UNIVERSITAT POLITÈCNICA DE CATALUNYA

FACULTAT D'INFORMÀTICA DE BARCELONA

Transformer Models

Big Data Seminars

Spring 2023

Authors:
León Villapún, Luis Alfredo, *email*: luis.alfredo.leon@estudiantat.upc.edu
Lorencio Abril, Jose Antonio, *email*: jose.antonio.lorencio@estudiantat.upc.edu

Contents

List of Figures

List of Tables

1 Introduction

Sequential data, which includes time series, images, and text, is a type of data where the order of elements carries important information. Analyzing sequential data has been a critical task in various fields, such as finance, healthcare, natural language processing, and computer vision. This article provides a state-of-the-art overview of Transformers, a breakthrough deep learning architecture for processing sequential data.

The history of sequential data analysis can be traced back to classical time series statistical methods, such as autoregressive (AR) and moving average (MA) models, used since the mid-20th century. With the advent of more complex data and the need for better predictive capabilities, researchers turned to more advanced techniques, such as recurrent neural networks (RNNs), first introduced as a learning method in [Ama72]. RNNs were later improved by long short-term memory networks (LSTMs), introduced in 1997 [HS97], and gated recurrent units (GRUs) in 2014 [Cho+14], with fewer parameters than LSTM and similar or improved performance than them in some use cases. These methods were designed to capture long-range dependencies and retain information from the past to create meaningful representations of the input data.

However, these approaches still faced limitations in capturing long-range dependencies and suffered from issues like vanishing and exploding gradients, making it difficult to establish a proper context when processing sequential data. Moreover, traditional machine learning methods were often insufficient for handling the complexity of sequential data, leading to the rise of deep learning techniques. Also, the deep learning approaches, leaded by recurrent models (RNNs, LSTMs and GRUs) process the data in a sequential manner, which difficults the parallelization of computations and, thus, the overall model performance in terms of response and training times.

In response to these challenges, the Transformer architecture was introduced in 2017 by Google Research team [Vas+17], revolutionizing the field of deep learning for sequential data. The Transformer is basically an architecture that enabled the powerful concept of the self-attention mechanism and position encoding to efficiently process and model long-range dependencies in data. This article will delve into the inner workings of Transformers, discussing their key advancements and the impact they have had on a wide range of applications.

As we explore the world of Transformers, we will uncover the reasons behind their success and the potential they hold for future developments in sequential data analysis. We also explain the main developments based on the Transformer model, such as the Large Language Models (LLMs) developed both by big technological companies and by the Open Source community, which were in fact introduced by Google Research in 2007 [Bra+07], but have received a huge boost thanks to the development of the Transformer.

2 The Transformer Model

The Transformer was first introduced by Google Research in the influential paper *Attention Is All You Need* [Vas+17]. Their major contribution was an architecture that combined previously used constructs to improve the natural language processing top approaches at the moment, specifically in language transduction.

2.1 Transformer Model Architecture

The Transformer model architecture, which is depicted using the original diagram in Figure 1, consisted in the following constructs:

- Encoder-decoder stacks: encoders are built with a multi-head attention layer and a fully-connected feed forward network, while decoders add a third component, which is another multi-head attention layer that takes as input the output of the encoder. They stacked 6 of these layers.

- The attention mechanism, implemented as a scaled dot-product between the vectors Query Q, Key K, and Value V. The softmax function is applied to the scaled dot-product of Q and K to generate attention weights, which are then multiplied by the Value matrix V. These vectors are projected to different sub-spaces, to leverage the multi-head attention mechanism. They used a total of 8 heads (i.e. projections). This process is depicted in Figure 2. The Query vector (Q) represents the part of the input sequence that we are focusing at a given moment, the Key vector (K) holds information about where we could find terms related to Q. Finally, V is the part of the input sequence in which we want to find these related terms. For example, in text processing, V could be the text previous to the currently processing word(s). Notice that these three vectors are trained during training time, focusing on the task at hand. The multi-head attention implies that several of these vectors are used in parallel, enabling for different tuples Q, K, V in each of the heads.

- Position-wise Feed-Forward Network: inside the encoder and the decoder, they used a FFN applied to each position separately and identically. The chosen function used was the composition of two linear functions L_1, L_2 and a ReLU activation function $ReLU$ in the form $L_1 \circ ReLu \circ L_2$.

- Embeddings and Softmax: the approach for converting text into an embedding space was already widely used as neural networks require numerical inputs. They applied a learned softmax activation function to convert the decoder output into predicted next-token probabilities.

- Positional Encoding: it is noticeable that the Transformer does not have recurrence nor convolution, and so to leverage the sequential nature of the data, a positional encoding is added to the input embeddings to provide information about the position of each token in the sequence. The positional encoding consists of sinusoidal functions with different frequencies based on each token's position in the input string, allowing the model to learn and use relative position information.

