# Big Bird: Transformers for Longer Sequences

**Manzil Zaheer,** **Guru Guruganesh,** **Avinava Dubey,**
**Joshua Ainslie,** **Chris Alberti,** **Santiago Ontanon,** **Philip Pham,**
**Anirudh Ravula,** **Qifan Wang,** **Li Yang,** **Amr Ahmed**
Google Research
{manzilz, gurug, avinavadubey}@google.com

## Abstract

Transformers-based models, such as BERT, have been one of the most successful deep learning models for NLP. Unfortunately, one of their core limitations is the quadratic dependency (mainly in terms of memory) on the sequence length due to their full attention mechanism. To remedy this, we propose, BIGBIRD, a sparse attention mechanism that reduces this quadratic dependency to linear. We show that BIGBIRD is a universal approximator of sequence functions and is Turing complete, thereby preserving these properties of the quadratic, full attention model. Along the way, our theoretical analysis reveals some of the benefits of having $O(1)$ global tokens (such as CLS), that attend to the entire sequence as part of the sparse attention mechanism. The proposed sparse attention can handle sequences of length up to 8x of what was previously possible using similar hardware. As a consequence of the capability to handle longer context, BIGBIRD drastically improves performance on various NLP tasks such as question answering and summarization. We also propose novel applications to genomics data.

## 1 Introduction

Models based on Transformers [92], such as BERT [22, 63], are wildly successful for a wide variety of Natural Language Processing (NLP) tasks and consequently are mainstay of modern NLP research. Their versatility and robustness are the primary drivers behind the wide-scale adoption of Transformers. The model is easily adapted for a diverse range of sequence based tasks – as a seq2seq model for translation [92], summarization [66], generation [15], etc. or as a standalone encoders for sentiment analysis [84], POS tagging [65], machine reading comprehension [94], etc. – and it is known to vastly outperform previous sequence models like LSTM [37]. The key innovation in Transformers is the introduction of a self-attention mechanism, which can be evaluated in parallel for each token of the input sequence, eliminating the sequential dependency in recurrent neural networks, like LSTM. This parallelism enables Transformers to leverage the full power of modern SIMD hardware accelerators like GPUs/TPUs, thereby facilitating training of NLP models on datasets of unprecedented size. This ability to train on large scale data has led to surfacing of models like BERT [22] and T5 [75], which pretrain transformers on large general purpose corpora and transfer the knowledge to down-stream task. The pretraining has led to significant improvement in low data regime downstream tasks [51] as well as tasks with sufficient data [102] and thus have been a major force behind the ubiquity of transformers in contemporary NLP.

The self-attention mechanism overcomes constraints of RNNs (namely the sequential nature of RNN) by allowing each token in the input sequence to attend independently to every other token in the sequence. This design choice has several interesting repercussions. In particular, the full self-attention have computational and memory requirement that is quadratic in the sequence length. We note that while the corpus can be large, the sequence length, which provides the context in many applications is very limited. Using commonly available current hardware and model sizes, this requirement translates to roughly being able to handle input sequences of length 512 tokens. This reduces its direct applicability to tasks that require larger context, like QA [60], document classification, etc.

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

However, while we know that self-attention and Transformers are useful, our theoretical understanding is rudimentary. What aspects of the self-attention model are necessary for its performance? What can we say about the expressivity of Transformers and similar models? Apriori, it was not even clear from the design if the proposed self-attention mechanism was as effective as RNNs. For example, the self-attention does not even obey sequence order as it is permutation equivariant. This concern has been partially resolved, as Yun et al. [105] showed that transformers are expressive enough to capture all continuous sequence to sequence functions with a compact domain. Meanwhile, Pérez et al. [72] showed that the full transformer is Turing Complete (i.e. can simulate a full Turing machine). Two natural questions arise: Can we achieve the empirical benefits of a fully quadratic self-attention scheme using fewer inner-products? Do these sparse attention mechanisms preserve the expressivity and flexibility of the original network?

In this paper, we address both the above questions and produce a sparse attention mechanism that improves performance on a multitude of tasks that require long contexts. We systematically develop BIGBIRD, an attention mechanism whose complexity is linear in the number of tokens (Sec. 2). We take inspiration from graph sparsification methods and understand where the proof for expressiveness of Transformers breaks down when full-attention is relaxed to form the proposed attention pattern. This understanding helped us develop BIGBIRD, which is theoretically as expressive and also empirically useful. In particular, our BIGBIRD consists of three main part:

- A set of $g$ global tokens that attend on all parts of the sequence.
- For each query $q_i$, a set of $r$ random keys that each query will attend to.
- A block of local neighbors $w$ so that each node attends on their local structure.

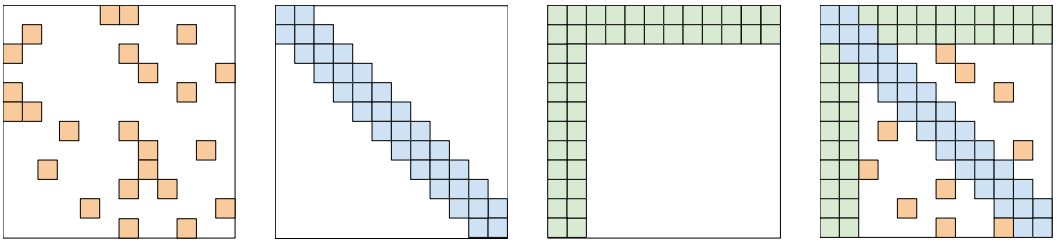
This leads to a high performing attention mechanism scaling to much longer sequence lengths (8x). To summarize, our main **contributions** are:

1. BIGBIRD satisfies all the known theoretical properties of full transformer (Sec. 3). In particular, we show that adding extra tokens allows one to express all continuous sequence to sequence functions with only $O(n)$-inner products. Furthermore, we show that under standard assumptions regarding precision, BIGBIRD is Turing complete.
2. Empirically, we show that the extended context modelled by BIGBIRD benefits variety of NLP tasks. We achieve *state of the art* results for question answering and document summarization on a number of different datasets. Summary of these results are presented in Sec. 4.
3. Lastly, we introduce a novel application of attention based models where long contexts are beneficial: extracting contextual representations of genomics sequences like DNA. With longer masked LM pretraining, BIGBIRD improves performance on downstream tasks such as promoter-region and chromatin profile prediction (Sec. 5).

### 1.1 Related Work

There have been a number of interesting attempts, that were aimed at alleviating the quadratic dependency of Transformers, which can broadly categorized into two directions. First line of work embraces the length limitation and develops method around it. Simplest methods in this category just employ sliding window [94], but in general most work fits in the following general paradigm: using some other mechanism select a smaller subset of relevant contexts to feed in the transformer and optionally iterate, i.e. call transformer block multiple time with different contexts each time. Most prominently, SpanBERT [42], ORQA [54], REALM [34], RAG [57] have achieved strong performance for different tasks. However, it is worth noting that these methods often require significant engineering efforts (like back prop through large scale nearest neighbor search) and are hard to train.

