AI

MDPI

Review

End-to-End Transformer-Based Models in Textual-Based NLP

Abir Rahali *,† **and Moulay A. Akhloufi** *,†

Perception, Robotics, and Intelligent Machines Research Group (PRIME), Department of Computer Science, Université de Moncton, Moncton, NB E1A 3E9, Canada

* Correspondence: ear4587@umoncton.ca (A.R.); moulay.akhloufi@umoncton.ca (M.A.A.)

† These authors contributed equally to this work.

Abstract: Transformer architectures are highly expressive because they use self-attention mechanisms to encode long-range dependencies in the input sequences. In this paper, we present a literature review on Transformer-based (TB) models, providing a detailed overview of each model in comparison to the Transformer's standard architecture. This survey focuses on TB models used in the field of Natural Language Processing (NLP) for textual-based tasks. We begin with an overview of the fundamental concepts at the heart of the success of these models. Then, we classify them based on their architecture and training mode. We compare the advantages and disadvantages of popular techniques in terms of architectural design and experimental value. Finally, we discuss open research, directions, and potential future work to help solve current TB application challenges in NLP.

Keywords: Transformers; deep learning; natural language processing; transfer learning

1. Introduction

NLP as the practical and applied aspect of computational linguistics [1] combines linguistics and artificial intelligence (AI). In other words, it is a technique for handling, deciphering, and understanding enormous amounts of text data. There are numerous NLP tasks that can be used in numerous domains and languages [2–4], including healthcare [5–8], financial services [9], and social media [10–14]. Massive datasets, deep neural network architectures, and specialized tools made it possible to solve problems more quickly and accurately. This increase in data accessibility across all domains has benefited deep neural networks (DNNs) [15] and improved the performance of the models. Nowadays, DNNs are frequently used to complete various tasks and serve as the fundamental building block of AI systems. Generally, DNNs have revolutionized image, text, and audio processing. By incorporating various types of networks, DNNs have significantly improved the state-of-the-art in a range of NLP tasks and applications. Recurrent neural networks (RNNs) enable computational models with multiple layers to learn input representations with different levels of abstraction [16], making them particularly effective at processing sequential data, such as text. RNNs also use recurrent cells to analyze sequential or time-series input. Long short-term memory units (LSTMs) [17], an RNN architecture that is frequently used, differ from traditional feed-forward neural networks [18] in that they have feedback connections. An LSTM is used to find patterns in input data sequences, such as numerical time-series data. Because they specify a temporal dimension to take time and sequence into account, RNNs set themselves apart from convolutional neural networks (CNNs) [19,20]. CNNs use a number of structural components, such as fully connected layers, convolution layers, and pooling layers, to automatically backpropagate the spatial hierarchies of input features in shift-invariant data, such as images. RNNs have the drawback of being slow and sequential. Additionally, they are unable to capture longer dependencies because of vanishing gradients. RNNs also assign the same weight to every word sequence relative to the currently processed word. Additionally, because sequence activations are collected in one vector, the learning process loses track of words that have already been fed to it.

Citation: Rahali, A.; Akhloufi, M.A. End-to-End Transformer-Based Models in Textual-Based NLP. *AI* **2023**, *4*, 54–110. https://doi.org/10.3390/ai4010004

Academic Editor: José Manuel Ferreira Machado

Received: 3 October 2022
Revised: 9 November 2022
Accepted: 17 December 2022
Published: 5 January 2023

Recent breakthroughs in the field of NLP have been made possible by a new deep learning (DL) architecture that has gained popularity. Vaswani et al. [21] suggested the concept of the Transformer. They significantly advanced the quality of the research on DL and NLP. The model architecture has demonstrated exceptional efficiency for typical NLP tasks. The Transformer is a kind of neural network that primarily makes use of the self-attention mechanism [22] to extract intrinsic features and has a great deal of promise for widespread use in AI applications. On a variety of NLP tasks, using the attention mechanism outperforms traditional CNN and RNN models [23–25]. The main goal of attention is to enable each word to react to inputs in a unique manner based on similarity scores. The model can use attention between sequences or even within one sequence in the case of self-attention. Transformer-based (TB) models have been used extensively to solve problems involving sequential data. Parallel to this, cutting-edge research studies have developed strategies to successfully overcome the limitations of convolutional DNN architectures. By paying attention to lengthy sequences, Transformers can easily achieve rapid learning, where the data that must be gradually learned over millions of training steps are stored in weight matrices. A Transformer can store information as key-value pairs in long-term memory and then later retrieve that information by creating a query that takes that information into account. By using attention as a method of information retrieval, the model is able to look up information that it has already seen. As a DNN model, the Transformer [21] also requires a large amount of data for training. However, these large datasets are not always accessible. This frequently happens with many challenging NLP tasks. Consider for example neural networks for machine translation, where it may be impossible to curate such vast datasets, particularly for languages with limited resources or for NLP domains where datasets are not frequently collected in large amounts. Examples of NLP tasks include text categorization, text generation and summarization, keyword searching, machine translation, named-entity recognition, information retrieval, and question answering [26,27]. DL models' high computational resource requirements are another disadvantage. Because of these barriers, researchers have questioned whether large, highly trained models can effectively transfer knowledge. The need for transfer learning is growing as more large models are developed.

Transfer learning has been widely utilized recently in NLP. The most popular technique for transfer learning is sequential fine-tuning. These trained language models (LM) for knowledge communication have helped to solve many textual-based NLP processing problems [28–31]. The process of language modeling entails learning a probability distribution over a collection of tokens taken from a predetermined vocabulary. Transfer learning is a procedure or activity where knowledge gained from unlabeled data can be applied to tasks with a small labeled dataset. NLP Transformers have become more accurate over time compared to machine learning (ML) and DL methods [32]. Transfer learning is an alternative to active learning and supervised learning that can be used to achieve high performance with less human supervision. For example, with a little adjustment, a system that has been trained to classify legal documents could also be used to classify financial records. For this, less training time and data will be needed. The focus theme of this survey runs throughout Transformers and its applications [33,34].

NLP uses machine learning algorithms to process speech and text. End-to-end speech processing techniques such as automatic speech recognition (ASR), speech translation (ST), and text-to-speech (TTS) have all made extensive use of sequence-to-sequence models. The recent incorporation of TB models in speech processing boosted the results. There are TB models that are significantly influencing speech-based tasks, such as TransformerTTS for text-to-speech proposed by Li et al. [35] and Wav2Vec 2.0 for speech recognition proposed by Baevski et al. [36]. It is an end-to-end speech recognizer that does not require data alignment nor the need to model word pronunciation. It does not matter if these tokens are graphemes, phonemes, word fragments, or other speech units; only one model converts the input's raw audio signal into the output's sequence of tokens. However, the main and

sole focus of our review is the text-related TB models for textual-based NLP tasks, covering only sequences in text format.

Among related works for Transformers survey papers, Tay et al. [37] proposed a study about the efficiency of Transformers both in terms of memory and computation. The taxonomy of efficient Transformer models, which is defined by technical innovation and major use cases, is their key contribution. By confronting the quadratic complexity issue of the self-attention mechanism, they modeled the developments and architectural innovations that improve the efficiency of Transformers and set lists of general improvements and various efficiency improvements in later parts. Lin et al. [38] provided a review of the Transformer and its variants by organizing the TB models according to their proposed taxonomy, which is mainly based on the methods used for improving the vanilla Transformer: architecture modification, pre-training, and applications. Kalyan at al. [39] proposed AMMUS, a survey for TB pre-trained models in NLP. The authors gave a brief overview of self-supervised learning and explained key concepts such as pre-training methods, pre-training tasks, embeddings, and downstream adaptation methods. They proposed taxonomy and provided a summary of various used benchmarks in the field. Gillioz et al. [40] proposed a summary of the most latest models along with auto-encoder architectures such as BERT. They also addressed auto-regressive models such as GPTs [41–43] and XLNet [44] as well as a number of post-BERT models such as ERNIEs [45,46].

In this survey, we highlight the most recent advances, summarize them, and place them in the appropriate category of TB models. We categorize and analyze the literature using our proposed taxonomy and provide a summary of various used benchmarks in the field. We focused only on recent papers with a reasonable number of citations because there are so many papers related to NLP. This review is distinguished by a broad and precise selection of contemporary TB with the goal of understanding its structures and providing an organized and comprehensive assessment of existing work from a variety of fields. We investigate TB models with specific applications in NLP with the goal of bridging research at various levels. We also go over the architectures of these models in depth and show how they relate to one another. This review is carried out as follows: Initially, we discuss the history of Transformer design and why it is more efficient than other models. Second, we explain the proposed taxonomy in detail by describing the TB architectures, including detailed explanations of each suggested model and its contributions. Third, we examine the TB applications in NLP in terms of task, domain, and language, each in its own area. Then, we go over the most important points, constraints, evaluations, and highlights from our review. Finally, we discuss several areas that could be further explored in the future.

2. Background

The evolution of different NLP models can be divided into three categories based on their primary architecture: RNNs, CNNs, and attention-based. RNNs can be regarded as traditional due to the recent replacement of recurrent-based models by parallelized architectures such as attention-based and CNN-based models. Transformers are an example of an attention-based model. TB language models have advanced and outperformed traditional language models at this point [47,48]. Surafel et al. [49] reported that the Transformer method generates the best performing multilingual models, outperforming corresponding bilingual models and RNNs. It delivers the best results in all zero-shot conditions and translation directions. Generally, NLP deals with sequence-to-sequence (S2S) tasks [50]. And an encoder-decoder model is used to carry out these tasks. The most frequently employed encoder and decoder architectures were RNNs. However, there are some drawbacks to these architectures. The limitations are described in this section, followed by a definition of the pre-training and fine-tuning framework, and an explanation of how the Transformer design has been used to address the limitations in the reviewed models. Next, we discuss the suggested taxonomy for this survey.

2.1. Recurrent Neural Network (RNN)

RNNs are a class of neural networks for processing sequential data that allow previous outputs to be used as inputs while maintaining hidden states, which is also known as sequence modeling. RNNs can handle input of any length, and the model size does not change as the input size does. Additionally, historical data and weights that have remained constant over time are taken into account during model computation. Text generation, machine translation, and time-series classification are a few examples of the tasks that RNNs excelled at.

RNNs, however, have a flaw in their architecture. RNNs, like many other ML algorithms, are enhanced using back-propagation, and the error diminishes significantly as it passes back through the recurrent layers because of their sequential design. To assist RNNs in overcoming their issues, numerous strategies have been proposed. One suggestion was to substitute the ReLU (Rectified Linear Unit) [15] for the sigmoid function. Another suggestion is to employ LSTMs, which are composed of a number of units, each with a different number of gates including the input, output, and forget gates, in the design. Additionally, every component produces a state that can be applied to the subsequent input in the chain.

In order to enhance the network's knowledge in both forward and backward directions, bidirectional LSTMs were created. Despite being able to handle lengthy sequence dependencies, this design has the obvious disadvantage of being incredibly slow because of the numerous parameters that must be trained. Gated recurrent networks (GRUs) [51] were created as a faster version of LSTMs. GRUs require only two gates, which accounts for their speed. Additionally, the GRU architecture performs better than LSTMs in some tasks, such as automatically detecting grammatical features of input texts.

Generally, RNNs' architectures including LSTMs and GRUs have the mentioned number of drawbacks: slow computation, the inability to take future inputs into account when determining the current state, and the additional challenge of accessing information from the past. Most importantly, RNN-based architectures are challenging to parallelize because the forward propagation graph is inherently sequential at each time step and can only be computed after the previous one. As a result, the runtime and memory cost are fixed and cannot be decreased because the states computed in the forward pass must be sorted before they are re-used during the backward pass.

2.2. The Encoder–Decoder Framework

The S2S model typically uses an encoder–decoder architecture. This architecture is made up of an encoder, which analyzes the input sequence and compresses the data into a context vector of a fixed length, and a decoder, which converts the encoded state of a fixed shape to a variable-length sequence. The main drawback of this fixed-length context vector design is that it is impossible to remember long sentences, whereas the decoder requires different information at different time steps. Figure 1 shows the main architecture of the encoder–decoder model with the attention mechanism.

There are two well-known problems with the traditional encoder–decoder structure. The encoder must first compress all of the input data into a single fixed-length vector before sending it to the decoder because sending lengthy and intricate input sequences with a single fixed-length vector runs the risk of losing information. Second, it is unable to depict the input–output sequence alignment that is necessary for tasks requiring a structured output, such as translation and summarization. Each output token in S2S tasks should be more influenced by particular segments of the input sequence. Although each output token is created by the decoder, it is unable to selectively concentrate on significant input tokens.

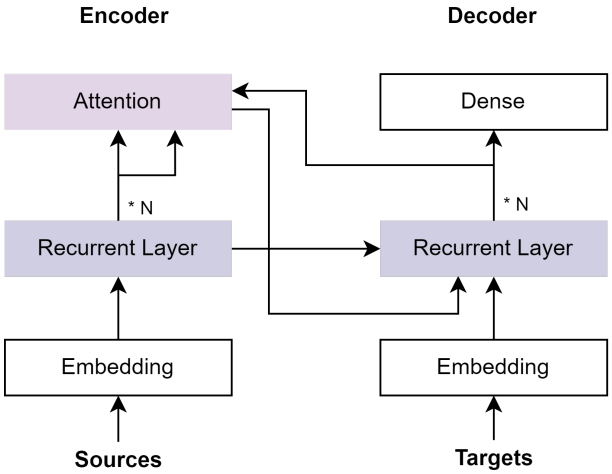

Figure 1. S2S encoder–decoder with attention architecture.

2.3. The Transformer

The Transformer was an idea put forth by Vaswani et al. [21] to address the shortcomings of both RNNs and encoder–decoder architectures. By substituting attention mechanisms for the RNNs in the S2S encoder–decoder, they improved the architecture. A very long-term memory is made possible by attention. The Transformer concentrates on every token that has ever been created in the past. Overall, this architecture consists of feed-forward layers (position-wise feed-forward), residual connections, and normalization layers, which are all stacked on top of one another as multiple multi-head attention layers.

2.3.1. Attentions Mechanism

The model uses the attention mechanism to concentrate on pertinent information based on what it is currently processing. The decoder state (Q: query) and the encoder hidden states (K: keys) were used to calculate the attention weights, which historically represented the relevance of the encoder hidden states (V: values) in processing the decoder state. As a result, a generalized attention model called $Attention(Q, K, V)$ uses a set of key-value pairs (K, V) and a query called Q, as Equation (1) shows.

$$Attention(Q, K, V) = F_{distribution}(F_{alignment}(K_i, Q)) * V_i \quad (1)$$

where the distribution function and the alignment function are denoted by $F_{distribution}$ and $F_{alignment}$, respectively. In order to create the attention weights, the distribution function maps the alignment scores from the alignment function. The logistic, sigmoid, and SoftMax methods are the distribution functions that are most frequently used in TB models. To learn the attention weights, an additional feed-forward neural network is incorporated into the architecture of the Transformer. This feed-forward network learns a particular attention weight as a function of the inputs of two states. The importance of the encoder hidden state for the decoder hidden state is measured by the alignment function. Additionally, it produces energy scores, which are later used to generate attention weights by the distribution function. There are many different kinds of alignment functions. Table 1 shows the formulations of various alignment functions.

In the original Transformer's [21] encoder–decoder framework, the dot product alignment is an example of a multiplicative alignment type used in the model. In cases where the key and query share the same vector space for representation, computing the cosine similarity or dot product between the key and query representations can be applied. To account for various representation lengths, a scaled dot product can be used, which normalizes the dot product by the representation vector length. General alignment extends the dot product to keys and queries with multiple representations by offering a learnable transformation matrix W that maps queries to the vector space of keys. Biased general alignment calculates the global significance of specific keys regardless of the query. The

general alignment is activated by adding a nonlinear activation layer, such as a hyperbolic tangent, rectifier linear unit, or scaled exponential linear unit. It is possible to match the key and query using a general kernel function rather than the dot product. Concatenate alignment is another option, in which keys and queries are combined to create a single shared representation. The computational time required for additive alignment is lowered by separating the contributions of the query and the key. This enables all key contributions to be computed in advance rather than having to do so for each query. Location-based alignment ignores keys and only uses queries Q. As a result, the key's position rather than its content is used to determine the alignment score for each key. The input to the function for feature alignment is derived from features such as mean and standard deviation when working with collections of items such as one-dimensional temporal sequences.

Table 1. List of common attention types used in reviewed TB models with corresponding alignment functions where *sim* is the measure of similarity between K_i and Q vectors and $||K_i||$ and $||Q||$ are the Euclidean norm of these vectors.

Type	Alignment	Score Function
Multiplicative Attention	Dot Product	$F_{alignment}(K_i, Q) = Q^T K_i$
	Scaled Dot Product	$F_{alignment}(K_i, Q) = \frac{Q^T K_i}{\sqrt{d_K}}$
	Cosine Similarity	$F_{alignment}(K_i, Q) = sim(K_i, Q) = \frac{K_i Q}{\|\|K_i\|\|\|\|Q\|\|}$
Addictive Attention	Concating	$F_{alignment}(K_i, Q) = w_{imp}^T F_{activation}(W[Q; K_i] + b)$
	Additive	$F_{alignment}(K_i, Q) = w_{imp}^T F_{activation}(W_1 Q + W_2 K_i + b)$
Scoring Attention	General	$F_{alignment}(K_i, Q) = Q^T W K_i$
	Biased General	$F_{alignment}(K_i, Q) = k_i(W_Q + b)$
	Activated General	$F_{alignment}(K_i, Q) = A(Q^T W K_i + b)$
Specific Attention	Location Based	$F_{alignment}(K_i, Q) = F_{alignment}(Q)$
	Feature Based	$F_{alignment}(K_i, Q) = w_{imp}^T F_{activation}(W_1 \phi_1(K) + W_2 \phi_2(K) + b)$

The original Transformer model used scaled dot product attention as an alignment function and SoftMax as a distribution function, so the attention is calculated as shown in Equation (2). Meanwhile, self-attention, also referred to as inner attention, is a mechanism for focusing attention that links various positions in a single sequence in order to create a representation of the sequence. Figure 2 shows the architecture of the self-attention mechanism, where d refers to the dimension of the keys K.

$$Attention(Q, K, V) = softmax(\frac{Q^T K}{\sqrt{d_K}}) * V \quad (2)$$

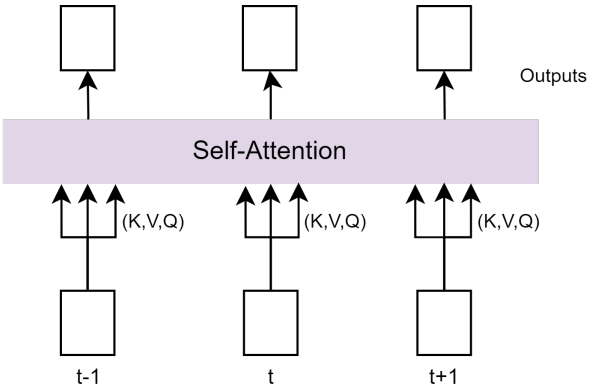

Figure 2. Self-attention architecture.

The multi-head attention layer [21] is composed of multiple attention heads. Each attention head computes attention over its inputs of V, K, and Q, while subjecting them to a linear transformation. As a result, the model can jointly pay attention to data from various representational sub-spaces. Instead of performing a single attention pass over the values, the multi-head attention block computes multiple attention-weighted sums. The multi-head attention applies various linear transformations to the V, K, and Q for each head of attention in order to learn diverse representations. An attention layer with multiple heads has N parallel attention layers, each of which is referred to as a head. The queries, keys, and values are projected onto three dense layers for each head with hidden sizes of q, k, and v before being fed into the attention layer, respectively. Another dense layer projects the results of these N heads after they have been concatenated. Figure 3 shows the architecture of the multi-head attention mechanism.

