Attention Mechanism in Neural Networks:
Where it Comes and Where it Goes

Derya Soydaner

Abstract A long time ago in the machine learning literature, the idea of incorporating a mechanism inspired by the human visual system into neural networks was introduced. This idea is named the *attention mechanism*, and it has gone through a long development period. Today, many works have been devoted to this idea in a variety of tasks. Remarkable performance has recently been demonstrated. The goal of this paper is to provide an overview from the early work on searching for ways to implement attention idea with neural networks until the recent trends. This review emphasizes the important milestones during this progress regarding different tasks. By this way, this study aims to provide a road map for researchers to explore the current development and get inspired for novel approaches beyond the attention.

Keywords Attention mechanism · Neural networks · Deep learning · Survey

1 Introduction

Human eye sees the world in an interesting way. We suppose as if we see the entire scene at once, but this is an illusion created by the subconscious part of our brain [1]. According to the *Scanpath* theory [2,3], when the human eye looks at an image, it can see only a small patch in high resolution. This small patch is called the *fovea*. It can see the rest of the image in low resolution which is called the *periphery*. To recognize the entire scene, the eye performs feature extraction based on the fovea. The eye is moved to different parts of the image until the information obtained from the fovea is sufficient for recognition [4]. These eye movements are called *saccades*. The eye makes successive fixations

Derya Soydaner
Department of Brain and Cognition, University of Leuven (KU Leuven), Leuven, Belgium
Tel.: +32-16710471
E-mail: derya.soydaner@kuleuven.be

until the recognition task is complete. This sequential process happens so quickly that we feel as if it happens all at once.

Biologically, this is called *visual attention system*. Visual attention is defined as the ability to dynamically restrict processing to a subset of the visual field [5]. It seeks answers for two main questions: *What* and *where* to look? Visual attention has been extensively studied in psychology and neuroscience; for reviews see [6,7,8,9,10]. Besides, there is a large amount of literature on modeling eye movements [11,12,13,14]. These studies have been a source of inspiration for many artificial intelligence tasks. It has been discovered that the attention idea is useful from image recognition to machine translation. Therefore, different types of attention mechanisms inspired from the human visual system have been developed for years. Since the success of deep neural networks has been at the forefront for these artificial intelligence tasks, these mechanisms have been integrated into neural networks for a long time.

This survey is about the journey of attention mechanisms used with neural networks. Researchers have been investigating ways to strengthen neural network architectures with attention mechanisms for many years. The primary aim of these studies is to reduce computational burden and to improve the model performance as well. Previous work reviewed the attention mechanisms from different perspectives [15], or examined them in context of natural language processing (NLP) [16,17]. However, in this study, we examine the development of attention mechanisms over the years, and recent trends. We begin with the first attempts to integrate the visual attention idea to neural networks, and continue until the most modern neural networks armed with attention mechanisms. One of them is the *Transformer*, which is used for many studies including the *GPT-3* language model [18], goes beyond convolutions and recurrence by replacing them with only attention layers [19]. Finally, we discuss how much more can we move forward, and what's next?

2 From the Late 1980s to Early 2010s: The Attention Awakens

The first attempts at adapting attention mechanisms to neural networks go back to the late 1980s. One of the early studies is the improved version of the *Neocognitron* [20] with selective attention [21]. This study is then modified to recognize and segment connected characters in cursive handwriting [22]. Another study describes *VISIT*, a novel model that concentrates on its relationship to a number of visual areas of the brain [5]. Also, a novel architecture named *Signal Channelling Attentional Network (SCAN)* is presented for attentional scanning [23].

Early work on improving the attention idea for neural networks includes a variety of tasks such as target detection [24]. In another study, a visual attention system extracts regions of interest by combining the bottom-up and top-down information from the image [25]. A recognition model based on selective attention which analyses only a small part of the image at each step, and combines results in time is described [4]. Besides, a model based on the

concept of selective tuning is proposed [26]. As the years go by, several studies that use the attention idea in different ways have been presented for visual perception and recognition [27, 28, 29, 30].

By the 2000s, the studies on making attention mechanisms more useful for neural networks continued. In the early years, a model that integrates an attentional orienting *where* pathway and an object recognition *what* pathway is presented [31]. A computational model of human eye movements is proposed for an object class detection task [32]. A serial model is presented for visual pattern recognition gathering Markov models and neural networks with selective attention on the handwritten digit recognition and face recognition problems [33]. In that study, a neural network analyses image parts and generates posterior probabilities as observations to the Markov model. Also, attention idea is used for object recognition [34], and the analysis of a scene [35]. An interesting study proposes to learn sequential attention in real-world visual object recognition using a Q-learner [36]. Besides, a computational model of visual selective attention is described to automatically detect the most relevant parts of a color picture displayed on a television screen [37]. The attention idea is also used for identifying and tracking objects in multi-resolution digital video of partially cluttered environments [38].

In 2010, the first implemented system inspired by the fovea of human retina was presented for image classification [39]. This system jointly trains a restricted Boltzmann machine (RBM) and an attentional component called *the fixation controller*. Similarly, a novel attentional model is implemented for simultaneous object tracking and recognition that is driven by gaze data [40]. By taking advantage of reinforcement learning, a novel recurrent neural network (RNN) is described for image classification [41]. *Deep Attention Selective Network (DasNet)*, a deep neural network with feedback connections that are learned through reinforcement learning to direct selective attention to certain features extracted from images, is presented [42]. Additionally, a deep learning based framework using attention has been proposed for generative modeling [43].

3 2015: The Rise of Attention

It can be said that 2015 is the golden year of attention mechanisms. Because the number of attention studies has grown like an avalanche after three main studies presented in that year. The first one proposed a novel approach for neural machine translation (NMT) [44]. As it is known, most of the NMT models belong to a family of encoder-decoders [45, 46], with an encoder and a decoder for each language. However, compressing all the necessary information of a source sentence into a fixed-length vector is an important disadvantage of this encoder-decoder approach. This usually makes it difficult for the neural network to capture all the semantic details of a very long sentence [1].

The idea that [44] introduced is an extension to the conventional NMT models. This extension is composed of an encoder and decoder as shown in
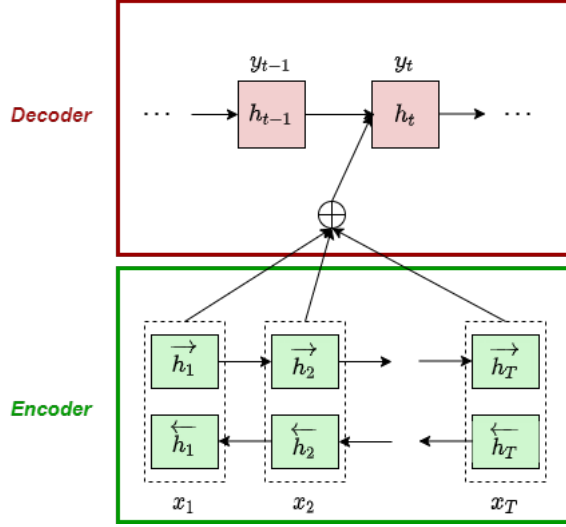
Fig. 1 The extension to the conventional NMT models that is proposed by [44]. It generates the t-th target word y_t given a source sentence $(x_1, x_2, ..., x_T)$.