Second line of work questions if full attention is essential and have tried to come up with approaches that do not require full attention, thereby reducing the memory and computation requirements. Prominently, Dai et al. [21], Sukhbaatar et al. [83], Rae et al. [74] have proposed auto-regresive models that work well for left-to-right language modeling but suffer in tasks which require bidirectional context. Child et al. [16] proposed a sparse model that reduces the complexity to $O(N\sqrt{N})$, Kitaev et al. [49] further reduced the complexity to $O(N \log(N))$ by using LSH to compute nearest neighbors. Ye et al. [104] proposed binary partitions of the data where as Qiu et al. [73] reduced complexity by using block sparsity. Recently, Longformer [8] introduced a localized sliding window based mask with few global mask to reduce computation and extended BERT to longer sequence based tasks. Finally, our work is closely related to and built on the work of Extended Transformers Construction [4]. This work was designed to encode structure in text for transformers. The idea of global tokens was used extensively by them to achieve their goals. Our theoretical work can be seen as providing a justification for the success of these models as well. It is important to note that most of the



(a) Random attention (b) Window attention (c) Global Attention (d) BIGBIRD

Figure 1: Building blocks of the attention mechanism used in BIGBIRD. White color indicates absence of attention. (a) random attention with $r = 2$, (b) sliding window attention with $w = 3$ (c) global attention with $g = 2$. (d) the combined BIGBIRD model.

aforementioned methods are heuristic based and empirically are not as versatile and robust as the original transformer, i.e. the same architecture do not attain SoTA on multiple standard benchmarks. (There is one exception of longformer which we include in all our comparisons, Sec. 4). Moreover, these approximations do not come with theoretical guarantees.

## 2 BIGBIRD Architecture

In this section, we describe the BIGBIRD model using the *generalised attention mechanism* that is used in each layer of transformer operating on an input sequence $\boldsymbol{X} = (\boldsymbol{x}_1, ..., \boldsymbol{x}_n) \in \mathbb{R}^{n \times d}$. The *generalized attention mechanism* is described by a directed graph $D$ whose vertex set is $[n] = \{1, \ldots, n\}$. The set of arcs (directed edges) represent the set of inner products that the attention mechanism will consider. Let $N(i)$ denote the out-neighbors set of node $i$ in $D$, then the $i^{\text{th}}$ output vector of the generalized attention mechanism is defined as

$$\text{ATTN}_D(\boldsymbol{X})_i = \boldsymbol{x}_i + \sum_{h=1}^{H} \sigma\left(Q_h(\boldsymbol{x}_i) K_h(\boldsymbol{X}_{N(i)})^T\right) \cdot V_h(\boldsymbol{X}_{N(i)}) \tag{AT}$$

where $Q_h, K_h : \mathbb{R}^d \to \mathbb{R}^m$ are query and key functions respectively, $V_h : \mathbb{R}^d \to \mathbb{R}^d$ is a value function, $\sigma$ is a scoring function (e.g. softmax or hardmax) and $H$ denotes the number of heads. Also note $X_{N(i)}$ corresponds to the matrix formed by only stacking $\{\boldsymbol{x}_j : j \in N(i)\}$ and not all the inputs.

If $D$ is the complete digraph, we recover the full quadratic attention mechanism of Vaswani et al. [92]. To simplify our exposition, we will operate on the adjacency matrix $A$ of the graph $D$ even though the underlying graph maybe sparse. To elaborate, $A \in [0, 1]^{n \times n}$ with $A(i, j) = 1$ if query $i$ attends to key $j$ and is zero otherwise. For example, when $A$ is the ones matrix (as in BERT), it leads to quadratic complexity, since all tokens attend on every other token. This view of self-attention as a fully connected graph allows us to exploit existing graph theory to help reduce its complexity. The problem of reducing the quadratic complexity of self-attention can now be seen as a *graph sparsification problem*. It is well-known that random graphs are expanders and can approximate complete graphs in a number of different contexts including in their spectral properties [80, 38]. We believe sparse random graph for attention mechanism should have two desiderata: small average path length between nodes and a notion of locality, each of which we discuss below.

Let us consider the simplest random graph construction, known as Erdős-Rényi model, where each edge is independently chosen with a fixed probability. In such a random graph with just $\tilde{\Theta}(n)$ edges, the shortest path between any two nodes is logarithmic in the number of nodes [17, 43]. As a consequence, such a random graph approximates the complete graph spectrally and its second eigenvalue (of the adjacency matrix) is quite far from the first eigenvalue [9, 10, 6]. This property leads to a rapid mixing time for random walks in the grpah, which informally suggests that information can flow fast between any pair of nodes. Thus, we propose a sparse attention where each query attends over $r$ random number of keys i.e. $A(i, \cdot) = 1$ for $r$ randomly chosen keys (see Fig. 1a).

The second viewpoint which inspired the creation of BIGBIRD is that most contexts within NLP and computational biology have data which displays a great deal of *locality of reference*. In this phenomenon, a great deal of information about a token can be derived from its neighboring tokens. Most pertinently, Clark et al. [19] investigated self-attention models in NLP tasks and concluded that that neighboring inner-products are extremely important. The concept of locality, proximity of tokens in linguistic structure, also forms the basis of various linguistic theories such as transformational-generative grammar. In the terminology of graph theory, clustering coefficient is a measure of locality

of connectivity, and is high when the graph contains many cliques or near-cliques (subgraphs that are almost fully interconnected). Simple Erdős-Rényi random graphs do not have a high clustering coefficient [85], but a class of random graphs, known as small world graphs, exhibit high clustering coefficient [95]. A particular model introduced by Watts and Strogatz [95] is of high relevance to us as it achieves a good balance between average shortest path and the notion of locality. The generative process of their model is as follows: Construct a regular ring lattice, a graph with $n$ nodes each connected to $w$ neighbors, $w/2$ on each side.

In other words we begin with a sliding window on the nodes. Then a random subset ($k\%$) of all connections is replaced with a random connection. The other (100 - $k$)% local connections are retained. However, deleting such random edges might be inefficient on modern hardware, so we retain it, which will not affect its properties. In summary, to capture these local structures in the context, in BIGBIRD, we define a sliding window attention, so that during self attention of width $w$, query at location $i$ attends from $i - w/2$ to $i + w/2$ keys. In our notation, $A(i, i-w/2 : i+w/2) = 1$ (see Fig. 1b). As an initial sanity check, we performed basic experiments to test whether these intuitions are sufficient in getting performance close to BERT like models, while keeping attention linear in the number of tokens. We found that random blocks and local window were insufficient in capturing all the context necessary to compete with the performance of BERT.

Table 1: Building block comparison @512

| Model | MLM | SQuAD | MNLI |
|---|---|---|---|
| BERT-base | 64.2 | 88.5 | 83.4 |
| Random (R) | 60.1 | 83.0 | 80.2 |
| Window (W) | 58.3 | 76.4 | 73.1 |
| R + W | 62.7 | 85.1 | 80.5 |

The final piece of BIGBIRD is inspired from our theoretical analysis (Sec. 3), which is critical for empirical performance. More specifically, our theory utilizes the importance of "global tokens" (tokens that attend to all tokens in the sequence and to whom all tokens attend to (see Fig. 1c). These global tokens can be defined in two ways:

- BIGBIRD-ITC: In internal transformer construction (ITC), we make some existing tokens "global", which attend over the entire sequence. Concretely, we choose a subset $G$ of indices (with $g := |G|$), such that $A(i, :) = 1$ and $A(:, i) = 1$ for all $i \in G$.
- BIGBIRD-ETC: In extended transformer construction (ETC), we include additional "global" tokens such as CLS. Concretely, we add $g$ global tokens that attend to all existing tokens. In our notation, this corresponds to creating a new matrix $B \in [0, 1]^{(N+g)\times(N+g)}$ by adding $g$ rows to matrix $A$, such that $B(i, :) = 1$, and $B(:, i) = 1$ for all $i \in \{1, 2, \dots g\}$, and $B(g + i, g + j) = A(i, j) \forall\, i, j \in \{1, \dots, N\}$. This adds extra location to store context and as we will see in the experiments improves performance.