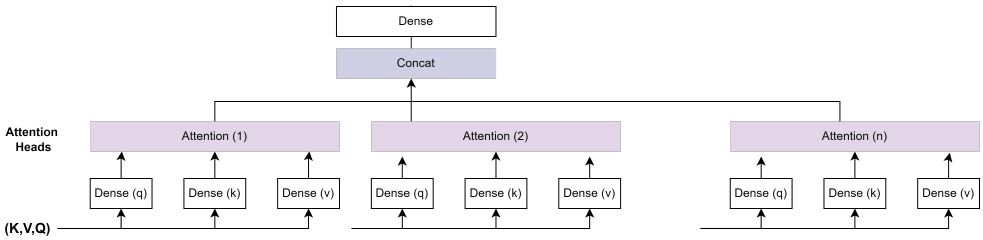

Figure 3. Multi-head attention architecture.

The masked multi-head attention network [21] performs a similar function to the decoder hidden state in machine translation architectures by attending over the previous decoder states. The inputs from future time-steps to the decoder are hidden, which is why this is known as the masked multi-head attention block. In both the encoder and decoder frameworks, the TB models make use of stacked multi-head self-attention blocks. For the multi-layer Transformer, the input vectors $\{x_i\}_{i=1}^{|x|}$ are first packed into $H^0 = [x_1, \ldots, x_{|x|}]$, and then translated into context-specific representations at various levels of abstraction $H^l = [h_1^l, \ldots, h_{|x|}^l]$ using an L-layer Transformer $H^l = Transformer_l(H^{l-1}), l \in [1, L]$. Each Transformer block aggregates the output vectors of the preceding layer using a number of self-attention heads. The output of a self-attention head A_l is calculated using the following Equations (3)–(5) for the lth Transformer layer:

$$Q = H^{l-1}, K = H^{l-1} * W_l^K, V = H^{l-1} * W_l^V \tag{3}$$

$$M_{i,j} = \begin{cases} -\infty, & \text{if prevent from attending} \\ 0, & \text{if allow to attend} \end{cases} \tag{4}$$

$$A_l = softmax(\frac{Q * K^T}{\sqrt{d_k}} + M) * V_l \tag{5}$$

2.3.2. Positional Encoding (PE)

Each sequence in language modeling has a fixed order of tokens in it. RNNs automatically encode their positions when they operate. Nevertheless, the attention mechanism ignores any details regarding the placement of each word. In contrast, attention-based models allow for the treatment of encoded words out of order, which could produce a randomization effect. Unlike recurrent networks, the multi-head attention network cannot naturally take advantage of the order of the words in the input sequence. One straightforward technique is to encode each word according to where it is in the current sequence. Following the embedding of each word using an embedding matrix, PE extracts the positions of each word using the following equations:

$$PE(pos_{word}, 2 * pos_{emb}) = \sin(\frac{pos_{word}}{\frac{10{,}000^{2*pos_{emb}}}{d_{model}}}) \quad (6)$$

$$PE(pos_{word}, 2 * pos_{emb} + 1) = \cos(\frac{pos_{word}}{\frac{10{,}000^{2*pos_{emb}}}{d_{model}}}) \quad (7)$$

where d_{model} is the embedding dimension and pos_{word} is the position within the sequence which takes the values $[0, \ldots, n-1]$ and pos_{emb} is the position within the embedding dimension which takes values from 0 to $d_{model} - 1$. PE handles the value position, as implied by its name, so the Transformer adds embeddings to the word embeddings that show the input position. PE vectorizes the input locations, which are then incorporated into the input embeddings. For instance, the Transformer-XL [52] model made use of relative positional encoding. Relative position encodings are a type of position embedding for TB models that seeks to exploit pairwise, relative positional information. On two levels, the model obtains relative positioning information: values and keys. In addition to the keys, the model is given relative location information. SoftMax's Vanilla self-attention functionality is unaffected. The values matrix is then subdivided into further relative positioning data. In other words, rather than just combining semantic and absolute positional embeddings, relative positional information is added to keys and values during attention calculation.

2.3.3. Position-Wise Feed-Forward Networks

The term position-wise refers to a feed-forward layer type that uses the same dense layers for each position in the sequence. Position-wise feed-forward layers are composed of two dense layers that apply to the last dimension. It accept input in the form of a three-dimensional shape (batch size, sequence length, and feature size). It consists of two dense layers that are applied to the final dimension; thus, position-wise, the same dense layers are applied to each position in the sequence. As shown in Figure 4, the position-wise feed-forward networks (FFN) block has two completely connected layers. The activation function at the hidden layer is usually set to ReLU activation. Sometimes, the GELU (Gaussian Error Linear Unit) [53] activation can be used instead of ReLU. The FFN function is defined in Equation (8) where W_1, W_2, b_1, b_2 are learnable parameters.

$$FFN(x, W_1, W_2, b_1, b_2) = max(0, x * W_1 + b_1) * W_2 + b_2 \quad (8)$$

2.3.4. Full Transformer Encoder–Decoder

There are two blocks in the encoder. The multi-head attention layer over the aforementioned inputs is one block. A straightforward feed-forward network is the other. There is a residual connection between each layer, which is followed by layer normalization. In DL, a normalization technique akin to batch normalization is called layer normalization [54]. In the Add and Layer Norm component of Figure 4, layer normalization comes after a residual connection. Because layer normalization normalizes across the feature dimension, it benefits from scale independence and batch size independence, just like batch normalization does. Layer normalization is efficient because inputs for tasks involving natural language processing frequently consist of variable-length sequences.

There are three blocks in the decoder. A stack of numerous identical layers bound together by residual connections and normalized along layer boundaries makes up the transformer decoder as well. Between the two blocks the encoder has, the decoder adds a third block known as the encoder–decoder attention layer. Even though the decoder only has one multi-head attention layer known as the masked multi-head attention network, it is very similar to the encoder in other ways. The decoder can review the entire sentence and pick out the details it needs to decode with the help of the attention mechanism. The encoder's hidden states are all accessible to the decoder through attention. However, since only one prediction for the following word needs to be made by the decoder, a complete sequence cannot be passed. However, in order to predict the subsequent word, it uses a

weighted sum of hidden states. Figure 4 shows the full architecture of the Transformer encoder–decoder blocks.

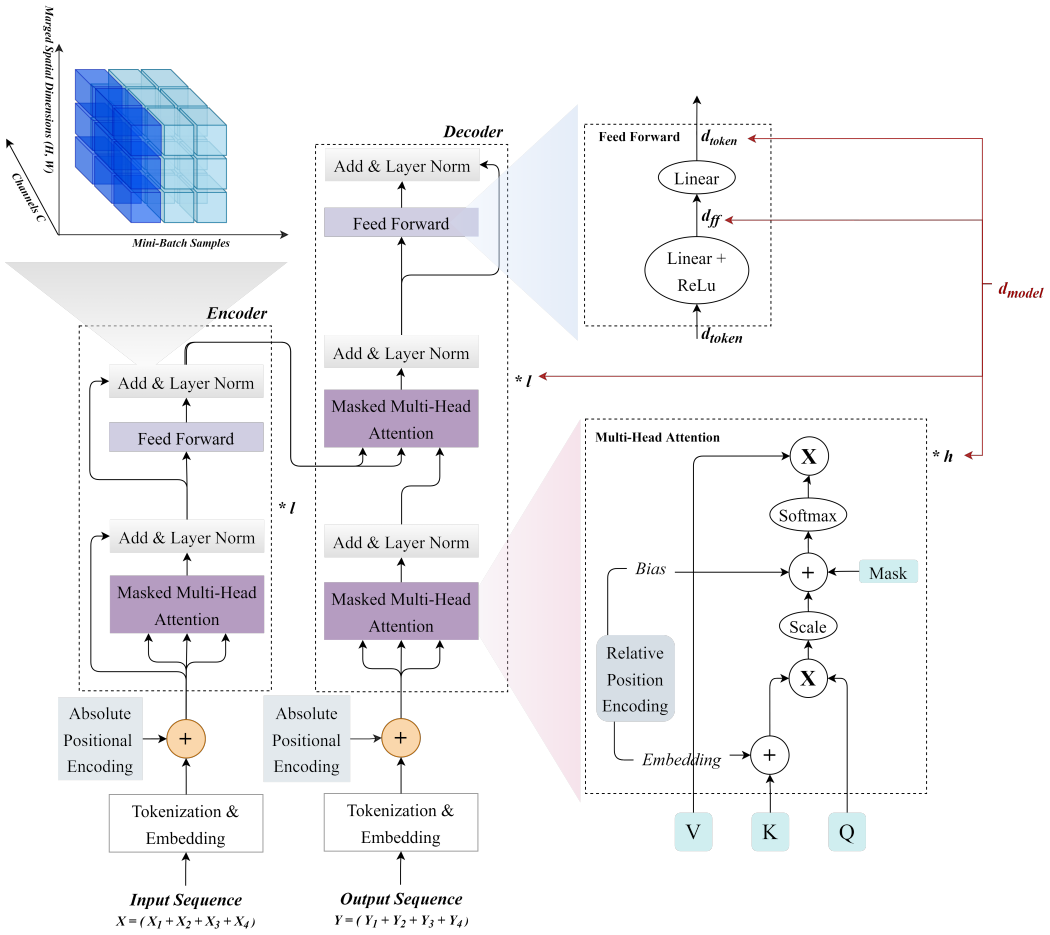

Figure 4. A diagram showing the detailed transformer architecture, which has 2 blocks for the encoder and 3 blocks for the decoder. Attention mechanisms, normalization, and feed-forward layers were the three main concepts on which we focused our attention. In the Feed-forward layer, the number of dimensions in the hidden layer d_ff is generally set to around four times that of the token embedding d_{model}. The model size $d_{model} = (l, d_{token}, d_{ff}, h)$ where l is the number of encoder/decoder layers, d_{token} is the dimensionality of token embeddings, d_{ff} is the intermediate dimensionality used by the feed-forward sublayer, and h is the number of attention heads in the attention sublayers. The initial Transformer model had a base configuration of $l = 6$, $d_{token} = 512$, $d_{ff} = 2048$, and $h = 8$.

2.4. The Pre-Train and Fine-Tune Framework

We describe the pre-training and fine-tuning framework used in the majority of TB models in this section. A number of advances in language representation learning had been made through the use of pre-training procedures. These pre-trained models have been extremely helpful for a variety of challenging NLP tasks, including those with little training data. Pre-training large models and reducing them to smaller ones for practical applications has become a standard procedure. Pre-training and fine-tuning phases are typically distinguished when using TB models. The model is trained on a sizable generic corpus during the pre-training phase. Then, the model is modified during the fine-tuning phase to fit a specific task or dataset. Modern TB models differ significantly from one another in terms of their pre-training model architectures and objectives. Typically, the following steps are taken to develop a new TB model: gathering a sizable dataset and then pre-training, which entails training the tokenizer and the model. After that, the trained model's fine-tuning phase for a downstream task will begin. When used to be

updated for a new, smaller dataset or task, this trained model is referred to as pre-trained. Consequently, we use the pre-trained model and modify its weights to fit a smaller dataset during the fine-tuning stage. The model's weights are updated in both pre-training and fine-tuning. Pre-training generally enables the development of a general representation that can later be useful in the specialization of a downstream task, especially when the data for the downstream task are extremely sparse and would not be sufficient to train a model. Algorithm 1 sums up the steps for the pre-training and fine-tuning framework.

Algorithm 1: Pre-training and fine-tuning steps for a TB model.

Data: You have a large dataset A and a small dataset B
Result: How to pre-train then fine-tune a TB language model m
while *Pre-training phase* **do**
    You select a TB model m and the pre-training objectives;
    You have a dataset A;
    Initialize some of m parameters;
    Train m on A;
end
while *Fine-tuning phase* **do**
    Let trained m on A as $m_{trained}$;
    You have a dataset B;
    Train $m_{trained}$ on B;
end

2.4.1. Pre-Training Procedure

The pre-training procedure includes training a tokenizer and then training a TB language model. Rather than training a model to look up fixed dictionaries, language models should be able to recognize text and learn from it. They accomplish this through the use of a system known as tokenization, in which text sequences are broken down into smaller parts, or tokens, and then fed as input into TB NLP models such as BERT [3]. Tokenization or word representations help downstream tasks such as document classification by allowing for cross-word generalization. The use of distributed representations as features is a type of semi-supervised learning in which supervised learning performance is supplemented by learning distributed representations from unlabeled data.

Traditional word embeddings, such as Word2vec [55] or GloVe [56], differ from the new contextualized embeddings used in TB models. For example, BERT [3] employs WordPiece tokens. The vocabularies of BERT-based models differ. The uncased base model, for example, has 994 tokens set aside for possible fine-tuning. The cased model has only 101 unused tokens because it requires more tokens to cover both uppercase and lowercase letters. Pre-trained BERT comes with a predefined vocabulary of size 30,522, which is also enforced on the pre-trained BERT tokenizer that was applied. The multilingual model mBERT [3] has only 100 unused tokens, but its total vocabulary is four times larger than that of uncased. More special characters are stored in the multilingual model to cover the words of 104 languages. The GPT2 [42] and RoBERTa [57] implementations share a vocabulary of 50,000 words. They use Byte Pair Encoding (BPE) word pieces with the special signaling character \u0120. In contrast to the preceding vocabularies, the XLM [58] employs a suffix signaling tag at the end of the word piece to indicate that this is the end of a word. XLM employs a BPE-based vocabulary that is shared by multiple languages and models. These pre-trained tokenizers can be used in a linear prediction model as features. The performance of the downstream systems that consume word representations can be used to evaluate them. Unfortunately, extrinsic and intrinsic evaluations do not always agree, and the best word representations for one downstream task may perform poorly on another.

The most commonly used tokenization types are: Byte Pair Encoding (BPE) [59], Byte Level BPE (bBPE) [42], SentencePiece [60] and WordPiece [61]. While it is true that a bBPE tokenizer trained on a huge monolingual corpus can tokenize any word of any language, it requires on average almost 70% of additional tokens when it is applied to a text in a language different from that used for its training. This information is key when it comes to choosing a tokenizer to train a natural language model such as a Transformer model. Tokenization in the context of Transformer models is divided into two steps. When creating a Transformer tokenizer, two files are typically generated: *merges.txt* and *vocab.json*. These are the two steps in the tokenization process. The first step is to read the dataset's input text in string format. The first file, *merges.txt*, is used to convert words or word fragments to tokens. After the tokens are generated, in the second step, the tokens are processed through the second file *vocab.json*, which is simply a mapping file from token-to-token ID. These token IDs are read by an embedding layer in the Transformer model during pre-training, which maps the token ID to a dense vector representation of that token. In order to currently and accurately detail the pre-training objectives for the TB models, we highlight accordingly the notations used to describe specific concepts in Table 2 including text corruption, sentence pair, sample efficient task and code-switched.

Definition 1. ***Text corruption:*** *We have four levels when processing a textual-based data: token, phrase, sentence and document. Text corruption refers to the changes made to the text during processing. It can occur at the token level through token deletion and token masking, at the phrase level through phrase infilling, at the sentence level through sentence permutation, or at the document level through document rotation. A corrupted token x is referred to as x^{corr}, and when the corruption is masking, we noted the masked x as x^{Masked}.*

Definition 2. ***Sentence pair:*** *Here, the input is a pair of sentences (x, y) where x and y are parallel sentences, i.e., x is a translation of y.*

Definition 3. ***Sample efficient task:*** *The sample efficient task avoids discrepancy between pre-training and fine-tuning stages by using 100% of tokens in each training sample for learning. We note a non-sample efficient task as a task that uses a smaller corruption rate: for example, the tasks that corrupt only 50% or 15% of the input tokens.*

Definition 4. ***Code-switched:*** *For a given parallel sentence pair (x, y), a code-switched sentence is generated by randomly substituting some phrases of x with their translations from y.*

While completing one or more predefined tasks, TB models learn various universal language representations. Self-supervised, pseudo-labeled data are used in pre-training tasks. The pseudo-labels are determined by the data attributes and the pre-training task definition. To give the model more training signals, a pre-training task should be difficult enough. Furthermore, a pre-training task ought to resemble a downstream task. The pre-training tasks employ various corruption techniques with a fixed corruption rate, such as masking, replacement, or swapping of tokens. Additionally, they can differ based on how the sequence should be handled. For example, on one hand, GPT1 [43] uses a left-to-right Transformer, while BERT [3] uses a bidirectional Transformer, where two special tasks have been developed for BERT pre-training: the Masked LM (MLM) task, in which the model predicts random masked words in a sentence, and the Next Sentence Prediction (NSP) task, in which the model predicts whether two sentences appear next to each other. Meanwhile, the pre-training tasks provide the model with general language understanding, which has been shown to improve ranking significantly. Table 2 shows the list of pre-training tasks along with its loss functions.

- **Masked Language Modeling (MLM):** Through the use of unique mask tokens, MLM masks the sentence. The SoftMax layer receives the masked token vectors from MLM and computes the probability distribution over the vocabulary before applying the

cross-entropy loss. MLM makes use of tokens from both settings. The original tokens are predicted by MLM using masked token vectors. MLM is approached as a task for classifying tokens at the token level over the masked tokens. It has two drawbacks. Since only 15% of the tokens in each training sample are masked token vectors, masked token prediction only uses a small portion of the training signal. Additionally, there is a pause between the pre-training and fine-tuning stages because the model only sees the unique mask token during pre-training.

- **Causal Language Modeling (CLM):** Alternatively known as unidirectional LM, CLM predicts the next word based on the context. Both a left-to-right and a right-to-left sequence can be handled by CLM. All of the words on the left are included in the context of a CLM that reads from left to right, whereas all of the words on the right are included in a CLM that reads from right to left. Each training sample used in CLM contains 100% tokens for learning. Because it cannot be used in both contexts, CLM has this major drawback. A bidirectional context cannot be used to train standard CLM because doing so would allow a token to see itself, which would make prediction easy.
- **Translation Language Modeling (TLM):** Alternatively known as XMLM and also referred to as cross-lingual MLM, in TLM, random masking is applied to tokens from both sentences. TLM aids the model's cross-linguistic mapping learning because the prediction of masked tokens requires context from both sentences. Only 15% of the tokens in each training sample is used by TLM.
- **Replaced Token Detection (RTD):** It indicates which tokens were replaced. Using the tokens produced by the generator model trained using MLM objectives, RTD corrupts the sentence. RTD is a token-level binary classification task that asks the model to decide whether or not each token has been replaced. For learning purposes, RTD uses 100% of the tokens in each training sample. The only problem with RTD is that it needs a separate generator to tamper with the input sequence, which is computationally expensive to train. Despite this, RTD is sample efficient.
- **Shuffled Token Detection (STD):** Identification of the shuffled tokens is a task that requires token-level discrimination. The words are randomly shuffled in STD with a 0.15 probability. For learning purposes, STD uses 100% of the tokens in each training sample.
- **Random Token Substitution (RTS):** It involves identifying the randomly substituted tokens. Here, 15% of the tokens in RTS are at random replaced with different tokens from the vocabulary. Since RTS is sample-efficient, it can corrupt the input sequence without the need for a separate generator model. In each training sample used in RTS for learning, tokens are used 100% of the time.
- **Swapped Language Modeling (SLM):** With a 0.15 probability, it tampers with the sequence by adding tokens that are randomly selected from the vocabulary. Since only 15% of the input tokens are used, SLM is not sample efficient.
- **lternate Language Modeling (ALM):** The task of cross-linguistic language model training is a pre-training task. Predicting the masked tokens in the code-switched sentences produced from parallel sentences is the goal of ALM. The settings for masking the tokens in ALM are the same as those in MLM. The model learns relationships between languages much more effectively by receiving pre-training on sentences that have been code-switched.
- **Sentence Boundary Objective (SBO):** Predicting the masked tokens using the span boundary tokens and position embeddings is part of the pre-training task. When performing tasks that require span-based extraction, such as entity extraction, coreference resolution, and question answering, SBO improves the model's performance. SBO only conceals token spans that are consecutive. Tokens representing span boundaries and position embeddings are used in the prediction of masked tokens.
- **Next Sentence Prediction (NSP):** It is a pre-training task at the sentence level that aids the model in understanding relationships between sentences. Finding consecutive sentences is a part of a binary sentence pair classification task. For training purposes,

the sentence pairs are produced so that 50% of the instances are consecutive and the other 50% are not. The topic and coherence predictions involved in NSP enable the model to learn sentence-level semantics. In order to promote learning, NSP only uses 50% of the tokens in each training sample.