Fig 1. The first part, encoder, is a *bidirectional RNN* (BiRNN) [47] that takes word vectors as input. The forward and backward states of BiRNN are computed. Then, an *annotation* a_j for each word x_j is obtained by concatenating these forward and backward hidden states. Thus, the encoder maps the input sentence to a sequence of annotations $(a_1, ..., a_{T_x})$. By using a BiRNN rather than conventional RNN, the annotation of each word can summarize both the preceding words and the following words. Besides, the annotation a_j can focus on the words around x_j because of the inherent nature of RNNs that representing recent inputs better.

In decoder, a weight α_{ij} of each annotation a_j is obtained by using its associated energy e_{ij} that is computed by a feedforward neural network f as in Eq. (1). This neural network f is defined as an *alignment model* that can be jointly trained with the proposed architecture. In order to reduce computational burden, a multilayer perceptron (MLP) with a single hidden layer is proposed as f. This alignment model tells us about the relation between the inputs around position j and the output at position i. By this way, the decoder applies an attention mechanism. As it is seen in Eq. (2), the α_{ij} is the output of softmax function:

$$e_{ij} = f(h_{i-1}, a_j) \tag{1}$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})} \tag{2}$$

Here, the probability α_{ij} determines the importance of annotation a_j with respect to the previous hidden state h_{i-1}. Finally, the context vector c_i is computed as a weighted sum of these annotations as follows [44]:

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} a_j \tag{3}$$

Based on the decoder state, the context and the last generated word, the target word y_t is predicted. In order to generate a word in a translation, the model searches for the most relevant information in the source sentence to concentrate. When it finds the appropriate source positions, it makes the prediction. By this way, the input sentence is encoded into a sequence of vectors and a subset of these vectors is selected adaptively by the decoder that is relevant to predicting the target [44]. Thus, it is no longer necessary to compress all the information of a source sentence into a fixed-length vector.

The second study is the first visual attention model in image captioning [48]. Different from the previous study [44], it uses a deep convolutional neural network (CNN) as an encoder. This architecture is an extension of the neural network [49] that encodes an image into a compact representation, followed by an RNN that generates a corresponding sentence. Here, the *annotation vectors* $a_i \in R^D$ are extracted from a lower convolutional layer, each of which is a D-dimensional representation corresponding to a part of the image. Thus, the decoder selectively focuses on certain parts of an image by weighting a subset of all the feature vectors [48]. This extended architecture uses attention for salient features to dynamically come to the forefront instead of compressing the entire image into a static representation.

The context vector c_t represents the relevant part of the input image at time t. The weight α_i of each annotation vector is computed similar to Eq. (2), whereas its associated energy is computed similar to Eq. (1) by using an MLP conditioned on the previous hidden state h_{t-1}. The remarkable point of this study is a new mechanism ϕ that computes c_t from the annotation vectors a_i corresponding to the features extracted at different image locations:

$$c_t = \phi(\{a_i\}, \{\alpha_i\}) \tag{4}$$

The definition of the ϕ function causes two variants of attention mechanisms: The *hard (stochastic)* attention mechanism is trainable by maximizing an approximate variational lower bound, i.e., by REINFORCE [50]. On the other side, the *soft (deterministic)* attention mechanism is trainable by standard backpropagation methods. The hard attention defines a location variable s_t, and uses it to decide where to focus attention when generating the *t-th* word. When the hard attention is applied, the attention locations are considered as intermediate latent variables. It assigns a multinoulli distribution parametrized by α_i, and c_t becomes a random variable. Here, $s_{t,i}$ is defined as a one-hot variable which is set to 1 if the *i-th* location is used to extract visual features [48]:

$$p(s_{t,i} = 1 | s_{j<t}, a) = \alpha_{t,i} \tag{5}$$

$$c_t = \sum_i s_{t,i} a_i \tag{6}$$

Whereas learning hard attention requires sampling the attention location s_t each time, the soft attention mechanism computes a weighted annotation vector similar to [44] and takes the expectation of the context vector c_t directly:

$$E_{p(s_t|\alpha)}[c_t] = \sum_{i=1}^{L} \alpha_{t,i} a_i \tag{7}$$

Furthermore, in training the deterministic version of the model, an alternative method namely *doubly stochastic attention*, is proposed with an additional constraint added to the training objective to encourage the model to pay equal attention to all parts of the image.

The third study should be emphasized presents two classes of attention mechanisms for NMT: the *global* attention that always attends to all source words, and the *local* attention that only looks at a subset of source words at a time [51]. These mechanisms derive the context vector c_t in different ways: Whereas the global attention considers all the hidden states of the encoder, the local one selectively focuses on a small window of context. In global attention, a variable-length alignment vector is derived similar to Eq. (2). Here, the current target hidden state h_t is compared with each source hidden state $\bar{h}_s$ by using a *score* function instead of the associated energy e_{ij}. Thus, the alignment vector whose size equals the number of time steps on the source side is derived. Given the alignment vector as weights, the context vector c_t is computed as the weighted average over all the source hidden states. Here, *score* is referred as a *content-based* function, and three different alternatives are considered [51].

On the other side, the local attention is differentiable. Firstly, an aligned position p_t is generated for each target word at a time t. Then, a window centered around the source position p_t is used to compute the context vector as a weighted average of the source hidden states within the window. The local attention selectively focuses on a small window of context, and obtains the alignment vector from the current target state h_t and the source states $\bar{h}_s$ in the window [51].

The introduction of these novel mechanisms in 2015 triggered the rise of attention for neural networks. Based on the proposed attention mechanisms, significant research has been conducted in a variety of tasks. In order to imagine the attention idea in neural networks better, two visual examples are shown in Fig. 2. A neural image caption generation task is seen in the top row that implements an attention mechanism [48]. Then, the second example shows how the attention mechanisms can be used for visual question answering [52]. Both examples demonstrate how attention mechanisms focus on parts of input images.
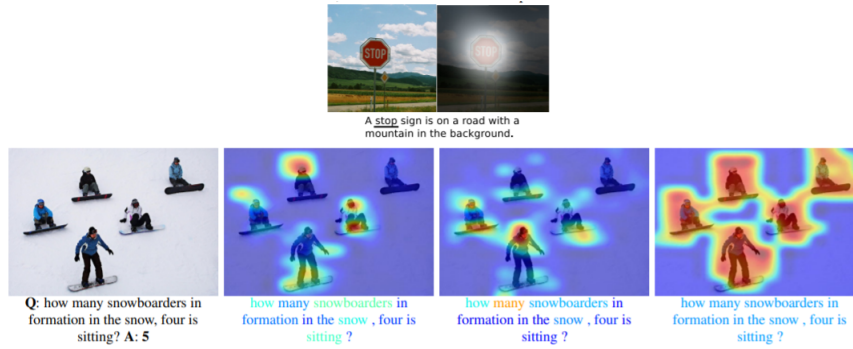
Fig. 2 Examples of the attention mechanism in visual. *(Top)* Attending to the correct object in neural image caption generation [48]. *(Bottom)* Visualization of original image and question pairs, and co-attention maps namely word-level, phrase-level and question-level, respectively [52].

