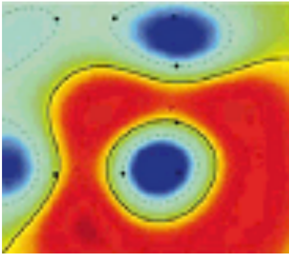
Technische Universität Berlin



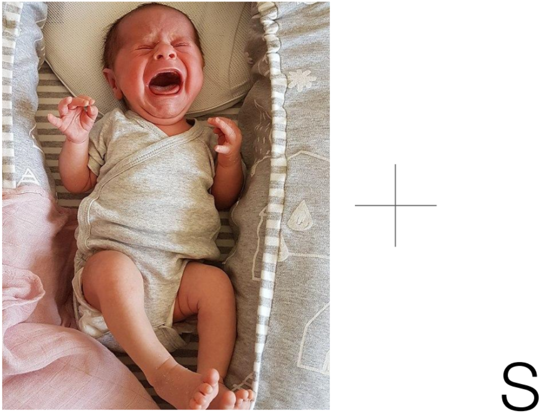
Lecture 2 | Attention

Outline

- Motivation
  - cognitive motivation/inspiration
  - machine-based attention
  - weighting of features/channels (CNNs, RNNs)
  - sequential modeling - Log-Likelihood (language, molecules)
- Introduction Attention Mechanisms
  - linear combination/ coefficients
  - simple non-linear combinations
  - encoder-decoder dot products
- Basic Transformer
  - QKV mechanism
  - self-attention
  - cross-attention, multi-modality
- Transformer Variants
  - Language Transformers
  - Vision Transformers
- Limitations and Discussion

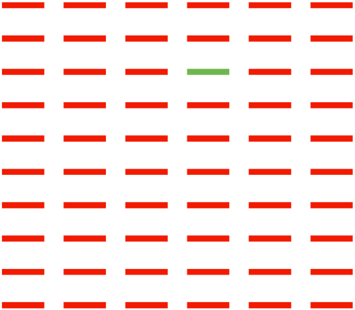
Attentional Capture

Attentional Capture



Attention Effects in Cognitive Science

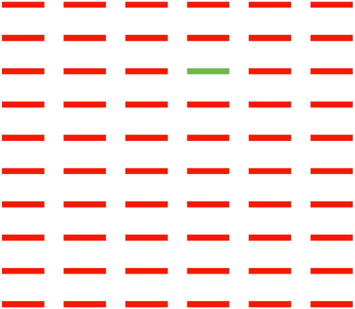
Pop-Out Effect



Time to find target is independent of display size, i.e., the number of items (involuntary, bottom-up)

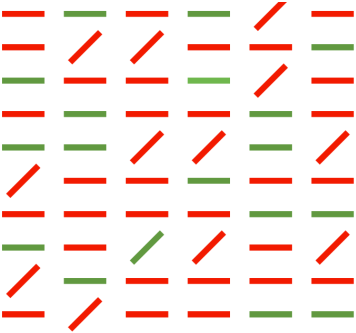
Attention Effects in Cognitive Science

Pop-Out Effect



Time to find target is independent of display size, i.e., the number of items (involuntary, bottom-up)

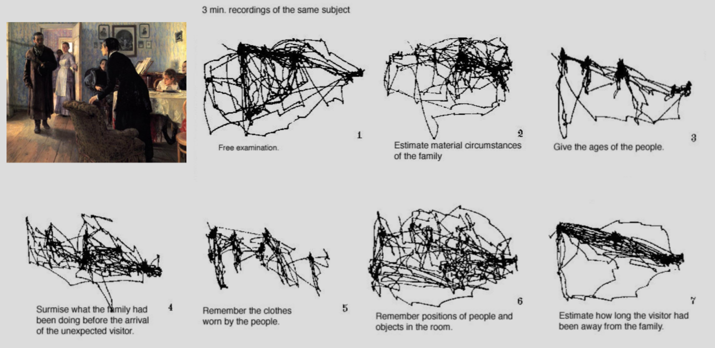
Serial Search



Time to find target is linear in display size (voluntary, top-down)

Saliency Maps and Saliency Modeling

- Data efficiency: How does the brain efficiently process visual scene understanding from massive retinal input ($\approx$ 10 to 100 million bits/s)?
- Attention mechanisms highlight such relevant (*salient*) regions and filters what regions to store and analyse in more detail
- Eye-tracking has been used since the 1940s to study human attention
- Yarbus (1967) [12]: Distribution of eye fixations is dependent on the questions asked about a presented scenery (task-specific saliency maps)



3 min. recordings of the same subject

Free examination.