The final attention mechanism for BIGBIRD (Fig. 1d) has all three of these properties: queries attend to $r$ random keys, each query attends to $w/2$ tokens to the left of its location and $w/2$ to the right of its location and they contain $g$ global tokens (The global tokens can be from existing tokens or extra added tokens). We provide implementation details in App. D.

## 3 Theoretical Results about Sparse Attention Mechanism

In this section, we will show that that sparse attention mechanisms are as powerful and expressive as full-attention mechanisms in two respects. First, we show that when sparse attention mechanisms are used in a standalone encoder (such as BERT), they are Universal Approximators of sequence to sequence functions in the style of Yun et al. [105]. We note that this property was also explored theoretically in contemporary work Yun et al. [106]. Second, unlike [106], we further show that sparse encoder-decoder transformers are Turning Complete (assuming the same conditions defined in [72]). Complementing the above positive results, we also show that moving to a sparse-attention mechanism incurs a cost, i.e. there is no free lunch. In Sec. 3.4, we show lower bounds by exhibiting a natural task where any sufficiently sparse mechanism will require polynomially more layers.

### 3.1 Notation

The complete Transformer *encoder* stack is nothing but the repeated application of a single-layer encoder (with independent parameters). We denote class of such Transformer encoders stack, defined using generalized encoder (Sec. 2), by $\mathcal{T}_D^{H,m,q}$ which consists of $H$-heads with head size $m$ and $q$ is the hidden layer size of the output network, and the attention layer is defined by the directed graph $D$.

The key difference between our proposed attention mechanism to that of Vaswani et al. [92], Yun et al. [105] is that we add a special token at the beginning of each sequence and assign it a special vector.

We will refer to this as $\boldsymbol{x}_0$. Therefore our graph $D$ will have vertex set $\{0\} \cup [n] = \{0, 1, 2, \ldots, n\}$. We will assume that this extra node and its respective vector will be dropped at the final output layer of transformer. To avoid cumbersome notation, we will still treat transformer as mapping sequences $\boldsymbol{X} \in \mathbb{R}^{n \times d}$ to $\mathbb{R}^{n \times d}$. We will also allow the transformer to append position embeddings $E \in \mathbb{R}^{d \times n}$ to matrix $X$ in the input layer.

Finally, we need to define the function class and distance measure for proving universal approximation property. Let $\mathcal{F}_{CD}$ denote the set of continuous functions $f : [0, 1]^{n \times d} \to \mathbb{R}^{n \times d}$ which are continuous with respect to the topology defined by $\ell_p$ norm. Recall for any $p \geq 1$, the $\ell_p$ distance is $d_p(f_1, f_2) = \left(\int \|f_1(X) - f_2(X)\|_p^p dX\right)^{1/p}$.

### 3.2 Universal Approximators

**Definition 1.** *The star-graph $S$ centered at $0$ is the graph defined on $\{0, \ldots, n\}$. The neighborhood of all vertices $i$ is $N(i) = \{0, i\}$ for $i \in \{1 \ldots n\}$ and $N(0) = \{1, \ldots n\}$.*

Our main theorem is that the sparse attention mechanism defined by any graph containing $S$ is a universal approximator:

**Theorem 1.** *Given $1 < p < \infty$ and $\epsilon > 0$, for any $f \in \mathcal{F}_{CD}$, there exists a transformer with sparse-attention, $g \in \mathcal{T}_D^{H,m,q}$ such that $d_p(f, g) \leq \epsilon$ where $D$ is any graph containing star graph $S$.*

To prove the theorem, we will follow the standard proof structure outlined in [105].

**Step 1: Approximate $\mathcal{F}_{CD}$ by piece-wise constant functions.** Since $f$ is a continuous function with bounded domain $[0, 1)^{n \times d}$, we will approximate it with a suitable piece-wise constant function. This is accomplished by a suitable partition of the region $[0, 1)$ into a grid of granularity $\delta$ to get a discrete set $\mathbb{G}_\delta$. Therefore, we can assume that we are dealing with a function $\bar{f} : \mathbb{G}_\delta \to \mathbb{R}^{n \times d}$, where $d_p(f, \bar{f}) \leq \frac{\epsilon}{3}$.

**Step 2: Approximate piece-wise constant functions by modified transformers.** This is the key step of the proof where the self-attention mechanism is used to generate a *contextual-mapping* of the input. Informally, a contextual mapping is a unique code for the pair consisting of a matrix $(\boldsymbol{X}, \boldsymbol{x}_i)$ and a column. Its uniqueness allows the Feed forward layers to use each code to map it to a unique output column.

The main technical challenge is computing the contextual mapping using only sparse attention mechanism. This was done in [105] using a "selective" shift operator which shift up entries that are in a specific interval. Key to their proof was the fact that the shift, was exactly the range of the largest entry to the smallest entry.

Creating a contextual mapping with a sparse attention mechanism is quite a challenge. In particular, because each query only attends to a few keys, it is not at all clear that sufficient information can be corralled to make a contextual embedding of the entire matrix. To get around this, we develop a sparse shift operator which shifts the entries of the matrices if they lie in a certain range. The exact amount of the shift is controlled by the directed sparse attention graphg $D$. The second key ingredient is the use of additional global token. By carefully applying the operator to a set of chosen ranges, we will show that each column will contain a unique mapping of the full mapping. Therefore, we can augment the loss of inner-products in the self attention mechanism by using multiple layers and an auxiliary global token.

**Step 3: Approximate modified transformers by original Transformers**: The final step is to approximate the modified transformers by the original transformer which uses ReLU and softmax.

We provide the full details in App. A.

### 3.3 Turing Completeness

Transformers are a very general class. In the original paper of Vaswani et al. [92], they were used in both an encoder and a decoder. While the previous section outlined how powerful just the encoders were, another natural question is to ask what the additional power of both a decoder along with an encoder is? Pérez et al. [72] showed that the full transformer based on a quadratic attention mechanism is Turing Complete. This result makes one unrealistic assumption, which is that the model works on arbitrary precision model. Of course, this is necessary as otherwise, Transformers are bounded finite state machines and cannot be Turing Complete.

It is natural to ask if the full attention mechanism is necessary. Or can a sparse attention mechanism also be used to simulate any Turing Machine? We show that this is indeed the case: we can use a sparse encoder and sparse decoder to simulate any Turing Machine.

To use the sparse attention mechanism in the transformer architecture, we need to define a suitable modification where each token only reacts to previous tokens. Unlike the case for BERT, where the entire attention mechanism is applied once, in full transformers, the sparse attention mechanism at decoder side is used token by token. Secondly the work of Pérez et al. [72], uses each token as a representation of the tape history and uses the full attention to move and retrieve the correct tape symbol. Most of the construction of Pérez et al. [72] goes through for sparse attentions, except for their addressing scheme to point back in history (Lemma B.4 in [72]). We show how to simulate this using a sparse attention mechanism and defer the details to App. B.