4 2015-2016: Attack of the Attention

During two years from 2015, the attention mechanisms were used for different tasks, and novel neural network architectures were presented applying these mechanisms. After the *memory networks* [53] that require a supervision signal instructing them how to use their memory cells, the introduction of the *neural Turing machine* [54] allows end-to-end training without this supervision signal, via the use of a content-based soft attention mechanism [1]. Then, *end-to-end memory network* [55] that is a form of *memory network* based on a recurrent attention mechanism is proposed.

In these years, an attention mechanism called *self-attention*, sometimes called *intra-attention*, was successfully implemented within a neural network architecture namely *Long Short-Term Memory-Networks (LSTMN)* [56]. It modifies the standard LSTM structure by replacing the memory cell with a memory network [53]. This is because memory networks have a set of key vectors and a set of value vectors, whereas LSTMs maintain a hidden vector and a memory vector [56]. In contrast to attention idea in [44], memory and attention are added *within* a sequence encoder in LSTMN. In order to compute a representation of a sequence, self-attention is described as relating different positions of it [19]. One of the first approaches of self-attention is applied for natural language inference [57].

Many attention-based models have been proposed for neural image captioning [58], abstractive sentence summarization [59], speech recognition [60, 61], automatic video captioning [62], neural machine translation [63], and recognizing textual entailment [64]. Different attention-based models perform visual question answering [65,66,67]. An attention-based CNN is presented for modeling sentence pairs [68]. A recurrent soft attention based model learns to focus selectively on parts of the video frames and classifies videos [69].

On the other side, several neural network architectures have been presented in a variety of tasks. For instance, *Stacked Attention Network (SAN)*

is described for image question answering [70]. *Deep Attention Recurrent Q-Network (DARQN)* integrates soft and hard attention mechanisms into the structure of Deep Q-Network (DQN) [71]. *Wake-Sleep Recurrent Attention Model (WS-RAM)* speeds up the training time for image classification and caption generation tasks [72]. *alignDRAW* model, an extension of the *Deep Recurrent Attention Writer (DRAW)* [73], is a generative model of images from captions using a soft attention mechanism [74]. *Generative Adversarial What-Where Network (GAWWN)* synthesizes images given instructions describing what content to draw in which location [75].

5 The Transformer: Return of the Attention

After the proposed attention mechanisms in 2015, researchers published studies that mostly modifying or implementing them to different tasks. However, in 2017, a novel neural network architecture, namely the *Transformer*, based entirely on *self-attention* was presented [19]. The Transformer achieved great results on two machine translation tasks in addition to English constituency parsing. The most impressive point about this architecture is that it contains neither recurrence nor convolution. The Transformer performs well by replacing the conventional recurrent layers in encoder-decoder architecture used for NMT with self-attention.

The Transformer is composed of encoder-decoder stacks each of which has six identical layers within itself. In Fig. 3, one encoder-decoder stack is shown to illustrate the model [19]. Each stack includes only attention mechanisms and feedforward neural networks. As this architecture does not include any recurrent or convolutional layer, information about the relative or absolute positions in the input sequence is given at the beginning of both encoder and decoder using *positional encodings*.

The calculations of self-attention are slightly different from the mechanisms described so far in this paper. It uses three vectors namely *query*, *key* and *value* for each word. These vectors are computed by multiplying the input with weight matrices W_q, W_k and W_v which are learned during training. In general, each value is weighted by a function of the query with the corresponding key. The output is computed as a weighted sum of the values. Based on this idea, two attention mechanisms are proposed: In the first one, called *scaled dot-product attention*, the dot products of the query with all keys are computed as given in the right side of Fig. 3. Each result is divided to the square root of the dimension of the keys to have more stable gradients. They pass into the softmax function, thus the weights for the values are obtained. Finally each softmax score is multiplied with the value as given in Eq. (8). The authors propose computing the attention on a set of queries simultaneously by taking queries and keys of dimension d_k, and values of dimension d_v as inputs. The keys, queries and values are packed together into matrices K, Q and V. Finally, the output matrix is obtained as follows [19]:
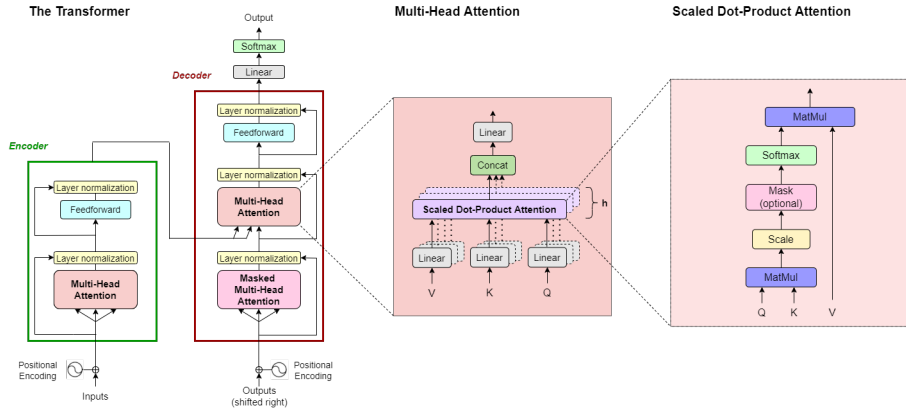
Fig. 3 The Transformer architecture and the attention mechanisms it uses in detail [19]. *(Left)* The Transformer with one encoder-decoder stack. *(Center)* Multi-head attention. *(Right)* Scaled dot-product attention.

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{8}$$

This calculation is performed by every word against the other words. This leads to having *values* of each word relative to each other. For instance, if the word x_2 is not relevant for the word x_1, then the softmax score gives low probability scores. As a result, the corresponding value is decreased. This leads to an increase in the value of relevant words, and those of others decrease. In the end, every word obtains a new value for itself.

As seen from Fig. 3, the Transformer model does not directly use scaled dot-product attention. But the attention mechanism it uses is based on these calculations. The second mechanism proposed, called the *multi-head attention*, linearly projects the queries, keys and values h times with different, learned linear projections to d_q, d_k and d_v dimensions, respectively [19]. The attention function is performed in parallel on each of these projected versions of queries, keys and values, i.e., *heads*. By this way, d_v-dimensional output values are obtained. In order to get the final values, they are concatenated and projected one last time as shown in the center of Fig. 3. By this way, the self-attention is calculated multiple times using different sets of query, key and value vectors. Thus, the model can jointly attend to information at different positions [19]:

$$MultiHead(Q, K, V) = Concat(head_1, ..., head_h)W^O \tag{9}$$
$$where\ head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$$

In the decoder part of the Transformer, *masked multi-head attention* is applied first to ensure that only previous word embeddings are used when trying to predict the next word in the sentence. Therefore, the embeddings that shouldn't be seen by the decoder are masked by multiplying with zero.

An interesting study examines the contribution made by individual attention heads in the encoder [76]. Also, there is an evaluation of the effects of self-attention on gradient propagation in recurrent networks [77]. For a deeper analysis of multi-head self-attention mechanism from a theoretical perspective see [78].

Self-attention has been used successfully in a variety of tasks including sentence embedding [79] and abstractive summarization [80]. It is shown that self-attention can lead to improvements to discriminative constituency parser [81], and speech recognition as well [82,83]. Also, the *listen-attend-spell* model [84] has been improved with the self-attention for acoustic modeling [85].