Estimate material circumstances of the family

Give the ages of the people.

Surmise what the family had been doing before the arrival of the unexpected visitor.

Remember the clothes worn by the people.

Remember positions of people and objects in the room.

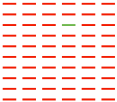
Estimate how long the visitor had been away from the family.

Feature Integration Theory

- Treisman & Gelade (1980) [9]: Visual processing of a scene consits of:
  - Disassembling into a set of feature representations such as color or orientation happening in parallel and pre-attentively
  - Combine feature maps into more complex objects requiring attentive mechanisms

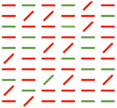
We have seen supporting evidence for this from our previous search experiment:

Pop-Out Effect



The panel is first disassembled into basic color, orientation or intensity maps, thus the target is unique with respect to a single feature dimension.
→ fast target pop-out

Serial Search



The target is a combination of multiple features (color and orientation), thus additional processing including attention mechanisms are needed to find the target.
→ slow target search

ML: Sequence Modeling

Sequence Modeling

Goal: Predicting or generating a sequence of data points. This sequence can be composed of any type of data, e.g. text, speech, images, or time series data.

$$\prod_{t=1}^{N} p(y_t \mid y_{1:t-1}, x_{1:T})$$

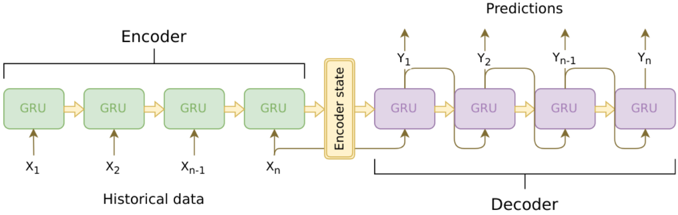
,
with input sequence $x_{1:T}$, previous elements in the target sequence $y_{1:t-1}$ and current target y_t and time step t.

Optimization: Typically, log-likelihood is used as a loss function during model optimization:

$$-\sum_{t=1}^{N} \log p(y_t \mid y_{1:t-1}, x_{1:T})$$

From RNNs to Attention-models

RNNs
seq2sec



Sequential processing All information for the decoder has to be represented into the *encoder state* or *context* vector. The hidden state changes as each new input is processed.

→ This is a problem for generating really long sequences!

Attention in Sequence Translation [1]

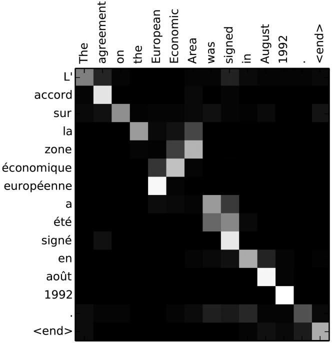
Attention Instead of allowing the decoder to directly look at input features from the encoder and let the model decide what to focus on for the current decoding step using an attention mechanisms that filters the input features:

$$\text{context}_l = \text{attention}_l \times \text{features}$$

This does not enforce a temporal structure of the sequence but it naturally emerges during training! Note flexibility in grouping and reordering tokens across languages:

- 'a été' → 'was'
- 'zone économique européenne' → 'European Economic Area'

Attention Interpretability "As side benefit, self-attention could yield more interpretable models." [10]
→ growing evidence that "attention is not explanation" [11]



Self-Attention

Assume inputs $\boldsymbol{x} = \{x_i, ..., x_N\}$ with $\boldsymbol{x} \in \mathbb{R}^{N \times D}$ and outputs $\boldsymbol{y} = \{y_i, ..., y_n\}$ with $\boldsymbol{y} \in \mathbb{R}^{N \times D}$ that are computed as a linear combination of all inputs:

$$y_i = \sum_j^N W_{ij} x_j \quad \text{with row-normalization} \quad \sum_j^N W_{ij} = 1$$

Attention weights W_{ij} are computed directly from the inputs (→ self-attention), e.g. using a dot-product:

$$w_{ij} = {x_i}^T x_j \quad \text{with softmax normalization} \quad W_{ij} = \frac{\exp w_{ij}}{\sum_j \exp w_{ij}}$$

So far we have not introduced any learnable parameters. Further, note that every x_i fulfills three different roles in this, every x_i is...

compared to all other data features to compute its own weights to later compute output y_i, e.g. in the above dot-product $w_{i1} = {x_i}^T x_1$.