- **Sentence Order Prediction (SOP):** It is a pre-training task at the sentence level that only considers sentence coherence. The training instances are generated using NSP in such a way that 50% of them are switched out while the other 50% are not. SOP only uses 50% of the tokens in each training sample to facilitate learning.
- **Sequence-to-Sequence LM (SSLM):** Both the left-side words in the predicted target sequence and every word in the input masked sequence are included in the context. The encoder inputs the masked sequence, and the decoder predicts the masked words sequentially from left to right. SSLM is sample inefficient because it only reconstructs the masked tokens. Only 15% of tokens are used by SSLM to facilitate learning in each training sample.
- **Denoising Auto Encoder (DAE):** It is an auto-encoder model that learns to forecast the original, uncorrupted data point as its output, after being fed a corrupted data point as input. By reassembling the original text from corrupted text, it aids the model's learning process. Models based on encoder–decoders can be trained using DAE. By offering more training signals for model learning, DAE is more sample effective. Due to the fact that DAE involves reconstructing the entire original text, it offers a stronger training signal. Each training sample used by DAE contains 100% tokens for learning.
- **Segment Level Recurrence (SLR):** When the model processes the following new segment, the representations calculated for the prior segment are fixed and cached for later use as an extended context. By allowing contextual information to cross segment boundaries, this additional connection increases the largest possible dependency length by N times, where N is the depth of the network. The context fragmentation problem is also solved by this recurrence mechanism, giving tokens at the beginning of a new segment the context they require.
- **Gap Sentences Generation (GSG):** Abstractive summarization is accomplished using this pre-training objective. S2S models extract gap sentences and use them for pre-training by creating sentences that are disconnected from the rest of the text and concealing entire sentences. Selecting specific sentences with apparent significance performs better than randomly selecting sentences for the generation.

Table 2. List of pre-training objectives tasks and their corresponding loss functions definition, along with the list of example TB models that used this pre-training objective.

Abbreviations	Objective Task	Loss Function	Design Highlights	Example
CLM	Casual Language Modeling	$Loss^{x}_{CLM} = -\frac{1}{\lvert x\rvert}\sum_{i=1}^{\lvert x\rvert}\log(P(\frac{x_i}{x_{<i}}))$ $x_{<i} = x_1, x_2, x_3, \ldots x_{i-1}$ $x = x_1, x_2, x_3, \ldots, x_N$ where N represents the number of tokens in the sequence.	Token-Level, Predict Next Token, Unidirectional, Sample Efficient	GPT1 [43], UniLM [62]
MLM	Masked Language Modeling	$Loss^{x}_{MLM} = -\frac{1}{\lvert M_x\rvert}\sum_{i\in M_x}\log(P(\frac{x_i}{x^{Masked}}))$ where M_x represents the masked token positions in x	Token-Level, Token Masking, Bidirectional, Not Sample Efficient	BERT [3], RoBERTa [57]
RTD	Replaced Token Detection	$Loss^{x}_{RTD} = -\frac{1}{\lvert x^{corr}\rvert}\sum_{i=1}^{\lvert x^{corr}\rvert}\log(P(\frac{d}{x^{corr}_i}))$ where $d \in \{1, 0\}$ represents whether the token is replaced or not.	Token-Level, Bidirectional, Token Replacement, Sample Efficient	ELECTRA [63]

Table 2. *Cont.*

Abbreviations	Objective Task	Loss Function	Design Highlights	Example
STD	Shuffled Token Detection	$Loss^{x}_{STD} = Entropy(Loss^{x}_{A} + Loss^{x}_{B})$ $Loss^{x}_{A} = \sum_{i=1}^{n} -1(x_i^{shuffled} = x_i) \log(D(x^{shuffled}, i))$ $Loss^{x}_{B} = -1(x_i^{shuffled} \neq x_i) \log(1 - D(x^{shuffled}, i))$ where $x^{shuffled} = [x_1, \ldots, x_n]$ and $h(x) = [h_1, \ldots, h_n]$ and $h_i^{'} = GELU(W_A^T h_i)]$ and $D(x^{shuffled}, i) = \alpha(W_B^T h_i^{'})$ Also, $\{W_A\}$ and $\{W_B\}$ are linear layers parameters and $GELU$ is the activation function [53]	Token-Level, Token Shuffling, Bidirectional, Sample Efficient	Panda et al. [64]
RTS	Random Token Substitution	$Loss^{x}_{RTS} = -\frac{1}{\lvert x^{corr} \rvert} \sum_{i=1}^{\lvert x^{corr} \rvert} \log(P(\frac{d}{x_i^{corr}}))$ where $d \in \{0, 1\}$ represents whether the token is randomly substituted or not.	Token-Level, Random Token Substituted, Sample Efficient	Di et al. [65]
SLM	Swapped Language Modeling	$Loss^{x}_{SLM} = -\frac{1}{\lvert R_x \rvert} \sum_{i \in R_x} \log(P(\frac{x_i}{x^{corr}}))$ where R_x represents the positions of swapped tokens.	Token-Level, Token Swapping, Not Sample Efficient	Di et al. [65]
TLM	Translation Language Modeling	$Loss^{x,y}_{TLM_A} = -\frac{1}{\lvert M_x \rvert} \sum_{i \in M_x} \log(P(\frac{x_i}{x^{Masked}}, y^{Masked}))$ $Loss^{x,y}_{TLM_B} = -\frac{1}{\lvert M_y \rvert} \sum_{i \in M_y} \log(P(\frac{x_i}{y^{Masked}}, x^{Masked}))$ $Loss^{x,y}_{TLM} = Loss^{x,y}_{TLM_A} + Loss^{x,y}_{TLM_B}$ where M_x and M_y represents masked positions and x^{Masked} and y^{Masked} represent the masked version of x and y.	Random Token Masking, Bidirectional, Not Sample Efficient	XLM [58], XNLG [66]
ALM	Alternate Language Modeling	$Loss^{(z(x,y))}_{ALM} = -\frac{1}{\lvert M \rvert} \sum_{i \in M} \log(P(\frac{z_i}{Z^{Masked}}))$ where z is the code-switched sentence generated from x and y, z^{Masked} represents the masked version of z and M represents the set of masked token positions in z^{Masked}.	Sentence-Level, Sentence Code-Switched, Bidirectional, Not Sample Efficient	Yang et al. [67]
SBO	Sentence Boundary Objective	$Loss^{(x,y)}_{SBO} = -\frac{1}{\lvert S \rvert} \sum_{i \in S} \log(P(\frac{x_i}{y_i}))$ $y_i = f(x_{s-1}, x_{e+1}, p_{s-e+1})$ where f is a two-layered feed-forward neural network, S represents the positions of tokens, $\lvert S \rvert$ represents the length of span, s and e represent the start and end positions of span, and p represents the position embedding.	Span-Level, Span Masking, Not Sample Efficient	SpanBERT [68]
NSP	Next Sentence Prediction	$Loss^{(x,y)}_{NSP} = -\log(P(\frac{d}{x,y}))$ Where $d \in \{1, 0\}$ represents whether the sentences are consecutive or not.	Sentence-Level, Consecutive Sentence Swapping, Not Sample Efficient	BERT [3]
SOP	Sentence Order Prediction	$Loss^{(x,y)}_{SOP} = -\log(P(\frac{d}{x,y}))$ where $d \in \{1, 0\}$ represents whether the sentences are swapped or not.	Sentence-Level, Sentence Swapping, Not Sample Efficient	ALBERT [69]
SSLM	Sequence To Sequence LM	$Loss^{x}_{SSLM} = -\frac{1}{\lvert l_s \rvert} \sum_{s=j} \log(P(\frac{x_s}{x^{Masked}, x_{<s-1}}))$ where x^{Masked} is the masked version of x with masked n-gram span and l_s represents the length of the masked n-gram span.	Token-Level, Unidirectional, Token Masking, Not Sample Efficient	T5 [70], mT5 [71], MASS [72]
DAE	Denoising Auto Encoder	$Loss^{x}_{DAE} = -\frac{1}{\lvert x \rvert} \sum_{i=1} \log(P(\frac{x_i}{x_{corr}, x_{<i}}))$ where x^{corr} is the corrupted version of x.	Sentence-Level, Sentence Reconstructing, Noise Reduction, Sample Efficient	BART [73]

2.4.2. Fine-Tuning Procedure

Definition 5. ***Downstream task:*** *A specific task to complete as part of self-supervised learning is a downstream task. Most frequently, this definition is related to transfer learning or self-directed learning. A model is pre-trained with a general dataset that does not represent the downstream task but enables the model to learn some general features in transfer learning in particular.*

The pre-trained model is then modified using the dataset that faithfully represents the issue that needs to be solved. A downstream task is what is referred to as in the context of self-supervised learning for the latter task. Word representations are updated from labeled data in the downstream task, which is when fine-tuning takes place. While fine-tuning involves adapting the model to a particular task or dataset, pre-training involves training the model on a sizable generic corpus. Both pre-training and fine-tuning scenarios technically involve changing the model weights. For example, the pre-trained model can be adjusted for a particular objective such as categorization or question answering. Pre-training, on the other hand, refers to the process of training a pre-training task representation method, such as an unsupervised learning task, especially when a large corpus is used and the target dataset is from a specific domain and contains a few unlabeled data points that may help the model adapt to the domain. The process of fine-tuning involves utilizing the target corpus with the intended task. The pre-training and fine-tuning framework used by the majority of TB models is described in this Algorithm 1. The three stages of the framework can be categorized as training on entirely new data, pre-training on representational techniques, and fine-tuning for a particular task. Training instability is frequently discovered when fine-tuning TB models. Even when using the same learning rate and batch size values as

hyperparameters, different random seeds can result in noticeably different results. The issue becomes even more obvious when using the large Transformers variants on small datasets. The instability of the TB fine-tuning process has been addressed in a number of ways since the introduction of BERT.

3. TB Models Applications and Architectures

We reviewed TB models from 2017 to 2022. Figure 5 shows the distribution of the reviewed papers per year, while Figure 6 shows the proposed taxonomy for this survey.

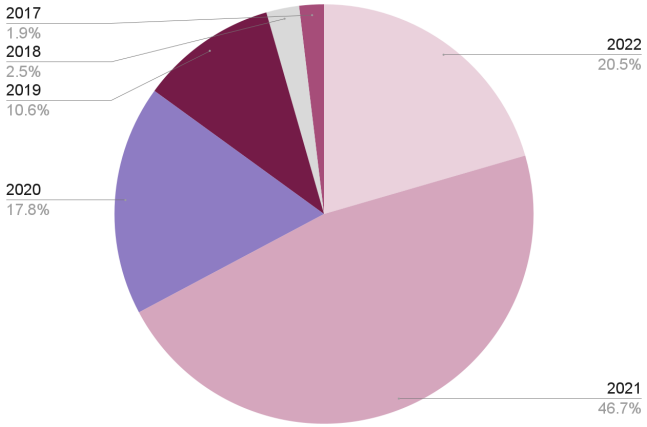

Figure 5. Reviewed papers distribution per year.

Based on the variations in the Transformer block's usage mode, different categories of Transformers can be created. There are three different kinds: encoder-only (also called auto-encoding, or AE), decoder-only (also called auto-regression, or AR), and encoder–decoder framework, also called S2S. By providing a list of examples and improvements for each architecture, this review analyzes the Transformers architectures for each TB model. We divided the taxonomy into applications and architectures as shown in Figure 6. Then, as previously described, the architectures are divided into AET, ART, and S2S.

The taxonomy for application-based models that we suggest in this survey is based on language, domain, and task applications. We detail the list of TB models used in each along with the methods used to evaluate them. We divided the applications into three categories: language (language), application domain (domain), and NLP downstream task (task). The main language-based factor is the existence of monolingual and multilingual models. Due to the fact that the majority of TB models are pre-trained, some specific fine-tuning is required for a number of downstream tasks, including question answering (QA), sentiment analysis (SA), document summarization (DS), machine translation (MT), topic modeling (TM), text classification (TC), text generation (TG), and text summarizing (TS). For the domain-based models' applications, textual-based NLP TB models can be used in a variety of fields, such as finance, health, and clinic applications, social media post classification or moderation, cyber-security, etc.

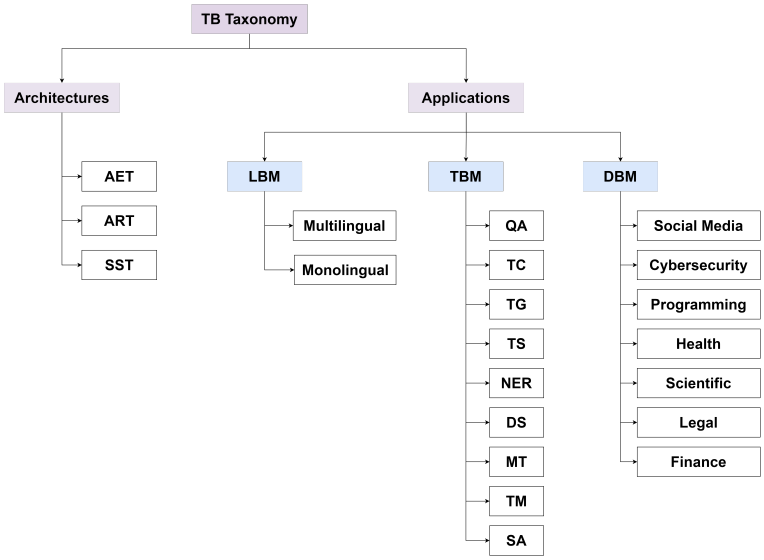

Figure 6. Reviewed TB models' proposed taxonomy. It separates them into models based on architectures and models based on applications. We defined three aspects of the applications: domain-based models' applications (DBM), downstream-based models' applications (TBM), and language-based models' applications (LBM).

3.1. Architecture-Based Models (ABM)

The field of NLP has seen great success with unsupervised representation learning. In most cases, these techniques pre-train neural networks on sizable unlabeled text corpora before fine-tuning the models or representations on subsequent tasks. Numerous unsupervised pre-training objectives have been researched in the literature under this common high-level concept. ABM include S2S Transformers (S2S), Auto-Encoding Transformers (AET), and Auto-Regressive Transformers (ART). In Figure 7, we show examples of TB models based on each ABM type with the most used downstream task.

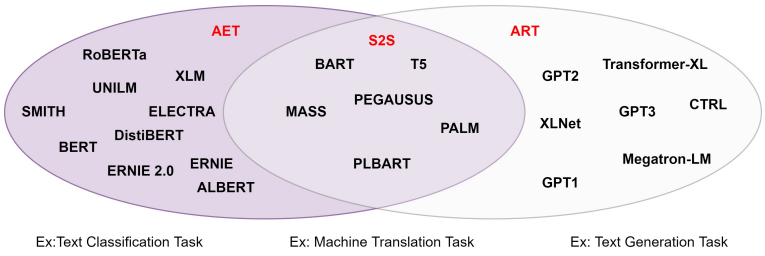

Figure 7. Diagram of the reviewed state-of-the-art TB models and examples of the mostly used downstream tasks.

3.1.1. Auto-Encoding Transformers (AET)

Definition 6. ***AET models:*** *These are TB models that use the encoder block of the Transformer architecture shown in Figure 8. AET models are comparable to the encoder in the original Transformer model in that they have unrestricted access to all inputs. The concept of input elimination serves as the foundation for this specific class of TB models: for instance, masking some of the words in a sentence and attempting to reconstruct the original text. BERT [3], RoBERTa [57], and ALBERT [69] are a few examples of this type that come to mind. These models' infrastructures frequently resemble the encoder portion of a Transformer because they aim to create bidirectional encoding representations of the entire sentences, which do not require any masking mechanisms and allow access to all input at any location. They can then be adjusted in subsequent tasks and produce excellent outcomes. Sentence classification, sequence labeling, and other tasks that lean more toward NLU are the most frequent downstream tasks for this TB type.*

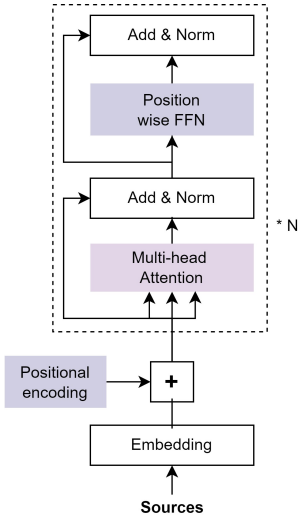

Figure 8. Encoder Block.

Typically, these models create a bidirectional representation of the entire sentence. Their most natural application is sentence classification or token classification, although they can be tuned and produce excellent results on many tasks such as text generation. First, we mention **BERT**, a bidirectional encoder representation model using the Transformer for Language Understanding (LU), introduced by [3]. This model has made improvements to the Transformer's architecture. As a result of its contribution, the unidirectionality constraint was lifted, and an MLM was substituted. After randomly masking some input tokens, the objective is to deduce the masked word's original vocabulary ID solely from its context. The representation can combine the left and right contexts with the help of the MLM to pre-train a deep bidirectional Transformer. BERT pre-trains text-pair representations through the use of the MLM and the next sentence prediction task (NSP). Numerous other models have been created based on the BERT model [5,57,68,74–82]. Lan et al. [69] proposed **ALBERT**, which stands for a light BERT. They developed a model that can match BERT's performance by altering a few scientific parameters in its architecture but only with a small fraction of the parameters and correspondingly lower computational cost. Lample et al. proposed **XLM** [58], which was a pre-trained TB architecture using multiple languages. The authors demonstrated the value of cross-lingual pre-training and suggested two approaches for learning cross-lingual language models: an unsupervised approach that only uses monolingual data and another supervised approach that uses parallel data with a new cross-lingual language model objective. CLM, MLM, and TLM are the three language modeling goals that have been put to the test in this method. The authors discovered that strong cross-lingual features are offered by both the CLM and MLM approach, which can be applied to pre-training models. **ELECTRA** a TB model proposed by Clark et al. [63]. It has a new pre-training method that trains the discriminator and the generator with two Transformers that replace tokens in the sequence. The ELECTRA contribution makes an effort to determine which tokens in the sequence the generator replaces. In addition, masking the input is replaced by the pre-training task known as replaced token detection (RTD). **RoBERTa** [57] is a BERT advancement that changes the pre-training procedure of BERT. The next sentence prediction objective was removed, longer sequences were used for training, and the masking pattern applied to the training data was dynamically changed. The model was also trained over a longer period of time with more data in larger batches. The authors also collect a sizable new dataset called CC-News [83] that is comparable in size to existing privately used datasets in order to more adequately account for training set size effects. **UNILM** [62] is a BERT-style bidirectional Transformer encoder. The model can be modified for tasks requiring the generation and comprehension of natural language. The model was pre-trained using three different language modeling tasks: unidirectional, bidirectional, and S2S prediction. Unified modeling is accomplished by utilizing a common

Transformer network and suitable self-attention masks to control what context the prediction is conditional on. In comparison to BERT, UNILM performs well on the GLUE [84] benchmark, SQUAD 2.0 [85], and the CoQA [86] question-answering tasks.