As soon as these self-attention mechanisms were proposed, they have been incorporated with deep neural networks for a wide range of tasks. For instance, a deep learning model learned a number of large-scale tasks from multiple domains with the aid of self-attention mechanism [86]. Novel self-attention neural models are proposed for cross-target stance classification [87] and NMT [88]. Another study points out that a fully self-attentional model can reach competitive predictive performance on ImageNet classification and COCO object detection tasks [89]. Besides, developing novel attention mechanisms has been carried out such as *area attention*, a novel mechanism that can be used along multi-head attention [90]. It attends to areas in the memory by defining the key of an area as the mean vector of the key of each item, and defining the value as the sum of all value vectors in the area.

When a novel mechanism is proposed, it is inevitable to incorporate it into the GAN framework [91]. *Self-Attention Generative Adversarial Networks (SAGANs)* [92] introduce a self-attention mechanism into convolutional GANs. Different from the traditional convolutional GANs, SAGAN generates high-resolution details using cues from all feature locations. Similarly, *Attentional Generative Adversarial Network (AttnGAN)* is presented for text to image generation [93]. On the other side, a machine reading and question answering architecture called *QANet* [94] is proposed without any recurrent networks. It uses self-attention to learn the global interaction between each pair of words whereas convolution captures the local structure of the text. In another study, *Gated Attention Network (GaAN)* controls the importance of each attention head's output by introducing gates [95]. Another interesting study introduces *attentive group convolutions* with a generalization of visual self-attention [96]. A deep transformer model is implemented for language modeling over long sequences [97].

5.1 Self-attention variants

In recent years, self-attention has become an important research direction within the deep learning community. Self-attention idea has been examined in different aspects. For example, self-attention is handled in a multi-instance learning framework [98]. The idea of *Sparse Adaptive Connection (SAC)* is presented for accelerating and structuring self-attention [99]. The research on

Table 1 Summary of Notation

Symbol	Definition
a	annotation
c	context vector
α	weight
e	energy
f	feedforward neural network
h	hidden state
ϕ	hard (stochastic) / soft (deterministic) attention
s	location variable
p	source position
K, Q, V	keys, queries and values matrices, respectively
W_q, W_k, W_v	weight matrices for queries, keys and values, respectively

improving self-attention continues as well [100, 101, 102]. Besides, based on the self-attention mechanisms proposed in the Transformer, important studies that modify the self-attention have been presented. Some of the most recent and prominent studies are summarized below.

Relation-aware self-attention It extends the self-attention mechanism by regarding representations of the relative positions, or distances between sequence elements [103]. Thus, it can consider the pairwise relationships between input elements. This type of attention mechanism defines vectors to represent the edge between two inputs. It provides learning two distinct edge representations that can be shared across attention heads without requiring additional linear transformations.

Directional self-attention (DiSA) A novel neural network architecture for learning sentence embedding named *Directional Self-Attention Network (DiSAN)* [104] uses *directional* self-attention followed by a *multi-dimensional* attention mechanism. Instead of computing a single importance score for each word based on the word embedding, multi-dimensional attention computes a feature-wise score vector for each token. To extend this mechanism to the self-attention, two variants are presented: The first one, called *multi-dimensional 'token2token' self-attention* generates context-aware coding for each element. The second one, called *multi-dimensional 'source2token' self-attention* compresses the sequence into a vector [104]. On the other side, directional self-attention produces context-aware representations with temporal information encoded by using positional masks. By this way, directional information is encoded. First, the input sequence is transformed to a sequence of hidden states by a fully connected layer. Then, multi-dimensional token2token self-attention is applied to these hidden states. Hence, context-aware vector representations are generated for all elements from the input sequence.

Reinforced self-attention (ReSA) A sentence-encoding model named *Reinforced Self-Attention Network (ReSAN)* uses *reinforced self-attention (ReSA)* that integrates soft and hard attention mechanisms into a single model. ReSA

selects a subset of head tokens, and relates each head token to a small subset of dependent tokens to generate their context-aware representations [105]. For this purpose, a novel hard attention mechanism called *reinforced sequence sampling (RSS)*, which selects tokens from an input sequence in parallel and trained via policy gradient, is proposed. Given an input sequence, RSS generates an equal-length sequence of binary random variables that indicates both the selected and discarded ones. On the other side, the soft attention provides reward signals back for training the hard attention. The proposed RSS provides a sparse mask to self-attention. ReSA uses two RSS modules to extract the sparse dependencies between each pair of selected tokens.

Outer product attention (OPA) *Self-Attentive Associative Memory (SAM)* is a novel operator based upon *outer product attention (OPA)* [106]. This attention mechanism is an extension of dot-product attention [19]. OPA differs using element-wise multiplication, outer product, and *tanh* function instead of *softmax*.

Bidirectional block self-attention (Bi-BloSA) Another mechanism, *bidirectional block self-attention (Bi-BloSA)* which is simply a *masked block self-attention (mBloSA)* with forward and backward masks to encode the temporal order information is presented [107]. Here, mBloSA is composed of three parts from its bottom to top namely *intra-block self-attention*, *inter-block self-attention* and *the context fusion*. It splits a sequence into several length-equal blocks, and applies an intra-block self-attention to each block independently. Then, inter-block self-attention processes the outputs for all blocks. This stacked self-attention model results a reduction in the amount of memory compared to a single one applied to the whole sequence. Finally, a feature fusion gate combines the outputs of intra-block and inter-block self-attention with the original input, to produce the final context-aware representations of all tokens.

Fixed multi-head attention The *fixed multi-head attention* proposes fixing the head size of the Transformer in the aim of improving the representation power [108]. This study emphasizes its importance by setting the head size of attention units to input sequence length.

Sparse sinkhorn attention It is based on the idea of differentiable sorting of internal representations *within* the self-attention module [109]. Instead of allowing tokens to only attend to tokens within the same block, it operates on block sorted sequences. Each token attends to tokens in the *sorted* block. Thus, tokens that may be far apart in the unsorted sequence can be considered. Additionally, a variant of this mechanism named *SortCut sinkhorn attention* applies a post-sorting truncation of the input sequence.

Adaptive attention span *Adaptive attention span* is proposed as an alternative to self-attention [110]. It learns the attention span of each head independently. To this end, a masking function inspired by [111] is used to control the attention span for each head. The purpose of this novel mechanism is to reduce the computational burden of the Transformer. Additionally, *dynamic attention span* approach is presented to dynamically change the attention span based on the current input as an extension [51,112].

5.2 Transformer variants

Different from developing novel self-attention mechanisms, several studies have been published in the aim of improving the performance of the Transformer. These studies mostly modify the model architecture. For instance, an additional recurrence encoder is preferred to model recurrence for Transformer directly [113]. In another study, a new weight initialization scheme is applied to improve Transformer optimization [114]. A novel positional encoding scheme is used to extend the Transformer to tree-structured data [115]. Investigating model size by handling Transformer width and depth for efficient training is also an active research area [116]. Transformer is used in reinforcement learning settings [117,118,119] and for time series forecasting in adversarial training setting [120].

