From Language to Vision

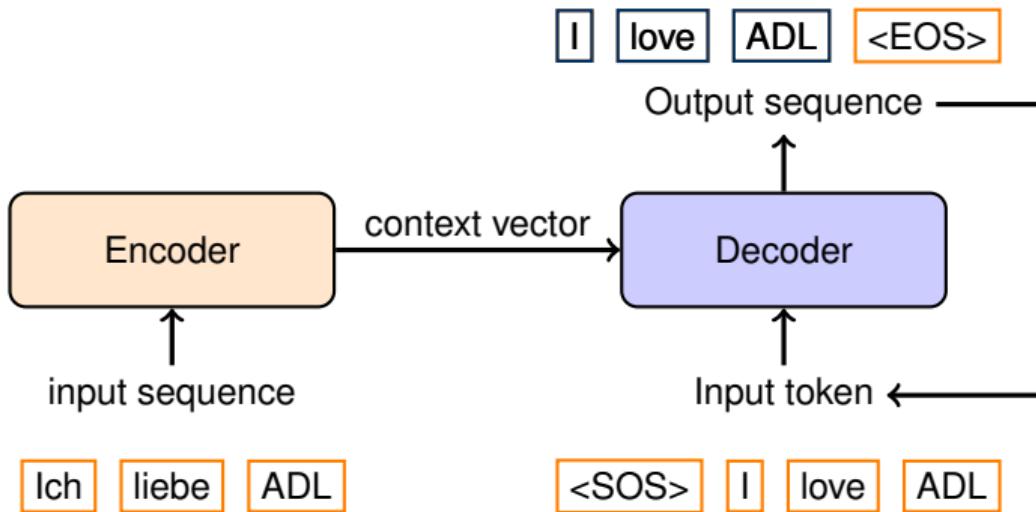
Recap: Sequence to Sequence



- Encoder network
 - Receives input sequence
 - Computes hidden states
- Decoder network:
 - Receives "**context vector**"
 - Computes own hidden states
 - Generates output sequence
- Encoder/Decoder split allows different length in input/output
- Output/input sequence in form of tokens

Recap: Sequence to Sequence (cont.)

Auto-regressive Nature



- Goal: create a vocabulary (unique set of tokens)
- Token-types:
 - Words, e.g., I-love-deep-learning
 - Huge vocabulary sizes needed
 - Out of vocabulary (OOV) words
 - Unknown token (UNK)
 - Word information lost
 - Same representation for every OOV word
 - Characters, e.g., I-l-o-v-e-d-e-e-p-l-e-a-r-n-i-n-g
 - + Limited vocabulary
 - + No OOV problem
 - Increased length of input/output sequences
 - Difficult to learn relationship between the characters to form meaningful words

Solution

Subwords tokenization, e.g., I-love-deep-learn-ing

- ✓ No OOV problem
 - ✓ Length of input and output sequences shorter compared to character tokenization
- How? → Byte Pair Encoding

Byte-pair Encoding

1. Append </w> + split words into characters
2. Initialize vocabulary with unique characters
3. Compute the character pair frequency
4. Merge most frequent pair in corpus
5. Add best pair to the vocabulary
6. Repeat steps 3 to 5 for N iterations

Corpus		
low	lower	newest
low	lower	newest
low	widest	newest
low	widest	newest
low	widest	newest

Corpus		
low </w>	lower </w>	newest </w>
low </w>	lower </w>	newest </w>
low </w>	widest </w>	newest </w>
low </w>	widest </w>	newest </w>
low </w>	widest </w>	newest </w>

Frequency
d-e (3)
l-o (7)
t-</w> (8)

1. Introduction to Large Language Models

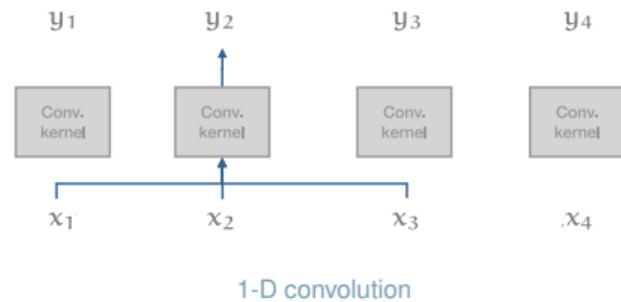
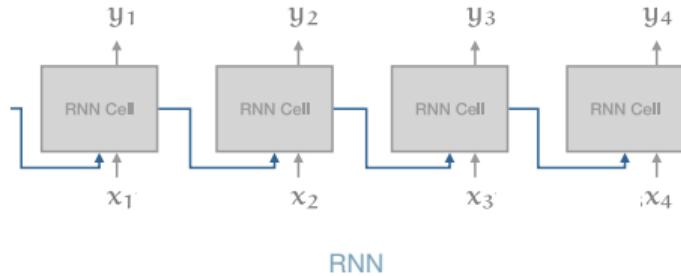
2. Transformers

3. Large Language Models

3.1 Training LLMs

3.2 From Language to Vision

Why do we want it?

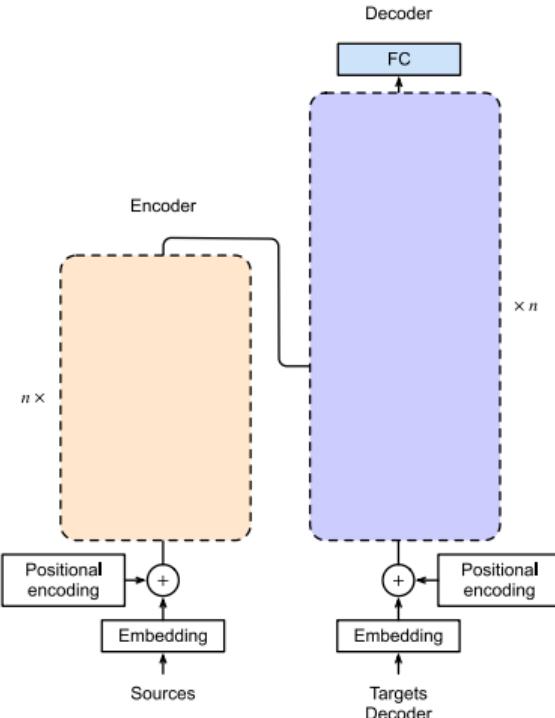


- Recurrent connection
- Allows information propagation along sequence
- Cannot compute cell at time t until having computed cell at $t - 1$

- Every output vector computable in parallel w. every output vector
- Much faster
- Limited in modeling long range dependencies

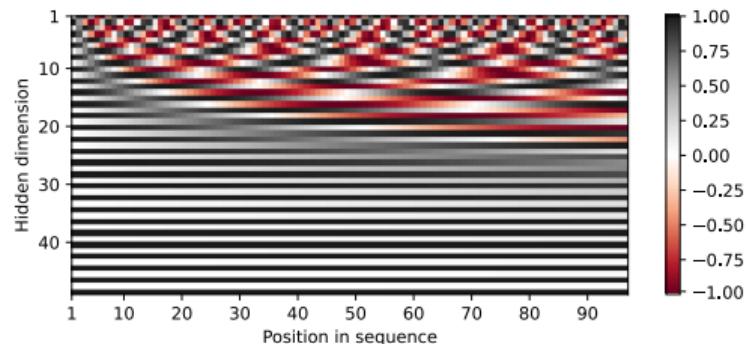
Sources:
<https://peterbloem.nl/blog/transformers>

Beginning

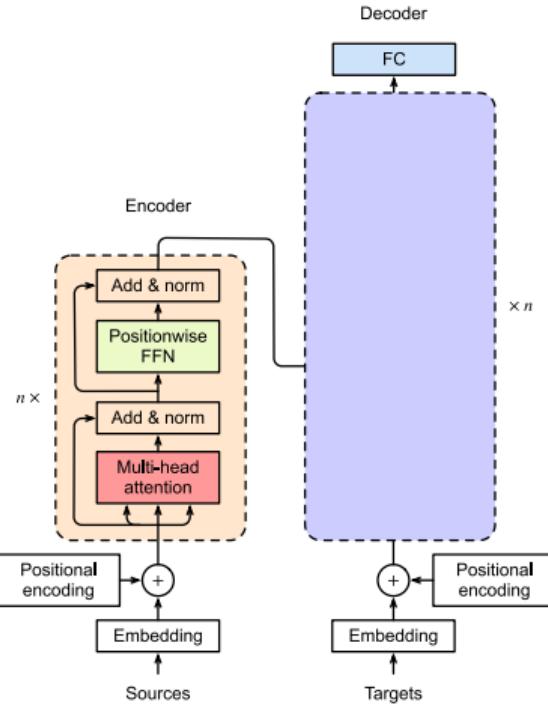


1. Tokens → linear layer → d -dimensional embedding ($d = 512$)
2. Permutation equivariant/invariant layers
→ Add position encoding

$$P_{\text{pos},i} = \begin{cases} \sin\left(\frac{\text{pos}}{10000^{i/d}}\right) & \text{if } i \bmod 2 = 0 \\ \cos\left(\frac{\text{pos}}{10000^{i/d}}\right) & \text{else} \end{cases}$$