Among all these constructs, the most critical is the self-attention mechanism, since they proved that a self-attention layer connects all positions in the input with a constant number of sequential operations, while a recurrent layer requires $O(n)$ such sequential operations. They also noted that self-attention layers are faster than recurrent layers when the input length is smaller than the embedding space's dimension. For larger input, they propose to analyze the input in batches of appropriate length, so effectively the Transformer has a 'context' of the dimension of the used embedding space at most. They also assert that the self-attention mechanism provides an inherent interpretability tool, since the weights of these layers could be used to check in which tokens the model is basing its output more strongly, which had been explored before in other applications of attention mechanisms, explained in Section 4.2.

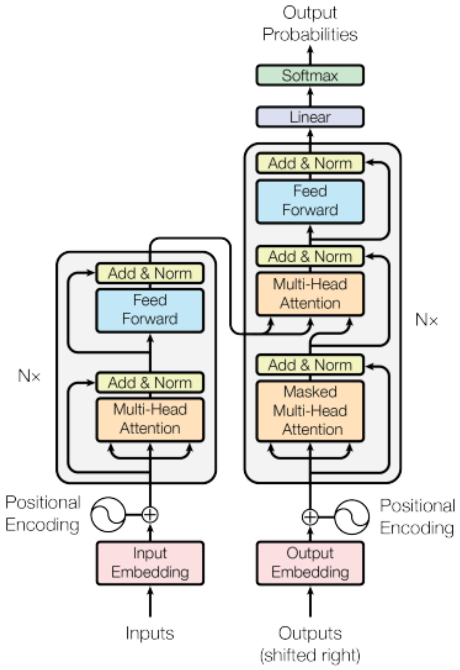


Figure 1: Transformer model architecture

Source: [Vas+17]

2.2 Training and Results

The authors conducted training experiments using two types of models: base models and big models. The base models were trained for 12 hours, while the big models were trained for around 84 hours. They used the Adam optimizer with specific hyperparameters and a computed learning rate. Several regularization techniques were employed, including residual dropout applied to output sublayers, embeddings, and positional encoding in both encoder and decoder stacks, as well as label smoothing. The big transformer model achieved impressive results on the WMT 2014 English-to-German and English-to-French translation tasks, outperforming previous models by a significant margin. These findings led to increased attention and further exploration of the Transformer model by the research community.

3 Research Areas

The advent of the Transformer architecture has inspired a multitude of researchers to explore novel methods for enhancing and refining this model. These approaches typically involve improvements or modifications to the Transformer, often overlapping and intersecting with each other. Nevertheless, [Lin+22] classifies the research endeavors pertaining to the Transformer into four primary directions[1]:

- Module-level improvements: This line of research concentrates on enhancing and modifying the Transformer in one or more of its individual components (such as multi-head attention, feed-forward neural networks, positional encoding, add and normalization layers).
- Architecture-level improvements: This area of investigation focuses on high-level improve-

[1] With a friendly explanation in [Gha23].

ments to the model, encompassing aspects like architecture modifications, recurrence and hierarchy, and adaptive computation time.

- Pretrained models: The evolution of transformers can also be studied by the lens of their implementation categories, including encoder-only, decoder-only, or encoder-decoder models.

- Applications: Finally, an interesting approach is to go through the research by their application use-case, such as text, vision, audio, multi-modal, etc. Please note that this approach will commonly apply some of the previously mentioned categories to a particular domain.

In this study we will review the state-of-the-art advancements in these fields, diving deeper into module-level improvements and pretrained models, particularly in the context of NLP, with a whole new tendency in the market with Large Language Models.

4 Module-Level Improvements

As outlined before, this line of research focuses on the improvement of the different components of the model. Therefore, we are going to delve into the advances conducted for these components, based partially in [Lin+22]. We focus in the advancements with regards to the attention mechanism, which can be considered the most important and differential module in the Transformer.

4.1 Attention

Self-attention could be considered as the core component of the model, since it is the component that allows the model to have memory to remember the past of the input sequence. This makes it a bottleneck when dealing with long sequences, since it then forces the model to perform several scans over the sequence, with appropriate context length. Not only this, but the vanilla self-attention mechanism is 'structure agnostic', in the sense that it does not assume any structural bias in the data. This kind of information need to be learnt at training time, making the model prone to overfit in small datasets. The Self-Attention mechanism, as we explained before, is achieved through a process like the one depicted in Figure 2.

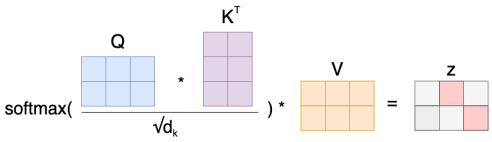


Figure 2: Vanilla Self-Attention Mechanism

Source: [Ala18]

The improvements for this mechanism have been done in different ways. In this paper, we deepen into *sparse attention*, *linearized attention*, *query prototyping* and *memory compression*, but other types of modifications to this module exist, from which we can highlight the *low-rank self-attention*, which replaces the attention matrix by a low-rank approximation; the *attention with prior*, which combines learnt attention vectors with some pre-defined attention 'prior distribution', thus introducing bias into the model through our beliefs about how attention works in each case; and the *improved multi-head mechanism*, which focuses on enhancing the multi-head

mechanism by ensuring that each of the heads does indeed capture different information for attention.