### 3.4 Limitations

We demonstrate a natural task which can be solved by the full attention mechanism in $O(1)$-layers. However, under standard complexity theoretic assumptions, this problem requires $\tilde{\Omega}(n)$-layers for any sparse attention layers with $\tilde{O}(n)$ edges (not just BIGBIRD). (Here $\tilde{O}$ hides poly-logarthmic factors). Consider the simple problem of finding the corresponding furthest vector for each vector in the given sequence of length $n$. Formally,

**Task 1.** Given $n$ unit vectors $\{u_1, \ldots, u_n\}$, find $f(u_1, \ldots, u_n) \to (u_{1^*}, \ldots, u_{n^*})$ where for a fixed $j \in [n]$, we define $j^* = \arg\max_k \|u_k - u_j\|_2^2$.

Finding vectors that are furthest apart boils down to minimize inner product search in case of unit vectors. For a full-attention mechanism with appropriate query and keys, this task is very easy as we can evaluate all pair-wise inner products.

The impossibility for sparse-attention follows from hardness results stemming from Orthogonal Vector Conjecture(OVC) [1, 2, 7, 97]. The OVC is a widely used assumption in fine-grained complexity. Informally, it states that one cannot determine if the minimum inner product among $n$ boolean vectors is 0 in subquadratic time. In App. C, we show a reduction using OVC to show that if a transformer $g \in \mathcal{T}_D^{H=1,m=2d,q=0}$ for any sparse directed graph $D$ can evaluate the Task 1, it can solve the orthogonal vector problem.

**Proposition 1.** *There exists a single layer full self-attention $g \in \mathcal{T}^{H=1,m=2d,q=0}$ that can evaluate Task 1, i.e. $g(u_1, ..., u_n) = [u_{1^*}, \ldots, u_{n^*}]$, but for any sparse-attention graph $D$ with $\tilde{O}(n)$ edges (i.e. inner product evaluations), would require $\tilde{\Omega}(n^{1-o(1)})$ layers.*
We give a formal proof of this fact in App. C.

## 4 Experiments: Natural Language Processing

In this section our goal is to showcase benefits of modeling longer input sequence for NLP tasks, for which we select three representative tasks. We begin with basic masked language modeling (MLM; Devlin et al. 22) to check if better contextual representations can be learnt by utilizing longer contiguous sequences. Next, we consider QA with supporting evidence, for which capability to handle longer sequence would allow us to retrieve more evidence using crude systems like TF-IDF/BM25. Finally, we tackle long document classification where discriminating information may not be located in first 512 tokens. Below we summarize the results for BIGBIRD using sequence length 4096[1], while we defer all other setup details including computational resources, batch size, step size, to App. E.

**Pretraining and MLM** We follow [22, 63] to create base and large versions of BIGBIRD and pretrain it using MLM objective. This task involves predicting a random subset of tokens which have been masked out. We use four standard data-sets for pretraining (listed in App. E.1, Tab. 9), warm-starting from the public RoBERTa checkpoint[2]. We compare performance in predicting the masked out tokens in terms of bits per character, following [8]. As seen in App. E.1, Tab. 10, both BIGBIRD and Longformer perform better than limited length RoBERTa, with BIGBIRD-ETC performing the best. We note that we trained our models on a reasonable $16GB$ memory/chip with batch size of 32-64. Our memory efficiency is due to efficient blocking and sparsity structure of the sparse attention mechanism described in Sec. 2.

[1] code available at http://goo.gle/bigbird-transformer

[2] https://github.com/pytorch/fairseq/tree/master/examples/roberta

Table 2: QA Dev results using Base size models. We report accuracy for WikiHop and F1 for HotpotQA, Natural Questions, and TriviaQA.

| Model | HotpotQA | | | NaturalQ | | TriviaQA | WikiHop |
|---|---|---|---|---|---|---|---|
| | Ans | Sup | Joint | LA | SA | Full | MCQ |
| RoBERTa | 73.5 | 83.4 | 63.5 | - | - | 74.3 | 72.4 |
| Longformer | 74.3 | 84.4 | 64.4 | - | - | 75.2 | 75.0 |
| BIGBIRD-ITC | **75.7** | 86.8 | 67.7 | 70.8 | 53.3 | **79.5** | **75.9** |
| BIGBIRD-ETC | 75.5 | **87.1** | **67.8** | **73.9** | **54.9** | 78.7 | **75.9** |

**Question Answering (QA)** We considered following four challenging datasets:

1. Natural Questions [52]: For the given question, find a short span of answer (SA) from the given evidences as well highlight the paragraph from the given evidences containing information about the correct answer (LA).
2. HotpotQA-distractor [101]: Similar to natural questions, it requires finding the answer (Ans) as well as the supporting facts (Sup) over different documents needed for multi-hop reasoning from the given evidences.
3. TriviaQA-wiki [41]: We need to provide an answer for the given question using provided Wikipedia evidence, however, the answer might not be present in the given evidence. On a smaller *verified* subset of question, the given evidence is guaranteed to contain the answer. Nevertheless, we model the answer as span selection problem in this case as well.
4. WikiHop [96]: Chose correct option from multiple-choice questions (MCQ), by aggregating information spread across multiple documents given in the evidences.

As these tasks are very competitive, multiple highly engineered systems have been designed specific each dataset confirming to respective output formats. For a fair comparison, we had to use some additional regularization for training BIGBIRD, details of which are provided in App. E.2 along with exact architecture description. We experiment using the base sized model and select the best configuration on the development set for each dataset (as reported in Tab. 2). We can see that BIGBIRD-ETC, with expanded global tokens consistently outperforms all other models. Thus, we chose this configuration to train a large sized model to be used for evaluation on the hidden test set.

In Tab. 3, we compare BIGBIRD-ETC model to top-3 entries from the leaderboard excluding BIGBIRD. One can clearly see the importance of using longer context as both Longformer and BIGBIRD outperform models with smaller contexts. Also, it is worth noting that BIGBIRD submission is a single model, whereas the other top-3 entries for Natural Questions are ensembles, which might explain the slightly lower accuracy in exact answer phrase selection.

**Classification** We experiment on datasets of different lengths and contents, specifically various document classification and GLUE tasks. Following BERT, we used one layer with cross entropy loss on top of the first [CLS] token. We see that gains of using BIGBIRD are more significant when we have longer documents and fewer training examples. For instance, using base sized model,

Table 3: Fine-tuning results on **Test** set for QA tasks. The Test results (F1 for HotpotQA, Natural Questions, TriviaQA, and Accuracy for WikiHop) have been picked from their respective leaderboard. For each task the top-3 leaders were picked not including BIGBIRD-etc. **For Natural Questions Long Answer (LA), TriviaQA Verified, and WikiHop, BIGBIRD-ETC is the new state-of-the-art**. On HotpotQA we are third in the leaderboard by F1 and second by Exact Match (EM).

| Model | HotpotQA | | | NaturalQ | | TriviaQA | | WikiHop |
|---|---|---|---|---|---|---|---|---|
| | Ans | Sup | Joint | LA | SA | Full | Verified | MCQ |
| HGN [26] | **82.2** | 88.5 | **74.2** | - | - | - | - | - |
| GSAN | 81.6 | 88.7 | 73.9 | - | - | - | - | - |
| ReflectionNet [32] | - | - | - | 77.1 | **64.1** | - | - | - |
| RikiNet [61] | - | - | - | 75.5 | 59.5 | - | - | - |
| Fusion-in-Decoder [39] | - | - | - | - | - | **84.5** | 90.3 | - |
| SpanBERT [42] | - | - | - | - | - | 79.1 | 86.6 | - |
| MRC-GCN [88] | - | - | - | - | - | - | - | 78.3 |
| MultiHop [14] | - | - | - | - | - | - | - | 76.5 |
| Longformer [8] | 81.2 | 88.3 | 73.2 | - | - | 77.3 | 85.3 | 81.9 |
| BIGBIRD-ETC | 81.2 | **89.1** | 73.6 | **77.7** | 57.8 | 80.9 | **90.8** | **82.3** |