Encoder



- Multi-Head Attention
- Normalization + residual connection
- Positionwise FFN

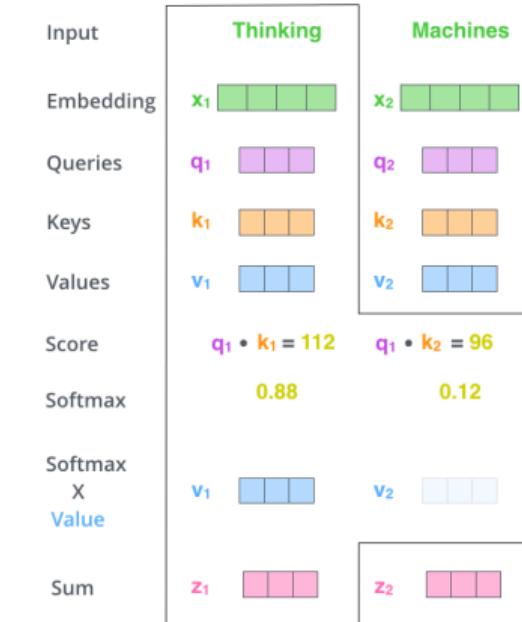
Sources:
https://d2l.ai/chapter_attention-mechanisms-and-transformers/transformer.html (adapted)
https://uvadlc-notebooks.readthedocs.io/en/latest/tutorial_notebooks/tutorial6/Transformers_and_MHAttention.html

Attention

- Each input token is translated into vector of same length using an **embedding** algorithm
- Self attention: For each token, compute
 - query **q**: what a token is "looking for"
 - key **k**: "description" for query
 - value **v**: potential "information"
 using trainable weights \mathbf{W}^q , \mathbf{W}^k , \mathbf{W}^v
- Alignment between query and key (scaled dot product) determines influence of elements in **v**:

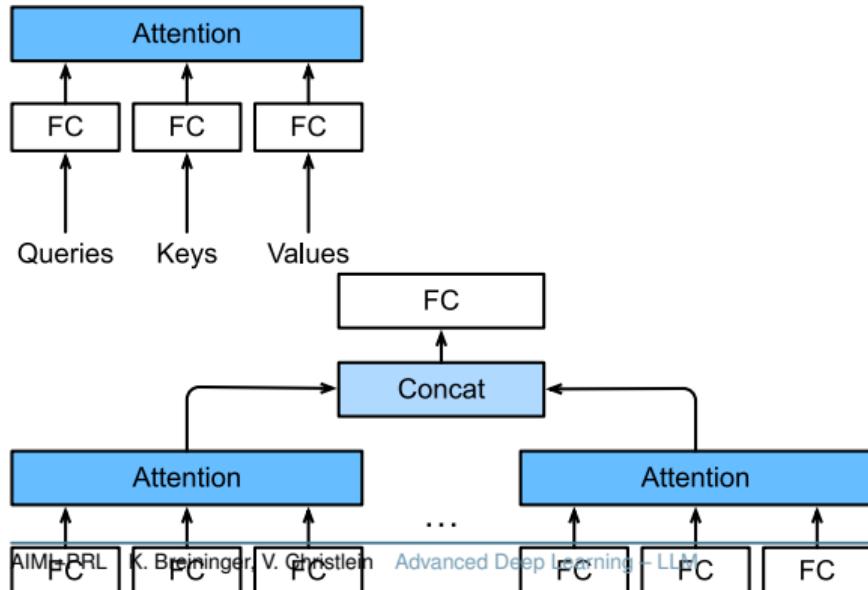
$$A(\mathbf{q}, \mathbf{k}, \mathbf{v}) = \text{softmax}\left(\frac{\mathbf{q}\mathbf{k}^T}{\sqrt{d_k}}\right)\mathbf{v}$$

- d_k : dimension of **q**, **k**, **v**



Source: <http://jalammar.github.io/illustrated-transformer/>

Attention – In Practice



- Stack N input embeddings: $\mathbf{X} \in \mathbb{R}^{N \times d}$
- $\mathbf{Q} = \mathbf{XW}^q | \mathbf{K} = \mathbf{XW}^k | \mathbf{V} = \mathbf{XW}^v | \{\mathbf{W}^q, \mathbf{W}^k, \mathbf{W}^v\} \in \mathbb{R}^{d \times d}$
- Attention-head \mathbf{H} :

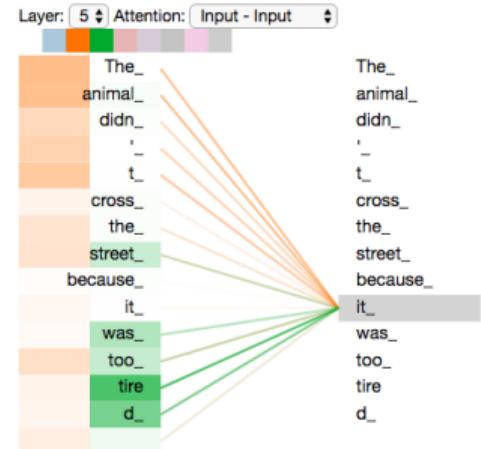
$$\mathbf{H} = \text{softmax}\left(\frac{\mathbf{QK}^T}{\sqrt{d_k}}\right)\mathbf{V} \in \mathbb{R}^{N \times d}$$

- $\mathbf{Q}_i = \mathbf{XW}_i^q | \mathbf{K}_i = \mathbf{XW}_i^k | \mathbf{V}_i = \mathbf{XW}_i^v | \{\mathbf{W}_i^q, \mathbf{W}_i^k, \mathbf{W}_i^v\} \in \mathbb{R}^{d \times d/h}$
- Attention-head \mathbf{H}_i :

$$\mathbf{H}_i = \text{softmax}\left(\frac{\mathbf{Q}_i\mathbf{K}_i^T}{\sqrt{d_k}}\right)\mathbf{V}_i \in \mathbb{R}^{N \times d/h}$$

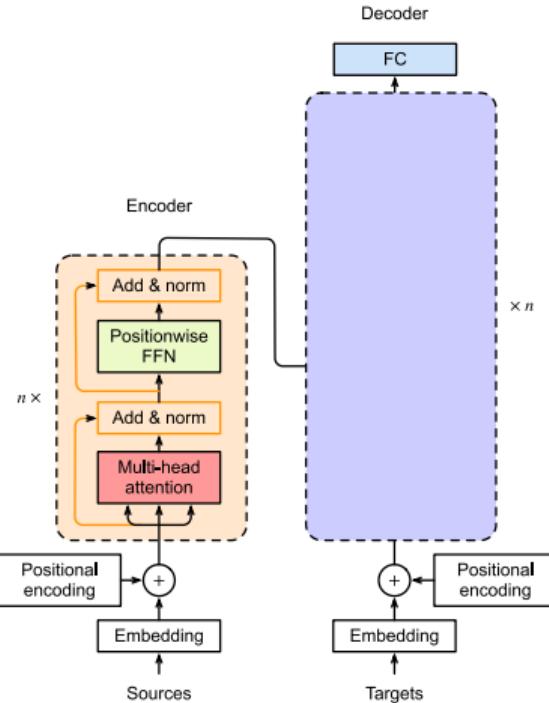
Attention – Advantages

- Provides some intepretability
- Inspecting attention distribution
- Significantly improves Neural Machine Translation performance
- Allows decoder to focus on certain parts of the source
- Helps w. vanishing gradients problem
- Provides shortcut between faraway states



Source: <https://jalammar.github.io/illustrated-transformer/>

Layer Normalization + Residual Connection

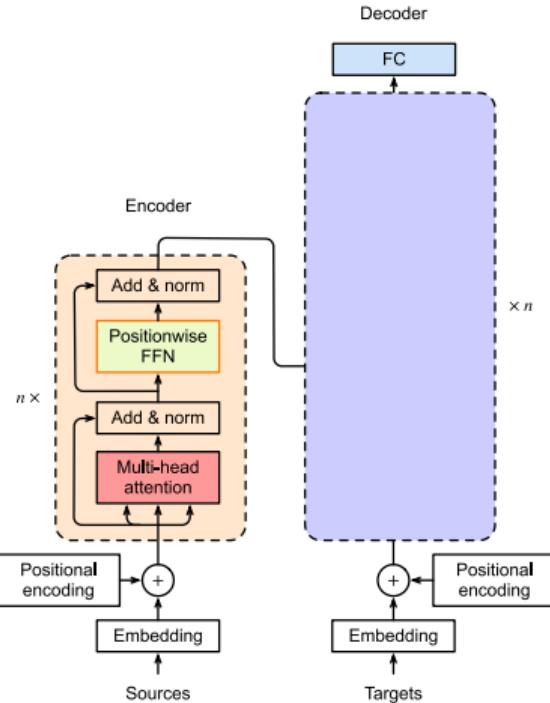


- Residual connection improves trainability
- Layer normalization (LN) [3] over embedding dimension
- Post-norm (original):
$$\text{LN}(\mathbf{X} + \mathbf{M}(\mathbf{X}))$$
- Pre-norm [4]:
$$\mathbf{X} + \mathbf{M}(\text{LN}(\mathbf{X}))$$