Definition 7. ***Distillation*** *is a method for shrinking a model by teaching a small model to closely resemble a larger teacher model. Therefore, when porting a model to less powerful hardware, such as a smartphone or a mini-laptop, distillation is necessary. In addition to being quicker and smaller, a distilled model achieves similar results.*

Sanh et al. [79] proposed **DistilBERT**, a TB model that is built on the BERT architecture. The model is less complex, quicker, and less costly to train than BERT. In order to shrink a BERT model by 40% during the pre-training stage, knowledge distillation is used. They implemented a triple loss that combines language modeling, distillation, and cosine-distance losses in order to make use of the inductive biases that larger models learned during pre-training. Yang et al. [87] proposed **SMITH** (Siamese Multi-depth Transformer-based Hierarchical Encoder), which is a TB paradigm for learning and matching document representations. It makes a number of design choices to adapt self-attention models for extended text inputs. A masked sentence block language modeling task is utilized for the model pre-training to capture sentence block relations inside a document, in addition to the basic MLM task at the word level used in BERT. From a series of sentence block representations, the document level Transformers learn the contextual representation for each sentence block and the final document representation. There are an interesting number of BERT-Based variations including mBART [88], DeBERTa [89], MobileBERT [90], Bort [91], DeeBERT [92], CuBERT [93], DynaBERT [94], TernaryBERT [95], I-BERT [96], ConvBERT [97], SqueezeBERT [98], MacBERT [99], BinaryBERT [100] and AutoTinyBERT [101]. Sun et al. [45] introduced **ERNIE**, a knowledge integration language representation model that outperforms Google's BERT in a variety of Chinese language tasks, and it is receiving high recognition from the NLP community. Following the original ERNIE, Baidu released the improved **ERNIE 2.0** [46], which outperforms not just the original ERNIE but also BERT and another pre-training model called XLNet [44]. ERNIE 2.0 [46] is a continuous pre-training system that integrates lexical, syntactic, and semantic information from big data in order to expand its already extensive knowledge base.

3.1.2. Auto-Regressive Transformers (ART)

Definition 8. ***ART models:*** *The GPT model family is by far the most well-known application of ART. Similar to how LMs are traditionally used, the main goal of ART models is to predict the next word based on the text that has already been read. Figure 9 illustrates how the infrastructure of ART is made up of the Transformer's decoder part, which is in contrast to the AET models previously mentioned. In order to prevent the attention calculations from seeing the content after a word, ART models rely on a masking mechanism in the training phase. Similar to other pre-trained TB models, ART can be tweaked to perform superbly on a variety of downstream tasks; natural language generation (NLG) tasks are where it shines the most. The decoder of the original Transformer model is analogous to ART, which covers the entire sentence with a mask. Text generation is the most obvious use case for those models, even though they can be altered and are excellent at many tasks.*

A typical example of such models is Extra Long Transformer or **Transformer-XL** [52]. It introduced the concept of segment-based recurrence to the deep self-attention network, in which Transformer-XL makes use of the hidden states acquired in earlier segments rather than computing the hidden states from scratch for each new segment. The previously used hidden states act as a memory for the current segment, creating a recurring link between them. As a result, because information can spread through recurrent connections, modeling extremely long-term dependency is made possible. The concept of relative positional encoding, which generalizes to attention lengths longer than those observed during training, is one of the Transformer-XL's most important new innovations. As

opposed to using a single embedding to represent each absolute position, the model computes an embedding that represents the separation between any two tokens.

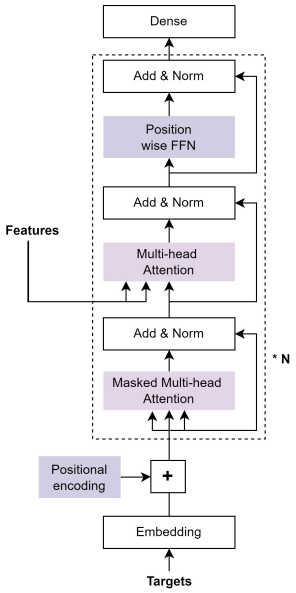

Figure 9. Decoder Block.

Yang et al. developed **XLNet** [44], an auto-regressive Transformer that attempts to balance the advantages of auto-encoding and auto-regressive language modeling while avoiding their drawbacks. As opposed to traditional auto-regressive models, which use a fixed forward or backward factorization order, XLNet maximizes the expected log-likelihood of a sequence of all feasible factorization order permutations. The context for each position can include tokens from both the left and the right thanks to the permutation operation. It is anticipated that each position will develop the ability to use contextual data from all positions, capturing bidirectional context. Additionally, XLNet incorporates the segment recurrence mechanism and relative encoding scheme of Transformer-XL into pre-training, which empirically improves performance, particularly for tasks involving a longer text sequence, and it is motivated by the most recent developments in auto-regressive language modeling. Another type of auto-regressive Transformer is the generative pre-trained Transformer developed by OpenAI. Radford et al proposed the **GPT1** [43] based on a TB architecture. This model's training process is divided into two steps. In order to learn the initial parameters of a neural network model, a language modeling objective is first applied to the unlabeled data. Second, a target task's parameters are adjusted using the corresponding supervised objective. **GPT2** [42] is a sizable TB-language model with 1.5 billion parameters that was developed using an 8 M web page training set. The straightforward goal of GPT2's training is to predict the following word from the text's previous words. This straightforward goal includes genuine examples of numerous tasks from various domains due to the diversity of the dataset. With more than 10 times the parameters and trained on more than 10 times the data as GPT1, GPT2 is a direct scale-up of GPT1. Brown et al. developed **GPT3** [41], the third generation of GPT, which has 175 billion parameter options. With the exception of the fact that GPT3 alternates between dense and locally banded sparse attention patterns in the layers of the Transformer, it uses the same architecture as GPT2, including the modified initialization, pre-normalization, and reversible tokenization. **CTRL** [102] is a 1.6 billion-parameter conditional Transformer language model that has been trained using control codes that define a domain, a sub-domain, entities, relationships between entities, dates, and task-specific behavior. Another model is MegatronLM proposed by Shoeybi et al. [103]. It is a bigger TB model created by the NVIDIA team working on applied DL research. A simple efficient intra-layer model with a parallel approach was used to train this TB model with up to 8.3 billion parameters

using 512 GPUs. They provided evidence that their techniques for training very large Transformer models are effective. The training of TB models with countless parameters is possible using these techniques. They demonstrated how massive language models can advance the state-of-the-art by training language models with 8.3 billion parameters (similar to GPT2) and 3.9 billion parameters (similar to BERT). They showed that in BERT-like models, the location of layer normalization needs to be carefully taken into account in order to achieve better performance as the model size grows.

3.1.3. S2S Transformers (S2S)

Definition 9. ***S2S models:*** *S2S makes use of the original Transformer's encoder and decoder for greater model flexibility as shown in Figure 10, either for translation tasks or by converting other tasks into S2S problems. An S2S model also unifies the NLU and NLG tasks, allowing for their solution within the same framework. It is adaptable to a range of NLG tasks, including translation and summarization, as well as NLU tasks. Currently, the most representative example of these models is the T5 [70].*

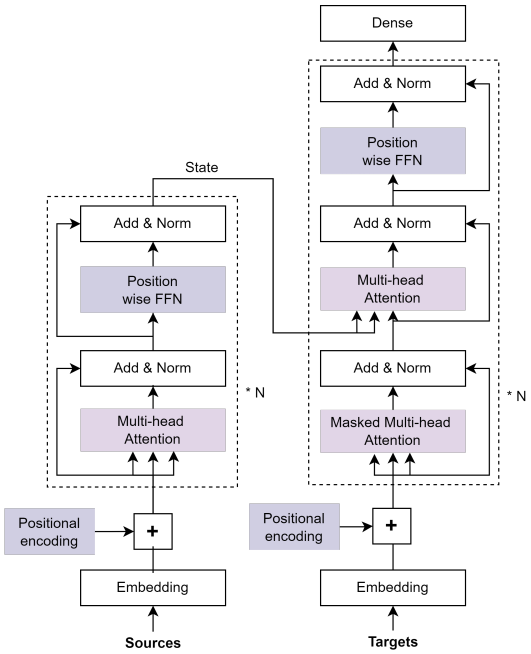

Figure 10. The Transformer architecture.

S2S can be fine-tuned to many tasks but their most natural applications are translation, summarization, and question answering. A typical example is **T5**, or Text-to-Text Transfer Transformer, developed by Raffel et al. [70]. It is a text-to-text architecture that is based on Transformers. For every task that comes after, including classification, question-answering, and translation, text input is fed to the model, and it has been taught to generate the desired text output. The same model's loss function and hyper-parameters can therefore be used for all of these different tasks. In addition to the bidirectional architecture of a causal decoder, T5 differs from BERT by substituting a number of different pre-training tasks for the fill-in-the-blank close task. Lewis et al. proposed **BART** [73], which is a S2S model that has been pre-trained using the typical TB neural machine translation architecture. It employs a typical S2S architecture with a left-to-right decoder such as GPT1 and a bidirectional encoder such as BERT. BART [73] uses a left-to-right decoder to reconstruct the original text after encoding a corrupted input sequence in both directions. The authors experimented with various corruption techniques before using sentence permutation and text infilling to train the model on the corrupted sentences. **PLBART** [104] is an S2S paradigm that can address a variety of issues related to the development and interpretation of programming languages in numerous experiments involving code generation, code

summarization, and code translation in seven programming languages that make use of the English language. PLBART uses the same architecture as the BART base; however, the original Transformer architecture was modified by including an additional normalizing layer on top of both the encoder and decoder in order to maintain training with FP16 precision [105]. **PEGAUSUS** [106] is a Transformer encoder–decoder model pre-trained using large document corpora and supervised GSG objective. Pre-training for S2S models for abstractive summarization is completed with extracted gap sentences. The authors found that creating gap sentences from a document and omitting entire sentences from it serves as an effective pre-training objective for subsequent summary tasks. In addition, leading or randomly choosing sentences leads to worse performance than sentences that are ostensibly significant. **PALM** [107] used the supervised GSG objective to pre-train a Transformer encoder–decoder on sizable document corpora. For abstractive summarization S2S models, the pre-training is carried out using extracted gap sentences. **MASS** [72] is a masked S2S generation model that reconstructs a sentence fragment from its remaining components. Each sentence is broken down into words; then, the sentence is reconstructed from randomly selected words.

3.2. Applications-Based Models (AppBM)

As detailed in the taxonomy, we divided the AppBM into three categories: namely, the language used (language), the application domain studied (domain), and the NLP downstream task investigated (task).

3.2.1. Language-Based Models (LBM)

In this section, we discuss language-specific TB models. The vast majority of the models in this library are monolingual. There are fewer and variously designed multilingual models available.

Multilingual

Multilingual TB models have the advantage of supporting cross-linguistic transfer learning. They can be trained on a particular task in one language and perform reasonably well on the same task in another, despite having only been pre-trained on monolingual tasks. There is disagreement regarding whether TB models can learn universal patterns across languages, which is one of the limitations [108]. The same architecture of BERT was pre-trained for the same task on 104 monolingual corpora of distinct languages in mBERT [3], which is a multilingual version of BERT. Good cross-lingual transfer skills were shown by mBERT. For instance, when mBERT is optimized for a classification task in English, it yields competitive results when evaluated in French [3]. Although to a lesser extent than between similar languages, this has also been shown to work between typologically different languages. The mT5 model presented by Xue et al. [71] is a multilingual variant of T5 pre-trained on the mC4 [71] dataset covering 101 languages. mT6 [109] is a multilingual pre-trained Text-to-Text Transformer with translation pairs that investigated three cross-lingual pre-training tasks: machine translation, translation pair span corruption, and translation span corruption. Both the pre-training tasks and the training objective are different in mT6 compared to mT5. On eight benchmarks, mT6 performs noticeably better than mT5. mBART [88] is an S2S denoising auto-encoder that was trained using the BART objective on sizable monolingual corpora in numerous languages. In contrast to earlier methods, which concentrated only on the encoder, decoder, or reconstructing portions of the text, mBART is one of the first techniques for pre-training an entire S2S model by denoising full texts in multiple languages. A number of other models have been created as extensions of the S2S cross-lingual TB model known as XLM. XLM-RoBERTa [110] was developed using the 100 language CommonCrawl dataset [111]. In terms of downstream tasks such as classification, sequence labeling, and question answering, it offers significant improvements over previously released multilingual models such as mBERT or XLM. XLM-R [110] is an XLM-based model architecture. XLM-R shows the possibility of training

one model for many languages while not sacrificing per-language performance. XLM-E [112] uses ELECTRA-style tasks to train the cross-lingual language model. Multilingual token detection and translation token detection are two pre-training tasks used by the model. Cross-lingual transferability is accomplished by XLM-E with a comparatively small transfer gap. The cross-lingual language model XLM-K [113] focuses on existing multilingual pre-training with two knowledge tasks, namely the masked entity prediction task and object entailment task. XLM-T [114,115] is a Twitter-based multilingual model. The multilingual baseline includes an XLM-R model that has been trained on millions of tweets in over thirty languages along with a starter code to fine-tune on a target task and a set of unified sentiment analysis Twitter datasets that each have an XLM-T [115] model tuned on them. Goyal et al. proposed XLM-RXL and XLM-RXXL [116], two larger multilingual masked language models, with 3.5B and 10.7B parameters. The two models outperformed XLM-R on the GLUE benchmark. MuRIL [117] is a BERT model that has been pre-trained on 17 Indian languages and their transliterated counterparts. Unicoder [118] is a multi-language encoder insensitive to different languages pre-trained on multiple cross-lingual tasks. IndicBERT [117] was trained using a large corpus of multilingual texts. The model is structured using an ALBERT model. IndicBERT has significantly fewer parameters than other publicly accessible models such as mBERT and XLM-R while still outperforming them on a variety of tasks.

Monolingual

A typical approach to modifying the models for the downstream tasks is to use downstream datasets for each new language to pre-train TB language models. There are several TB models that are monolingual. The main strategy is to adjust an all-encompassing TB language model for a particular language, as in the case of the BERT architecture. BERT is only trained on English datasets, unlike the multilingual BERT model mBERT, which uses the original BERT architecture and training objectives but is trained on corpora of up to 104 languages. There are several other models trained using BERT architecture including IndonesianBERT, IndonesianBERTLite [2] and IndoBERT [119] for the Indonesian language. The same approach is used for other languages such as RomanianBERT [4], FlauBERT [120], HerBERT [121], KLUE-BERT [122], AraBERT [123], PhoBERT [124], CamemBERT [125], SweedishBERT [126], PolishBERT [127], BERTje [128], FinnishBERT [129], ALBERTO [130], PortugueseBERT [131], RuBERT [132], BanglaBERT [133], ARBERT [134], MARBERT [134], ParsBERT [135], BERT-SentiX [12], CT-BERT [13]. In addition, for a GPT-based pre-trained model, we found AraGPT2 [136], a Transformer architecture-based model for Arabic language generation. The largest publicly accessible collection of filtered Arabic corpora served as the model's training data with a total size of 77 GB with 8.8 billion words [136]. The perplexity measure, which gauges how well a probability model predicts a sample, was used to assess the model's performance. Taking into account the inherent biases of the dataset, ARAGPT2, similar to many ML models, has ethical ramifications and can be abused (e.g., by automatically creating false news). A detector model that is responsible for detecting output produced by ARAGPT2 is also released to aid in the detection of model abuse. For the Indian language, Aniruddha et al. [137] proposed Meta-ED, which is a pre-trained model based on GPT2 architecture. The RoBERTa-based pre-trained models include KLUE-RoBERTa [122], WangchanBERTa [138], RoBERTa-Twitter [14] and RoBERTa-Reviews [10]. Several NLP tasks have been successfully completed using trained language models. Recent studies have shown that models pre-trained on monolingual corpora outperform models pre-trained on multilingual corpora on tasks in the same language. When pre-trained on a Portuguese corpus, BERT models have already demonstrated improved performance for Portuguese tasks. One of the reasons to carry out a comparable pre-training with the T5 model is that it can produce text. As a result, it can carry out tasks such as summarization, answering abstract questions, and translation that a BERT model cannot. PTT5 [139] improved the original T5 model on Portuguese language tasks by pre-training it on BrWac [140], which is a large corpus of web pages in Brazilian Portuguese.

The model is tested using pre-training on tasks involving named entity recognition and sentence entailment prediction in Portuguese. Results demonstrate that monolingual pre-training significantly boosts the model's performance. Using an ELECTRA pre-trained model, we found SweedishELECTRA [126] for Sweedish and AraELECTRA [141] for Arabic language. Using an XLM pre-trained model, we found XLM-R-Twitter [114]. Using a BART pre-trained model, we found IndoBART [142]. Using an ALBERT pre-trained model, we found KoreALBERT [143] and SweedishALBERT [126]. In addition, RobeCzech [144] and BETO [145] are monolingual TB models based on other state-of-the-art architectures.

3.2.2. Domain-Based Models (DBM)

This section examines the TB models according to the application domain. The taxonomy demonstrates that there are currently eight large domains: programming, health, social media, science, law, finance, and cybersecurity.

Social Media

Because of the increased connectivity between individuals, the popularity of social media and microblogging platforms is still having unrecognized effects on our lives. The numerous instances of abusive language phenomena overshadow any potential advantages. HateBERT [11] is a language model that has already been trained to recognize English language abuse. The main goal of this contribution is to identify ways to solve that issue. HateBERT consistently outperforms generic BERT when it comes to detecting offensive language on state-of-the-art hate speech datasets such as OffensEval 2019 [146], AbusEval [147], and HatEval [148]. According to the cross-dataset experiments, HateBERT successfully captures each abusive language phenomenon against which it has been tuned. Bertweet [149] and BertweetCovid19 [149] are two widely used language models that have already been trained on English tweets. BERTweet is trained based on the RoBERTa pre-training procedure. The corpus used to pre-train BERTweet contains 850 M English tweets, including 5 M tweets about the COVID-19 pandemic and 845 M tweets streamed from 01/2012 to 08/2019. In addition, BERT-SentiX [12] and CT-BERT [13] used BERT as a pre-trained model for social media-related fine-tuning. Despite the fact that the fine-tuning steps can be more varied for the RoBERTa-base model, however, the fundamental strategy of using a more focused dataset for social media has not changed. Rahali et al. [150] studied misogyny detection on social media in different languages and platforms using TB models including BERT, mBERT, XLNet, and RoBERTa in comparison to BiLSTM for the same task. RoBERTa-Twitter [14] is a RoBERTa-base model trained on 58 M tweets, which were described and evaluated on the TweetEval benchmark [14]. RoBERTa-Reviews [10] and RoBERTa-News [10] used the same approach. The majority of downstream tasks using TB models in relation to social media and entertainment can be classified to toxic content prevention tasks [151–155], recommendation tasks [156–158] and fake news detection tasks [159–162].

Programming

Therefore, the state-of-the-art in NLP already demonstrates reliability in carrying out tasks based on the semantic content of natural language. The first methods for applying concepts from the NLP field to the understanding of the program code field start to emerge. However, current methods treat software code as plain text, ignoring the syntactical features that programming languages provide. Additionally, the majority of current methods are designed for a particular downstream task, which limits the range of situations in which they can be used. Text data can be processed using NLP for a number of different tasks. We discovered that these models can also exhibit comparable advantages. There are several works for software code processing such as CodeGPT [163], GraphCodeBERT [164], CodeGPT-adapted [163], CoText [165], PLBART [104] and CodeBERT [166].