- Sparse attention: based in the observation that attention matrices are usually very sparse across most data points [Chi+19], this approach tries to introduce a structural bias to reduce the complexity of the attention matrix. Instead of computing the attention matrix through the previously explained process, we just define a structure that make this process easier, i.e., we define the attention matrix as

$$\hat{A}_{i,j} = \begin{cases} Q_i \cdot K_j^T & \text{if token } i \text{ attends to token } j, \\ -\infty & \text{if token } i \text{ does not attend to token } j. \end{cases}$$

  Sparse attentions has basically two perspectives:

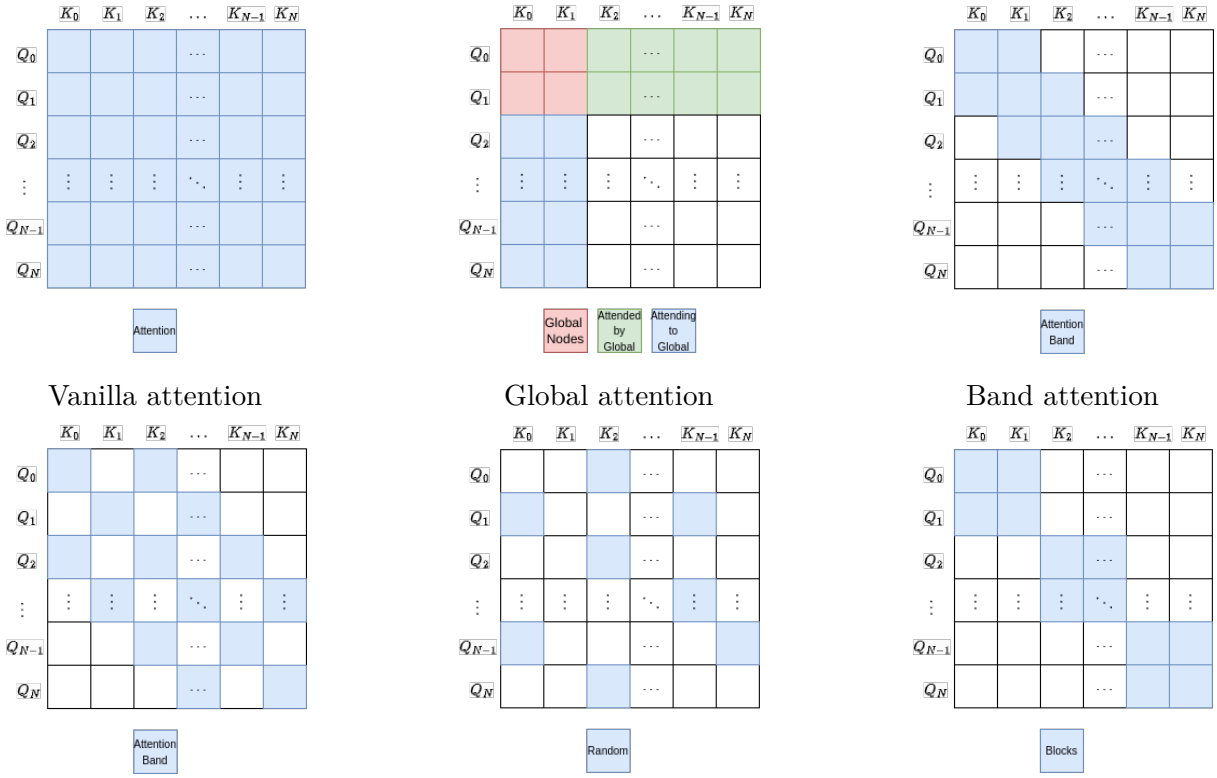
  – Position-based: this approach tries to simplify the attention matrix by using predefined patterns, which can be atomic, when they are directly detected, or composed, which are combinations of other patterns.

    There exist basically five kinds of atomic patterns, depicted in Figure 3, which are the following:

    1. Global attention: global nodes are added, so that attention is focused in these instead of in the full sequence. Each of this nodes are allowed to attend to the entirety of the sequence.
    2. Band attention: attention is restricted to neighbouring parts of the sequence, similarly to a sliding window.
    3. Dilated attention: a variation of band attention, in which nodes can attend to a diladed window with gaps. This can increase the width of the attention, while keeping the complexity still.
    4. Random attention: attention edges are sampled randomly.
    5. Block local attention: the input sequence is divided into blocks, and attention is only done at the block level.

    As for sparse patterns, they usually involve more than one of the atomic patterns, depicted in Figure 4. These are:

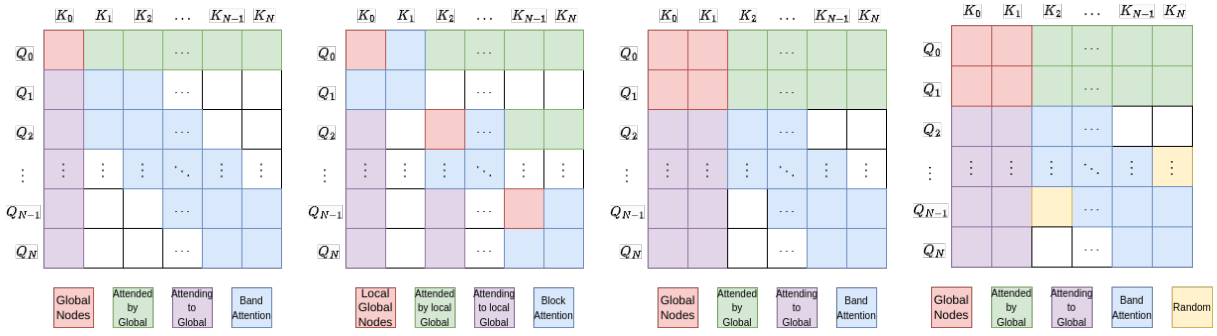
    1. Star-Transformer [Guo+19]: combines band attention and global attention, where pairs of non-adjacent nodes are connected through a single global node, and neighbouring nodes are directly connected.
    2. Longformer [BPC20]: combines band attention, block local global attention and some band attention heads in upper layers of the model are replaced by diladed attention, to augment the perception field.
    3. Extended Transformer Construction (ETC) [Ain+20]: combines band attention and global attention, together with a masked mechanism.
    4. BigBird [Zah+21]: adds random attention to the ETC to approximate a full-input attention.



Vanilla attention | Global attention | Band attention

Dilated attention | Random attention | Block attention

Figure 3: Atomic patterns for sparse position-based attention.



Star-Transformer | Longformer | ETC | BigBird

Figure 4: Composed patterns for sparse position-based attention.

Not only this, but some attention mechanisms have been developed for specific datatypes. Among these, we highlight the BP-Transformer [Ye+19] and the Fourier Sparse Attention for Transformer (FSAT) [ZZT22] for text data; and the Image Transformer [Par+18] and Axial Transformer [Ho+19] for images.

– Content-based: this research line tries to perform attention by means of a sparse graph built using the input content. The Routing Transformer [Roy+21] uses a k-means approach to cluster the queries and keys, so that queries only attend to those keys lying in the same cluster. The Reformer [KKL20] selects key-value pairs by means of LSH, i.e., a hashing scheme that makes use of locality. There is a bucket for each query, and the query only attends to those key-value pairs that are hashed into its bucket. Sparse Adaptive Connection [Li+20] trains a LSTM network using reinforcement learning to construct attention edges between tokens. Sparse Sinkhorn Attention [Tay+20] uses a sorting network with Sinkhorn normalization to assigned previously grouped of queries and keys between them. The Energon [Zho+23] uses a mix-precision multiround filtering to dynamically identify which key-value pairs are important at runtime.

- Linearized Attention: in [Kat+20] the authors developed a new formulation for the attention, which enabled to create an iterative implementation that speeds the computation of the attention matrix up from $O(n^2)$ to $O(n)$. This method is based in an alternative formulation, using a linear dot-product and making use of the associativity of matrix products, by applying a function $\phi(x)$ to Q and K and changing the operators order. This idea is depicted in Figure 5.

Vanilla attention Linear attention

Figure 5: Linearizing attention.

There are alternative and innovative approaches for linearizing the computation of the attention, which we summarize in Table 1.

Method	Main Idea	Reference
Performer / Random Feature Attention	Selects random orthogonal features, obtaining an unbiased estimator of the attention.	[Cho+22] [Pen+21]
Fast Weight Programmers	Train a second network that learns how our model computes its weight. The second model uses lightweight operations.	[SIS21]
Momentum Transformer	Introduced the concept of Momentum, that enables to compute the attention in a Gradient Descent manner.	[Ngu+22]
Flowformer	Interpret attention as a flow of information and leverage the flow conservation property.	[Wu+22]

Table 1: Alternative approaches for linearized attention.

Class	Method	Main Idea	Reference
Query Prototyping	Clustered Attention Mechanism	Cluster queries, compute attention between centroids.	[VKF20]
	ProbSparse Attention	Attend only to top-K queries, according to a metric.	[Zho+21]
	RACP	Refine previously computed prototypes based on an attention score.	[Wan+22]
	Trajectory Attention	Take into account the time dimension in video transformers.	[Pat+21]
Memory Compression	Memory Compressed Attention	Use a convolution kernel to summarize key-value pairs.	[Liu+18]
	Set Transformer/Luna	Use external trainable nodes for summarization.	[Lee+19] [Ma+21]
	Linformer	Use linear projection to reduce keys and values dimensionality.	[Wan+20]
	Poolingformer	Uses max pooling and convolution for decreasing the amount of keys and pairs.	[Zha+22]
	LFEformer	Uses a dynamic sliding window that changes its size based on the embedded network layers.	[Wei+23]

Table 2: Query Prototyping and Memory Compression.

- Query Prototyping and Memory Compression: query prototyping refers to reducing the number of queries, while memory compression is the reduction of key-value pairs. Both approaches aim at reducing the complexity of the attention mechanism. The basic idea for each method is depicted in Figure 6. In Table 2, we summarize the principal approaches in this line of research.