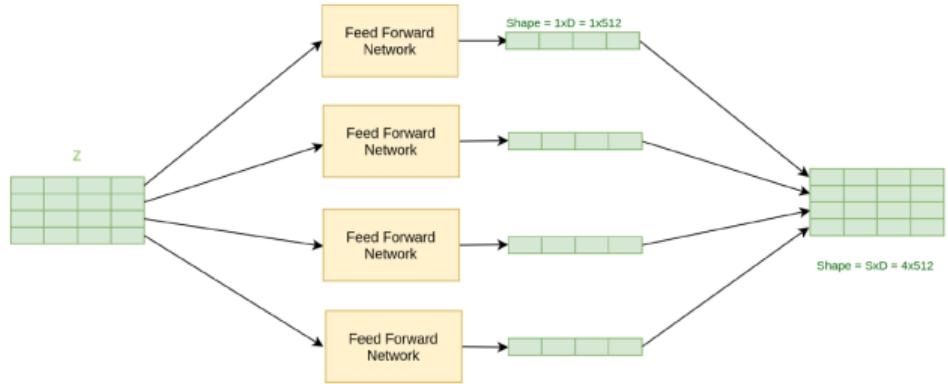
Sources:

https://d2l.ai/chapter_attention-mechanisms-and-transformers/transformer.html (adapted)

Positionwise Feed Forward



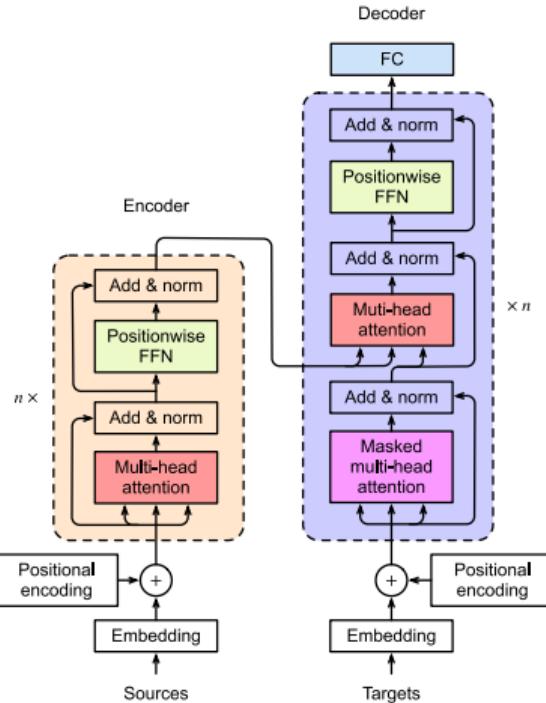
- Applied to each position separately and identically



- $\text{FFN}(\mathbf{X}) = \text{FC}_2(\text{ReLU}(\text{FC}_1(\mathbf{X}))$
 FC_1 : 2048 nodes, FC_2 : 512 nodes

Sources:
https://d2l.ai/chapter_attention-mechanisms-and-transformers/transformer.html (adapted)
<https://mlwhiz.com/blog/2020/09/20/transformers/>

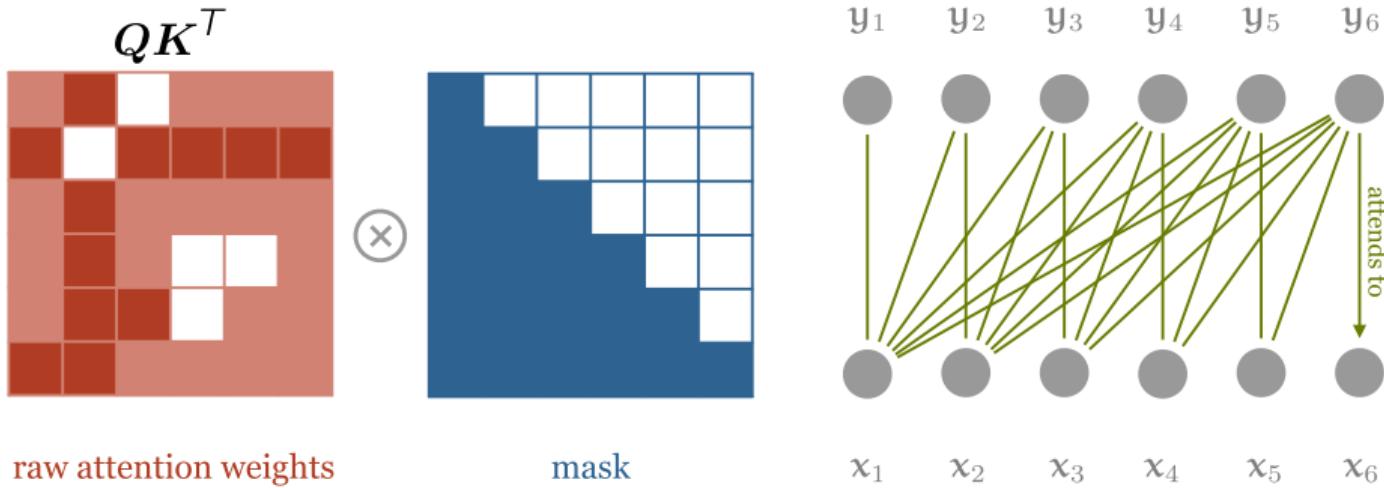
Decoder



- Masked self-attention
- Cross-attention

Sources:
https://d2l.ai/chapter_attention-mechanisms-and-transformers/transformer.html (adapted)

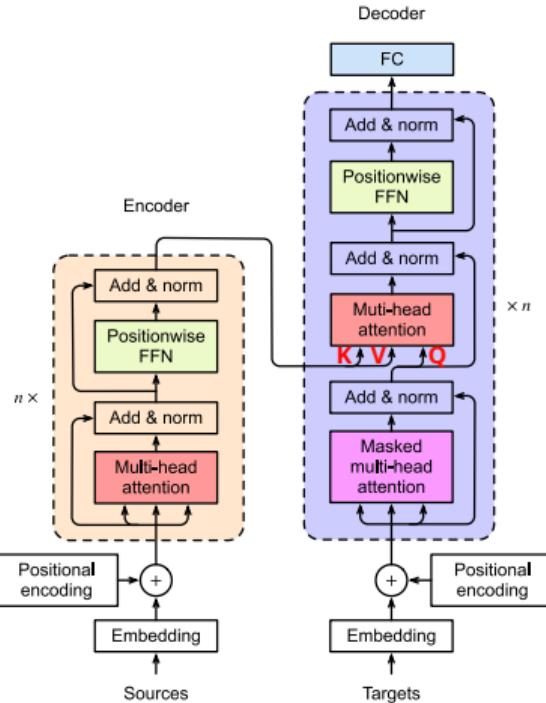
Masked Self-Attention



- Elements can only attend to elements that **precede** them in the sequence
- Note that the actually the masked out elements (the white squares) are set to $-\infty$
- Also needed in the encoder (+decoder) in case of different input lengths!

Source: <https://peterbloem.nl/blog/transformers>

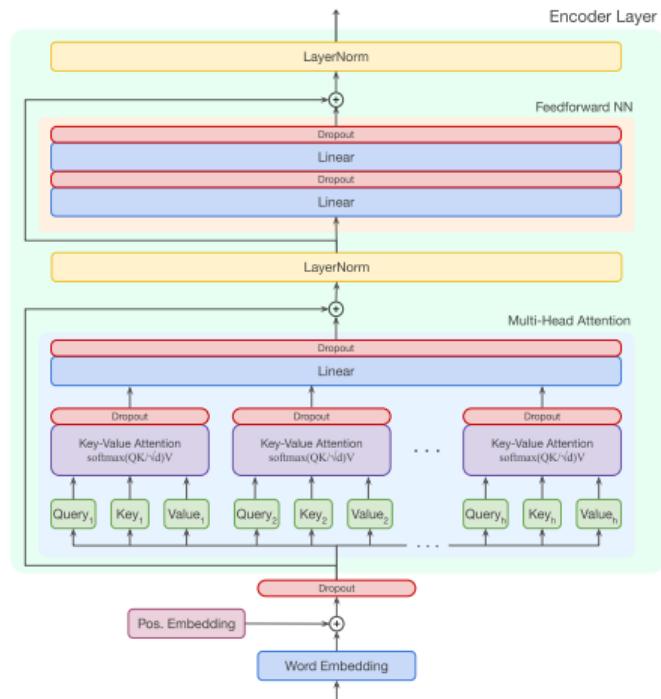
Cross-Attention



- Cross-attention:
 - Source 1: **Q**
 - Source 2: **K** and **V**

Sources:
https://d2l.ai/chapter_attention-mechanisms-and-transformers/transformer.html (adapted)

Addendum: Dropout



Dropout ($p = 0.1$):