Health

The rapid adoption of electronic health records (EHR) encourages the application of data-driven modeling to enhance patient care management and delivery. Particularly, cutting-edge DL techniques such as RNNs [167], CNNs [168] and Transformer [21] are being used in novel applications to predict future medical events. These DL models can gradually extract pertinent features from large patient samples and demonstrate promising model performance for various predictive tasks. For example, BioBERT [5], which was trained on a dataset with more than 200,000 patient records, shows promising results for predicting the diagnosis and medication of subsequent medical visits. In addition, ClinicalBERT [6], BlueBERT [7] and PubMedBERT [8] are other proposed architectures for health-related TB applications. The majority of DL models need to be trained on large amounts of high-quality data. Even though the large-scale EHR database contains millions of patient records, these records are frequently not fully applicable for a variety of reasons, such as a lack of cases for rare diseases and conditions, restricted access to the entire database due to privacy concerns, and difficulties in data cleaning and merging. These limitations slow down the data collection process, decreasing the likelihood that data-hungry DL models will appear and obstruct the advancement of healthcare computing and care delivery. Kalyan et al. [169] proposed AMMU, a survey of TB biomedical pretrained LM. There are several other approaches including healthcare monitoring [170,171], LM for health records analysis [172–180], health documents classification [181–184], TB medical management approaches [185], named entity recognition (NER) for medial records [186–188], question answering [189], document summarization [190,191], system recommendation [192] and relation extraction [193–195].

Scientific

In comparison to earlier works, pre-trained Transformers such as SciBERT [75] or BioBERT [5] have excelled in scientific NER. Despite this success, they typically classify the first subtoken representation of each word to label sequences, which can result in less-than-ideal fine-tuning for NER. This issue has been addressed in some work by designing NER as a span-based classification rather than a sequence labeling. For example, MathBERT [196], SciBERT [75] and OAG-BERT [197] showed good performance on downstream tasks using this technique. For geology, using geological reports, some TB models were pre-trained to solve NLP-related tasks such as NER. Among these studies, we found the study proposed by Liu et al. [198]. They trained a BERT-based model for NER tasks. Additionally, there are other approaches for chemical-related papers [199], scientific papers analysis [200,201], biomedical related papers [202,203], scientific experiment prediction [204], scientific articles classification [205], scientific NER and information retrieval [206–209]. and scientific documents summarization [77,210].

Legal

ALeaseBERT [211] and LegalBERT [212] are a family of BERT models for the legal field that aims to support applications in computational law, legal technology, and legal NLP research. They gathered 12 GB of diverse English legal text from a variety of fields (such as legislation, court cases, and contracts), scraped from publicly accessible resources, to pre-train the various iterations of Legal-BERT. When performing tasks that are domain-specific, sub-domain variants that use general Legal-BERT outperform those that use BERT out of the box. LegalBERT is a lightweight model that performs competitively that was pre-trained on legal data and is 33% the size of the BERT base. There are other legal-related studies for different NLP downstream tasks including pre-trained language models for different languages [213–218], legal documents summarization [219–222], legal documents classification [223–229], key information extraction and NER [230–232], legal sentiment analysis [233,234], legal question answering [235–237], and legal judgment prediction [238–243].

Finance

Traditional financial product recommendation algorithms ignore the effects of time in series data and are based on the static characteristics of consumers and financial products, which results in low-quality recommendations. By utilizing the benefits of Transformer in processing time series, Lian et al. [244] suggested an R-Transformer (Recommendation system based on Transformer) to address this issue. Users' and financial products' states are mined using two R-Transformer networks based on time series, and the inner product of users' and financial products' states is used as the final score. FinBERT [9] is a sentiment analysis model. FinBERT has already been trained to evaluate the tone of the financial text. It is created by further honing the BERT language model for financial sentiment classification by training it on a large financial corpus in the finance domain. Goel et al. [245] suggested a TB BERT architecture that gathers contextual data from a collection of raw texts created specifically for the FinNLP workshop in the finance domain and performs a classification operation to group domain terms into a predetermined number of class labels. The suggested model's feature extractor is a TF-IDF vectorizer that provides the model with a word-level relevance in addition to taking contextual BERT embeddings into account. Other studies focused on financial forecasting [246–253], financial terms classification [245,254–256], generation of finance reports [257], financial reports summarization [247,258–261], sentiment analysis for financial documents [262–271], financial terms extraction [262,272–274], NER for financial terms [254,275–278] , topic modeling for financial documents [279,280] and financial language models [9,281,282].

Cybersecurity

Cybersecurity is a crucial issue, and recent developments in NLP have accelerated the adoption of cybersecurity solutions. In particular, Malware categorization, a crucial and difficult issue in information security, is one of many tasks where TB models are used for this subject. TB models that may be trained on data such as opcode sequences, API calls, and byte n-grams, among many others, are used in modern malware classification techniques [283–288]. Other approaches focused on cybersecurity data analysis [289,290], anomaly detection [291–294], fake cyber threat generation [295] and NER [296,297].

3.2.3. Task-Based Models (TBM)

The majority of downstream textual-based NLP tasks, including text classification (TC), text generation (TG), automatic summarization (DS), machine translation (MT), named entity recognition (NER), relationship extraction (RE), sentiment analysis (SA), and topic segmentation (TM), have shown TB models to be effective. Downstream tasks are very dependent on the selected datasets and benchmarks used to test the TB models. We review first the NLP standard benchmarks; second, we review the evaluation metrics and finally we review the TB models used in each downstream task.

Benchmarks

The field of NLP encompasses a wide range of research activities, including sentiment analysis, text summarization, question answering, and machine translation. To accomplish these tasks, researchers work to create various ML and DL models. The SOTA dataset variants are presented in Table 3.

Table 3. NLP benchmarks list and sizes.

Benchmark	Task	Variants	Size	Split	Ref
RACE	Reading Comprehension		24.26 MiB	test 1045 train 18,728 validation 1021	[298]
GLUE	Sentiment Analysis	SST-2	7.09 MiB	test 1821 train 67,349 validation 872	[299]
		CoLA	368.14 KiB	test 1063 train 8551 validation 1043	[300]
		MRPC	1.43 MiB	test 1725 train 3668 validation 408	[301]
		QQP	57.73 MiB	test 390,965 train 363,849 validation 40,430	[302]
	Natural Language Inference	MNLI	298.29 MiB	test 9796 train 392,702 validation 9815	[303]
		QNLI	10.14 MiB	test 5463 train 104,743 validation 5463	[304]
		WNLI	28.32 KiB	test 146 train 635 validation 71	[305]
	Semantic Textual Similarity	RTE	680.81 KiB	test 3000 train 2490 validation 277	[306]
		STS-B	784.05 KiB	test 1379 train 5749 validation 1500	[307]
SQuAD	Question Answering	SQuAD2.0 SQuAD1.1	94.04 MiB	train 87,599 validation 10,570	[85]

- RACE [298] is a large-scale reading comprehension dataset with more than 28,000 passages and nearly 100,000 questions. The dataset is collected from English examinations in China, which are designed for middle school and high school students. The dataset can be used for training and test sets for machine comprehension.
- The GLUE [84] (General Language Understanding Evaluation) benchmark is a collection of nine natural language understanding tasks, including single-sentence tasks CoLA and SST-2, similarity and paraphrasing tasks MRPC, STS-B and QQP, and natural language inference tasks MNLI, QNLI, RTE, and WNLI 2.1.
- SQuAD [85] (Stanford Question Answering Dataset) is a reading comprehension dataset, consisting of questions posed by crowdworkers on a set of Wikipedia articles, where the answer to every question is a segment of text, or span, from the corresponding reading passage, or the question might be unanswerable.

The models use the pre-training and then fine-tuning procedure, as described in the background section. The NLP benchmarks are commonly used to evaluate models, with each dataset serving as a reference for a specific downstream task. The model's pre-training can be single or multitask specific. To investigate the differences in the performance of the TB models on these benchmarks further, we present in Table 4 a comparison of corresponding experimental results on these datasets for important and widely used TB

models. Table 4 displays the outcomes of the GLUE, SQuAD, and RACE benchmarks used to evaluate the most recent TB models. Similar to the GLUE paper selected metrics, we present the matthews correlation (mc) coefficient for CoLA and we report F1 for MRPC, QQP and SQuAD. We also present Pearson correlation results for STS-B. For every other task, we provide the accuracy (Acc) results. It will be much easier to compare TB models if all cutting-edge architectures have the same number of layers and hyperparameters and are trained on the same dataset. However, this condition is not available because, as stated in previous sections, each model is trained on a different dataset and has a unique architecture. However, some models, such as TinyBERT [74], share the same architectures as the BERT base. TinyBERT learns from the BERT base through complex distillation procedures.

For explicit reasoning tasks with a longer context, such as SQuQD [85] and RACE [298], the performance gain of XLNet is typically greater. The Transformer-XL backbone in XLNet may be responsible for this advantage in handling more complex situations. The effectiveness of the suggested word structural objective (WSO) was demonstrated by the StructBERT model with structural pre-training, which consistently outperformed the original BERT model. The loss and accuracy of the masked token prediction were largely unaffected by the augmented shuffled token prediction in StructBERT's structural objective. Even though RoBERTa and BERT largely share the same architecture, RoBERTa consistently outperforms BERT in several tasks. RoBERTa outperforms XLNET by 0.6 points on the SQuAD datasets [85] in terms of F1 scores. The comparison Table 4 demonstrates how well the T5 model performed on the GLUE and SQuAD datasets. Moreover, it achieved a stellar F1 score of 88.9 on the SuperGLUE language benchmark [308], which is a dataset based on GLUE with a new set of more difficult language comprehension tasks. T5 outperformed the previous state-of-the-art, ALBERT, by more than one point on the F1 score for SQuAD [85]. According to the GLUE tasks' average score, we can see that ELECTRA largely outperformed GPT1 and BERT. The ELECTRA model is made from the ground up. The comparison table's findings demonstrated that ELECTRA was only slightly more effective at CoLA than the other TB models. Similar to ELECTRA's pre-training task of identifying fake marijuana, CoLA's objective is to distinguish between sentences that are and are not grammatically correct. This could contribute to the explanation of ELECTRA's performance. Performance on sentence pair tasks such as MNLI, QQP, SNLI, and SQuAD was significantly improved by including the sentence structure objective. The WSO was particularly crucial for tasks such as CoLA and SST-2 that only called for a single sentence. It illustrates how understanding the relationship between sentences for subsequent tasks is influenced by pre-training inter-sentence structures. The correlation between grammatical error and improvement in the CoLA task for ELECTRA was greater than 5%. Pre-training assessments of the acceptability of a single sentence were more precise due to the model's capacity to reconstruct word order.

Table 4. List of evaluation results for examples of TB state-of-the-art models on NLP benchmarks, where N referes to the number of parameters of the TB model.

N	TB Models	GLUE										SQuAD		RACE
		MNLI-m (*Acc*)	MNLI-mm (*Acc*)	QQP (*F1*)	QNLI (*Acc*)	SST-2 (*Acc*)	CoLA (*MC*)	STS-B (*Spearman Correlation*)	MRPC (*F1*)	RTE (*Acc*)	WNLI (*Acc*)	SQuAD 1.1 (*F1*)	SQuAD 2.0 (*F1*)	
14.5 M	$BERT_{tiny}$	75.4	74.9	66.5	84.8	87.6	19.5	77.1	83.2	62.6	-	-	-	-
29.2 M	$BERT_{small}$	77.6	77.0	68.1	86.4	89.7	27.8	77.0	83.4	61.8	-	-	-	-
110 M	$BERT_{base}$	84.6	83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	-	88.5	-	-
335 M	$BERT_{large}$	86.7	85.9	72.1	92.7	94.9	60.5	86.5	89.3	70.1	-	90.9	81.9	-
235 M	$ALBERT_{ensembles}$	91.3	91.0	74.2	-	97.1	69.1	92.5	93.4	89.2	91.8	94.1	88.1	82.3
335 M	$SpanBERT$	88.1	−87.7	71.9	94.3	94.8	64.3	89.9	90.9	79.0	65.1	94.6	88.7	-
15.1 M	$MobileBERT_{tiny}$	81.5	81.6	68.9	89.5	91.7	46.7	80.1	87.9	65.1	65.1	-	-	-
356 M	$RoBERTa_{large}$	90.2	90.2	92.2	94.7	96.4	68.0	92.4	90.9	86.6	89.0	94.6	89.4	83.2
14.5 M	$TinyBERT$	82.5	81.8	71.3	87.7	92.6	44.1	80.4	86.4	66.6		-	-	-
340 M	$StructBERT_{large}$	88.0	-	74.1	95.7	95.2	65.3	90.3	92.0	83.1	65.1	-	-	-
110 M	$StructBERT_{base}$	85.5	-	72.0	92.6	94.7	57.2	88.5	89.9	76.9	65.1	90.6	-	-
52.2 M	$DistilBERT$	78.9	78.0	68.5	85.2	91.4	32.8	76.1	82.4	54.1	65.1	-	-	-
335 M	$ELECTRA_{large}$	91.3	90.8	90.8	95.8	97.1	71.7	92.5	90.7	89.8	92.5	94.9	91.4	-
550 M	XLM	89.1	88.5	73.2	94.0	95.6	62.9	88.8	90.7	76.0	71.9	-	-	-
260 B	$ERNIE2.0$	92.3	91.7	75.2	97.3	97.8	75.5	93.0	93.9	92.6	95.9	-	-	-
340 M	$UNILM$	87.0	85.9	-	92.7	94.5	61.1	-	-	70.9	-	-	83.4	
11 B	$T5_{xxlarge}$	92.2	91.9	75.1	96.9	97.5	71.6	93.1	92.8	92.8	94.5	96.2	-	-
140 M	$BART$	89.9	90.1	92.5	94.9	96.6	62.8	91.2	90.4	87.0	-	94.6	89.2	-
117 M	$GPT1$	82.1	81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	-	-	-	-
1.5 B	$GPT2$	82.1	81.4	70.3	88.1	91.3	45.4	82.0	82.3	56.0	-	-	-	59.0
117 M	$XLNet_{base}$	85.8	85.4	-	-	92.6	-	-	-	-	-	-	81.3	66.0
360 M	$XLNet_{large}$	89.8	91.0	91.8	93.9	95.6	63.6	91.8	89.2	83.8	92.5	94.5	88.8	81.8
530 M	$MegatronLM$	91.4	91.4	92.7	-	-	-	-	-	-	-	95.5	91.2	89.5

Evaluation Metrics

The most difficult component of NLP is determining the performance of these models for various jobs. Because the cost function or evaluation criteria are well defined in some ML jobs, measuring model performance is easy. For example, in regression, the mean absolute error (MAE) or mean square error (MSE) can be determined using established libraries, while in classification, the most commonly used metrics are classification accuracy and F1-score. However, this is not the case for all NLP models, because in the case of TB tasks, the ground truth or result can change. As a result, we concentrated on automatic evaluation metrics for several types of TB models. First, we looked at AET models and analyzed metrics relating to its most commonly utilized downstream task TC. Second, we looked at the ART model evaluation and we detailed the metrics used for TG downstream task evaluations. Third, we examined the MT metrics for S2S model evaluation. The metrics are typically applied to similar downstream tasks for each TB type, such as NER, DS, and so on.

A. Metrics for TC Evaluation

Here are various measures for evaluating the TB models on TC tasks. Certain measures, such as precision–recall, are helpful for a variety of activities. Using alternative measures for performance evaluation, instead of properly evaluating the TB model and relying solely on the accuracy, might cause issues when the model is applied to unobserved data and result in inaccurate predictions. Table 5 shows the formulations for the listed evaluation metrics for TC.

- **Accuracy [309]:** Simply put, accuracy reflects how frequently the classifier predicts correctly. Accuracy is determined by dividing the total number of forecasts by the proportion of true predictions.
- **Precision [310]:** Precision reveals how many of the labels that were predicted with accuracy ended up being positive. When false positives are more problematic than false negatives, precision is helpful.
- **Recall (Sensitivity) [310]:** Recall describes how many of the actual positive cases our model was able to properly anticipate. When false negative is more important than false positive, it is a valuable metric. In medical situations, it is crucial because even if a false alarm is raised, the real positive cases should not go unnoticed. The proportion of true positives to all other positive results is known as a recall for a label.
- **F1 Score [310]:** It provides a synthesis of the precision and recall measurements. it reaches its optimum when precision and recall are equal. The harmonic mean of recall and precision is the F1 Score.
- **AUC-ROC [311]:** A probability curve called the Receiver Operator Characteristic (ROC) separates the "signal" from the "noise" by plotting the TPR (True Positive Rate) versus the FPR (False Positive Rate) at different threshold values. A classifier's capacity to distinguish between classes is measured by the area under the curve (AUC).
- **Confusion Matrix:** A performance indicator for ML classification issues when the output can be two or more classes is the confusion matrix. It is a table with combinations of values that were expected and actual.

Table 5. List of TC evaluation metrics formulations, where true positive (TP), true negative (TN), false positive (FP), and false negative (FN) represent the number of predicted samples.

Metric	Formula
Accuracy [309]	$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$
Precision [310]	$Precision = \frac{TP}{TP+FP}$
Recall [310]	$Recall = \frac{TP}{TP+FN}$
F1-Score [310]	$F1\text{-}Score = 2 * \frac{Precision*Recall}{Precision+Recall}$

B. Metrics for TG Evaluation

It is crucial to assess the generated text's quality from a broad standpoint for any TG model. Among these are the text's factuality, which provides a sense of how accurately the generated text reflects the facts mentioned in the context, the text's fluency, which conveys how fluid the language in the output text is, and the text's diversity. These evaluation-related factors can be measured by either a machine or a person. Since humans are the best judges of the natural language texts produced by NLG systems, human-based evaluation metrics are reliable. Manual evaluation of the generated text is completed by assigning a score to each component. The automatic evaluation metric is the alternative type of evaluation. The similarity of the NLG model-generated texts to the corresponding reference texts in benchmarking datasets is typically used as the basis for automatic evaluation metrics for TG. Lexical, syntactic, and semantic criteria can all be used in the metrics calculation.

- **BLEU [312]:** In NLP tasks, the metric known as BLEU is frequently used to assess how generated texts from an NLG model differ from reference texts. Its value was in the neighborhood of 0.0 and 1.0. The value of BLEU is 1.0 if two sentences perfectly match one another. The BLEU value is 0.0 when there is absolutely no overlap between them. In particular, BLEU counts the number of n-gram matches between the reference text and the generated text.
- **ROUGE [313]:** Recall-oriented understanding for sting evaluation is known by its acronym, ROUGE. It is an automated summary evaluation method that uses a set of indicators to gauge the effectiveness of the texts that are generated automatically. In order to determine how closely the automatically generated texts resemble a set of reference texts, the system compares the generated texts with the reference texts, counts the overlapping basic units (n-grams), and calculates the corresponding score. The denominator is the only distinction between $BLEU_n$ and $ROUGE_n$ as shown in Table 6. The total number of n-grams in the texts generated is the denominator of BLEU-n, which focuses on precision. The total number of n-grams found in the reference texts serves as the denominator because ROUGE-n is recall-oriented. Better recall-oriented quality is indicated by a higher $ROUGE_n$ value.
- **Perplexity (PPL) [314]:** The advantages and disadvantages of a linguistic probability model can be evaluated using a word-level technique PPL. A Language Probability Model (LPM) is a probability distribution on a given text, i.e., the likelihood of the $n+1$th word given the n preceding words in the text. When a language probability model is trained on the training set of reference texts for the TG task, it is applied to predict the generated text. The generated text is more fluid when the PPL value is lower.
- **Distinct-n [315]:** An n-gram-based statistic called distinct-n is used in various scenes that aim to increase the diversity of generated texts. The diversity rises as the value increases, where the total number of n-grams in the generated texts serves as the denominator and the numerator is the number of n-grams that appear just once in the generated texts. The value of Distinct-n is 1 when the total number of n-grams and the count of unique n-grams are equal.

Table 6. List of mathematical formulations for the automatic metrics calculation for the task of TG evaluation.

Metric	Formula
BLUE [312]	$BLUE_n = \frac{\sum_{x \in G} \sum_{n-gram \in x} Count_{match}(n-gram)}{sum_{n-gram \in x} \sum_{x \in G} Count(n-gram)}$ where x is the generated text sequence to be compared. *match* indicates that the n-gram appears in both the generated and reference texts. n is the number of n-grams in the sequence. G is an abbreviation to generated text.