Besides, many Transformer variants have been presented in the recent past. *COMmonsEnse Transformer (COMET)* is introduced for automatic construction of commonsense knowledge bases [121]. *Evolved Transformer* applies neural architecture search for a better Transformer model [122]. *Transformer Autoencoder* is a sequential autoencoder for conditional music generation [123]. *CrossTransformer* takes a small number of labeled images and an unlabeled query, and computes distances between spatially-corresponding features to infer class membership [124]. *DEtection TRansformer (DETR)* is a new design for object detection systems [125], and *Deformable DETR* is an improved version that achieves better performance in less time [126]. *FLOw-bAsed TransformER (FLOATER)* emphasizes the importance of position encoding in the Transformer, and models the position information via a continuous dynamical model [127]. *Disentangled Context (DisCo) Transformer* simultaneously generates all tokens given different contexts by predicting every word in a sentence conditioned on an arbitrary subset of the rest of the words [128]. *Generative Adversarial Transformer (GANsformer)* is presented for visual generative modeling [129].

Recent work has demonstrated significant performance on NLP tasks. In *OpenAI GPT*, there is a left-to-right architecture, where every token can only attend to previous tokens in the self-attention layers of the Transformer [130]. *GPT-2* [131] and *GPT-3* [18] models have improved the progress. In addition to these variants, some prominent Transformer-based models are summarized below.

Universal Transformer A generalization of the Transformer model named the *Universal Transformer* [132] iteratively computes representations H^t at step t for all positions in the sequence in parallel. To this end, it uses the scaled dot-product attention in Eq. (8) where d is the number of columns of Q, K and V. In the Universal Transformer, the multi-head self-attention with k heads is used. The representations H^t is mapped to queries, keys and values with affine projections using learned parameter matrices $W^Q \in \Re^{d \times d/k}$, $W^K \in \Re^{d \times d/k}$, $W^V \in \Re^{d \times d/k}$ and $W^O \in \Re^{d \times d}$ [132]:

$$MultiHead(H^t) = Concat(head_1, ..., head_k)W^O \qquad (10)$$
$$where \; head_i = Attention(H^t W_i^Q, H^t W_i^K, H^t W_i^V)$$

Image Transformer *Image Transformer* [133] demonstrates that self-attention based models can also be well-suited for images instead of text. This Transformer type restricts the self-attention mechanism to attend to local neighborhoods. Thus, the size of images that the model can process is increased. Its larger receptive fields allow the Image Transformer to significantly improve the model performance on image generation as well as image super-resolution.

Transformer-XL This study aims to improve the fixed-length context of the Transformer [19] for language modeling. *Transformer-XL* [134] makes modeling very long-term dependency possible by reusing the hidden states obtained in previous segments. Hence, information can be propagated through the recurrent connections. In order to reuse the hidden states without causing temporal confusion, Transformer-XL uses relative positional encodings. Based on this architecture, a modified version named *the Gated Transformer-XL (GTrXL)* is presented in the reinforcement learning setting [135].

Tensorized Transformer *Tensorized Transformer* [136] compresses the multi-head attention in Transformer. To this end, it uses a novel self-attention model *multi-linear attention* with Block-Term Tensor Decomposition (BTD) [137]. It builds a *single-block attention* based on the Tucker decomposition [138]. Then, it uses a multi-linear attention constructed by a BTD to compress the multi-head attention mechanism. In Tensorized Transformer, the factor matrices are shared across multiple blocks.

BERT The **B**idirectional **E**ncoder **R**epresentations from **T**ransformers *(BERT)* aims to pre-train deep bidirectional representations from unlabeled text [139]. BERT uses a multilayer bidirectional Transformer as the encoder. Besides, inspired by the Cloze task [140], it has a *masked language model* pre-training objective. BERT randomly masks some of the tokens from the input, and predicts the original vocabulary id of the masked word based only on its context. This model can pre-train a deep bidirectional Transformer. In all layers, the

pre-training is carried out by jointly conditioning on both left and right context. BERT differs from the left-to-right language model pre-training from this aspect.

Recently, BERT model has been examined in detail. For instance, the behaviour of attention heads are analysed [141]. Various methods have been investigated for compressing [142, 143], pruning [144], and quantization [145]. Also, BERT model has been considered for different tasks such as coreference resolution [146]. A novel method is proposed in order to accelerate BERT training [147].

Furthermore, various BERT variants have been presented. *ALBERT* aims to increase the training speed of BERT, and presents two parameter reduction techniques [148]. Similarly, *PoWER-BERT* [149] is developed to improve the inference time of BERT. This scheme is also used to accelerate ALBERT. Also, *TinyBERT* is proposed to accelerate inference and reduce model size while maintaining accuracy [150]. In order to obtain better representations, *SpanBERT* is proposed as a pre-training method [151]. As a robustly optimized BERT approach, *RoBERTa* shows that BERT was significantly undertrained [152]. Also, *DeBERTa* improves RoBERTa using the disentangled attention mechanism [153]. On the other side, *DistilBERT* shows that it is possible to reach similar performances using much smaller language models pre-trained with knowledge distillation [154]. *StructBERT* proposes two novel linearization strategies [155]. *Q-BERT* is introduced for quantizing BERT models [156], *BioBERT* is for biomedical text mining [157], and *RareBERT* is for rare disease diagnosis [158].

Since 2017 when the Transformer was presented, research directions have generally focused on novel self-attention mechanisms, adapting the Transformer for various tasks, or making them more understandable. In one of the most recent studies, NLP becomes possible in the mobile setting with *Lite Transformer*. It applies *long-short range attention* where some heads specialize in the local context modeling while the others specialize in the long-distance relationship modeling [159]. A deep and light-weight Transformer *DeLighT* [160] and a hypernetwork-based model namely *HyperGrid Transformers* [161] perform with fewer parameters. *Graph Transformer Network* is introduced for learning node representations on heterogeneous graphs [162] and different applications are performed for molecular data [163] or textual graph representation [164]. Also, *Transformer-XH* applies *eXtra Hop attention* for structured text data [165]. *AttentionXML* is a tree-based model for extreme multi-label text classification [166]. Besides, attention mechanism is handled in a Bayesian framework [167]. For a better understanding of Transformers, an identifiability analysis of self-attention weights is conducted in addition to presenting *effective attention* to improve explanatory interpretations [168]. Lastly, *Vision Transformer (ViT)* processes an image using a standard Transformer encoder as used in NLP by interpreting it as a sequence of patches, and performs well on image classification tasks [169].

5.3 What about complexity?

All these aforementioned studies undoubtedly demonstrate significant success. But success not make one great. The Transformer also brings a very high computational complexity and memory cost. The necessity of storing attention matrix to compute the gradients with respect to queries, keys and values causes a non-negligible quadratic computation and memory requirements. Training the Transformer is a slow process for very long sequences because of its quadratic complexity. There is also time complexity which is quadratic with respect to the sequence length. In order to improve the Transformer in this respect, recent studies have been conducted to improve this issue. One of them is *Linear Transformer* which expresses the self-attention as a linear dot-product of kernel feature maps [170]. Linear Transformer reduces both memory and time complexity by changing the self-attention from the softmax function in Eq. (8) to a feature map based dot-product attention. Its performance is competitive with the vanilla Transformer architecture on image generation and automatic speech recognition tasks while being faster during inference. On the other side, *FMMformers* which use the idea of the *fast multipole method (FMM)* [171] outperform the linear Transformer by decomposing the attention matrix into near-field and far-field attention with linear time and memory complexity [172].