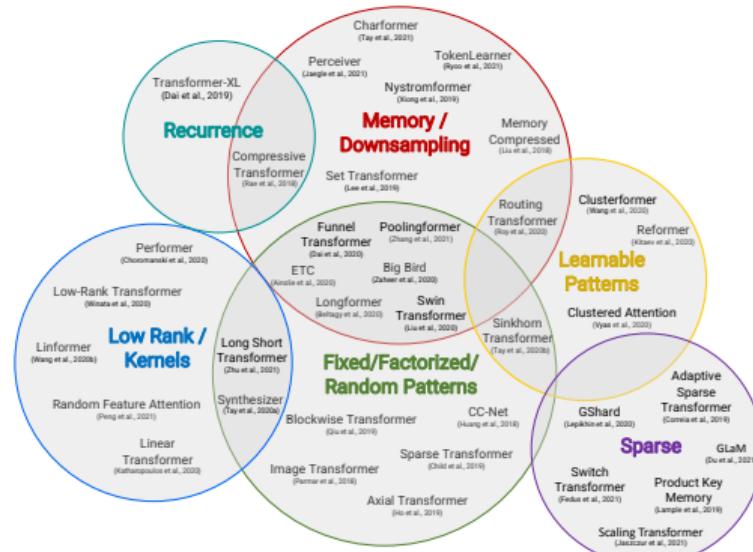
- After positional encoding
- After attention blocks
- Before normalization
- In-between positionwise FFN

Source: <https://aniruddhadeb.com/articles/2023/properly-illustrated-transformer.html> (adapted)

- More powerful representation than CNNs
- **But** typically a huge amount of training data necessary!
- Slow inference due to auto-regressive nature
- Non-auto-regressive versions exist [5], [6] (but rather uncommon)
 - Attention mechanism: $\mathcal{O}(n^2)$ computational and memory complexity

Efficient Transformers

- Many “efficient” transformers exist
- No adoption in LLMs so far!
- Possible reasons:¹
 - Potential speed-up only useful for large input sequences
 - New methods introduce additional constraints → less universal
 - Reported efficiency measures don’t translate in actual computational cost and time savings



¹<https://www.stateof.ai/>

1. Introduction to Large Language Models

2. Transformers

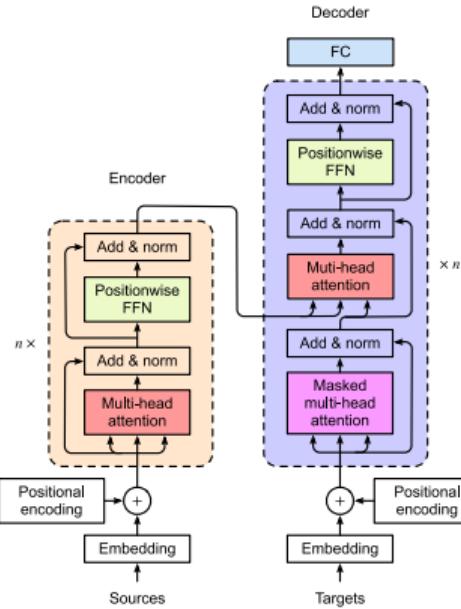
3. Large Language Models

3.1 Training LLMs

3.2 From Language to Vision

BERT [8]

- Encoder-only
- Goals:
 - Classification (e.g., sentiment)
 - Questions and answering
 - Summarization
 - Named entity recognition
- Out: Word embeddings



GPT [9]

- Decoder-only
- Goals:
 - Translation
 - Generation
- Out: Next words w. probabilities

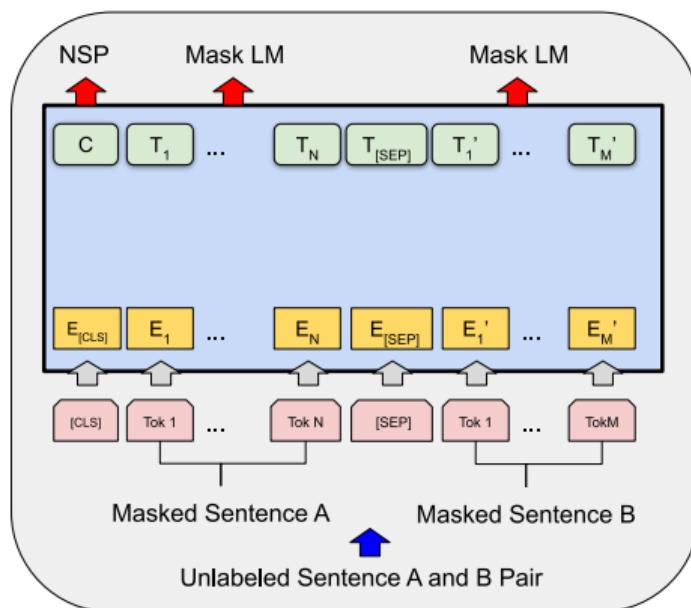
Training LLMs

- Standard cross-entropy
- Efficient training possible:
 - Teacher forcing: instead of decoder output use ground truth output
 - Hidden states of all time steps computed simultaneously in the attention heads
- Label smoothing [10] ($\eta = 0.1$)
- ADAM w. learning rate warmup

Pre-training

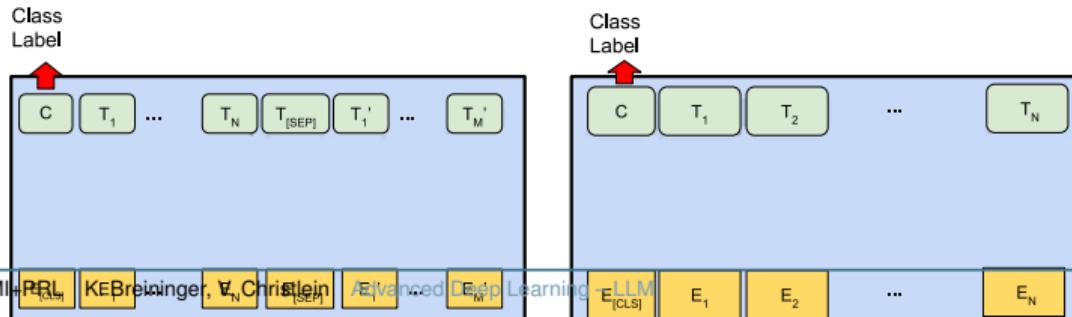
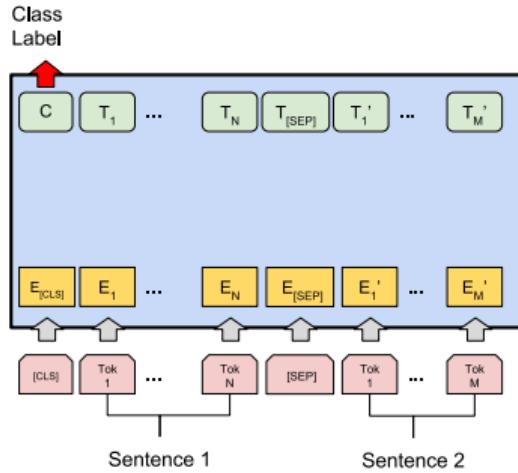
Bidirectional Transformers (BERT)

- Masking:
 - Mask some words out
For each word: randomly replace/keep it
 - Goal: predict the masked words
- Next sequence classification:
 - Sample two 256 word sequences from
 - (1) random places or
 - (2) following each other
 - Goal: predict (1) or (2)



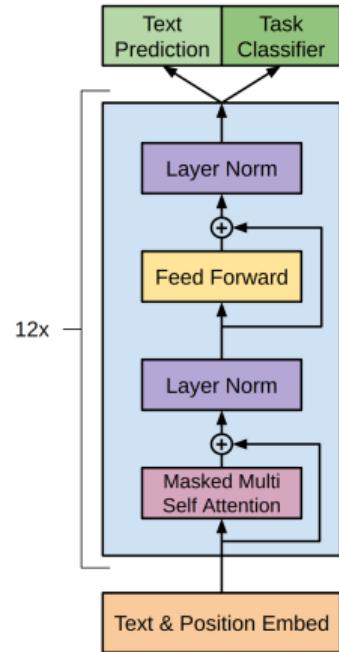
Source: Devlin et al. 2019 [8] (adapted)