Table 4: Summarization ROUGE score for long documents.

| | Model | Arxiv | | | PubMed | | | BigPatent | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | R-1 | R-2 | R-L | R-1 | R-2 | R-L | R-1 | R-2 | R-L |
| Prior Art | SumBasic [68] | 29.47 | 6.95 | 26.30 | 37.15 | 11.36 | 33.43 | 27.44 | 7.08 | 23.66 |
| | LexRank [25] | 33.85 | 10.73 | 28.99 | 39.19 | 13.89 | 34.59 | 35.57 | 10.47 | 29.03 |
| | LSA [98] | 29.91 | 7.42 | 25.67 | 33.89 | 9.93 | 29.70 | - | - | - |
| | Attn-Seq2Seq [86] | 29.30 | 6.00 | 25.56 | 31.55 | 8.52 | 27.38 | 28.74 | 7.87 | 24.66 |
| | Pntr-Gen-Seq2Seq [77] | 32.06 | 9.04 | 25.16 | 35.86 | 10.22 | 29.69 | 33.14 | 11.63 | 28.55 |
| | Long-Doc-Seq2Seq [20] | 35.80 | 11.05 | 31.80 | 38.93 | 15.37 | 35.21 | - | - | - |
| | Sent-CLF [82] | 34.01 | 8.71 | 30.41 | 45.01 | 19.91 | 41.16 | 36.20 | 10.99 | 31.83 |
| | Sent-PTR [82] | 42.32 | 15.63 | 38.06 | 43.30 | 17.92 | 39.47 | 34.21 | 10.78 | 30.07 |
| | Extr-Abst-TLM [82] | 41.62 | 14.69 | 38.03 | 42.13 | 16.27 | 39.21 | 38.65 | 12.31 | 34.09 |
| | Dancer [31] | 42.70 | 16.54 | 38.44 | 44.09 | 17.69 | 40.27 | - | - | - |
| Base | Transformer | 28.52 | 6.70 | 25.58 | 31.71 | 8.32 | 29.42 | 39.66 | 20.94 | 31.20 |
| | + RoBERTa [76] | 31.98 | 8.13 | 29.53 | 35.77 | 13.85 | 33.32 | 41.11 | 22.10 | 32.58 |
| | + Pegasus [108] | 34.81 | 10.16 | 30.14 | 39.98 | 15.15 | 35.89 | 43.55 | 20.43 | 31.80 |
| | BIGBIRD-RoBERTa | 41.22 | 16.43 | 36.96 | 43.70 | 19.32 | 39.99 | 55.69 | 37.27 | 45.56 |
| Large | Pegasus (Reported) [108] | 44.21 | 16.95 | 38.83 | 45.97 | 20.15 | 41.34 | 52.29 | 33.08 | 41.75 |
| | Pegasus (Re-eval) | 43.85 | 16.83 | 39.17 | 44.53 | 19.30 | 40.70 | 52.25 | 33.04 | 41.80 |
| | BIGBIRD-Pegasus | **46.63** | **19.02** | **41.77** | **46.32** | **20.65** | **42.33** | **60.64** | **42.46** | **50.01** |

BIGBIRD improves state-of-the-art for Arxiv dataset by about $\mathbf{5}\%$ **points**. On Patents dataset, there is improvement over using simple BERT/RoBERTa, but given the large size of training data the improvement over SoTA (which is not BERT based) is not significant. Note that this performance gain is not seen for much smaller IMDb dataset. Along with experimental setup detail, we present detailed results in App. E.4 which show competitive performance.

### 4.1 Encoder-Decoder Tasks

For an encoder-decoder setup, one can easily see that both suffer from quadratic complexity due to the full self attention. We focus on introducing the sparse attention mechanism of BIGBIRD only at the encoder side. This is because, in practical generative applications, the length of output sequence is typically small as compared to the input. For example for text summarization, we see in realistic scenarios (c.f. App. E.5 Tab. 18) that the median output sequence length is $\sim 200$ where as the input sequence's median length is $> 3000$. For such applications, it is more efficient to use sparse attention mechanism for the encoder and full self-attention for the decoder.

**Summarization** Document summarization is a task of creating a short and accurate summary of a text document. We used three long document datasets for testing our model details of which are mention in Tab. 18. In this paper we focus on abstractive summarization of long documents where using a longer contextual encoder should improve performance. The reasons are two fold: First, the salient content can be evenly distributed in the long document, not just in first 512 tokens, and this is by design in the BigPatents dataset [78]. Second, longer documents exhibit a richer discourse structure and summaries are considerably more abstractive, thereby observing more context helps. As has been pointed out recently [76, 108], pretraining helps in generative tasks, we warm start from our general purpose MLM pretraining on base-sized models as well as utilizing state-of-the-art summarization specific pretraining from Pegasus [108] on large-sized models. The results of training BIGBIRD sparse encoder along with full decoder on these long document datasets are presented in Tab. 4. We can clearly see modeling longer context brings significant improvement. Along with hyperparameters, we also present results on shorter but more widespread datasets in App. E.5, which show that using sparse attention does not hamper performance either.

## 5 Experiments: Genomics

There has been a recent upsurge in using deep learning for genomics data [87, 107, 13], which has resulted in improved performance on several biologically-significant tasks such as promoter site prediction [71], methylation analysis [55], predicting functional effects of non-coding variant [110], etc. These approaches consume DNA sequence fragments as inputs, and therefore we believe longer input sequence handling capability of BIGBIRD would be beneficial as many functional effects

in DNA are highly non-local [12]. Furthermore, taking inspiration from NLP, we learn powerful contextual representations for DNA fragments utilizing abundant unlabeled data (e.g. human reference genome, Saccharomyces Genome Database) via MLM pretraining. Next, we showcase that our long input BIGBIRD along with the proposed pretraining significantly improves performances in two downstream tasks. Detailed experimental setup for the two tasks are provided in App. F.

**Pre-training and MLM** As explored in Liang [58], instead of operating on base pairs, we propose to first segment DNA into tokens so as to further increase the context length (App. F, Fig. 7). In particular, we build a byte-pair encoding [50] table for the DNA sequence of size 32K, with each token representing 8.78 base pairs on average. We learn contextual representation of these token on the human reference genome (GRCh37)[3] using MLM objective. We then report the bits per character (BPC) on a held-out set in Tab. 5. We find that attention based contextual representation of DNA does improve BPC, which is further improved by using longer context.

Table 5: MLM BPC

| Model | BPC |
|---|---|
| SRILM [58] | 1.57 |
| BERT (sqln. 512) | 1.23 |
| BIGBIRD (sqln. 4096) | **1.12** |

**Promoter Region Prediction** Promoter is a DNA region typically located upstream of the gene, which is the site of transcription initiation. Multiple methods have been proposed to identify the promoter regions in a given DNA sequence [100, 59, 11, 99, 71], as it is an important first step in understanding gene regulation. The corresponding machine learning task is to classify a given DNA fragment as promoter or non-promoter sequence. We use the dataset compiled by Oubounyt et al. [71] which was built from Eukaryotic Promoter Database (EPDnew) [24] [4]. We finetuned the pretrained BIGBIRD model from above, using the training data and report F1 on test dataset. We compare our results to the previously reported best method in Tab. 6. We see that BIGBIRD achieve nearly perfect accuracy with a $5\%$ jump from the previous best reported accuracy.