→ *query*

compared to all other data feature to compute weights to compute their output, e.g. $w_{1i} = {x_1}^T x_i$.

→ *key*

used as a feature to compute the actual output y_i, e.g. via the contribution of $W_{1i} x_i$ for y_1.

→ *value*

Self-Attention - Query, Key and Value

Let us assign new variables to represent each of these different roles:

$$\rightarrow \textit{query}\colon\ q_i = x_i \qquad \Big| \qquad \rightarrow \textit{key}\colon\ k_i = x_i \qquad \Big| \qquad \rightarrow \textit{value}\colon\ v_i = x_i$$

In addition, we introduce learnable parameters $W_{\{k,q,v\}}$ to allow for projections of the data that can adapt to these distinct roles:

$$\textit{query}\colon\ q_i = W_q x_i \qquad \Big| \qquad \textit{key}\colon\ k_i = W_k x_i \qquad \Big| \qquad \textit{value}\colon\ v_i = W_v x_i$$

Resulting in

$$w_{ij} = q_i^T k_j / \sqrt{D} \quad \text{and} \quad y_i = \textstyle\sum_j W_{ij} v_j \quad \text{with} \quad W_{ij} = \text{softmax } w_{ij},$$

with scaling factor $\sqrt{D}$ to counteract large dot-product magnitudes that would result in small gradients in the subsequent softmax computation.

Intuition Matching a query to a set of available keys using dot-products (e.g. in databases) which eventually returns a matched value.

Building Transformers

Multi-head self-attention

We can further introduce multiple attention heads that each use different projection matrices. Similar to different convolutional kernels in CNNs, different attention heads indexed by $r = 1...R$ allow to extract different data features:

$$\textit{query}:\ q_i^r = W_q^r x_i \quad \Big| \quad \textit{key}:\ k_i^r = W_k^r x_i \quad \Big| \quad \textit{value}:\ v_i^r = W_v^r x_i$$

The outputs y_i^r of different heads are then concatenated and projected to lower dimensionality using matrix W_y:

$$y_i = W_y \text{concat}\left[y_i^1, ..., y_i^R\right]$$

Building Transformers

Multi-head self-attention

We can further introduce multiple attention heads that each use different projection matrices. Similar to different convolutional kernels in CNNs, different attention heads indexed by $r = 1...R$ allow to extract different data features:

$$\textit{query}:\ q_i^r = W_q^r x_i \quad \Big| \quad \textit{key}:\ k_i^r = W_k^r x_i \quad \Big| \quad \textit{value}:\ v_i^r = W_v^r x_i$$

The outputs y_i^r of different heads are then concatenated and projected to lower dimensionality using matrix W_y:

$$y_i = W_y \text{concat}\,[y_i^1, ..., y_i^R]$$

Positional encoding

So far, we have looked at sets of sequences, thus changing the order of x_i in $\boldsymbol{x}$ does not change the extracted features after the self-attention layers (→ *permutation-invariance)*. To overcome this, positional embeddings p_t are *added* to the initial token embeddings. A common choice are sinusoidal encodings. The index of the token $t = 1...T$ is positionally embedded using sine functions of different frequencies:

$$p_t = [\sin \omega_1 t, \sin \omega_2 t, ..., \sin \omega_D t] \ \text{ with } \ \omega = 2\pi f, \tag{1}$$

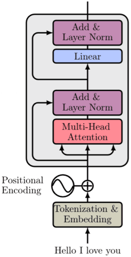
where one frequency corresponds to one embedding dimension D.

Building Transformers

Transformer block

Transformers consist of several processing blocks, i.e.

1. self-attention block (multi-head self-attention, residual connections and layer normalization)
2. a subsequent projection block (Linear MLP layer(s) and again layer normalization).



Vanillla Transformer [10]

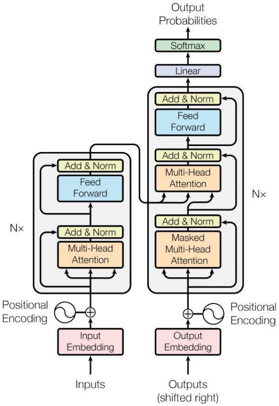
Task Typical scenario for Transformers are sequence-to-sequence tasks, e.g. machine translation, which use an encoder-decode structure to encode the input sentence and also the translated sentence during training time.