Fine-tuning

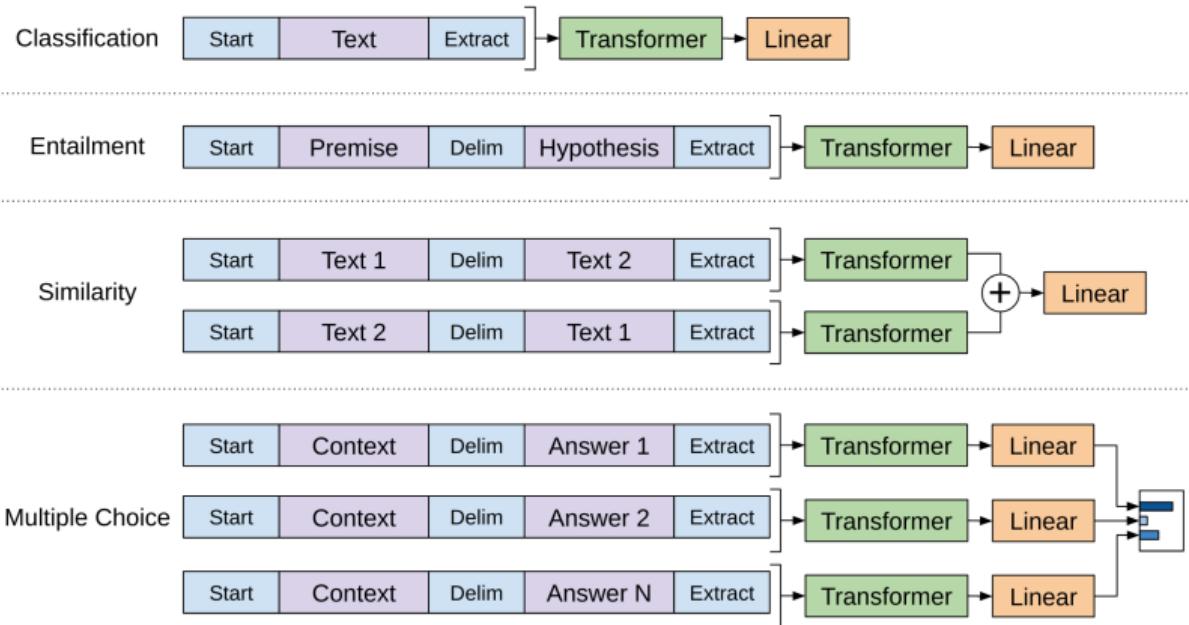


Generative Pre-trained Transformer (GPT)

- Auto-regressive model
- Use masked multi-head attention
- Goal: predict next words



Source: Radford et al. 2018 [9]



Source: Radford et al. 2018 [9]

GPT-2 [4]

- Pure decoder (no additional layers)
- 10x larger than GPT-1 (48 transformer blocks)
- 10x more data for training
- Pre-norm LayerNorm
- Additional LayerNorm after final attention block
- Modified initialization: scale weights of residual layers by factor $1/\sqrt{N}$ where N : number of residual layers
- Context size: 1024 (instead of 512)

GPT-3 [11]

- 100x larger than GPT-2
- 500 billion word dataset (crawled)
- Trend continues with GPT-4

	GPT	GPT-2	GPT-3
Parameters	117 Million	1.5 Billion	175 Billion
Decoder Layers	12	48	96
Context Token Size	512	1024	2048
Hidden Layer	768	1600	12288
Batch Size	64	512	3.2M

GPT 4 [12]

A lot of benchmarks, but...

???

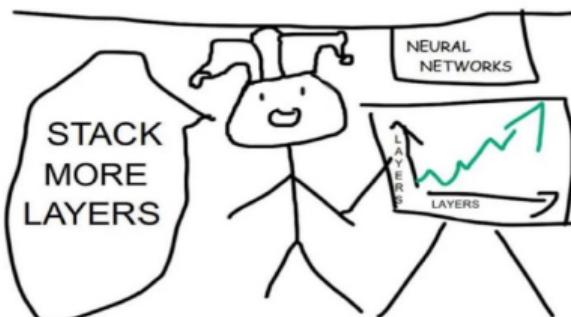
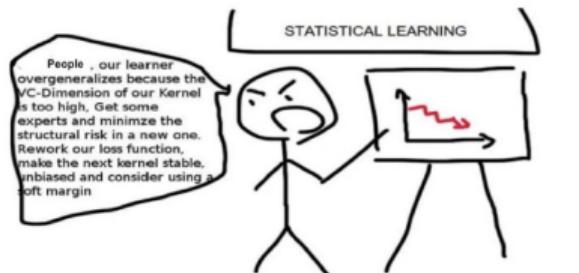
but now also with memes!

“We were wrong. Flat out, we were wrong.”

Ilya Sutskever, OpenAI chief scientist & co-founder
(about sharing too much information open source)

User

Can you explain why this is funny. Think about it step-by-step.



GPT-4

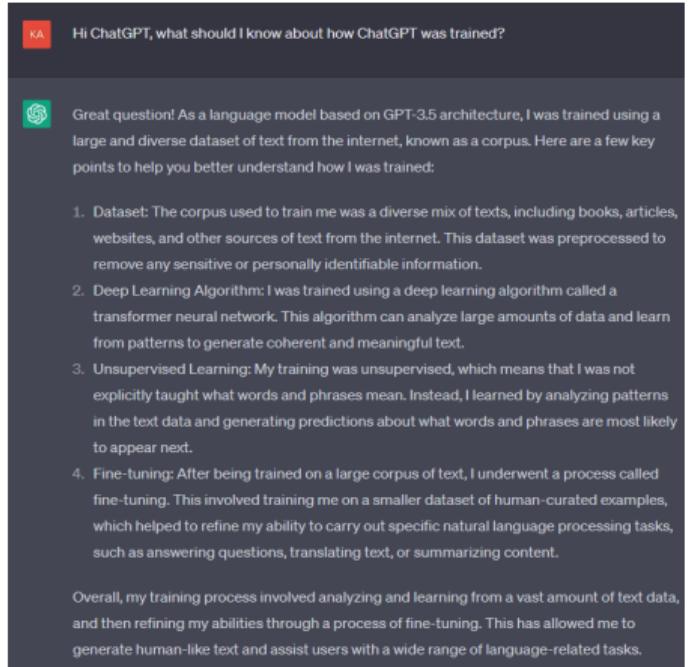
The comic is satirizing the difference in approaches to improving model performance between statistical learning and neural networks.

In statistical learning, the character is shown to be concerned with overfitting and suggests a series of complex and technical solutions, such as minimizing structural risk, reworking the loss function, and using a soft margin.

In contrast, the neural networks character simply suggests adding more layers to the model. This is often seen as a common solution to improving performance in neural networks, but it's also considered a simplistic and brute-force approach.

Source: Adapted from [12]

- Fine-tuned GPT-3.5 (and GPT-4 for ChatGPT Plus)
- Uses **reinforcement learning by human feedback** (RLHF)
- Adversarial training to avoid jailbreaking

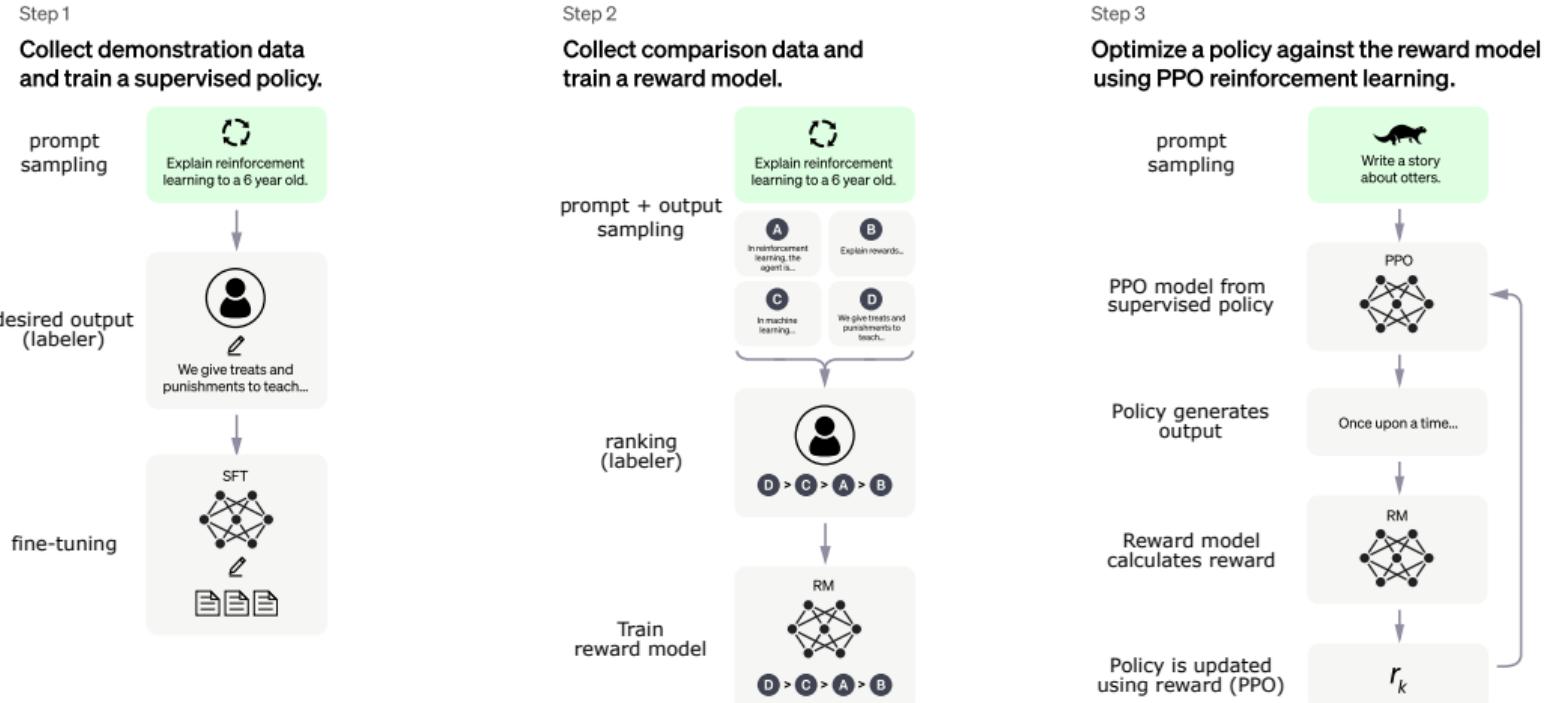


KA Hi ChatGPT, what should I know about how ChatGPT was trained?