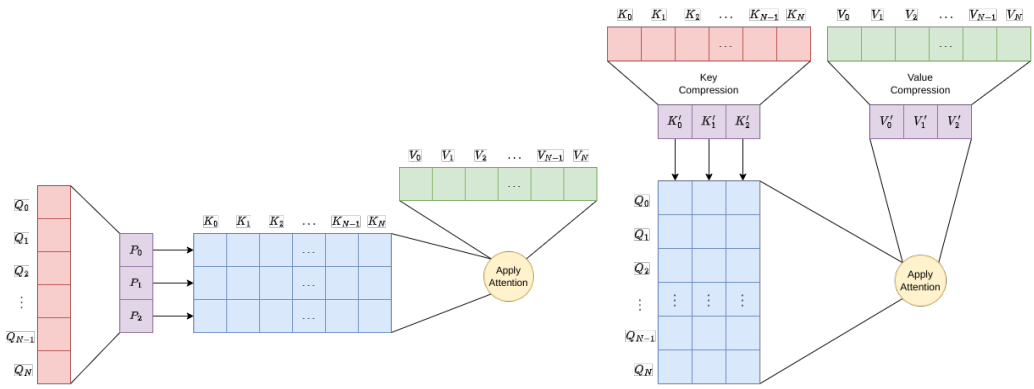


Figure 6: Query prototyping (left) and Memory Compression (right).

4.2 Interpretability

As we outlined in Section 2, the authors claimed that the multi-head attention mechanism provides inherently a simple way to interpret which parts of the input is the model using for creating the output. This had been explored before in [Xu+15]. This approach, however, is quite basic, and several studies have developed more complex interpretability schemes for Transformer

based models. For example, in [RT18], they used the attention weights to extract dependency relations between the different representations of the encoders, which enabled them to conclude that lower layers tend to learn more about the syntax of the language, while the higher layers tend to learn more about the semantics.

Later, in [Voi+19] a more advanced approach was introduced, in which they assessed the importance of each head in each encoder layer leveraging the approach proposed in [Din+17] to later characterize each head's roles as positional, syntactic and rare-words-focus. Then, they pruned heads using a regularization approach. They found out that most heads are not important for translations tasks, important heads have one or more specialized and interpretable functions in the model, and these functions correspond to attention to neighbouring words and to tokens in specific syntactic relationships.

More recently, in [CGW21], the researchers developed an advanced technique for determining which parts of an image were used by a Transformer-based computer vision (Vision Transformers, which were introduced in [Par+18]) model to classify the image. Their method assigns local relevance based on the Deep Taylor Decomposition principle and then propagates these relevancy scores through the layers of the model, obtaining clear and consistent visualizations, as well as state-of-the-art results in some segmentation metrics, and also on the Movie Reviews reasoning task [ZE08].

5 Pretrained Models

As mentioned in [Han+21], in recent years, the question of finding and refining techniques to train deep learning models with relatively short datasets has posed a challenge for researchers. The prevalent approach in contemporary studies involves the utilization of pretrained models, as they offer a cost-effective and versatile framework for diverse applications. A pretrained model represents a machine learning model that has undergone prior training with a broader objective than the target task. The primary benefit of employing such models lies in their capacity for fine-tuning to cater to specific tasks, thereby endowing them with both domain-specific and general-purpose contexts. For instance, ImageNet has become the standard pretraining dataset repository for computer vision tasks [Den+09]. As a prominent case, the natural language processing domain has highly benefited from this approach, and we can trace their use on the Transformer with its precursor, the Word2Vec algorithm [Mik+13]. Particularly, the term *Large Language Model*[2] has been coined to refer to the families of models used for natural language representation trained on a big enough corpus[3], independently of their use of the Transformer architecture or not. However, as we will see, the current state-of-the-art models use this approach.

The birth of the Transformer quickly brought three main variations to the table, each of these with different applications and goals: encoder-only, decoder-only, and encoder-decoder models. It's worth noting that the popularity of the Transformer architecture has made it difficult to keep track of the vast amount of new variations on the models, but the work in [Yan+23] and [Ama23] does a great contribution at addressing all these family trees in a structured manner.

[2] [SG05] already define LLMs in terms of using neural networks, and the size of parameter and corpus data.

[3] There is no agreed upon threshold for classifying models as LLMs. With technological progress, most current models can be considered LLMs due to their use of deep learning and billions of parameters. For instance, [SG05] definition limited LLMs to 600M parameters at the most, yet GPT-3[Bro+20] trained with 300 billion parameters.

5.1 Encoder-only models

These kind of models are not considered generative by nature, since they are omitting the decoder part of the Transformer. They utilize the encoder to learn the context and then perform tasks like classification. Oftentimes these kind of models are useful in cases where the context can be bidirectional, this means that the right side of the analysed text is not masked as in other approaches. This is the innovation used by [Dev+19] with Google's BERT, the most representative model with these conditions. Other pretrained models inspired by BERT include RoBERTa [Liu+19], DistillBERT [San+19] (a faster implementation utilizing distillation[4]), and DeBERTa [He+20]. Although it is important to remark the massive utilisation of BERT-style models in 2020 and 2021, in recent years they have lost popularity against other families such as GPT, which we will review as well.