Table 6. *Cont.*

Metric	Formula
ROUGE [313]	$ROUGE_n = \frac{\sum_{x \in G} \sum_{n-gram \in x} Count_{match}(n-gram)}{sum_{n-gram \in x} \sum_{x \in R} Count(n-gram)}$ where R is an abbreviation to referenced text.
PPL [314]	Perplexity is defined as the exponentiated average negative log-likelihood of a sequence. If we have a tokenized sequence $W = (w_0, w_1, w_2, \ldots, w_n)$, then the perplexity of W is: $PPL(W) = exp\left\{-\frac{1}{t}\sum_i^t \log p(w_i \mid w_{<i})\right\}$ where n is the number of words in the generated text. i the i_{th} word in the generated text. $p(w_i)$ is the probability of the i_{th} word in an LM trained with reference texts.
DISTINCT [315]	$DISTINCT_n = \frac{Count_{unique}(n-gram)}{Count(n-gram)}$ where n is the number of n-grams that appear once in the generated texts and the denominator is the total number of n-grams in the generated text.

There are also other word token level and sentence level evaluation metrics for TG such as Word Mover's Distance (WMD) [316], MEANT 2.0 [317], editing distance [318], and TESLA [319]. Semantic-based metrics are designed to evaluate texts with various lexical structures but the same semantic content. Semantic similarity is more difficult to measure and requires more consideration than lexical-based or syntactic similarity. Among the semantic metrics, we list semantic textual similarity (STS) [320] and deep semantic similarity model (DSSM) [321].

C. Metrics for MT Evaluation

The development of TB and MT models depends heavily on MT evaluation. Along with BLUE, there exist other evaluation metrics for MT tasks. We briefly describe the evaluation measures taken into account for TB MT models. We list the most widely used metrics.

- **TER [322]:** It counts how many edits, including insertions, deletions, shifts, and substitutions, were necessary to convert the MT output into the reference.
- **CHRF [323]:** Instead of using the word n-grams, it compares the MT output with the reference using the character n-grams. This facilitates matching word morphological variants.
- **YISI-1 [324]:** Using contextual word embeddings such as BERT determines the semantic similarity of phrases in the MT output with the reference.
- **YISI-2 [325]:** It is identical to YISI-1, with the exception that it calculates the degree to which the MT output and the source are comparable using cross-lingual embeddings.
- **Enhanced Sequential Inference Model (ESIM) [326]:** It is a trained neural model that first determines sentence representations from BERT embeddings and then determines how similar the two strings are to one another.
- **Word Error Rate (WER MWER) [327]:** This is the accepted measurement for automatic speech recognition evaluation. WER is calculated by dividing the length of the reference translation by the Levenshtein distance [328] between the words of the system output and the words of the reference translation. In order to determine the best alignment between the MT output and the reference translation, the Levenshtein distance is calculated using dynamic programming, with each word in the MT output aligning to either 1 or 0 words in the reference translation and vice versa.
- **NIST [329]:** As a better alternative to BLEU, the NIST precision measure was developed. Ngram occurrences should be given more weight based on their significance, and unwanted consequences of the shortness penalty of BLEU should be minimized. The frequency of the n-gram in the references is used to calculate significance. Regard-

ing BLEU, numerous reference translations are pooled, although NIST believes that infrequent n-grams are more significant than those that occur frequently. The brevity penalty is intended to counteract BLEU's small favoritism of brief candidates.

- **Metric for Evaluation of Translation with Explicit ORdering (METEOR) [330]:** It is a test that was created expressly to address a number of BLEU's recognized flaws. BLEU is typically a precision-oriented measure, whereas METEOR is recall-oriented. Unlike BLEU, which only computes precision, METEOR computes both recall and precision before combining the two to compute the harmonic mean, heavily favoring recall.
- **Position-Independent Error Rate (PER) [331]:** It is an effort to overcome the WER's word-ordering restriction by treating the reference and hypothesis as bags of words. This allows words from the hypothesis to be aligned to terms in the reference regardless of position. As a result, it is guaranteed that the PER of an MT output will be lower than or equal to the WER of the MT output. The drawback of this type is that it cannot tell a correct translation from one where the words have been jumbled.

Mathur et al. [332] suggested a review of the metrics for MT's automatic evaluation. They highlighted their key recommendations, which include switching from employing BLEU or TER to CHRF, YISI-1, or ESIM for MT evaluation. Furthermore, it was advised to ensure that significant empirical conclusions were validated by manual evaluation rather than relying just on minor variations in evaluation metrics.

NLP Downstream Tasks

It has been widely accepted that learning representations of TB models in NLP are beneficial for a variety of NLP tasks. For example, machine translation is used to create fluent text from one natural language into another while maintaining meaning. Rule-based, statistical, and neural machine translation techniques, among others, are employed. Other downstream tasks, as proposed in our taxonomy, are listed in this section, along with detailed examples of TB approaches used in different domains.

A. Text Classification (TC)

Text classification is significant because it serves as the foundation for other concepts. In recent years, techniques for text classification that use TB models have seen a lot of success. TB techniques are now the method of choice for classifying texts due to their success. For this task, there exist several studies including different languages [333,334], long documents classification [335–337], and financial documents classification [254]. For scientific documents classification, we found SCIBERT [75], which is a pre-trained language model based on BERT that was developed to address the lack of high-quality, large-scale labeled scientific data. To enhance performance on subsequent scientific NLP tasks, SCIBERT makes use of unsupervised pre-training on a sizable multi-domain corpus of scientific publications. With datasets from various scientific fields, they assess a range of tasks, such as sentence classification, dependency parsing, and sequence tagging. ClinicalBert [76] is a BERT-based model that creates and assesses clinical note representations using bidirectional Transformers. It outperforms baselines on 30-day hospital readmission prediction using both discharge summaries and the first few days of intensive care unit notes, and it unearths high-quality relationships between medical concepts as judged by humans. In addition, BioBERT [5] is a language representation model for a specific domain that has been pre-trained on extensive biomedical corpora. When pre-trained on biomedical corpora, it outperforms BERT and previous state-of-the-art models in a variety of biomedical text mining tasks with nearly the same architecture across tasks. BERT-based models were used wildly in different domains, among the interesting approaches are the applications of TB models in malware classification. For example, MalBERT [286,338] used the pre-trained BERT model to classify malware based on extracted text features from the source codes of the applications. Murat et al. [339] proposed a bidirectional Transformer (BiTransformer), built from two Transformer encoder blocks, that makes use of bidirectional position encod-

ing to account for the forward and backward position information of text data. The model is employed to assess the downstream task of text classification's attention mechanisms.

B. Question Answering (QA)

Numerous studies using the Transformer architecture have been used for question-answering systems. TOD-BERT [82], a pre-trained task-oriented dialogue BERT, combines nine datasets for task-oriented multi-turn human–human dialogue. In order to simulate the response selection task, they propose a contrastive objective function and incorporate user and system tokens into the masked language modeling. In addition, DIALOGPT [340] is a dialogue-generative pre-trained Transformer. The model is a neural conversational response generation model trained on 147 million conversations. It increases the Transformer's capacity to deliver a performance that is comparable to a human's in single-turn dialogue settings both in terms of automatic and human evaluation. SOLOIST [341] is a technique that creates task-oriented dialog systems at scale using transfer learning. They parameterized a dialog system by combining various dialog modules into a single neural model using a TB auto-regressive language model. Shao et al. [27] designed a TB neural network to select the best answer. The Transformer uses a bidirectional long short-term memory (BiLSTM) to collect both global information and sequential features from the question or answer sentence. This TB network differs from the original Transformer in that it prioritizes sentence embedding over the S2S task. Using a variety of NLP techniques, several automated question-answering systems were developed to search through unstructured documents such as social media posts to find the relevant data, analyze it, and choose the best part to respond to the question [27,82,340,341].

C. Document Summarization (DS)

The automatic text summarization transfer learning using pre-trained word embedding models has shown promising results. HIBERT was proposed by [77]; it is a shorthand for Hierarchical Bidirectional Encoder Representations from Transformers. HIBERT is primarily used for document encoding and as a pre-training method with unlabeled data. On a version of the *New York Times* dataset and the CNN *Daily Mail* dataset, it performs better than its randomly initialized counterpart by 1.25 ROUGE and 2.0 ROUGE, respectively. Lamsiyah et al. [342] fine-tuned a BERT model on supervised intermediate tasks from GLUE benchmark datasets using single-task and multi-task fine-tuning methods. Khandelwal et al. [343] showed that using a single pre-trained Transformer for S2S tasks simplifies the model, reduces the number of parameters, and removes the non-pre-trained encoder–decoder attention weights. Liu et al. [344] applied pre-trained BERT for text summarization. It is a document-level encoder and proposed a general framework for both abstractive and extractive summarization. Zhang et al. [345] proposed a two-stage model based on the S2S paradigm. The model incorporates reinforcer objectives into the learning process and uses BERT on both the encoder and decoder sides. A text can be summarized to produce a condensed version that highlights its main ideas and points. Both abstractive and extractive summarization techniques are used in most cases [77,342,344]. For the Arabic language, Ameen et al. [346] proposed a TB hybrid approach to the Arabic text summarization task.

D. Text Generation (TG)

In AI, where its techniques are frequently used to avoid over-fitting and enhance the generalization of deep neural network models, TG has been extensively studied. The Transformers have been the subject of recent studies to accomplish this task. Among the interesting works, Kumar et al. [26] studied different types of pre-trained TB models, such as GPT2 and BERT, and BART, for conditional TG and show that pre-pending the class labels to text sequences provides a simple yet effective way to condition the pre-trained models for TG. They investigated the differences in data diversity between various pre-trained model-based TG techniques and the degree to which label information is preserved. In addition, Shao et al. [27] suggested the use of conditional BERT contextual generation, which is a method for enhancing data for labeled sentences. By introducing a new conditional MLM, they convert BERT into a conditional BERT. Contextual TG can be improved by using the

trained conditional BERT. ART models such as GPT and S2S models such as T5 are widely used for TG tasks. Several other approaches using TB models were proposed [347–352].

E. Name Entities Recognition (NER)

NER is the task of identifying entities from a text. It identifies names in text and classifies them into predefined groups. It is possible to extract entities such as the names of businesses, locations, dates, numbers, people, and other textual components [353]. Li et al. [354] proposed FLAT, a TB model for Chinese NER, that transforms the lattice structure into a flat structure made up of spans. Each span represents the pair of a character or latent word and the location of that character or word in the original lattice. A method for identifying entities based on their names is called NER. Zhang et al. [275] proposed FinBERT-MRC, which is a BERT-based model for financial NER using BERT under the machine reading comprehension paradigm. Long et al. [355] suggested a flexible method for entity dictionary integration called dictionary-fused BERT. In order to obtain contextualized word and entity representations, the system uses a logit matrix to create a robust loss function and adds an auxiliary task that involves an on-top binary classification to determine whether the token is a mentioned word or not. Jarrar et al. [356] suggested Wojood, an entity corpus with Arabic names using BERT recognition. With the help of the pre-trained ARaBERT, they used the corpus to train a nested NER model based on multi-task learning. There are several studies for NER in different languages including Chinese [357–360], Arabic [361], Spanish [362,363], etc. Liu et al. [198] suggested a two-stage method for fine-tuning BERT for the geological domain knowledge for the NER task. They used in the first fine-tuning stage a pre-trained BERT model, and in the second stage, a small number of samples to complete the NER task for geological reports, based on GeoBERT. Based on ALBERT architecture, Kezhou et al. [364] proposed a model that combines ALBERT with BiLSTM and Conditional Random Field (CRF) to form the ALBERT-BiLSTM-CRF model.

F. Topic Modeling (TM)

Information retrieval, document classification, document summarization, and exploratory text analysis using large text corpora have all been successfully accomplished using topic modeling [365]. Topic modeling refers to a group of NLP algorithms that help us identify semantic topics or patterns in a set of documents. The hidden topics that are present in the corpus are identified using different approaches. Rukhma et al. [366] proposed ZeroBERTo, which is a model that uses an unsupervised clustering step to obtain a compressed data representation before the classification task. Aaron et al. [367] suggested a BERT-based model for the TM of a consumer Twitter discussion relating to telehealth for mental health or substance abuse during pre-pandemic versus pandemic time periods.

G. Machine Translation (MT)

Elaffendi et al. [368] suggested a PIA (polynomial inherent attention) model, based on the Transformer architecture and assessed its effects on the MT task. Dongxing et al. [369] seek to enhance the TB MT strategies. In order to avoid a low-rank bottleneck, they suggested the interacting-head attention mechanism, which selects the right number of heads and induces deeper and wider interactions among the attention heads by low-dimension computations in various sub-spaces of all the tokens. Cheikh et al. [370] suggested two MT systems based on S2S models with attention and TB architectures, French to Wolof and Wolof to French models. They concentrated on French datasets to solve the low-resource MT solution job [371]. For TB models for MT in the programming domain, there are few studies [372], while for the task of Chinese language translation, several works were proposed [373–375].

H. Sentiment Analysis (SA)

Dimple et al. [376] suggested KEAHT, which is a knowledge-enriched attention-based hybrid TB model for social SA, by enhancing the explicit knowledge of lexicalized domain ontology and latent dirichlet allocation (LDA) topic modeling. They used BERT to train the corpus. This method can precisely resolve complex text problems and offers the facility of an attention mechanism. Rolandos et al. [377] used the TB model to address the issue of locating the aforementioned ironic and sarcastic posts. A neural network approach has

been developed on top of the model for SA. Several other studies were conducted as part of the social media sentiment analysis [378–382].

4. Discussion

Transformers had a significant impact on how ML is used in various tasks involving NLP. S2S models, ART models, and AET models are the overall types of these Transformers. In this paper, we reviewed these models in detail. We started by studying the idea of encoder–decoder architectures. These techniques perform superbly across a variety of domains. However, there are still a few challenges to be resolved, as Transformer's potential for textual-based NLP tasks has not been fully investigated. As we detailed in this review, each new TB model builds on the models that came before it and suggests modifications and improvements at the level of applications and architecture to solve specific issues. We can generally address some Transformer architecture drawbacks; among these challenges, we can list:

- Attention is limited to handling fixed-length text sequences. Before being entered as input into the system, the text must be divided into a predetermined number of segments or chunks. This may lead to the loss of context as a result of text chunking.
- The inability of TB models to process extended sequences is mainly caused by the computational and memory cost of the self-attention module.

In this section, we go over these difficulties in detail and offer a summary of the proposed approaches to solve them. The two levels of discussion are TB model architecture evaluation and TB model application evaluation.

4.1. Architecture Level

Many different architectures have been developed, changing the fundamental characteristics of TB models. The fact that these extensions use a wide range of terms to describe every component of the model is what unites them. Related studies categorize some models as auto-regressive, while others are classified as auto-encoding or S2S. These three encoder–decoder architectures' similarities and differences are discussed in Section 3. In order to provide the necessary context, we went over the fundamentals of encoder–decoder architectures. A brief discussion of traditional Transformer architecture is also included. After that, we discussed auto-encoding, auto-regressive, and S2S models.

4.1.1. TB Model Configurations Evaluation

- **RQ1:** Does the model size have an impact on the model's performance?
- **RQ2:** Does the number of blocks and attention layers have an effect on the model's performance?

The Transformer's parameters, such as the quantity of decoder and encoder layers, etc., are very flexible, and the outcomes would probably be enhanced with better training and tuning of these parameters. Regularization, load balancing, and fine-tuning hyper-parameters are all necessary to comprehend the dynamics of expert models' fine-tuning. After reviewing all TB models, it is clear that there is a consistent link, demonstrating that for all TB models, stronger pre-training resulted in better downstream outcomes. We also discovered that TB models perform similarly in the small to medium model size regime for fixed upstream perplexity. The larger models, such as the Switch-Transformer [383] regime, may not always well convey their confusion from the upstream to the downstream fine-tuning on the SuperGLUE task. The best compute-efficient training strategy is the exact opposite of the widespread practice of reducing model size. On the other hand, training TB models on a tight budget should not drastically decrease model size. However, instead, training with the original size and stopping the training very early is more efficient. To put it another way, it is possible to increase the model size while sacrificing convergence, which forces the reevaluation of the implicit assumption that models must be trained until convergence. Ontanon et al. [384] provided an empirical evaluation of the Transformer model

design space. The results show that when compared to a baseline transformer, changing the configuration can result in significant improvements in compositional generalization. They provided the configurations required to achieve this goal.

Large models are very helpful even on datasets with 4000 labeled examples. Going from the 110 M model to the 340 M parameters model is beneficial as in the case of the GPT models family. In DL, using more computing by expanding the size of the model, increasing the size of the dataset, or the number of training steps frequently results in greater accuracy. This is particularly true in light of the recent popularity of unsupervised pre-training techniques such as BERT-based models, which enable training on very sizable models and datasets. Large-scale training is unfortunately very computationally expensive, especially without the hardware capabilities of significant industry research labs. Therefore, obtaining high accuracy while staying within one's hardware budget and training time is typically the goal of advanced research to improve the TB models. Large models seem to be unworkable for the majority of training budgets. Instead, using models with small hidden sizes or few layers is the go-to tactic for maximizing training efficiency because these models run faster and consume less memory. Growing the size of a TB model can enhance the effectiveness of training and inference, demonstrating that the rule to train on a large dataset then compresses—Train Large, Then Compress [385]—is an efficient recommendation. This discovery raises a number of intriguing new queries, such as *why larger models converge more quickly and compress more effectively?* These details about the TB models and pre-trained LM, in general, are still under investigation by the research community.

4.1.2. Tokenization Evaluation

- **RQ1:** What is the effect of preprocessing techniques on TB models?
- **RQ2:** What differentiates the tokenization of the TB models from the traditional methods?

TB models use different tokenization methods described in Section 2. **For example:** *The GPT family [41–43] of models process text using tokens, which are common sequences of characters found in text. The models are excellent at producing the following token in a series of tokens because they are aware of the statistical relationships between these tokens.* The context representation of the sequences in TB models limits how well the TB models perform on many NLP tasks. We can differentiate the TB tokenization methods from the traditional ones in three aspects:

- First, TB models' success is due to a pre-training task that was self-supervised and promoted general language comprehension without considering the particular requirements of ranking tasks. There is a continuum between the initial self-supervised training task and the final interaction ranker. Isolating ranking-aware pre-training tasks may result in gains in both effectiveness and efficiency, especially when there is a dearth of data on the target task. By ranking tasks, we mean the downstream tasks that require the model to iterate during fine-tuning and learn by ranking patterns in each iteration until achieving the best results. In other terms, the ranking task is the reduction of the loss function used in each pre-training objective.
- Second, TB models combine a lengthy sequence with numerous layers, but it is not clear what value this rich semantics adds in terms of ranking. **For example:** *the deep layers of BERT resulted in some, albeit modest, performance improvements, but it is still unclear how exactly the model learns to accurately understand the patterns. Some refer to the MLM pre-training step. The masking method is what differentiate BERT from the original Transformer encoder block, but even the masking mechanism is not well explained in term of what patterns are masked and how the model learns these patterns.*
- Third, TB models create distinct sequence representations by employing a different form of tokenization than feature extraction techniques such as stemming and lemmatizing. **For example,** *BERT tokenizer's constrained vocabulary affects a large number of*

long-tail tokens, resulting in significant efficiency gains at the expense of a negligible drop in effectiveness.

4.1.3. Improving the Transformer Architecture

- **RQ1:** How to improve the TB model's time, training speed, memory, complexity, and efficiency?
- **RQ2:** What techniques were proposed to improve the Transformer architecture?