Great question! As a language model based on GPT-3.5 architecture, I was trained using a large and diverse dataset of text from the internet, known as a corpus. Here are a few key points to help you better understand how I was trained:

1. Dataset: The corpus used to train me was a diverse mix of texts, including books, articles, websites, and other sources of text from the internet. This dataset was preprocessed to remove any sensitive or personally identifiable information.
2. Deep Learning Algorithm: I was trained using a deep learning algorithm called a transformer neural network. This algorithm can analyze large amounts of data and learn from patterns to generate coherent and meaningful text.
3. Unsupervised Learning: My training was unsupervised, which means that I was not explicitly taught what words and phrases mean. Instead, I learned by analyzing patterns in the text data and generating predictions about what words and phrases are most likely to appear next.
4. Fine-tuning: After being trained on a large corpus of text, I underwent a process called fine-tuning. This involved training me on a smaller dataset of human-curated examples, which helped to refine my ability to carry out specific natural language processing tasks, such as answering questions, translating text, or summarizing content.

Overall, my training process involved analyzing and learning from a vast amount of text data, and then refining my abilities through a process of fine-tuning. This has allowed me to generate human-like text and assist users with a wide range of language-related tasks.



Source: Adapted from [13].

Sure...

KA

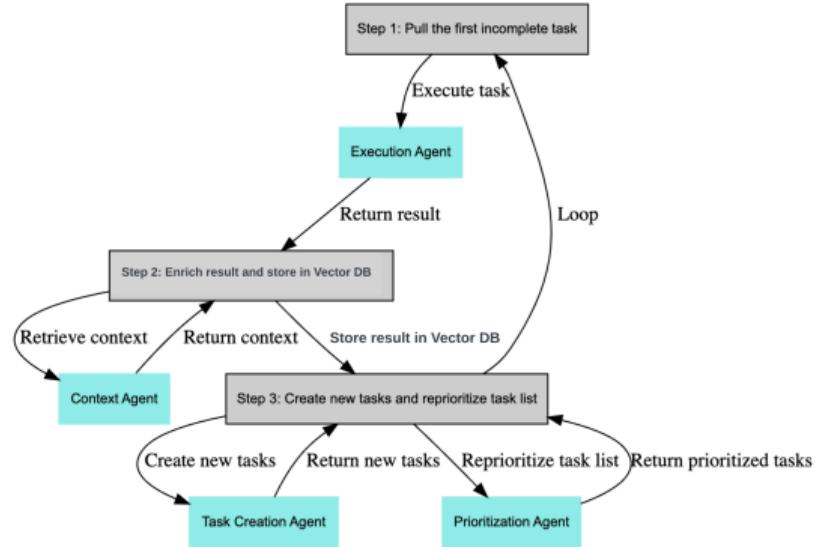
What is the most interesting development in machine learning right now - from your perspective?



As an AI language model, I do not have personal perspectives or opinions. However, one of the most interesting developments in machine learning right now is the field of GPT-3-based natural language processing, where models are trained on massive amounts of data and can generate human-like language and responses to complex prompts, with potential applications in language translation, content creation, and conversational AI.

AutoGPT,^a BabyAGI,^b AgentGPT^c

- Chains together LLM outputs to achieve goal
 - Creates task lists and processes them subsequently
 - Basis of many other apps/tools
 - Try (variations): godmode.space, agentgpt.reworkd.ai/, beta.nando.ai/goalgpt.php
- ! Warning: still based on LLM! Halluzinations!



^a<https://github.com/Significant-Gravitas/Auto-GPT> (MIT)

^b<https://github.com/yoheinakajima/babyagi> (MIT)

^c<https://github.com/reworkd/AgentGPT> (GPL v3)

Source: <https://github.com/yoheinakajima/babyagi>

- Fun: Bot for ordering pizza:
<https://twitter.com/rogerhamilton/status/1646557324188712966?s=20>
- poe.com – Try different LLMs (ChatGPT, Sage, ...), now also enables agents
- durable.co/uncody.com – Build a website in 30 sec
- aomni.com – Agent for research (AutoGPT-based)
- humanornot.ai – Turing test chat-game where you have to decide if it's a machine or not you are talking to
- docu-ask.com/docliffe.com – Upload document and ask questions about it
- magicslides.app – Creates google slides just from some text
- tldv.io – Meeting recorder that transcribes and summarizes calls
- Not enough? Have a look here: supertools.therundown.ai or here: topapps.ai

- Bard AI (Google, uses LaMDA)
- Perplexity AI (cites sources)
- Chatsonic, builds on ChatGPT but can access internet
- Colossal Chat (based on LLaMA model)
- ...

Open Source Models:

- LLaMA model (open source, GPL v.3)
- StableVicuna (open source, instruction fine-tuned LLAMA, CC-BY-NC-SA)
- Dolly (open source, Apache v.2)

Problem: difficult to use LLMs efficiently and effectively

Possible solutions

1. Fine-tune distilled small models, e.g., DistilBERT [14]
2. Prompt modifications:
 - In-context learning
 - Hard prompt tuning
 - Soft prompt tuning/Prefix-tuning
3. Adapter methods, e.g., LLaMA Adapter [15], LLM-Adapters [16]
4. Reparametrization, e.g., LoRA [17], DyLoRA [18]

Give examples



```
1 Translate the following German sentences into English:  
2  
3 Example 1:  
4 German: "Ich liebe Eis."  
5 English: "I love ice cream."  
6  
7 Example 2:  
8 German: "Draußen ist es stürmisch und regnerisch"  
9 English: "It's stormy and rainy outside."  
10  
11 Translate this sentence:  
12 German: "Wo ist die naechste Supermarkt?"
```

Change discrete input tokens



```
1) "Translate the English sentence '{english_sentence}' into German: {german_translation}"  
2) "English: '{english_sentence}' | German: {german_translation}"  
3) "From English to German: '{english_sentence}' -> {german_translation}"
```

- Concat embeddings of the input tokens with trainable tensor [19] or full

From Language to Vision

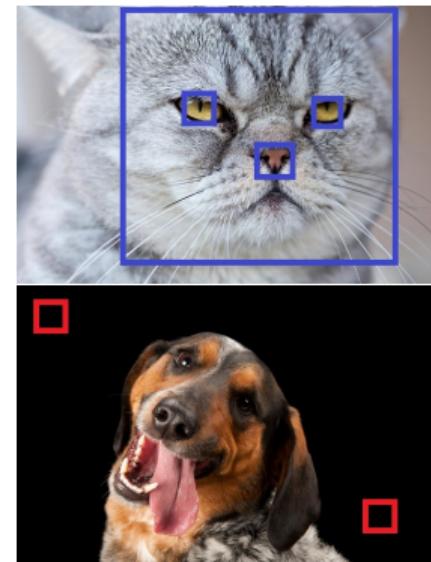
CNNs incorporate **inductive bias**

- Hierarchical organization
- Local connectivity
- Translational equivariance
- Reduces what the network can represent
- Receptive field strongly linked to network depth

Question: **Can we get rid of this restriction?**

Answer: **Yes***

*Terms and conditions apply



Vision Transformer (ViT)

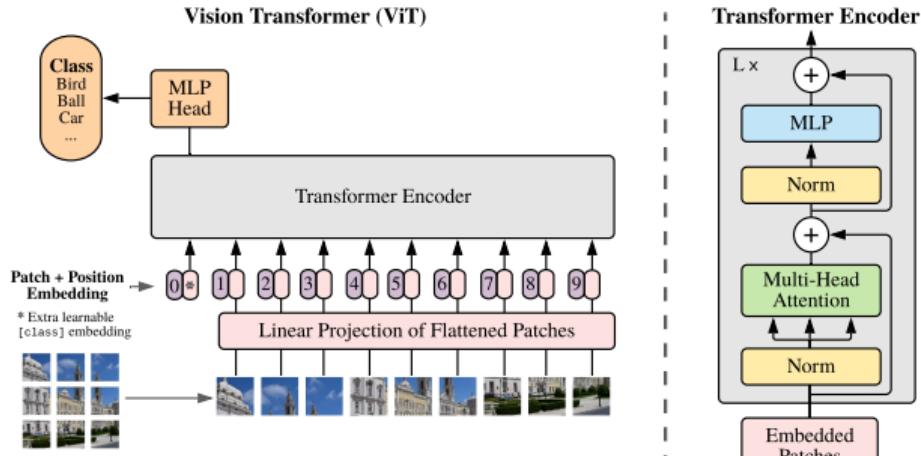
“An image is worth 16×16 words” [21]

Core idea: Images are also just “sequences”

- Separate images into **patches**
- Transform patches to **tokens**
- Encode patch-tokens using Transformer

Main parameters:

- Size of input patches:
16×16 input patches - ViT-X/16
- Transformer parameters:
Layers, hidden size, MLP size, heads ...
→ViT-Base (86M)
→ViT-Large (307M)
→ViT-Huge (632M)

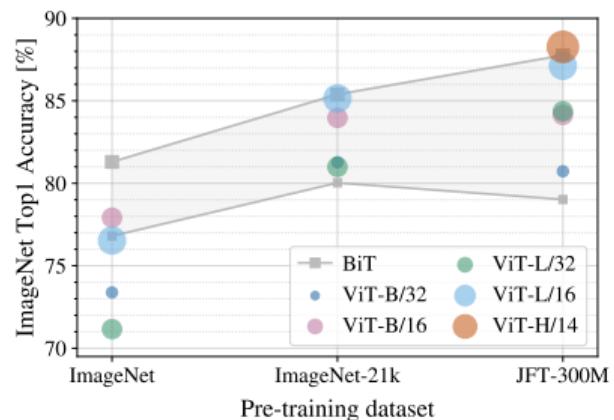


Source: Adapted from [21]

“An image is worth 16×16 words” [21]

Core insight: It works!