5.2 Encoder-decoder models

The original vanilla Transformer from [Vas+17] proposes this architecture, taking into consideration an innovative attention layer that captures important context into the model. The fact that we also have a decoding layer makes this family of Transformers generative. This family is commonly implemented for translation, summarization, or question answering tasks. The most representative models from this family include BART [Lew+20] and T5 [Raf+20]. However, current state-of-the-art developments have extended these ideas with models such as GLM [Zen+22] and UL2 [Tay+23].

5.3 Decoder-only models

The benefit of having an encoding layer can be inherently expressed inside the decoder itself, making the models that explore this idea encode the data directly in the hidden states of the decoder. This allows for tasks like text generation and completion and is currently the most used approach with technologies such as the famous GPT subfamily, which originated with the work from [RN18]. The current state-of-the-art models include GPT-4 [Ope23], LLaMA [Tou+23], and Bard, a lightweight version of the original LaMDA [Coh+22].

6 Final Discussion

Transformer models are a booming technology, and ever since the release of [Vas+17] in 2017 we have observed a massive interest from both academia and industry in this area. This can be exemplified with platforms like ChatGPT or HuggingFace. Transformer models are a revolutionary technology and we personally think that they are marking a new era in Machine Learning. The architectural and structural variations that we have discussed in this paper are only a fraction of all the possible research areas that have awaken the interest of the academia. We can say that focus now is both in improving model performance while at the same time reducing costs of training. This is undelving interesting model variations and techniques, however all based on the same original idea: Transformers.

The technologies developed under this architecture are evolving at such a fast rate that it is difficult to distinguish between state-of-the-art and already obsolete approaches. This poses significant opportunities as well as threats. On one side, we can observe that the democratization

[4]This approach uses a "master" model's output as input to a "student" model. This saves computation time as the student is able to capture most of the information from the original, with labels generated by the master LLM. The most recent improvements include the work from [Hsi+23].

of Machine Learning and Artificial Intelligence is underway, which will be beneficial in the long term. However, we cannot put aside the fact that many of the advancements proposed will require formal research and approval by the community, which can be hard to achieve with so many variations appearing day by day. In our opinion, this is an exciting time to truly delve into the research of such an amazing topic.

References

[Ain+20] Joshua Ainslie et al. "ETC: Encoding Long and Structured Inputs in Transformers". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, Nov. 2020, pp. 268–284. DOI: 10.18653/v1/2020.emnlp-main.19. URL: https://aclanthology.org/2020.emnlp-main.19.

[Ala18] Jay Alammar. *The Illustrated Transformer*. Ed. by Github. [Online; posted 27-June-2018]. 2018. URL: https://jalammar.github.io/illustrated-transformer/.

[Ama72] S.-I. Amari. "Learning Patterns and Pattern Sequences by Self-Organizing Nets of Threshold Elements". In: *IEEE Transactions on Computers* C-21.11 (1972), pp. 1197–1206. DOI: 10.1109/T-C.1972.223477.

[Ama23] Xavier Amatriain. *Transformer models: an introduction and catalog*. Feb. 2023. DOI: 10.48550/arXiv.2302.07730.

[BPC20] Iz Beltagy, Matthew E. Peters, and Arman Cohan. *Longformer: The Long-Document Transformer*. 2020. arXiv: 2004.05150 [cs.CL].

[Bra+07] Thorsten Brants et al. "Large language models in machine translation". In: (2007).

[Bro+20] Tom B. Brown et al. "Language Models are Few-Shot Learners". In: *CoRR* abs/2005.14165 (2020). arXiv: 2005.14165. URL: https://arxiv.org/abs/2005.14165.

[CGW21] Hila Chefer, Shir Gur, and Lior Wolf. "Transformer Interpretability Beyond Attention Visualization". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2021, pp. 782–791.

[Chi+19] Rewon Child et al. *Generating Long Sequences with Sparse Transformers*. 2019. arXiv: 1904.10509 [cs.LG].

[Cho+14] Kyunghyun Cho et al. "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1724–1734. DOI: 10.3115/v1/D14-1179. URL: https://aclanthology.org/D14-1179.

[Cho+22] Krzysztof Choromanski et al. *Rethinking Attention with Performers*. 2022. arXiv: 2009.14794 [cs.LG].

[Coh+22] Aaron Daniel Cohen et al. "LaMDA: Language Models for Dialog Applications". In: *arXiv*. 2022.

[Den+09] Jia Deng et al. "Imagenet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255.

[Dev+19] Jacob Devlin et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *ArXiv* abs/1810.04805 (2019).

[Din+17] Yanzhuo Ding et al. "Visualizing and Understanding Neural Machine Translation". In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vancouver, Canada: Association for Computational Linguistics, July 2017, pp. 1150–1159. DOI: 10.18653/v1/P17-1106. URL: https://aclanthology.org/P17-1106.

[Gha23] Soran Ghaderi. *The Map of Transformers*. Ed. by TowardsDataScience. [Online; posted 19-April-2023]. 2023. URL: https://towardsdatascience.com/the-map-of-transformers-e14952226398.

[Guo+19] Qipeng Guo et al. "Star-Transformer". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota:

Association for Computational Linguistics, June 2019, pp. 1315–1325. DOI: 10.18653/v1/N19-1133. URL: https://aclanthology.org/N19-1133.