Another suggestion made in response to the Transformer's quadratic nature is The *Reformer* that replaces dot-product attention by one that uses *locality-sensitive hashing* [173]. It reduces the complexity but one limitation of the Reformer is its requirement for the queries and keys to be identical. *Set Transformer* aims to reduce computation time of self-attention from quadratic to linear by using an attention mechanism based on sparse Gaussian process literature [174]. *Routing Transformer* aims to reduce the overall complexity of attention by learning dynamic sparse attention patterns by using *routing attention with clustering* [175]. It applies k-means clustering to model sparse attention matrices. At first, queries and keys are assigned to clusters. The attention scheme is determined by considering only queries and keys from the same cluster. Thus, queries are routed to keys belonging to the same cluster [175].

Sparse Transformer introduces sparse factorizations of the attention matrix by using *factorized self-attention*, and avoids the quadratic growth of computational burden [176]. It also shows the possibility of modeling sequences of length one million or more by using self-attention in theory. In the Transformer, all the attention heads with the softmax attention assign a non-zero weight to all context words. *Adaptively Sparse Transformer* replaces softmax with α-entmax which is a differentiable generalization of softmax allowing low-scoring words to receive precisely zero weight [177]. By means of context-dependent sparsity patterns, the attention heads become flexible in the Adaptively Sparse Transformer. *Random feature attention* approximates softmax attention with random feature methods [178]. *Skyformer* replaces softmax with a Gaussian kernel and adapts Nyström method [179]. A sparse atten-

tion mechanism named *BIGBIRD* aims to reduce the quadratic dependency of Transformer-based models to linear [180]. Different from the similar studies, BIGBIRD performs well for genomics data alongside NLP tasks such as question answering.

Music Transformer [181] shows that self-attention can also be useful for modeling music. This study emphasizes the infeasibility of the relative position representations introduced by [103] for long sequences because of the quadratic intermediate relative information in the sequence length. Therefore, this study presents an extended version of relative attention named *relative local attention* that improves the relative attention for longer musical compositions by reducing its intermediate memory requirement to linear in the sequence length. A softmax-free Transformer (*SOFT*) is presented to improve the computational efficiency of ViT. It uses Gaussian kernel function instead of the dot-product similarity [182].

Additionally, various approaches have been presented in *Hierarchical Visual Transformer* [183], *Long-Short Transformer (Transformer-LS)* [184], *Perceiver* [185], and *Performer* [186]. Image Transformer based on the cross-covariance matrix between keys and queries is applied [187], and a new vision Transformer is proposed [188]. Furthermore, a Bernoulli sampling attention mechanism decreases the quadratic complexity to linear [189]. A novel linearized attention mechanism performs well on object detection, instance segmentation, and stereo depth estimation [190]. A study shows that kernelized attention with relative positional encoding can be calculated using Fast Fourier Transform and it leads to get rid of the quadratic complexity for long sequences [191]. A linear unified nested attention mechanism namely *Luna* uses two nested attention functions to approximate the softmax attention in Transformer to achieve linear time and space complexity [192].

6 Concluding Remarks: A New Hope

Inspired by the human visual system, the attention mechanisms in neural networks have been developing for a long time. In this study, we examine this duration beginning with its roots up to the present time. Some mechanisms have been modified, or novel mechanisms have emerged in this period. Today, this journey has reached a very important stage. The idea of incorporating attention mechanisms into deep neural networks has led to state-of-the-art results for a large variety of tasks. Self-attention mechanisms and *GPT-n* family models have become a new hope for more advanced models. These promising progress bring the questions whether the attention could help further development, replace the popular neural network layers, or could be a better idea than the existing attention mechanisms? It is still an active research area and much to learn we still have, but it is obvious that more powerful systems are awaiting when neural networks and attention mechanisms join forces.

Conflict of interest

The author declares that she has no conflict of interest.

References

1. I. Goodfellow, Y. Bengio, A. Courville, The MIT Press (2016)
2. D. Noton, L. Stark, Scientific American **224(6)**, 34 (1971)
3. D. Noton, L. Stark, Vision Research **11**, 929 (1971)
4. E. Alpaydın, Advances in Neural Information Processing Systems 8 pp. 771–777 (1995)
5. S. Ahmad, Advances in Neural Information Processing Systems 4 pp. 420–427 (1991)
6. M. Posner, S. Petersen, Annual Review of Neuroscience **13(1)**, 25 (1990)
7. C. Bundesen, Psychological Review **97(4)**, 523 (1990)
8. R. Desimone, J. Duncan, Annual Review of Neuroscience **18(1)**, 193 (1995)
9. M. Corbetta, G. Shulman, Nature Reviews Neuroscience **3(3)**, 201 (2002)
10. S. Petersen, M. Posner, Annual Review of Neuroscience **35**, 73 (2012)
11. R. Rimey, C. Brown, Technical Report, University of Rochester (1990)
12. B. Sheliga, L. Riggio, G. Rizzolatti, Experimental Brain Research **98(3)**, 507 (1994)
13. B. Sheliga, L. Riggio, G. Rizzolatti, Experimental Brain Research **105(2)**, 261 (1995)
14. J. Hoffman, B. Subramaniam, Perception and Psychophysics **57(6)**, 787 (1995)
15. S. Chaudhari, et al., ACM Transactions on Intelligent Systems and Technology (TIST) pp. 1–32 (2021)
16. A. Galassi, et al., IEEE Transactions on Neural Networks and Learning Systems (2020)
17. J. Lee, et al., ACM Transactions on Knowledge Discovery from Data (TKDD) **13(6)**, 1 (2019)
18. T. Brown, et al., Advances in Neural Information Processing Systems 33 pp. 1877–1901 (2020)
19. A. Vaswani, et al., Advances in Neural Information Processing Systems 30 pp. 5998–6008 (2017)
20. K. Fukushima, Biological Cybernetics **36**, 193 (1980)
21. K. Fukushima, Applied Optics **26(23)**, 4985 (1987)
22. K. Fukushima, T. Imagawa, Neural Networks **6(1)**, 33 (1993)
23. E. Postma, H.V. den Herik, P. Hudson, Neural Networks **10(6)**, 993 (1997)
24. J. Schmidhuber, R. Huber, International Journal of Neural Systems pp. 125–134 (1991)
25. R. Milanese, et al., IEEE Computer Society Conference on Computer Vision and Pattern Recoginition, Seattle, WA, USA pp. 781–785 (1994)
26. J. Tsotsos, et al., Artificial Intelligence **78(1-2)**, 507 (1995)
27. S. Culhane, J. Tsotsos, Proceedings of the 11th IAPR International Conference on Pattern Recognition, The Hague, Netherlands pp. 36–40 (1992)
28. D. Reisfeld, H. Wolfson, Y. Yeshurun, International Journal of Computer Vision **14(2)**, 119 (1995)
29. I. Rybak, et al., Vision Research **38(15-16)**, 2387 (1998)
30. J. Keller, et al., Pattern Analysis and Applications **2(3)** (1999)
31. F. Miau, L. Itti, Proceedings of the 23rd Annual International Conference of the IEEE Engineering in Medicine and Biology Society, Istanbul, Turkey pp. 789–792 (2001)
32. W. Zhang, et al., Advances in Neural Information Processing Systems 19 pp. 1609–1616 (2006)
33. A. Salah, E. Alpaydın, L. Akarun, IEEE Transactions on Pattern Analysis and Machine Intelligence **24(3)**, 420 (2002)
34. D. Walther, et al., International Workshop on Biologically Motivated Computer Vision, Springer, Berlin, Heidelberg pp. 472–479 (2002)
35. K. Schill, et al., Journal of Electronic Imaging **10(1)**, 152 (2001)
36. L. Paletta, G. Fritz, C. Seifert, International Conference on Machine Learning (2005)
37. O.L. Meur, et al., IEEE Transactions on Pattern Analysis and Machine Intelligence **28(5)**, 802– (2006)