- SOTA for various image recognition benchmarks ...
- ... when pre-training on large-scale (!) datasets (JFT-300M)
- More efficient pre-training compared to (large) CNNs



	Ours-JFT (ViT-H/14)	BiT-L (ResNet152x4)	Noisy Student (EfficientNet-L2)
ImageNet	88.55 \pm 0.04	87.54 ± 0.02	88.4
ImageNet ReaL	90.72 \pm 0.05	90.54	90.55
CIFAR-10	99.50 \pm 0.06	99.37 ± 0.06	—
CIFAR-100	94.55 \pm 0.04	93.51 ± 0.08	—
Oxford-IIIT Pets	97.56 \pm 0.03	96.62 ± 0.23	—
Oxford Flowers-102	99.68 \pm 0.02	99.63 ± 0.03	—
VTAB (19 tasks)	77.63 \pm 0.23	76.29 ± 1.70	—
TPUv3-core-days	2.5k	9.9k	12.3k

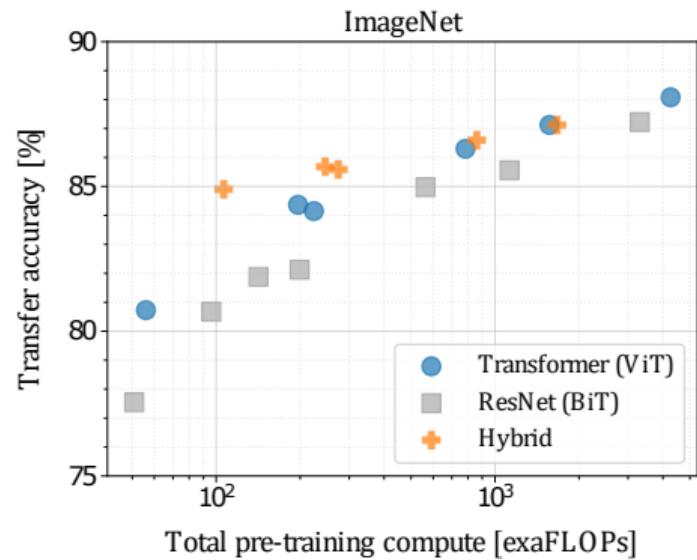
Excerpt of Tab. 2 from [21]

Source: Adapted from [21]

“An image is worth 16×16 words” [21]

Additional insights:

- Hybrid models possible, boost performance for smaller data regimes
- Position-embeddings can be successfully learned
- Efficient implementations (LLMs) can be re-used

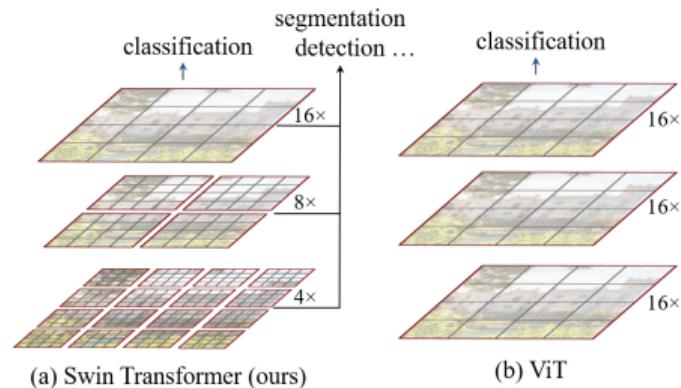


Source: Adapted from [21].

Beyond re-using Transformers

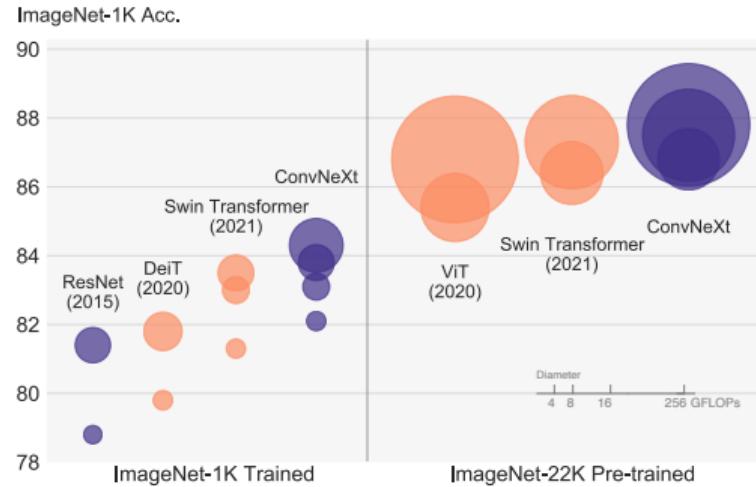
Swin-Transformers [23]

- Vanilla ViTs use uniform “resolution”
→ Can give a lot of freedom but data hungry
- Introduce **inductive bias** again:
Hierarchical representation
- Hierarchy allows tasks at multiple scales
 - Classification
 - Object detection
 - Segmentation
- Similar concepts: SegFormer [22]



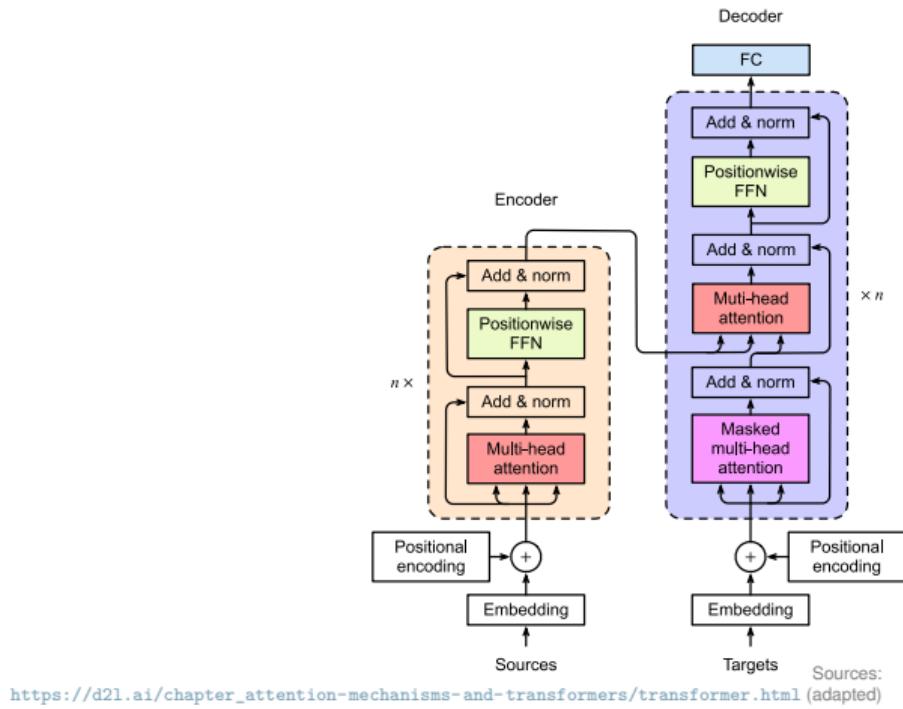
Beyond Transformers?