The inefficiency of TB models in processing extended sequences, which is primarily caused by the self-attention module's computational and memory cost, is one of the most challenging aspects of using them. It is also challenging to train the Transformer on small-scale data due to its flexible architecture, which enables it to make few assumptions about the structural bias of the input data. A number of Transformer variants, or **X-formers**, have been suggested recently to address these problems. These *X-formers* improve the first Transformer architecture in a number of different ways. When it comes to improving model efficiency, lightweight attention strategies such as sparse attention variations and divide-and-conquer techniques such as recurrent and hierarchical mechanisms are employed. Additionally, several models improved model generalization by adding structural bias, regularization, or pre-training on vast amounts of unlabeled data. Additionally, techniques for model adaptation were suggested to customize the Transformer for particular downstream tasks and applications. There may be one or more problems that existing X-formers can address. In order to avoid overfitting on small datasets, sparse attention versions, for instance, introduce a structural prior to the input data while reducing computing complexity.

Among the TB models that made modifications to the standard architecture to enhance the model's time, training speed, memory, complexity, or efficiency, we present first, the **Sparse Transformer** [386], which is a TB architecture that uses the attention matrix's sparse factorization to save time and memory. A reorganized residual block and weight initialization, a collection of sparse attention kernels that effectively compute portions of the attention matrix, and updates to attention weights made during the backward pass to save memory are additional changes made to the Transformer architecture. Zihao et al. proposed the **BP-Transformer (BPT)** [387] to achieve better harmony between self-attention capability and computational complexity. By using binary partitioning (BP), the architecture divides the input sequence into various multi-scale spans. As the relative distance grows, it incorporates an inductive bias that shifts attention from fine-grained to coarse-grained context information. A graph neural network with multi-scale spans can be compared to BPT by saying that it has nodes. Kitaev et al. proposed the **reformer** [388], which is a TB architecture that aims to increase productivity. Its complexity is changed from dot-product attention to one that employs locality-sensitive hashing. Reformers also substitute reversible residual layers for conventional residuals. Meanwhile, **Performers** [389] is a variant of the linear attention-based Transformer that uses the FAVOR+ (Fast Attention Via Positive Orthogonal Random Features) mechanism to greatly increase the space and time complexity of Transformers [390]. This mechanism opens up new directions in the study of Transformers and the function of non-sparsifying attention mechanisms by effectively and impartially estimating the original SoftMax-based Transformer with linear space and time complexity. **Rotary Transformer** [391] or RoFormer is built using a rotary position embedding (RoPE) for the Transformer structure. The only absolute position coding that can currently be used for linear attention is ROPE, which has strong theoretical foundations. Although Transformer models have been successful in a number of tasks, their high memory and computing resource requirements prevent their implementation on devices with limited resources, such as mobile phones. In this section, we examine the research conducted on Transformer model compression and acceleration for effective implementation. This includes knowledge distillation, network quantization, network pruning, low-rank decomposition, and compact architecture design. **Transformer-XL** is a new architecture that permits natural language comprehension outside of a context with a

fixed length while maintaining temporal coherence. Its two main innovations are a positional encoding system and a segment-level recurrence mechanism. Its main advantages over the Transformer model include its ability to capture longer-term dependency and resolve the context fragmentation issue. The results of the experiments demonstrate that compared to RNNs or the original Transformer, Transformer-XL takes much longer to learn dependency.

4.2. Application Level

The power of the TB models is explained by the fact that different combinations and layer implementations can be used, which means that the application range was not constrained. TB models were initially applied to NLP-related tasks before being expanded to include other tasks. Transformers are strong sequence models, and they have a wide range of business applications in a number of different industries. Finance, biology, cybersecurity, healthcare, and tasks involving languages are included among these industries, as detailed in Sections 4 and 5.

4.2.1. Benchmarks Evaluation

- **RQ1:** Are the currently used benchmarks to evaluate the TB models efficient?
- **RQ2:** Pre-raining vs. fine-tuning datasets for TB models?

Not only the selection of the optimal automatic metrics for evaluation for each model and downstream task is important but also the datasets used to train the models; their size, relevance, and annotations are important to determine how effective the TB model is for the specific task. The identification of benchmark units is crucial in the practice of research management and evaluation. Generally, in order to comprehend and track a research body's growth and performance, researchers frequently use benchmark units as points of comparison. A set of test programs using various measurement techniques was implemented for the benchmark evaluation. For the purpose of this review, we listed the most common NLP benchmarks used to evaluate the downstream tasks of the TB models. These benchmarks are well annotated and widely used by both academia and industry research groups. Usually, ML and TB models use very large datasets for the pre-training phase, given the nature of this task as a supervised one. While for the fine-tuning, smaller, task-specific datasets are used to train and evaluate the TB models, as explained in Section 2.

4.2.2. Downstream Tasks Evaluation

- **RQ1:** What is the difference between fine-tuning pre-trained TB models and using feature-based approaches?
- **RQ2:** What techniques can be used to improve the fine-tuning phase?

Fine-tuning and feature-based approaches are two ways to apply previously trained language representations to downstream NLP tasks. On the one hand, the fine-tuning method trains on the downstream tasks by simply fine-tuning all the pre-trained parameters, and on the other hand, it introduces minimal task-specific parameters. The feature-based approach, on the other hand, employs task-specific architectures that use the trained representations as input features to learn the task. However, sometimes removing some of these weights and re-initializing them during the fine-tuning process aids in obtaining better fine-tuning results. It is difficult to pinpoint the main reason for the unstable and subpar performance of the TB model. Typically, these issues are more common in settings with large models and small datasets. Associated data characteristics and downstream tasks' characteristics can also be important. Along with hyperparameter tuning, applying some of the sophisticated fine-tuning methods can improve outcomes. Table 7 shows some fine-tuning techniques used mostly in AET models such as BERT, RoBERTa, etc., to improve the traditional TB fine-tuning process.

Table 7. List of methods to improve the fine-tuning phase for the TB models.

Method	Highlights	Examples
Debiasing Omission (DO) [392]	The most popular optimizer for fine-tuning BERT is BERTADAM, which is a modified version of the ADAM first-order stochastic optimization technique. The difference between it and the original ADAM algorithm is the absence of a bias correction step. One of the main causes of BERT fine-tuning instability is the bias correction omission, which affects the learning rate, particularly early in the process.	BertADAM [392]
Re-Initializing Transformer Layers (RTL) [393]	It is the re-initialization of the top N layers of the transformer. Selecting the ideal number of top layers is crucial because increasing the number of re-initializations past the ideal point may lead to subpar results.	[394] [395]
Utilizing Intermediate Layers (UIL) [396]	The semantic information in the intermediate layers is ignored in existing BERT-based works, which only use the final output layer of BERT. This approach recommends investigating the possibility of using BERT intermediate layers to improve the effectiveness of fine-tuning BERT.	[396] [397] [398]
Layer-wise Learning Rate Decay (LLRD) [399]	With the learning rate distribution method known as LLRD, the top layers learn at higher rates than the bottom layers, which learn at lower rates. In order to achieve this, the learning rate is first set for the top layer and then decreased layer by layer from top to bottom using a multiplicative decay rate.	XLNet [44] ELECTRA [63]
Mixout Regularization (MR) [400]	Dropout [401] and DropConnect [402] inspired Mixout, which is a stochastic regularization technique. At every training iteration, the pre-trained value for each model parameter is substituted. It is shown that this constrains the fine-tuned model from deviating too much from the pre-trained initialization in order to achieve the goal of preventing catastrophic forgetting.	[400] [403]
Pre-trained Weight Decay (PWD) [404]	The regularization technique known as weight decay (WD) is common. By deducting a fixed amount from the model's pre-trained parameters, this technique is modified for fine-tuning pre-trained models. Pre-trained weight decay in Transformer fine-tuning performs better than conventional weight decay and stabilizes fine-tuning.	[405], BioBERT [5], DeFormer [406]
Stochastic Weight Averaging (SWA) [407]	A method of deep neural network training that uses a modified learning rate schedule and averages the weights of the networks iteratively traversed. Weights are averaged to produce larger optima and better generalization.	ALBERT [408]
Learning Rate Warm-up Steps (LRWS) [409]	A linear schedule has two phases when warming-up steps are included. The optimizer's initial learning rates are set during the warm-up phase, which comes first. The learning rates therefore begin to increase linearly from 0 to the initialized learning rates. Next comes the normal phase, when the steps start to linearly decrease until they reach 0.	EfficientBERT [410]

4.2.3. Multilinguality Evaluation

- **RQ1:** How efficient are the cross-lingual TB models?
- **RQ2:** Is using a monolingual model better than relying on a multilingual one?

English-language studies have dominated the field of LBM for TB models. It has been suggested that using multilingual text for pre-training can effectively target multiple languages using cross-lingual shared representations. The power of pre-training may now be applied to many other languages thanks to multilingual LMs such as mBERT, XLM, XLM-R, mT5, etc., given their success with zero-shot transfer learning. Cross-lingual encoders are often evaluated using either zero-shot cross-lingual transfer in supervised downstream tasks or unsupervised cross-lingual textual similarity. Based on the experiment results by Conneau et al. [110], it is preferable to use high-capacity multilingual TB models trained on much larger pre-training data instead of multilingual ones with lower capacities. While for monolingual tasks, it is preferable to use monolingual models. They compared the performance of the state-of-the-art monolingual models, BERT and RoBERTa, with the state-of-the-art multilingual models mBERT, XLM-R base, and XLM-R on state-of-the-art NLP benchmarks. They show that XLM-R performs superior to XLM-R base, which performs superior to mBERT. The number of parameters of mBERT, XLM-R base, and XLM-R are, respectively, 172 M, 270 M, and 559 M. On most tasks, none of the multilingual TB models outperformed the monolingual ones. However, the number of parameters in BERT and RoBERTa is equally 335 M. For the size of the pre-training datasets for TB models, Liu et al. [411] demonstrated that pre-training mBERT on larger corpora, as opposed to smaller corpora, improves cross-lingual transfer learning. Additionally, Lauscher et al. [412] demonstrated that for some tasks, the amount of data collected in the target language for pre-training the multilingual TB models has a meaningful impact on zero-shot transfer performance.

5. Conclusions

The aim of this survey is to give the NLP research community an up-to-date review of TB models, helping to boost and highlight the most important aspects of these models at both the application and architectural levels. In this article, we reviewed the literature on TB models, paying particular attention to their variety and to the description of their various architectures. We explained each model compared to the Transformer's typical architecture. We divided the existing models into three categories: auto-encoding Transformers, auto-regressive Transformers, and S2S Transformers, as explained in our taxonomy. We examined the applications, downstream tasks, as well as languages of the TB models. We detailed the model's limitations on different levels and showed the intended methods for improving TB approaches based on differences in usage modes and architectural changes. We provided thorough explanations of many examples of these effective TB models. Finally, we provided examples of the Transformers being used in the NLP field. We concluded that as in NLP, the TB applications in the listed domains and others may enhance innovative discoveries and AI-powered products.

Author Contributions: A.R. and M.A.A. contributed equally to this work. All authors have read and agreed to the published version of the manuscript.

Funding: This research was enabled in part by support provided by the Natural Sciences and Engineering Research Council of Canada (NSERC), funding reference number RGPIN-2018-06233.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AE	Auto-Encoding
AET	Auto-Encoding Transformers
AI	Artificial Intelligence
ALM	Alternate Language Modeling
AR	Auto-Regression
ART	Auto-Regression Transformers
bBPE	Byte Level Byte Pair Encoding
BERT	Bidirectional Encoder Representations from Transformers
BPE	Byte Pair Encoding
CLM	Causal Language Modeling
CNN	Convolutional Neural Networks
DAE	Denoising Auto Encoder
DBM	Domain-Based Models
DL	Deep Learning
DNN	Deep Neural Networks
DS	Document Summarization
GELU	Gaussian Error Linear Unit
GRU	Gated Recurrent Networks
GSG	Gap Sentences Generation
K	Key
LBM	Language-Based Models
LM	Language Model
LSTM	Long Short-Term Memory
MAE	Mean Absolute Error
ML	Machine Learning
MLM	Masked Language Model
MSE	Mean Square Error
MT	Machine Translation
NER	Named Entity Recognition
NLP	Natural Language Processing
NSP	Next Sentence Prediction
PE	Positional Encoding
Q	Query
QA	Question Answering
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Networks
RTD	Replaced Token Detection
RTS	Random Token Substitution
SA	Sentiment Analysis
SBO	Sentence Boundary Objective
SLM	Swapped Language Modeling
SLR	Segment Level Recurrence
SOP	Sentence Order Prediction
SSLM	Sequence-to-Sequence LM
STD	Shuffled Token Detection
TB	Transformer-Based
TBM	Task Based Models
TC	Text Classification
TG	Text Generation
TLM	Translation Language Modeling
V	Value
WSO	Word Structural Objective

References

1. Mitkov, R. *The Oxford Handbook of Computational Linguistics*; Oxford University Press: Oxford, UK, 2022.
2. Wilie, B.; Vincentio, K.; Winata, G.I.; Cahyawijaya, S.; Li, X.; Lim, Z.Y.; Soleman, S.; Mahendra, R.; Fung, P.; Bahar, S.; et al. Indonlu: Benchmark and resources for evaluating indonesian natural language understanding. *arXiv* **2020**, arXiv:2009.05387.
3. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
4. Dumitrescu, S.D.; Avram, A.M.; Pyysalo, S. The birth of Romanian BERT. *arXiv* **2020**, arXiv:2009.08712.
5. Lee, J.; Yoon, W.; Kim, S.; Kim, D.; Kim, S.; So, C.H.; Kang, J. BioBERT: A pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics* **2020**, *36*, 1234–1240. [CrossRef]
6. Alsentzer, E.; Murphy, J.R.; Boag, W.; Weng, W.H.; Jin, D.; Naumann, T.; McDermott, M. Publicly available clinical BERT embeddings. *arXiv* **2019**, arXiv:1904.03323.
7. Peng, Y.; Yan, S.; Lu, Z. Transfer learning in biomedical natural language processing: An evaluation of BERT and ELMo on ten benchmarking datasets. *arXiv* **2019**, arXiv:1906.05474.
8. Gu, Y.; Tinn, R.; Cheng, H.; Lucas, M.; Usuyama, N.; Liu, X.; Naumann, T.; Gao, J.; Poon, H. Domain-specific language model pretraining for biomedical natural language processing. *ACM Trans. Comput. Healthc. (HEALTH)* **2021**, *3*, 1–23. [CrossRef]
9. Yang, Y.; Uy, M.C.S.; Huang, A. FinBERT: A Pretrained Language Model for Financial Communications. *arXiv* **2020**, arXiv:2006.08097.
10. Gururangan, S.; Marasovic, A.; Swayamdipta, S.; Lo, K.; Beltagy, I.; Downey, D.; Smith, N.A. Don't stop pretraining: Adapt language models to domains and tasks. *arXiv* **2020**, arXiv:2004.10964.
11. Caselli, T.; Basile, V.; Mitrovic, J.; Granitzer, M. Hatebert: Retraining bert for abusive language detection in english. *arXiv* **2010**, arXiv:2010.12472.
12. Zhou, J.; Tian, J.; Wang, R.; Wu, Y.; Xiao, W.; He, L. Sentix: A sentiment-aware pre-trained model for cross-domain sentiment analysis. In Proceedings of the 28th International Conference on Computational Linguistics, Online, 8–13 December 2020; pp. 568–579.
13. Muller, M.; Salathe, M.; Kummervold, P.E. Covid-twitter-bert: A natural language processing model to analyse covid-19 content on twitter. *arXiv* **2020**, arXiv:2005.07503.
14. Barbieri, F.; Camacho-Collados, J.; Neves, L.; Espinosa-Anke, L. Tweeteval: Unified benchmark and comparative evaluation for tweet classification. *arXiv* **2020**, arXiv:2010.12421.
15. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning (Adaptive Computation and Machine Learning Series)*; The MIT Press Cambridge: Cambridge, MA, USA, 2016; Volume 19, pp. 226, 305–307.
16. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]
17. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef]
18. Glorot, X.; Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, PMLR 9:249-256, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010.
19. LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [CrossRef]
20. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **2012**, *25*, 1097–1105. [CrossRef]
21. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is all you need. In Proceedings of the Advances in Neural Information Processing Systems 30 (NIPS 2017), Long Beach, CA, USA, 4–9 December 2017; pp. 5998–6008.
22. Shaw, P.; Uszkoreit, J.; Vaswani, A. Self-attention with relative position representations. *arXiv* **2018**, arXiv:1803.02155.
23. Chorowski, J.; Bahdanau, D.; Serdyuk, D.; Cho, K.; Bengio, Y. Attention-based models for speech recognition. *arXiv* **2015**, arXiv:1506.07503.
24. Firat, O.; Cho, K.; Bengio, Y. Multi-way, multilingual neural machine translation with a shared attention mechanism. *arXiv* **2016**, arXiv:1601.01073.
25. Choi, H.; Cho, K.; Bengio, Y. Fine-grained attention mechanism for neural machine translation. *Neurocomputing* **2018**, *284*, 171–176. [CrossRef]
26. Kumar, V.; Choudhary, A.; Cho, E. Data augmentation using pre-trained transformer models. *arXiv* **2020**, arXiv:2003.02245.
27. Shao, T.; Guo, Y.; Chen, H.; Hao, Z. Transformer-Based Neural Network for Answer Selection in Question Answering. *IEEE Access* **2019**, *7*, 26146–26156. [CrossRef]
28. Kowsher, M.; Sobuj, M.S.I.; Shahriar, M.F.; Prottasha, N.J.; Arefin, M.S.; Dhar, P.K.; Koshiba, T. An Enhanced Neural Word Embedding Model for Transfer Learning. *Appl. Sci.* **2022**, *12*, 2848. [CrossRef]
29. Bensoltane, R.; Zaki, T. Towards Arabic aspect-based sentiment analysis: A transfer learning-based approach. *Soc. Netw. Anal. Min.* **2022**, *12*, 7. [CrossRef]
30. Prottasha, N.J.; Sami, A.A.; Kowsher, M.; Murad, S.A.; Bairagi, A.K.; Masud, M.; Baz, M. Transfer Learning for Sentiment Analysis Using BERT Based Supervised Fine-Tuning. *Sensors* **2022**, *22*, 4157. [CrossRef] [PubMed]
31. Sasikala, S.; Ramesh, S.; Gomathi, S.; Balambigai, S.; Anbumani, V. Transfer learning based recurrent neural network algorithm for linguistic analysis. *Concurr. Comput. Pract. Exp.* **2022**, *34*, e6708. [CrossRef]