38. S. Gould, et al., International Joint Conference on Artificial Intelligence (IJCAI) pp. 2115–2121 (2007)
39. H. Larochelle, G. Hinton, Advances in Neural Information Processing Systems 23 pp. 1243–1251 (2010)
40. L. Bazzani, et al., International Conference on Machine Learning (2011)
41. V. Mnih, et al., Advances in Neural Information Processing Systems 27 pp. 2204–2212 (2014)
42. M. Stollenga, et al., Advances in Neural Information Processing Systems 27 pp. 3545–3553 (2014)
43. Y. Tang, N. Srivastava, R. Salakhutdinov, Advances in Neural Information Processing Systems 27 (2014)
44. D. Bahdanau, K. Cho, Y. Bengio, International Conference on Learning Representations (2015)
45. I. Sutskever, O. Vinyals, Q. Le, Advances in Neural Information Processing Systems 27 pp. 3104–3112 (2014)
46. K. Cho, et al., Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) pp. 1724–1734 (2014)
47. M. Schuster, K. Paliwal, IEEE Transactions on Signal Processing **45(11)**, 2673 (1997)
48. K. Xu, et al., International Conference on Machine Learning pp. 2048–2057 (2015)
49. O. Vinyals, et al., In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition pp. 3156–3164 (2015)
50. R. Williams, Machine Learning **8(3-4)**, 229 (1992)
51. M.T. Luong, H.P..C. Manning, Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal pp. 1412–1421 (2015)
52. J. Lu, et al., Advances in Neural Information Processing Systems 29 (2016)
53. J. Weston, S. Chopra, A. Bordes, International Conference on Learning Representations (2014)
54. A. Graves, G. Wayne, I. Danihelka, arXiv preprint arXiv:1410.5401 (2014)
55. S. Sukhbaatar, et al., Advances in Neural Information Processing Systems 28 pp. 2440–2448 (2015)
56. J. Cheng, L. Dong, M. Lapata, Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing pp. 551–561 (2016)
57. A. Parikh, et al., Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Austin, Texas pp. 2249–2255 (2016)
58. Q. You, et al., In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV pp. 4651–4659 (2016)
59. A. Rush, S. Chopra, J. Weston, Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal pp. 379–389 (2015)
60. D. Yu, et al., Interspeech pp. 17–21 (2016)
61. J. Chorowski, et al., Advances in Neural Information Processing Systems 28 pp. 577–585 (2015)
62. M. Zanfir, E. Marinoiu, C. Sminchisescu, In Asian Conference on Computer Vision, Springer, Cham pp. 104—-119 (2016)
63. Y. Cheng, et al., Proceedings of the 25th International Joint Conference on Artificial Intelligence (2016)
64. T. Rockt International Conference on Learning Representations (2016)
65. Y. Zhu, et al., Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition pp. 4995–5004 (2016)
66. K. Chen, et al., arXiv preprint arXiv:1511.05960 (2015)
67. H. Xu, K. Saenko, In European Conference on Computer Vision pp. 451–466 (2016)
68. W. Yin, et al., Transactions of the Association for Computational Linguistics **4**, 259 (2016)
69. S. Sharma, R. Kiros, R. Salakhutdinov, International Conference on Learning Representations (2016)
70. Z. Yang, et al., In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition pp. 21–29 (2016)
71. I. Sorokin, et al., arXiv preprint arXiv:1512.01693 (2015)
72. J. Ba, et al., Advances in Neural Information Processing Systems 28 pp. 2593–2601 (2015)

73. K. Gregor, et al., International Conference on Machine Learning pp. 1462–1471 (2015)
74. E. Mansimov, et al., International Conference on Learning Representations (2016)
75. S. Reed, et al., Advances in Neural Information Processing Systems 29 pp. 217–225 (2016)
76. E. Voita, et al., In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy pp. 5797–5808 (2019)
77. G. Kerg, et al., Advances in Neural Information Processing Systems 33 (2020)
78. J.B. Cordonnier, A. Loukas, M. Jaggi, International Conference on Learning Representations (2020)
79. Z. Lin, et al., International Conference on Learning Representations (2017)
80. R. Paulus, C. Xiong, R. Socher, International Conference on Learning Representations (2018)
81. N. Kitaev, D. Klein, In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Long papers) pp. 2676–2686 (2018)
82. D. Povey, et al., IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE pp. 5874–5878 (2018)
83. A. Vyas, et al., Advances in Neural Information Processing Systems 33 (2020)
84. W. Chan, et al., IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Shanghai pp. 4960—-4964 (2016)
85. M. Sperber, et al., In proceedings of Annual Conference of the International Speech Communication Association (InterSpeech) pp. 3723–3727 (2018)
86. L. Kaiser, et al., arXiv preprint arXiv:1706.05137 (2017)
87. C. Xu, et al., Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Short papers), Melbourne, Australia pp. 778–783 (2018)
88. S. Maruf, A. Martins, G. Haffari, Proceedings of NAACL-HLT, Minneapolis, Minnesota pp. 3092–3102 (2019)
89. P. Ramachandran, et al., Advances in Neural Information Processing Systems 32 pp. 68–80 (2019)
90. Y. Li, et al., International Conference on Machine Learning (2019)
91. I. Goodfellow, et al., Advances in Neural Information Processing Systems 27 pp. 2672–2680 (2014)
92. H. Zhang, et al., International Conference on Machine Learning pp. 7354–7363 (2019)
93. T. Xu, et al., In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) pp. 1316–1324 (2018)
94. A. Yu, et al., International Conference on Learning Representations (2018)
95. J. Zhang, et al., Conference on Uncertainty in Artificial Intelligence (2018)
96. D. Romero, et al., International Conference on Machine Learning (2020)
97. R. Al-Rfou, et al., AAAI Conference on Artificial Intelligence **33**, 3159 (2019)
98. J. Du, et al., Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing pp. 2216–2225 (2018)
99. X. Li, et al., Advances in Neural Information Processing Systems 33 (2020)
100. B. Yang, et al., AAAI Conference on Artificial Intelligence **33**, 387 (2019)
101. B. Yang, et al., Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium pp. 4449–4458 (2018)
102. Proceedings of the IEEE International Conference on Computer Vision pp. 3286–3295
103. P. Shaw, J. Uszkoreit, A. Vaswani, Proceedings of NAACL-HLT, New Orleans, Louisiana pp. 464–468 (2018)
104. T. Shen, et al., AAAI Conference on Artificial Intelligence pp. 5446–5455 (2018)
105. T. Shen, et al., In Proceedings of the 27th International Joint Conference on Artificial Intelligence, (IJCAI-18) pp. 4345–4352 (2018)
106. H. Le, T. Tran, S. Venkatesh, International Conference on Machine Learning (2020)
107. T. Shen, et al., International Conference on Learning Representations (2018)
108. S. Bhojanapalli, et al., International Conference on Machine Learning (2020)
109. Y. Tay, et al., International Conference on Machine Learning (2020)
110. S. Sukhbaatar, et al., Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy pp. 331–335 (2019)
111. Y. Jernite, et al., International Conference on Learning Representations (2017)
112. R. Shu, H. Nakayama, In Proceedings of the First Workshop on Neural Machine Translation, Vancouver, Canada pp. 1–10 (2017)