Table 6: Comparison.

| Model | F1 |
|---|---|
| CNNProm [91] | 69.7 |
| DeePromoter [71] | 95.6 |
| BIGBIRD | **99.9** |

**Chromatin-Profile Prediction** Non-coding regions of DNA do not code for proteins. Majority of diseases and other trait associated single-nucleotide polymorphism are correlated to non-coding genomic variations [110, 46]. Thus, understanding the functional effects of non-coding regions of DNA is a very important task. An important step in this process, as defined by Zhou and Troyanskaya [110], is to predict large-scale chromatin-profiling from non-coding genomic sequence. To this effect, DeepSea [110], compiled 919 chromatin-profile of 2.4M non-coding variants from Encyclopedia of DNA Elements (ENCODE)[5] and Roadmap Epigenomics projects[6]. The corresponding ML task is to predict, for a given non-coding region of DNA, these 919 chromatin-profile including 690 transcription factors (TF) binding profiles for 160 different TFs, 125 DNase I sensitivity (DHS) profiles and 104 histone-mark (HM) profiles. We jointly learn 919 binary classifiers to predict these functional effects from sequence of DNA fragments. On held-out chromosomes, we compare AUC with the baselines in Tab. 7 and see that we significantly improve on performance on the harder task HM, which is known to have longer-range correlations [27] than others.

Table 7: Chromatin-Profile Prediction

| Model | TF | HM | DHS |
|---|---|---|---|
| gkm-SVM [30] | 89.6 | - | - |
| DeepSea [110] | 95.8 | 85.6 | **92.3** |
| BIGBIRD | **96.1** | **88.7** | 92.1 |

## 6 Conclusion

We propose BIGBIRD: a sparse attention mechanism that is linear in the number of tokens. BIGBIRD satisfies a number of theoretical results: it is a universal approximator of sequence to sequence functions and is also Turing complete. Theoretically, we use the power of extra global tokens preserve the expressive powers of the model. We complement these results by showing that moving to sparse attention mechanism do incur a cost. Empirically, BIGBIRD gives *state-of-the-art* performance on a number of NLP tasks such as question answering and long document classification. We further introduce attention based contextual language model for DNA and fine-tune it for down stream tasks such as promoter region prediction and predicting effects of non-coding variants.

[3] https://www.ncbi.nlm.nih.gov/assembly/GCF_000001405.13/
[4] https://epd.epfl.ch/human/human_database.php?db=human
[5] https://www.encodeproject.org/
[6] http://www.roadmapepigenomics.org/

## Broader Impacts

**Inference Efficiency:** Quadratic attention mechanisms cannot capture long range dependencies which exist in natural text and other datasets. Moreover, there is a growing concern in the ML community about the resource and energy requirement training large scale systems [81]. Moreover, that sparse, computationally efficient systems, like BIGBIRD, can capture long range dependencies in an energy efficient way without losing expressive power.

**Wide Applicability:** Beyond the impact of our model on NLP tasks that require longer context, our proposed contextualized representations of DNA using attention based models, should help in better modeling effects of longer sequences of DNA. Our effort continues a long line of research that bridges the gap between computational models designed for NLP and those for computational biology.

## References


[1] A. Abboud, V. V. Williams, and O. Weimann. Consequences of faster alignment of sequences. In *International Colloquium on Automata, Languages, and Programming*, pages 39–51. Springer, 2014.

[2] A. Abboud, A. Backurs, and V. V. Williams. Tight hardness results for lcs and other sequence similarity measures. In *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, pages 59–78. IEEE, 2015.

[3] J. Abreu, L. Fred, D. Macêdo, and C. Zanchettin. Hierarchical attentional hybrid neural networks for document classification. In *International Conference on Artificial Neural Networks*, pages 396–402. Springer, 2019.

[4] J. Ainslie, S. Ontanon, C. Alberti, P. Pham, A. Ravula, and S. Sanghai. Etc: Encoding long and structured data in transformers. *arXiv preprint arXiv:2004.08483*, 2020.

[5] C. Alberti, K. Lee, and M. Collins. A bert baseline for the natural questions. *arXiv preprint arXiv:1901.08634*, 2019.

[6] J. Alt, R. Ducatez, and A. Knowles. Extremal eigenvalues of critical erd\h {o} sr\'enyi graphs. *arXiv preprint arXiv:1905.03243*, 2019.

[7] A. Backurs and P. Indyk. Edit distance cannot be computed in strongly subquadratic time (unless seth is false). In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 51–58, 2015.

[8] I. Beltagy, M. E. Peters, and A. Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

[9] F. Benaych-Georges, C. Bordenave, A. Knowles, et al. Largest eigenvalues of sparse inhomogeneous erdős–rényi graphs. *Annals of Probability*, 47(3):1653–1676, 2019.

[10] F. Benaych-Georges, C. Bordenave, A. Knowles, et al. Spectral radii of sparse random matrices. In *Annales de l'Institut Henri Poincaré, Probabilités et Statistiques*, volume 56, pages 2141–2161. Institut Henri Poincaré, 2020.

[11] R. Bharanikumar, K. A. R. Premkumar, and A. Palaniappan. Promoterpredict: sequence-based modelling of escherichia coli $\sigma$70 promoter strength yields logarithmic dependence between promoter strength and sequence. *PeerJ*, 6:e5862, 2018.

[12] S. Buldyrev, A. Goldberger, S. Havlin, R. Mantegna, M. Matsa, C.-K. Peng, M. Simons, and H. Stanley. Long-range correlation properties of coding and noncoding dna sequences: Genbank analysis. *Physical Review E*, 51(5):5084, 1995.

[13] A. Busia, G. E. Dahl, C. Fannjiang, D. H. Alexander, E. Dorfman, R. Poplin, C. Y. McLean, P.-C. Chang, and M. DePristo. A deep learning approach to pattern recognition for short dna sequences. *BioRxiv*, page 353474, 2019.

[14] J. Chen, S.-t. Lin, and G. Durrett. Multi-hop question answering via reasoning chains. *arXiv preprint arXiv:1910.02610*, 2019.

[15] Y.-C. Chen, Z. Gan, Y. Cheng, J. Liu, and J. Liu. Distilling the knowledge of bert for text generation. *arXiv preprint arXiv:1911.03829*, 2019.

[16] R. Child, S. Gray, A. Radford, and I. Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.

[17] F. Chung and L. Lu. The average distances in random graphs with given expected degrees. *Proceedings of the National Academy of Sciences*, 99(25):15879–15882, 2002.

[18] C. Clark and M. Gardner. Simple and effective multi-paragraph reading comprehension. *arXiv preprint arXiv:1710.10723*, 2017.

[19] K. Clark, U. Khandelwal, O. Levy, and C. D. Manning. What does bert look at? an analysis of bert's attention. *arXiv preprint arXiv:1906.04341*, 2019.

[20] A. Cohan, F. Dernoncourt, D. S. Kim, T. Bui, S. Kim, W. Chang, and N. Goharian. A discourse-aware attention model for abstractive summarization of long documents. *arXiv preprint arXiv:1804.05685*, 2018.