Encoder

(i) embedding and positional encoding of input token

(ii) subsequently pass through all $N = 6$ Transformer blocks

(iii) route encoder output to the encoder



Decoder

(i) embedding and positional encoding of output token (partially masked)

(ii) subsequently pass through all $N = 6$ Transformer blocks

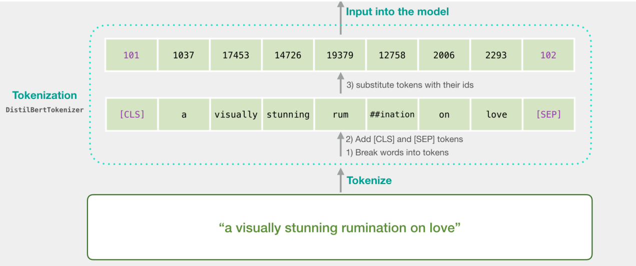
(iii) Transformer encoder blocks additionally received inputs from the encoder that inform the translation

(iv) final linear layer and softmax over all tokens

(v) select most likely next translated token and feed back to the decoder

Transformers in Practice

Tokenization As a first step, input sequences are tokenized into a set of basic elements that are part of the vocabulary and for which an embedding is computed.



Input into the model

101	1037	17453	14726	19379	12758	2006	2293	102

3) substitute tokens with their ids

Tokenization

DistilBertTokenizer

[CLS]	a	visually	stunning	rum	##ination	on	love	[SEP]

2) Add [CLS] and [SEP] tokens

1) Break words into tokens

Tokenize

"a visually stunning rumination on love"

Note: A [CLS] token is added at the beginning of each sentence and a [SEP] token is added at the end respectively. The output representation of the [CLS] token is typically used as the input to the readout/classification layer.

Linear Transformers

Linear Attention The softmax results in quadratic complexity ($\rightarrow \mathrm{Q}^T K$). We can rewrite the self-attention using an arbitrary similarity score computed using kernel functions ϕ:

$$
\begin{aligned}
y_i &= \mathrm{softmax}(\frac{q_i^T k_j}{\sqrt{D}}) v_i \\
&= \frac{\phi(q_i)^T \sum_j^N \phi(k_j) v_j^T}{\phi(q_i)^T \sum_j^N \phi(k_j)}
\end{aligned}
$$

This allows autoregressive Transformer inference with linear complexity and constant memory [3].

Transformers as RNNs For *autoregressive* Transformers it can be shown that they are in fact RNN models. Autoregressive refers to the assumption that only past values are used to predict future values, i.e. if the sum only goes until $N = i$ in above summations) [3].

Random Feature Attention We can further build on this idea to replace the softmax and use random feature methods to approximate the softmax function. [7]

Transformer-family

More data RoBERTa, GPT
Compression DistilBERT, Electra
Architecture/Domain

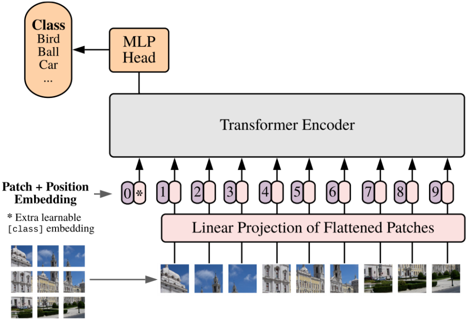
- NLP: BERT, GPT-2/3/etc.,
- Vision: Vision Transformer (ViT), Image GPT, ConvBERT
- Predict molecular properties: Chemformer
- Speech: UniSpeech, Wav2Vec2-Conformer

Multi-Tasks/Domains T5 (text multitasking), MuIT (image multitasking), UnIT (multimodal), CLIP (text-image), ChatGPT (multi-tasks, multi-modal)

- Multilingual: mBERT, MBart, XLM-family

Vision Transformers [2]

Convolution-free
Instead of using convolutional layers represent an image as a sequence of image patches (tokens).


Class
Bird
Ball
Car
...
MLP Head
Transformer Encoder
Patch + Position Embedding
* Extra learnable [class] embedding
0
*
1
2
3
4
5
6
7
8
9
Linear Projection of Flattened Patches

Vision Transformers achieve comparable or higher classification performance than best-performing CNNs, some insights into the differences include [8, 5]:

- ViT can aggregate global information early due to self-attention (instead of only building them in later layers as CNNs)
- Using shape information much more than texture information (unlike CNNs that heavily rely on texture information).
- Skip connections propagate representations from lower to higher layers and have been found to play an important role in ViT.

Discussion on Transformers

Inductive Biases In comparison, Transformers have less strong inductive biases as compared to CNNs or RNNs. Thus, their solution space is less constrained but as a result large amounts of training data are typically required. The design of Transformers still implements assumptions about the structure of the problem via:

- **Structure of Self-Attention**: Assign specific roles to different representations (e.g. k controls what information is selected from v, while v is responsible for the value of this information).
- **Bi-directional training**: Allows to attend to future and past tokens.
- **Positional Encoding**: Allows Transformers to encode absolute and relative information about token position.