[Han+21] Xu Han et al. "Pre-trained models: Past, present and future". In: *AI Open* 2 (2021), pp. 225–250. ISSN: 2666-6510. DOI: https://doi.org/10.1016/j.aiopen.2021.08.002. URL: https://www.sciencedirect.com/science/article/pii/S2666651021000231.

[He+20] Pengcheng He et al. "Deberta: Decoding-enhanced bert with disentangled attention". In: *arXiv preprint arXiv:2006.03654* (2020).

[Ho+19] Jonathan Ho et al. *Axial Attention in Multidimensional Transformers*. 2019. arXiv: 1912.12180 [cs.CV].

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-Term Memory". In: *Neural Computation* 9.8 (Nov. 1997), pp. 1735–1780. ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735. eprint: https://direct.mit.edu/neco/article-pdf/9/8/1735/813796/neco.1997.9.8.1735.pdf. URL: https://doi.org/10.1162/neco.1997.9.8.1735.

[Hsi+23] Cheng-Yu Hsieh et al. *Distilling Step-by-Step! Outperforming Larger Language Models with Less Training Data and Smaller Model Sizes*. 2023. arXiv: 2305.02301 [cs.CL].

[Kat+20] Angelos Katharopoulos et al. "Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention". In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, 13–18 Jul 2020, pp. 5156–5165. URL: https://proceedings.mlr.press/v119/katharopoulos20a.html.

[KKL20] Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. "Reformer: The Efficient Transformer". In: *International Conference on Learning Representations*. 2020. URL: https://openreview.net/forum?id=rkgNKkHtvB.

[Lee+19] Juho Lee et al. "Set Transformer: A Framework for Attention-based Permutation-Invariant Neural Networks". In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, Sept. 2019, pp. 3744–3753. URL: https://proceedings.mlr.press/v97/lee19d.html.

[Lew+20] Mike Lewis et al. "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension". In: Jan. 2020, pp. 7871–7880. DOI: 10.18653/v1/2020.acl-main.703.

[Li+20] Xiaoya Li et al. "SAC: Accelerating and Structuring Self-Attention via Sparse Adaptive Connection". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 16997–17008. URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/c5c1bda1194f9423d744e0ef67df94ee-Paper.pdf.

[Lin+22] Tianyang Lin et al. "A survey of transformers". In: *AI Open* 3 (2022), pp. 111–132. ISSN: 2666-6510. DOI: https://doi.org/10.1016/j.aiopen.2022.10.001. URL: https://www.sciencedirect.com/science/article/pii/S2666651022000146.

[Liu+19] Yinhan Liu et al. *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. 2019. arXiv: 1907.11692 [cs.CL].

[Liu+18] Peter J. Liu* et al. "Generating Wikipedia by Summarizing Long Sequences". In: *International Conference on Learning Representations*. 2018. URL: https://openreview.net/forum?id=Hyg0vbWC-.

[Ma+21] Xuezhe Ma et al. *Luna: Linear Unified Nested Attention*. 2021. arXiv: 2106.01540 [cs.LG].

[Mik+13] Tomas Mikolov et al. *Efficient Estimation of Word Representations in Vector Space*. 2013. arXiv: 1301.3781 [cs.CL].

[Ngu+22] Tan Minh Nguyen et al. "Momentum Transformer: Closing the Performance Gap Between Self-attention and Its Linearization". In: *Proceedings of Mathematical and Scientific Machine Learning*. Ed. by Bin Dong et al. Vol. 190. Proceedings of Machine Learning Research. PMLR, 15–17 Aug 2022, pp. 189–204. URL: https://proceedings.mlr.press/v190/nguyen22a.html.

[Ope23] OpenAI. *GPT-4 Technical Report*. 2023. arXiv: 2303.08774 [cs.CL].

[Par+18] Niki Parmar et al. "Image Transformer". In: *International Conference on Machine Learning*. 2018.

[Pat+21] Mandela Patrick et al. "Keeping Your Eye on the Ball: Trajectory Attention in Video Transformers". In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato et al. Vol. 34. Curran Associates, Inc., 2021, pp. 12493–12506. URL: https://proceedings.neurips.cc/paper_files/paper/2021/file/67f7fb873eaf29526a11a9b7ac33bfac-Paper.pdf.

[Pen+21] Hao Peng et al. "Random Feature Attention". In: *International Conference on Learning Representations*. 2021. URL: https://openreview.net/forum?id=QtTKTdVrFBB.

[RN18] Alec Radford and Karthik Narasimhan. "Improving Language Understanding by Generative Pre-Training". In: 2018.

[Raf+20] Colin Raffel et al. "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer". In: *J. Mach. Learn. Res.* 21 (2020), 140:1–140:67. URL: http://jmlr.org/papers/v21/20-074.html.

[RT18] Alessandro Raganato and Jörg Tiedemann. "An Analysis of Encoder Representations in Transformer-Based Machine Translation". In: *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. Brussels, Belgium: Association for Computational Linguistics, Nov. 2018, pp. 287–297. DOI: 10.18653/v1/W18-5431. URL: https://aclanthology.org/W18-5431.