[21] Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. V. Le, and R. Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv:1901.02860*, 2019.

[22] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[23] L. Dong, N. Yang, W. Wang, F. Wei, X. Liu, Y. Wang, J. Gao, M. Zhou, and H.-W. Hon. Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems*, pages 13042–13054, 2019.

[24] R. Dreos, G. Ambrosini, R. Cavin Périer, and P. Bucher. Epd and epdnew, high-quality promoter resources in the next-generation sequencing era. *Nucleic acids research*, 41(D1):D157–D164, 2013.

[25] G. Erkan and D. R. Radev. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research*, 22:457–479, 2004.

[26] Y. Fang, S. Sun, Z. Gan, R. Pillai, S. Wang, and J. Liu. Hierarchical graph network for multi-hop question answering. *arXiv preprint arXiv:1911.03631*, 2019.

[27] L. A. Gates, C. E. Foulds, and B. W. O'Malley. Histone marks in the 'driver's seat': functional roles in steering the transcription cycle. *Trends in biochemical sciences*, 42(12):977–989, 2017.

[28] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1243–1252. JMLR. org, 2017.

[29] S. Gehrmann, Y. Deng, and A. M. Rush. Bottom-up abstractive summarization. *arXiv preprint arXiv:1808.10792*, 2018.

[30] M. Ghandi, D. Lee, M. Mohammad-Noori, and M. A. Beer. Enhanced regulatory sequence prediction using gapped k-mer features. *PLoS computational biology*, 10(7), 2014.

[31] A. Gidiotis and G. Tsoumakas. A divide-and-conquer approach to the summarization of academic articles. *arXiv preprint arXiv:2004.06190*, 2020.

[32] M. Gong. *ReflectionNet*, 2020 (accessed June 3, 2020). URL https://www.microsoft.com/en-us/research/people/migon/.

[33] S. Gray, A. Radford, and D. P. Kingma. Gpu kernels for block-sparse weights. *arXiv preprint arXiv:1711.09224*, 3, 2017.

[34] K. Guu, K. Lee, Z. Tung, P. Pasupat, and M.-W. Chang. Realm: Retrieval-augmented language model pre-training. *arXiv preprint arXiv:2002.08909*, 2020.

[35] J. He, L. Wang, L. Liu, J. Feng, and H. Wu. Long document classification from local word glimpses via recurrent attention learning. *IEEE Access*, 7:40707–40718, 2019.

[36] K. M. Hermann, T. Kocisky, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom. Teaching machines to read and comprehend. In *Advances in neural information processing systems*, pages 1693–1701, 2015.

[37] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[38] S. Hoory, N. Linial, and A. Wigderson. Expander graphs and their applications. *Bulletin of the American Mathematical Society*, 43(4):439–561, 2006.

[39] G. Izacard and E. Grave. Leveraging passage retrieval with generative models for open domain question answering. *arXiv preprint arXiv:2007.01282*, 2020.

[40] Y. Jiang, J. Petrak, X. Song, K. Bontcheva, and D. Maynard. Team bertha von suttner at semeval-2019 task 4: Hyperpartisan news detection using elmo sentence representation convolutional network. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 840–844, 2019.

[41] M. Joshi, E. Choi, D. S. Weld, and L. Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[42] M. Joshi, D. Chen, Y. Liu, D. S. Weld, L. Zettlemoyer, and O. Levy. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8: 64–77, 2020.

[43] E. Katzav, O. Biham, and A. K. Hartmann. Distribution of shortest path lengths in subcritical erdős-rényi networks. *Physical Review E*, 98(1):012301, 2018.

[44] W. J. Kent, C. W. Sugnet, T. S. Furey, K. M. Roskin, T. H. Pringle, A. M. Zahler, and D. Haussler. The human genome browser at ucsc. *Genome research*, 12(6):996–1006, 2002.

[45] U. Khandelwal, K. Clark, D. Jurafsky, and L. Kaiser. Sample efficient text summarization using a single pre-trained transformer. *arXiv preprint arXiv:1905.08836*, 2019.

[46] E. Khurana, Y. Fu, D. Chakravarty, F. Demichelis, M. A. Rubin, and M. Gerstein. Role of non-coding sequence variants in cancer. *Nature Reviews Genetics*, 17(2):93, 2016.

[47] J. Kiesel, M. Mestre, R. Shukla, E. Vincent, P. Adineh, D. Corney, B. Stein, and M. Potthast. Semeval-2019 task 4: Hyperpartisan news detection. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 829–839, 2019.

[48] B. Kim, H. Kim, and G. Kim. Abstractive summarization of reddit posts with multi-level memory networks. *arXiv preprint arXiv:1811.00783*, 2018.

[49] N. Kitaev, L. Kaiser, and A. Levskaya. Reformer: The efficient transformer. In *International Conference on Learning Representations*, 2019.

[50] T. Kudo and J. Richardson. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*, 2018.

[51] V. Kumar, A. Choudhary, and E. Cho. Data augmentation using pre-trained transformer models. *arXiv preprint arXiv:2003.02245*, 2020.

[52] T. Kwiatkowski, J. Palomaki, O. Redfield, M. Collins, A. Parikh, C. Alberti, D. Epstein, I. Polosukhin, J. Devlin, K. Lee, et al. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.

[53] J.-S. Lee and J. Hsiang. Patent classification by fine-tuning bert language model. *World Patent Information*, 61:101965, 2020.

[54] K. Lee, M.-W. Chang, and K. Toutanova. Latent retrieval for weakly supervised open domain question answering. *arXiv preprint arXiv:1906.00300*, 2019.

[55] J. J. Levy, A. J. Titus, C. L. Petersen, Y. Chen, L. A. Salas, and B. C. Christensen. Methylnet: an automated and modular deep learning approach for dna methylation analysis. *BMC bioinformatics*, 21(1):1–15, 2020.

[56] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

[57] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *arXiv preprint arXiv:2005.11401*, 2020.

[58] W. Liang. Segmenting dna sequence into words based on statistical language model. *Nature Precedings*, pages 1–1, 2012.

[59] H. Lin, Z.-Y. Liang, H. Tang, and W. Chen. Identifying sigma70 promoters with novel pseudo nucleotide composition. *IEEE/ACM transactions on computational biology and bioinformatics*, 2017.

[60] J. Lin, D. Quan, V. Sinha, K. Bakshi, D. Huynh, B. Katz, and D. R. Karger. What makes a good answer? the role of context in question answering. In *Proceedings of the Ninth IFIP TC13 International Conference on Human-Computer Interaction (INTERACT 2003)*, pages 25–32, 2003.

[61] D. Liu, Y. Gong, J. Fu, Y. Yan, J. Chen, D. Jiang, J. Lv, and N. Duan. Rikinet: Reading wikipedia pages for natural question answering. *arXiv preprint arXiv:2004.14560*, 2020.

[62] Y. Liu and M. Lapata. Text summarization with pretrained encoders. *arXiv preprint arXiv:1908.08345*, 2019.

[63] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[64] A. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 142–150, 2011.

[65] L. Martin, B. Muller, P. J. O. Suárez, Y. Dupont, L. Romary, É. V. de la Clergerie, D. Seddah, and B. Sagot. Camembert: a tasty french language model. *arXiv preprint arXiv:1911.03894*, 2019.