- ConvNets vs. Transformers: Whose Visual Representations are More Transferable? [24]
- ConvNeXt: A Convnet for the 2020s [25]
- Designing BERT for Convolutional Networks [26]



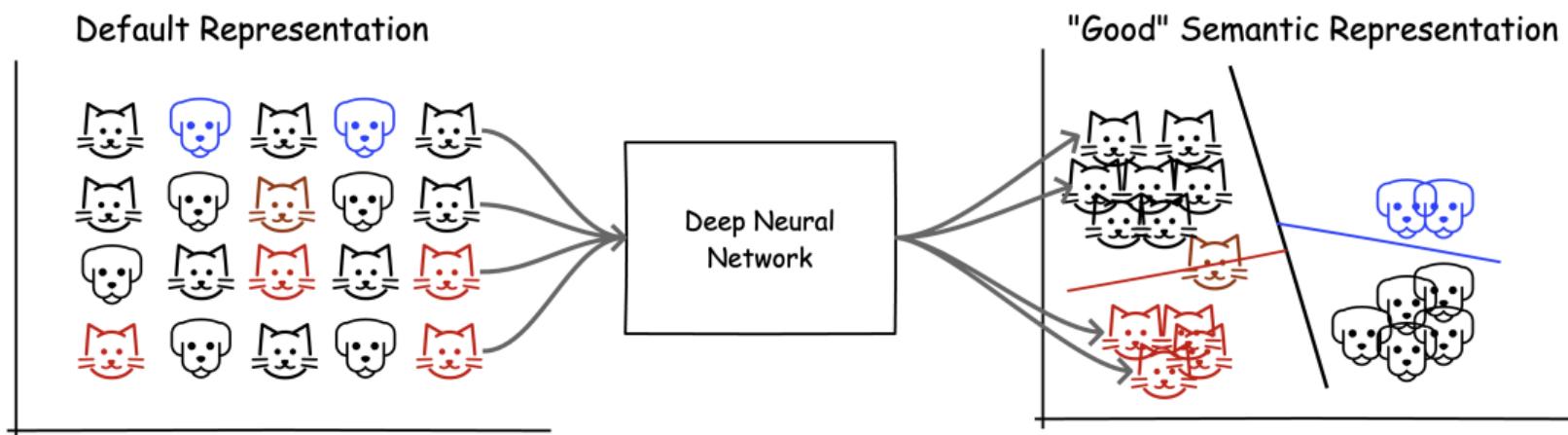
Source: Adapted from [25]

- Sequence to sequence
- Transformer architecture
 - Attention
 - Positional encoding
 - Masking
- BERT vs GPT
- Vision Transformer



NEXT TIME
ADVANCED
ON\DEEP LEARNING

Next: Representation Learning



Source: <https://blog.fastforwardlabs.com/2020/11/15/representation-learning-101-for-software-engineers.html>

- What is a sequence to sequence model?
- How must a text input be processed?
- What is self-attention?
- Why do we need positional encoding?
- What's the role of masking?
- What's the difference between BERT and GPT?
- How to train LLMs?
- How to use Transformers in Vision?

- Tokenization:
<https://www.analyticsvidhya.com/blog/2020/05/what-is-tokenization-nlp/>
- Jay Alammar: The Illustrated Transformer,
<http://jalammar.github.io/illustrated-transformer/>
- Lilian Weng: The Transformer Family,
<https://lilianweng.github.io/posts/2020-04-07-the-transformer-family/>
- Transformer models: an introduction and catalog – 2023 Edition, <https://amatriain.net/blog/transformer-models-an-introduction-and-catalog-2d1e9039f376/>
- Sebastian Raschka: DL course, <https://sebastianraschka.com/blog/2021/dl-course.html#l19-self-attention-and-transformer-networks>
- Lucas Beyers: Transformers, <https://www.youtube.com/watch?v=UpfcyzoZ644>
- Jurafsky and Martin: “Speech and Language Processing”,
<https://web.stanford.edu/~jurafsky/slp3/>

- https://d2l.ai/chapter_attention-mechanisms-and-transformers/transformer.html
- https://uvadlc-notebooks.readthedocs.io/en/latest/tutorial_notebooks/tutorial6/Transformers_and_MHAttention.html
- The annotated transformer: <http://nlp.seas.harvard.edu/annotated-transformer/>
- Karpathy:
 - minGPT: <https://github.com/karpathy/minGPT>
 - nanoGPT: <https://github.com/karpathy/nanoGPT>

References

-
- [1] NLLB Team, Marta R. Costa-jussà, James Cross, et al.
No Language Left Behind: Scaling Human-Centered Machine Translation. 2022. arXiv: 2207.04672 [cs.CL].
 - [2] John Jumper, Richard Evans, Alexander Pritzel, et al. “Highly accurate protein structure prediction with AlphaFold”. In: Nature 596.7873 (Aug. 2021), pp. 583–589.
 - [3] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. “Layer normalization”. In: Neural Information Processing Systems (NeurIPS). 2016.
 - [4] Alec Radford, Jeffrey Wu, Rewon Child, et al.
Language models are unsupervised multitask learners. 2019.
 - [5] Jiatao Gu, James Bradbury, Caiming Xiong, et al. “Non-Autoregressive Neural Machine Translation”. In: International Conference on Learning Representations. 2018.

-
- [6] Fei Huang, Tianhua Tao, Hao Zhou, et al. “On the Learning of Non-Autoregressive Transformers”. In: Proceedings of the 39th International Conference on Machine Learning. Vol. 162. Proceedings of Machine Learning Research. PMLR, 17–23 Jul 2022, pp. 9356–9376.
 - [7] Yi Tay, Mostafa Dehghani, Dara Bahri, et al. “Efficient Transformers: A Survey”. In: ACM Comput. Surv. 55.6 (Dec. 2022).
 - [8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics. Minneapolis, Minnesota: Association for Computational Linguistics, June 2019, pp. 4171–4186.
 - [9] Alec Radford, Karthik Narasimhan, Tim Salimans, et al. Improving language understanding by generative pre-training. 2018.

-
- [10] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, et al. "Rethinking the Inception Architecture for Computer Vision". In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). June 2016, pp. 2818–2826.
 - [11] Tom Brown, Benjamin Mann, Nick Ryder, et al. "Language Models are Few-Shot Learners". In: Advances in Neural Information Processing Systems. Vol. 33. Curran Associates, Inc., 2020, pp. 1877–1901.
 - [12] OpenAI. GPT-4 Technical Report. 2023. arXiv: 2303.08774 [cs.CL].
 - [13] OpenAI. Introducing ChatGPT. <https://openai.com/blog/chatgpt>. Accessed: 25.04.2023. 2022.
 - [14] Victor Sanh, Lysandre Debut, Julien Chaumond, et al. "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter". In: 5th Workshop on Energy Efficient Machine Learning and Cognitive Computing - NeurIPS 2019. 2020. arXiv: 1910.01108 [cs.CL].

-
- [15] Renrui Zhang, Jiaming Han, Aojun Zhou, et al.
LLaMA-Adapter: Efficient Fine-tuning of Language Models with Zero-init Attention. 2023. arXiv: 2303.16199 [cs.CV].
 - [16] Zhiqiang Hu, Yihuai Lan, Lei Wang, et al.
LLM-Adapters: An Adapter Family for Parameter-Efficient Fine-Tuning of Large Language Models. 2023. arXiv: 2304.01933 [cs.CL].
 - [17] Edward J Hu, yelong shen, Phillip Wallis, et al. “LoRA: Low-Rank Adaptation of Large Language Models”. In: International Conference on Learning Representations. 2022.
 - [18] Mojtaba Valipour, Mehdi Rezagholizadeh, Ivan Kobyzev, et al.
DyLoRA: Parameter Efficient Tuning of Pre-trained Models using Dynamic Search-Free Low-Rank Ad
2023. arXiv: 2210.07558 [cs.CL].

-
- [19] Brian Lester, Rami Al-Rfou, and Noah Constant. “The Power of Scale for Parameter-Efficient Prompt Tuning”. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 3045–3059.
 - [20] Xiang Lisa Li and Percy Liang. “Prefix-Tuning: Optimizing Continuous Prompts for Generation”. In: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th Online: Association for Computational Linguistics, Aug. 2021, pp. 4582–4597.
 - [21] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, et al. “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”. In: 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3–7. OpenReview.net, 2021.

-
- [22] Enze Xie, Wenhui Wang, Zhiding Yu, et al. “SegFormer: Simple and Efficient Design for Semantic Segmentation with Transformers”. In: Advances in Neural Information Processing Systems. Vol. 34. Curran Associates, Inc., 2021, pp. 12077–12090.
 - [23] Ze Liu, Yutong Lin, Yue Cao, et al. “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows”. In: 2021 IEEE/CVF International Conference on Computer Vision (ICCV) (2021), pp. 9992–10002.
 - [24] Hong-Yu Zhou, Chixiang Lu, Sibei Yang, et al. “ConvNets vs. Transformers: Whose Visual Representations are More Transferable?” In: 2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW). 2021, pp. 2230–2238.
 - [25] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, et al. “A ConvNet for the 2020s”. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2022, pp. 11966–11976.

-
- [26] Keyu Tian, Yi Jiang, qishuai diao, et al. “Designing BERT for Convolutional Networks: Sparse and Hierarchical Masked Modeling”. In: [The Eleventh International Conference on Learning Representations](#). 2023.
 - [27] Ashish Vaswani, Noam Shazeer, Niki Parmar, et al. “Attention is All you Need”. In: [Advances in Neural Information Processing Systems](#). Vol. 30. Curran Associates, Inc., 2017.
 - [28] Yi Tay, Mostafa Dehghani, Samira Abnar, et al. “Long Range Arena : A Benchmark for Efficient Transformers”. In: [International Conference on Learning Representations](#). 2021.