[Roy+21] Aurko Roy et al. "Efficient Content-Based Sparse Attention with Routing Transformers". In: *Transactions of the Association for Computational Linguistics* 9 (Feb. 2021), pp. 53–68. ISSN: 2307-387X. DOI: 10.1162/tacl_a_00353. eprint: https://direct.mit.edu/tacl/article-pdf/doi/10.1162/tacl_a_00353/1923932/tacl_a_00353.pdf. URL: https://doi.org/10.1162/tacl%5C_a%5C_00353.

[San+19] Victor Sanh et al. "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter". In: *ArXiv* abs/1910.01108 (2019).

[SIS21] Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. *Linear Transformers Are Secretly Fast Weight Programmers*. 2021. arXiv: 2102.11174 [cs.LG].

[SG05] Holger Schwenk and Jean-Luc Gauvain. "Training neural network language models on very large corpora". In: *Proceedings of human language technology conference and conference on empirical methods in natural language processing*. 2005, pp. 201–208.

[Tay+20] Yi Tay et al. "Sparse Sinkhorn Attention". In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, 13–18 Jul 2020, pp. 9438–9447. URL: https://proceedings.mlr.press/v119/tay20a.html.

[Tay+23] Yi Tay et al. *UL2: Unifying Language Learning Paradigms*. 2023. arXiv: 2205.05131 [cs.CL].

[Tou+23] Hugo Touvron et al. *LLaMA: Open and Efficient Foundation Language Models*. 2023. arXiv: 2302.13971 [cs.CL].

[Vas+17] Ashish Vaswani et al. "Attention is All You Need". In: 2017. URL: https://arxiv.org/pdf/1706.03762.pdf.

[Voi+19] Elena Voita et al. *Analyzing Multi-Head Self-Attention: Specialized Heads Do the Heavy Lifting, the Rest Can Be Pruned*. 2019. arXiv: 1905.09418 [cs.CL].

[VKF20] Apoorv Vyas, Angelos Katharopoulos, and François Fleuret. *Fast Transformers with Clustered Attention*. 2020. arXiv: 2007.04825 [cs.LG].

[Wan+20] Sinong Wang et al. *Linformer: Self-Attention with Linear Complexity*. 2020. arXiv: 2006.04768 [cs.LG].

[Wan+22] Xingmei Wang et al. "RACP: A network with attention corrected prototype for few-shot speaker recognition using indefinite distance metric". In: *Neurocomputing* 490 (2022), pp. 283–294. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2021.11.092. URL: https://www.sciencedirect.com/science/article/pii/S092523122101804X.

[Wei+23] Guangyong Wei et al. "LFEformer: Local Feature Enhancement Using Sliding Window With Deformability for Automatic Speech Recognition". In: *IEEE Signal Processing Letters* 30 (2023), pp. 180–184. DOI: 10.1109/LSP.2023.3241558.

[Wu+22] Haixu Wu et al. *Flowformer: Linearizing Transformers with Conservation Flows*. 2022. arXiv: 2202.06258 [cs.LG].

[Xu+15] Kelvin Xu et al. "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention". In: *Proceedings of the 32nd International Conference on Machine Learning*. Ed. by Francis Bach and David Blei. Vol. 37. Proceedings of Machine Learning Research. Lille, France: PMLR, July 2015, pp. 2048–2057. URL: https://proceedings.mlr.press/v37/xuc15.html.

[Yan+23] Jingfeng Yang et al. "Harnessing the Power of LLMs in Practice: A Survey on ChatGPT and Beyond". In: (2023). arXiv: 2304.13712 [cs.CL].

[Ye+19] Zihao Ye et al. *BP-Transformer: Modelling Long-Range Context via Binary Partitioning*. 2019. arXiv: 1911.04070 [cs.CL].

[Zah+21] Manzil Zaheer et al. *Big Bird: Transformers for Longer Sequences*. 2021. arXiv: 2007.14062 [cs.LG].

[ZE08] Omar Zaidan and Jason Eisner. "Modeling annotators: A generative approach to learning from annotator rationales". In: *Proceedings of the 2008 conference on Empirical methods in natural language processing*. 2008, pp. 31–40.

[Zen+22] Aohan Zeng et al. *GLM-130B: An Open Bilingual Pre-trained Model*. 2022. arXiv: 2210.02414 [cs.CL].

[Zha+22] Hang Zhang et al. *Poolingformer: Long Document Modeling with Pooling Attention*. 2022. arXiv: 2105.04371 [cs.CL].

[Zho+21] Haoyi Zhou et al. "Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting". In: 35 (May 2021), pp. 11106–11115. DOI: 10.1609/aaai.v35i12.17325. URL: https://ojs.aaai.org/index.php/AAAI/article/view/17325.

[Zho+23] Zhe Zhou et al. "Energon: Toward Efficient Acceleration of Transformers Using Dynamic Sparse Attention". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 42.1 (2023), pp. 136–149. DOI: 10.1109/TCAD.2022.3170848.

[ZZT22] Yimeng Zhuang, Jing Zhang, and Mei Tu. "Long-range Sequence Modeling with Predictable Sparse Attention". In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 234–243. DOI: 10.

18653/v1/2022.acl-long.19. URL: https://aclanthology.org/2022.acl-long.19.