These underline again that instead of making strong assumptions, the typical approach in Transformers is to give the model the flexibility to learn from the data.

Discussion on Transformers

Complexity

Layer	(a) Complexity per Layer	(b) Sequential Operations	(c) Max. Path Length
Self-Attention	$O(N^2 \cdot D)$	$O(1)$	$O(1)$
Recurrent	$O(N \cdot D^2)$	$O(N)$	$O(N)$
Convolutional	$O(K \cdot N \cdot D^2)$	$O(1)$	$O(\log_k N)$

N: sequence length, D: hidden dimension, K: kernel size
Max. Path Length: shortest path between first encoder input and last decoder output

(a) Self-Attention layers are more computationally efficient than recurrent and convolutional layers when the sequence length is smaller than the size of the hidden dimension (when is this realistic?).

(b) Self-attention and convolutional are feed-forward using a fixed number of serial computations, in RNNs the number of recurrent layers grow with sequence length.

(c) Self-attention reduces maximal path length between long range dependencies of input and output (attending all elements at the same time).

Discussion on Transformers

In general: Trade-off between the number of sequential operations and decoding complexity: In Transformers, number of sequential computations are independent of sequence length during encoding, but decoding complexity grows with the sequence length!

Computational limitations Performant modern language models are typically trained for several weeks by big corporations or institutions. (→ **carbon footprint** [6])

- Reproducibility and transparency are often not granted, while application have seen vastly growing distribution.
- This may have effects on innovation and progress in the field, which has previously been driven by open source and open data.

Parallelization Can positions of the sequence be processed in parallel?

- RNNs can memorize but operations can not be parallelized.
- CNNs can be parallelized but are not designed to store information.
- Transformers can be parallelized in parts (the encoder and self-attention, the decoder only during training) and are able to memorize.

Discussion on Transformers

Interpretability

- Learned attention units are not directly interpretable ('Attention is not explanation.' [11]).
- Open conceptual questions: how to analyze and validate generative Transformers?

Generalization

- Limited abilities to reason beyond the input.
- Limited to domain-specific knowledge of training material. (But, promising results for zero-shot and few-shot language transfer have been reported. [4])

Bibliography I

[1] D. Bahdanau, K. Cho, and Y. Bengio.
Neural machine translation by jointly learning to align and translate.
In Y. Bengio and Y. LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[2] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby.
An image is worth 16x16 words: Transformers for image recognition at scale.
In *9th International Conference on Learning Representations, ICLR 2021*, 2021.

[3] A. Katharopoulos, A. Vyas, N. Pappas, and F. Fleuret.
Transformers are rnns: Fast autoregressive transformers with linear attention.
In *Proceedings of the International Conference on Machine Learning (ICML)*, 2020.

[4] A. Lauscher, V. Ravishankar, I. Vulić, and G. Glavaš.
From zero to hero: On the limitations of zero-shot language transfer with multilingual Transformers.
In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4483–4499, Online, Nov. 2020. Association for Computational Linguistics.

[5] M. Naseer, K. Ranasinghe, S. Khan, M. Hayat, F. Khan, and M.-H. Yang.
Intriguing properties of vision transformers.
In A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021.

[6] D. Patterson, J. Gonzalez, Q. Le, C. Liang, L.-M. Munguia, D. Rothchild, D. So, M. Texier, and J. Dean.
Carbon emissions and large neural network training, 2021.

[7] H. Peng, N. Pappas, D. Yogatama, R. Schwartz, N. A. Smith, and L. Kong.
Random feature attention.
CoRR, abs/2103.02143, 2021.

[8] M. Raghu, T. Unterthiner, S. Kornblith, C. Zhang, and A. Dosovitskiy.
Do vision transformers see like convolutional neural networks?
Advances in Neural Information Processing Systems, 34:12116–12128, 2021.

Bibliography II

[9] A. M. Treisman and G. Gelade.
A feature-integration theory of attention.
Cognitive Psychology, 12(1):97–136, 1980.

[10] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin.
Attention is all you need.
In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[11] S. Wiegreffe and Y. Pinter.
Attention is not not explanation.
In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 11–20, Hong Kong, China, 2019. Association for Computational Linguistics.

[12] A. L. Yarbus.
Eye Movements and Vision.
Plenum. New York., 1967.