113. J. Hao, et al., Proceedings of NAACL-HLT, Minneapolis, Minnesota pp. 1198–1207 (2019)
114. X. Huang, et al., International Conference on Machine Learning (2020)
115. V. Shiv, C. Quirk, Advances in Neural Information Processing Systems 32 pp. 12,081–12,091 (2019)
116. Z. Li, et al., International Conference on Machine Learning (2020)
117. Y. Hoshen, Advances in Neural Information Processing Systems 30, Long Beach, CA, USA (2017)
118. S. Hu, et al., International Conference on Learning Representations (2021)
119. E. Parisotto, R. Salakhutdinov, International Conference on Learning Representations (2021)
120. S. Wu, et al., Advances in Neural Information Processing Systems 33 (2020)
121. A. Bosselut, et al., Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (2019)
122. D. So, C. Liang, Q. Le, International Conference on Machine Learning (2019)
123. K. Choi, et al., International Conference on Machine Learning (2020)
124. C. Doersch, A. Gupta, A. Zisserman, Advances in Neural Information Processing Systems 33 pp. 21,981–21,993 (2020)
125. N. Carion, et al., European Conference on Computer Vision pp. 213—-229 (2020)
126. X. Zhu, et al., International Conference on Learning Representations (2021)
127. X. Liu, et al., International Conference on Machine Learning pp. 6327–6335 (2020)
128. J. Kasai, et al., International Conference on Machine Learning (2020)
129. D. Hudson, L. Zitnick, International Conference on Machine Learning pp. 4487–4499 (2021)
130. A. Radford, et al., Technical Report, OpenAI (2018)
131. A. Radford, et al., OpenAI blog p. 9 (2019)
132. M. Dehghani, et al., International Conference on Learning Representations (2019)
133. N. Parmar, International Conference on Machine Learning (2018)
134. Z. Dai, et al., Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics pp. 2978–2988 (2019)
135. E. Parisotto, International Conference on Machine Learning (2020)
136. X. Ma, et al., Advances in Neural Information Processing Systems 32 pp. 2232–2242 (2019)
137. L. Lathauwer, SIAM Journal on Matrix Analysis and Applications **30(3)**, 1033 (2008)
138. L. Tucker, Psychometrika **31(3)**, 279 (1966)
139. J. Devlin, et al., Proceedings of NAACL-HLT 2019 pp. 4171–4186 (2019)
140. W. Taylor, Journalism Bulletin **30(4)**, 415 (1953)
141. K. Clark, et al., arXiv preprint arXiv:1906.04341 (2019)
142. S. Sun, et al., Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, Hong Kong, China pp. 4323–4332 (2019)
143. W. Wang, et al., Advances in Neural Information Processing Systems 33 (2020)
144. J. McCarley, R. Chakravarti, A. Sil, arXiv preprint arXiv:1910.06360 (2020)
145. O. Zafrir, et al., The 5th Workshop on Energy Efficient Machine Learning and Cognitive Computing - NeurIPS (2019)
146. M. Joshi, et al., In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing pp. 5803–5808 (2019)
147. L. Gong, et al., International Conference on Machine Learning pp. 2337–2346 (2019)
148. Z. Lan, et al., International Conference on Learning Representations (2020)
149. S. Goyal, et al., International Conference on Machine Learning (2020)
150. X. Jiao, et al., arXiv preprint arXiv:1909.10351 (2019)
151. M. Joshi, et al., Transactions of the Association for Computational Linguistics **8**, 64 (2020)
152. Y. Liu, et al., arXiv preprint arXiv:1907.11692 (2019)
153. P. He, et al., International Conference on Learning Representations (2021)
154. V. Sanh, et al., the 5th Workshop on Energy Efficient Machine Learning and Cognitive Computing - NeurIPS (2019)

155. W. Wang, et al., International Conference on Learning Representations (2020)
156. S. Shen, et al., AAAI Conference on Artificial Intelligence **34**, 8815 (2020)
157. J. Lee, et al., Bioinformatics **36(4)**, 1234 (2020)
158. P. Prakash, et al., AAAI Conference on Artificial Intelligence **35**, 453 (2021)
159. Z. Wu, et al., International Conference on Learning Representations (2020)
160. S. Mehta, et al., International Conference on Learning Representations (2021)
161. Y. Tay, et al., International Conference on Learning Representations (2021)
162. S. Yun, et al., International Conference on Learning Representations (2018)
163. Y. Rong, et al., Advances in Neural Information Processing Systems 33 (2020)
164. J. Yang, et al., Advances in Neural Information Processing Systems 34 (2021)
165. C. Zhao, et al., International Conference on Learning Representations (2020)
166. R. You, et al., Advances in Neural Information Processing Systems 32 (2019)
167. X. Fan, et al., Advances in Neural Information Processing Systems 33 (2020)
168. G. Brunner, et al., International Conference on Learning Representations (2020)
169. A. Dosovitskiy, et al., International Conference on Learning Representations (2021)
170. A. Katharopoulos, et al., International Conference on Machine Learning (2020)
171. L. Greengard, V. Rokhlin, Journal of Computational Physics **73(2)**, 325– (1987)
172. T. Nguyen, et al., Advances in Neural Information Processing Systems 34 (2021)
173. N. Kitaev, L. Kaiser, A. Levskaya, International Conference on Learning Representations (2020)
174. J. Lee, et al., International Conference on Machine Learning pp. 3744–3753 (2019)
175. A. Roy, et al., Transactions of the Association for Computational Linguistics pp. 53–68 (2020)
176. R. Child, et al., arXiv preprint arXiv:1904.10509 (2019)
177. G. Correia, V. Niculae, A. Martins, Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing pp. 2174–2184 (2019)
178. H. Peng, et al., International Conference on Learning Representations (2021)
179. Y. Chen, et al., Advances in Neural Information Processing Systems 34 (2021)
180. M. Zaheer, et al., Advances in Neural Information Processing Systems 33 (2020)
181. C.Z. Huang, et al., International Conference on Learning Representations (2019)
182. J. Lu, et al., Advances in Neural Information Processing Systems 34 (2021)
183. Z. Pan, et al., Proceedings of the IEEE/CVF International Conference on Computer Vision pp. 377–386 (2021)
184. C. Zhu, et al., Advances in Neural Information Processing Systems 34 (2021)
185. A. Jaegle, et al., International Conference on Machine Learning pp. 4651–4664 (2021)
186. K. Choromanski, et al., International Conference on Learning Representations (2021)
187. A. El-Nouby, et al., Advances in Neural Information Processing Systems 34 (2021)
188. Q. Yu, et al., Advances in Neural Information Processing Systems 34 (2021)
189. Z. Zeng, et al., International Conference on Machine Learning pp. 12,321–12,332 (2021)
190. Z. Shen, et al., Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision pp. 3531–3539 (2021)
191. S. Luo, et al., Advances in Neural Information Processing Systems 34 (2021)
192. X. Ma, et al., Advances in Neural Information Processing Systems 34 (2021)