[66] D. Miller. Leveraging bert for extractive text summarization on lectures. *arXiv preprint arXiv:1906.04165*, 2019.

[67] S. Narayan, S. B. Cohen, and M. Lapata. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. *arXiv preprint arXiv:1808.08745*, 2018.

[68] A. Nenkova and L. Vanderwende. The impact of frequency on summarization. *Microsoft Research, Redmond, Washington, Tech. Rep. MSR-TR-2005*, 101, 2005.

[69] M. L. Olson, L. Zhang, and C.-N. Yu. Adapting pretrained language models for long document classification. *OpenReview*, 2019.

[70] A. v. d. Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[71] M. Oubounyt, Z. Louadi, H. Tayara, and K. T. Chong. Deepromoter: Robust promoter predictor using deep learning. *Frontiers in genetics*, 10, 2019.

[72] J. Pérez, J. Marinković, and P. Barceló. On the turing completeness of modern neural network architectures. *arXiv preprint arXiv:1901.03429*, 2019.

[73] J. Qiu, H. Ma, O. Levy, S. W.-t. Yih, S. Wang, and J. Tang. Blockwise self-attention for long document understanding. *arXiv preprint arXiv:1911.02972*, 2019.

[74] J. W. Rae, A. Potapenko, S. M. Jayakumar, and T. P. Lillicrap. Compressive transformers for long-range sequence modelling. *arXiv preprint arXiv:1911.05507*, 2019.

[75] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*, 2019.

[76] S. Rothe, S. Narayan, and A. Severyn. Leveraging pre-trained checkpoints for sequence generation tasks. *arXiv preprint arXiv:1907.12461*, 2019.

[77] A. See, P. J. Liu, and C. D. Manning. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*, 2017.

[78] E. Sharma, C. Li, and L. Wang. Bigpatent: A large-scale dataset for abstractive and coherent summarization. *arXiv preprint arXiv:1906.03741*, 2019.

[79] P. Shaw, J. Uszkoreit, and A. Vaswani. Self-attention with relative position representations. *arXiv preprint arXiv:1803.02155*, 2018.

[80] D. A. Spielman and S.-H. Teng. Spectral sparsification of graphs. *SIAM Journal on Computing*, 40(4): 981–1025, 2011.

[81] E. Strubell, A. Ganesh, and A. McCallum. Energy and policy considerations for deep learning in nlp. *arXiv preprint arXiv:1906.02243*, 2019.

[82] S. Subramanian, R. Li, J. Pilault, and C. Pal. On extractive and abstractive neural document summarization with transformer language models. *arXiv preprint arXiv:1909.03186*, 2019.

[83] S. Sukhbaatar, E. Grave, P. Bojanowski, and A. Joulin. Adaptive attention span in transformers. *arXiv preprint arXiv:1905.07799*, 2019.

[84] C. Sun, L. Huang, and X. Qiu. Utilizing bert for aspect-based sentiment analysis via constructing auxiliary sentence. *arXiv preprint arXiv:1903.09588*, 2019.

[85] D. Sussman. *Lecture Notes for Boston University MA 882 Spring 2017*, 2017 (accessed June 3, 2020). URL http://math.bu.edu/people/sussman/MA882_2017/2017-01-26-Lecture-2.html.

[86] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[87] A. Tampuu, Z. Bzhalava, J. Dillner, and R. Vicente. Viraminer: Deep learning on raw dna sequences for identifying viral genomes in human samples. *PloS one*, 14(9), 2019.

[88] Z. Tang, Y. Shen, X. Ma, W. Xu, J. Yu, and W. Lu. Multi-hop reading comprehension across documents with path-based graph convolutional network. *arXiv:2006.06478*, 2020.

[89] T. Thongtan and T. Phienthrakul. Sentiment classification using document embeddings trained with cosine similarity. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 407–414, 2019.

[90] T. H. Trinh and Q. V. Le. A simple method for commonsense reasoning. *arXiv preprint arXiv:1806.02847*, 2018.

[91] R. K. Umarov and V. V. Solovyev. Recognition of prokaryotic and eukaryotic promoters using convolutional deep learning neural networks. *PloS one*, 12(2), 2017.

[92] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[93] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

[94] Z. Wang, P. Ng, X. Ma, R. Nallapati, and B. Xiang. Multi-passage bert: A globally normalized bert model for open-domain question answering. *arXiv preprint arXiv:1908.08167*, 2019.

[95] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world'networks. *nature*, 393(6684): 440–442, 1998.

[96] J. Welbl, P. Stenetorp, and S. Riedel. Constructing datasets for multi-hop reading comprehension across documents. *Transactions of the Association for Computational Linguistics*, 6:287–302, 2018.

[97] R. Williams. A new algorithm for optimal 2-constraint satisfaction and its implications. *Theoretical Computer Science*, 348(2-3):357–365, 2005.

[98] S. Wiseman, S. M. Shieber, and A. M. Rush. Challenges in data-to-document generation. *arXiv preprint arXiv:1707.08052*, 2017.

[99] X. Xiao, Z.-C. Xu, W.-R. Qiu, P. Wang, H.-T. Ge, and K.-C. Chou. ipsw (2l)-pseknc: A two-layer predictor for identifying promoters and their strength by hybrid features via pseudo k-tuple nucleotide composition. *Genomics*, 111(6):1785–1793, 2019.

[100] Y. Yang, R. Zhang, S. Singh, and J. Ma. Exploiting sequence-based features for predicting enhancer–promoter interactions. *Bioinformatics*, 33(14):i252–i260, 2017.

[101] Z. Yang, P. Qi, S. Zhang, Y. Bengio, W. W. Cohen, R. Salakhutdinov, and C. D. Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*, 2018.

[102] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5754–5764, 2019.

[103] Z. Yao, S. Cao, W. Xiao, C. Zhang, and L. Nie. Balanced sparsity for efficient dnn inference on gpu. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5676–5683, 2019.

[104] Z. Ye, Q. Guo, Q. Gan, X. Qiu, and Z. Zhang. Bp-transformer: Modelling long-range context via binary partitioning. *arXiv preprint arXiv:1911.04070*, 2019.

[105] C. Yun, S. Bhojanapalli, A. S. Rawat, S. J. Reddi, and S. Kumar. Are transformers universal approximators of sequence-to-sequence functions? *arXiv preprint arXiv:1912.10077*, 2019.

[106] C. Yun, Y.-W. Chang, S. Bhojanapalli, A. S. Rawat, S. J. Reddi, and S. Kumar. $o(n)$ connections are expressive enough: Universal approximability of sparse transformers. In *Advances in Neural Information Processing Systems*, 2020.

[107] H. Zhang, C.-L. Hung, M. Liu, X. Hu, and Y.-Y. Lin. Ncnet: Deep learning network models for predicting function of non-coding dna. *Frontiers in genetics*, 10, 2019.

[108] J. Zhang, Y. Zhao, M. Saleh, and P. J. Liu. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. *arXiv preprint arXiv:1912.08777*, 2019.

[109] X. Zhang, J. Zhao, and Y. LeCun. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657, 2015.

[110] J. Zhou and O. G. Troyanskaya. Predicting effects of noncoding variants with deep learning–based sequence model. *Nature methods*, 12(10):931–934, 2015.

[111] Y. Zhu, R. Kiros, R. Zemel, R. Salakhutdinov, R. Urtasun, A. Torralba, and S. Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *IEEE international conference on computer vision*, pages 19–27, 2015.