32. Taneja, K.; Vashishtha, J. Comparison of Transfer Learning and Traditional Machine Learning Approach for Text Classification. In Proceedings of the IEEE 2022 9th International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, India, 23–25 March 2022; pp. 195–200.
33. Qiao, Y.; Zhu, X.; Gong, H. BERT-Kcr: Prediction of lysine crotonylation sites by a transfer learning method with pre-trained BERT models. *Bioinformatics* **2022**, *38*, 648–654. [CrossRef]
34. Qasim, R.; Bangyal, W.H.; Alqarni, M.A.; Ali Almazroi, A. A fine-tuned BERT-based transfer learning approach for text classification. *J. Healthc. Eng.* **2022**, *2022*, 3498123. [CrossRef]
35. Li, N.; Liu, S.; Liu, Y.; Zhao, S.; Liu, M. Neural speech synthesis with transformer network. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; Volume 33, pp. 6706–6713.
36. Baevski, A.; Zhou, Y.; Mohamed, A.; Auli, M. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 12449–12460.
37. Tay, Y.; Dehghani, M.; Bahri, D.; Metzler, D. Efficient transformers: A survey. *arXiv* **2020**, arXiv:2009.06732.
38. Lin, T.; Wang, Y.; Liu, X.; Qiu, X. A survey of transformers. *arXiv* **2021**, arXiv:2106.04554.
39. Kalyan, K.S.; Rajasekharan, A.; Sangeetha, S. Ammus: A survey of transformer-based pretrained models in natural language processing. *arXiv* **2021**, arXiv:2108.05542.
40. Gillioz, A.; Casas, J.; Mugellini, E.; Khaled, O.A. Overview of the Transformer-based Models for NLP Tasks. In Proceedings of the 2020 15th Conference on Computer Science and Information Systems (FedCSIS), Sofia, Bulgaria, 6–9 September 2020; pp. 179–183. [CrossRef]
41. Brown, T.B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language models are few-shot learners. *arXiv* **2020**, arXiv:2005.14165.
42. Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I. Language models are unsupervised multitask learners. *OpenAI Blog* **2019**, *1*, 9.
43. Radford, A.; Narasimhan, K.; Salimans, T.; Sutskever, I. *Improving Language Understanding by Generative Pre-Training*; Online, OpenAI. 2018. Available online: https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf (accessed on 2 October 2022).
44. Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J.; Salakhutdinov, R.R.; Le, Q.V. Xlnet: Generalized autoregressive pretraining for language understanding. In Proceedings of the NIPS'19: Proceedings of the 33rd International Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; pp. 5753–5763.
45. Sun, Y.; Wang, S.; Li, Y.; Feng, S.; Chen, X.; Zhang, H.; Tian, X.; Zhu, D.; Tian, H.; Wu, H. ERNIE: Enhanced Representation through Knowledge Integration. *arXiv* **2019**, arXiv:1904.09223.
46. Sun, Y.; Wang, S.; Li, Y.; Feng, S.; Tian, H.; Wu, H.; Wang, H. ERNIE 2.0: A Continual Pre-training Framework for Language Understanding. *arXiv* **2020**, arXiv:1907.12412.
47. Wang, Z.; Ma, Y.; Liu, Z.; Tang, J. R-transformer: Recurrent neural network enhanced transformer. *arXiv* **2019**, arXiv:1907.05572.
48. Parisotto, E.; Song, F.; Rae, J.; Pascanu, R.; Gulcehre, C.; Jayakumar, S.; Jaderberg, M.; Kaufman, R.L.; Clark, A.; Noury, S.; et al. Stabilizing transformers for reinforcement learning. In Proceedings of the International Conference on Machine Learning, PMLR, Virtual Event, 13–18 July 2020; pp. 7487–7498.
49. Lakew, S.M.; Cettolo, M.; Federico, M. A comparison of transformer and recurrent neural networks on multilingual neural machine translation. *arXiv* **2018**, arXiv:1806.06957.
50. Sutskever, I.; Vinyals, O.; Le, Q.V. Sequence to sequence learning with neural networks. In Proceedings of the 27th International Conference on Neural Information Processing Systems, Montreal, QC, Canada, 8–13 December 2014; pp. 3104–3112.
51. Chung, J.; Gulçehre, Ç.; Cho, K.; Bengio, Y. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *arXiv* **2014**, arXiv:1412.3555.
52. Dai, Z.; Yang, Z.; Yang, Y.; Carbonell, J.; Le, Q.V.; Salakhutdinov, R. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv* **2019**, arXiv:1901.02860.
53. Hendrycks, D.; Gimpel, K. Gaussian error linear units (gelus). *arXiv* **2016**, arXiv:1606.08415.
54. Ba, J.L.; Kiros, J.R.; Hinton, G.E. Layer normalization. *arXiv* **2016**, arXiv:1607.06450.
55. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient estimation of word representations in vector space. *arXiv* **2013**, arXiv:1301.3781.
56. Pennington, J.; Socher, R.; Manning, C.D. Glove: Global vectors for word representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; pp. 1532–1543.
57. Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv* **2019**, arXiv:1907.11692.
58. Lample, G.; Conneau, A. Cross-lingual language model pretraining. *arXiv* **2019**, arXiv:1901.07291.
59. Sennrich, R.; Haddow, B.; Birch, A. Neural machine translation of rare words with subword units. *arXiv* **2015**, arXiv:1508.07909.
60. Kudo, T.; Richardson, J. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv* **2018**, arXiv:1808.06226.
61. Wu, Y.; Schuster, M.; Chen, Z.; Le, Q.V.; Norouzi, M.; Macherey, W.; Krikun, M.; Cao, Y.; Gao, Q.; Macherey, K.; et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv* **2016**, arXiv:1609.08144.

62. Dong, L.; Yang, N.; Wang, W.; Wei, F.; Liu, X.; Wang, Y.; Gao, J.; Zhou, M.; Hon, H.W. Unified language model pre-training for natural language understanding and generation. In Proceedings of the 33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, BC, Canada, 8–14 December 2019; pp. 13063–13075.
63. Clark, K.; Luong, M.T.; Le, Q.V.; Manning, C.D. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv* **2020**, arXiv:2003.10555.
64. Panda, S.; Agrawal, A.; Ha, J.; Bloch, B. Shuffled-token detection for refining pre-trained roberta. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop, Online, 6–11 June 2021; pp. 88–93.
65. Di Liello, L.; Gabburo, M.; Moschitti, A. Efficient pre-training objectives for transformers. *arXiv* **2021**, arXiv:2104.09694.
66. Chi, Z.; Dong, L.; Wei, F.; Wang, W.; Mao, X.L.; Huang, H. Cross-lingual natural language generation via pre-training. *Artif. Intell.* **2020**, *34*, 7570–7577. [CrossRef]
67. Yang, J.; Ma, S.; Zhang, D.; Wu, S.; Li, Z.; Zhou, M. Alternating language modeling for cross-lingual pre-training. In Proceedings of the AAAI Conference on Artificial Intelligence, New York Hilton Midtown, NY, USA, 7–12 February 2020; Volume 34, pp. 9386–9393.
68. Joshi, M.; Chen, D.; Liu, Y.; Weld, D.S.; Zettlemoyer, L.; Levy, O. Spanbert: Improving pre-training by representing and predicting spans. *Trans. Assoc. Comput. Linguist.* **2020**, *8*, 64–77. [CrossRef]
69. Lan, Z.; Chen, M.; Goodman, S.; Gimpel, K.; Sharma, P.; Soricut, R. Albert: A lite bert for self-supervised learning of language representations. *arXiv* **2019**, arXiv:1909.11942.
70. Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; Liu, P.J. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv* **2019**, arXiv:1910.10683.
71. Xue, L.; Constant, N.; Roberts, A.; Kale, M.; Al-Rfou, R.; Siddhant, A.; Barua, A.; Raffel, C. mT5: A massively multilingual pre-trained text-to-text transformer. *arXiv* **2020**, arXiv:2010.11934.
72. Song, K.; Tan, X.; Qin, T.; Lu, J.; Liu, T.Y. MASS: Masked Sequence to Sequence Pre-training for Language Generation. In Proceedings of the International Conference on Machine Learning (ICML), Long Beach, CA, USA, 9–15 June 2019.
73. Lewis, M.; Liu, Y.; Goyal, N.; Ghazvininejad, M.; Mohamed, A.; Levy, O.; Stoyanov, V.; Zettlemoyer, L. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv* **2019**, arXiv:1910.13461.
74. Jiao, X.; Yin, Y.; Shang, L.; Jiang, X.; Chen, X.; Li, L.; Wang, F.; Liu, Q. Tinybert: Distilling bert for natural language understanding. *arXiv* **2019**, arXiv:1909.10351.
75. Beltagy, I.; Lo, K.; Cohan, A. SciBERT: A pretrained language model for scientific text. *arXiv* **2019**, arXiv:1903.10676.
76. Huang, K.; Altosaar, J.; Ranganath, R. Clinicalbert: Modeling clinical notes and predicting hospital readmission. *arXiv* **2019**, arXiv:1904.05342.
77. Zhang, X.; Wei, F.; Zhou, M. HIBERT: Document Level Pre-training of Hierarchical Bidirectional Transformers for Document Summarization. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; pp. 5059–5069. [CrossRef]
78. Goyal, S.; Choudhary, A.R.; Chakaravarthy, V.; ManishRaje, S.; Sabharwal, Y.; Verma, A. PoWER-BERT: Accelerating BERT inference for Classification Tasks. *arXiv* **2020**, arXiv:2001.08950.
79. Sanh, V.; Debut, L.; Chaumond, J.; Wolf, T. DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter. *arXiv* **2019**, arXiv:1910.01108.
80. Liu, W.; Zhou, P.; Zhao, Z.; Wang, Z.; Deng, H.; Ju, Q. FastBERT: A Self-distilling BERT with Adaptive Inference Time. *arXiv* **2020**, arXiv:2004.02178.
81. Wu, X.; Lv, S.; Zang, L.; Han, J.; Hu, S. Conditional BERT contextual augmentation. In *International Conference on Computational Science*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 84–95.
82. Wu, C.S.; Hoi, S.; Socher, R.; Xiong, C. Tod-bert: Pre-trained natural language understanding for task-oriented dialogues. *arXiv* **2020**, arXiv:2004.06871.
83. Mackenzie, J.; Benham, R.; Petri, M.; Trippas, J.R.; Culpepper, J.S.; Moffat, A. CC-News-En: A Large English News Corpus. In Proceedings of the 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, 19–23 October 2020; pp. 3077–3084.
84. Wang, A.; Singh, A.; Michael, J.; Hill, F.; Levy, O.; Bowman, S. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, Brussels, Belgium, 1 November 2018; Association for Computational Linguistics: Brussels, Belgium, 2018; pp. 353–355. [CrossRef]
85. Rajpurkar, P.; Zhang, J.; Lopyrev, K.; Liang, P. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP), Austin, TX, USA, 1–5 November 2016; pp. 2383–2392.
86. Reddy, S.; Chen, D.; Manning, C.D. CoQA: A Conversational Question Answering Challenge. *Trans. Assoc. Comput. Linguist.* **2019**, *7*, 249–266. [CrossRef]
87. Yang, L.; Zhang, M.; Li, C.; Bendersky, M.; Najork, M. Beyond 512 Tokens: Siamese Multi-depth Transformer-based Hierarchical Encoder for Long-Form Document Matching. In Proceedings of the 29th ACM International Conference on Information & Knowledge Management, Online, 19–23 October 2020; pp. 1725–1734.

88. Liu, Y.; Gu, J.; Goyal, N.; Li, X.; Edunov, S.; Ghazvininejad, M.; Lewis, M.; Zettlemoyer, L. Multilingual Denoising Pre-training for Neural Machine Translation. *Trans. Assoc. Comput. Linguist.* **2020**, *8*, 726–742. [CrossRef]
89. He, P.; Liu, X.; Gao, J.; Chen, W. DeBERTa: Decoding-enhanced BERT with Disentangled Attention. *arXiv* **2021**, arXiv:2006.03654.
90. Sun, Z.; Yu, H.; Song, X.; Liu, R.; Yang, Y.; Zhou, D. MobileBERT: A Compact Task-Agnostic BERT for Resource-Limited Devices. *arXiv* **2020**, arXiv:2004.02984.
91. de Wynter, A.; Perry, D. Optimal Subarchitecture Extraction For BERT. *arXiv* **2020**, arXiv:2010.10499.
92. Xin, J.; Tang, R.; Lee, J.; Yu, Y.; Lin, J.J. DeeBERT: Dynamic Early Exiting for Accelerating BERT Inference. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Online, 5–10 July 2020; pp. 2246–2251
93. Kanade, A.; Maniatis, P.; Balakrishnan, G.; Shi, K. Learning and Evaluating Contextual Embedding of Source Code. In Proceedings of the 37th International Conference on Machine Learning (ICML), Online, 13–18 July 2020; pp. 5110–5121
94. Hou, L.; Huang, Z.; Shang, L.; Jiang, X.; Liu, Q. DynaBERT: Dynamic BERT with Adaptive Width and Depth. *arXiv* **2020**, arXiv:2004.04037.
95. Zhang, W.; Hou, L.; Yin, Y.; Shang, L.; Chen, X.; Jiang, X.; Liu, Q. TernaryBERT: Distillation-aware Ultra-low Bit BERT. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Online, 16–20 November 2020.
96. Kim, S.; Gholami, A.; Yao, Z.; Mahoney, M.W.; Keutzer, K. I-BERT: Integer-only BERT Quantization. *arXiv* **2021**, arXiv:2101.01321.
97. Jiang, Z.; Yu, W.; Zhou, D.; Chen, Y.; Feng, J.; Yan, S. ConvBERT: Improving BERT with Span-based Dynamic Convolution. *arXiv* **2020**, arXiv:2008.02496.
98. Iandola, F.N.; Shaw, A.E.; Krishna, R.; Keutzer, K. SqueezeBERT: What can computer vision teach NLP about efficient neural networks? In Proceedings of the SustaiNLP: Workshop on Simple and Efficient Natural Language Processing, Online, 20 November 2020; pp. 124–135.
99. Cui, Y.; Che, W.; Liu, T.; Qin, B.; Wang, S.; Hu, G. Revisiting Pre-Trained Models for Chinese Natural Language Processing. *arXiv* **2020**, arXiv:2004.13922.
100. Bai, H.; Zhang, W.; Hou, L.; Shang, L.; Jin, J.; Jiang, X.; Liu, Q.; Lyu, M.R.; King, I. BinaryBERT: Pushing the Limit of BERT Quantization. *arXiv* **2021**, arXiv:2012.15701.
101. Yin, Y.; Chen, C.; Shang, L.; Jiang, X.; Chen, X.; Liu, Q. AutoTinyBERT: Automatic Hyper-parameter Optimization for Efficient Pre-trained Language Models. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), Online, 2–5 August 2021; pp. 5146–5157.
102. Keskar, N.S.; McCann, B.; Varshney, L.R.; Xiong, C.; Socher, R. Ctrl: A conditional transformer language model for controllable generation. *arXiv* **2019**, arXiv:1909.05858.
103. Shoeybi, M.; Patwary, M.; Puri, R.; LeGresley, P.; Casper, J.; Catanzaro, B. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv* **2019**, arXiv:1909.08053.
104. Ahmad, W.U.; Chakraborty, S.; Ray, B.; Chang, K.W. Unified pre-training for program understanding and generation. *arXiv* **2021**, arXiv:2103.06333.
105. Abdelfattah, A.; Tomov, S.; Dongarra, J. Investigating the benefit of FP16-enabled mixed-precision solvers for symmetric positive definite matrices using GPUs. In *Computational Science—ICCS 2020. ICCS 2020*; Lecture Notes in Computer Science; Springer Nature: Cham, Switzerland 2020; Volume 12138, pp. 237–250.. [CrossRef]
106. Zhang, J.; Zhao, Y.; Saleh, M.; Liu, P. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In Proceedings of the 37th International Conference on Machine Learning, Online, 13–18 July 2020; Article No.: 1051, pp. 11328–11339
107. Bi, B.; Li, C.; Wu, C.; Yan, M.; Wang, W.; Huang, S.; Huang, F.; Si, L. Palm: Pre-training an autoencoding & autoregressive language model for context-conditioned generation. *arXiv* **2020**, arXiv:2004.07159.
108. Gaschi, F.; Plesse, F.; Rastin, P.; Toussaint, Y. Multilingual Transformer Encoders: A Word-Level Task-Agnostic Evaluation. In Proceedings of the WCCI2022—IEEE World Congress on Computational Intelligence, Padoue, Italy, 18–23 July 2022; pp. 1–8.
109. Chi, Z.; Dong, L.; Ma, S.; Mao, S.H.X.L.; Huang, H.; Wei, F. mt6: Multilingual pretrained text-to-text transformer with translation pairs. *arXiv* **2021**, arXiv:2104.08692.
110. Conneau, A.; Khandelwal, K.; Goyal, N.; Chaudhary, V.; Wenzek, G.; Guzman, F.; Grave, E.; Ott, M.; Zettlemoyer, L.; Stoyanov, V. Unsupervised cross-lingual representation learning at scale. *arXiv* **2019**, arXiv:1911.02116.
111. Patel, J.M. Introduction to common crawl datasets. In *Getting Structured Data from the Internet*; Apress: Berkeley, CA, USA, 2020; pp. 277–324. [CrossRef]
112. Chi, Z.; Huang, S.; Dong, L.; Ma, S.; Singhal, S.; Bajaj, P.; Song, X.; Wei, F. XLM-E: Cross-lingual language model pre-training via ELECTRA. *arXiv* **2021**, arXiv:2106.16138.
113. Jiang, X.; Liang, Y.; Chen, W.; Duan, N. XLM-K: Improving Cross-Lingual Language Model Pre-Training with Multilingual Knowledge. *arXiv* **2021**, arXiv:2109.12573.
114. Barbieri, F.; Anke, L.E.; Camacho-Collados, J. Xlm-t: A multilingual language model toolkit for twitter. *arXiv* **2021**, arXiv:2104.12250.
115. Barbieri, F.; Espinosa-Anke, L.; Camacho-Collados, J. XLM-T: Multilingual Language Models in Twitter for Sentiment Analysis and Beyond. In Proceedings of the Language Resources and Evaluation Conference (LREC), Marseille, France, 20–25 June 2022; pp. 20–25.
116. Goyal, N.; Du, J.; Ott, M.; Anantharaman, G.; Conneau, A. Larger-scale transformers for multilingual masked language modeling. *arXiv* **2021**, arXiv:2105.00572.

117. Khanuja, S.; Bansal, D.; Mehtani, S.; Khosla, S.; Dey, A.; Gopalan, B.; Margam, D.K.; Aggarwal, P.; Nagipogu, R.T.; Dave, S.; et al. Muril: Multilingual representations for indian languages. *arXiv* **2021**, arXiv:2103.10730.
118. Huang, H.; Liang, Y.; Duan, N.; Gong, M.; Shou, L.; Jiang, D.; Zhou, M. Unicoder: A universal language encoder by pre-training with multiple cross-lingual tasks. *arXiv* **2019**, arXiv:1909.00964.
119. Koto, F.; Rahimi, A.; Lau, J.H.; Baldwin, T. IndoLEM and IndoBERT: A benchmark dataset and pre-trained language model for Indonesian NLP. *arXiv* **2020**, arXiv:2011.00677.
120. Le, H.; Vial, L.; Frej, J.; Segonne, V.; Coavoux, M.; Lecouteux, B.; Allauzen, A.; Crabbe, B.; Besacier, L.; Schwab, D. Flaubert: Unsupervised language model pre-training for french. *arXiv* **2019**, arXiv:1912.05372.
121. Rybak, P.; Mroczkowski, R.; Tracz, J.; Gawlik, I. KLEJ: Comprehensive benchmark for polish language understanding. *arXiv* **2020**, arXiv:2005.00630.
122. Park, S.; Moon, J.; Kim, S.; Cho, W.I.; Han, J.; Park, J.; Song, C.; Kim, J.; Song, Y.; Oh, T.; et al. Klue: Korean language understanding evaluation. *arXiv* **2021**, arXiv:2105.09680.
123. Antoun, W.; Baly, F.; Hajj, H. Arabert: Transformer-based model for arabic language understanding. *arXiv* **2020**, arXiv:2003.00104.
124. Nguyen, D.Q.; Nguyen, A.T. PhoBERT: Pre-trained language models for Vietnamese. *arXiv* **2020**, arXiv:2003.00744.
125. Martin, L.; Muller, B.; Suarez, P.J.O.; Dupont, Y.; Romary, L.; de La Clergerie, E.V.; Seddah, D.; Sagot, B. CamemBERT: A tasty French language model. *arXiv* **2019**, arXiv:1911.03894.
126. Malmsten, M.; Borjeson, L.; Haffenden, C. Playing with Words at the National Library of Sweden–Making a Swedish BERT. *arXiv* **2020**, arXiv:2007.01658.
127. Dadas, S.; Perelkiewicz, M.; Poswiata, R. Pre-training polish transformer-based language models at scale. In Proceedings of the Artificial Intelligence and Soft Computing: 19th International Conference, ICAISC 2020, Zakopane, Poland, 12–14 October 2020; Proceedings Part II, pp. 301–314.. [CrossRef]
128. de Vries, W.; van Cranenburgh, A.; Bisazza, A.; Caselli, T.; van Noord, G.; Nissim, M. Bertje: A dutch bert model. *arXiv* **2019**, arXiv:1912.09582.
129. Virtanen, A.; Kanerva, J.; Ilo, R.; Luoma, J.; Luotolahti, J.; Salakoski, T.; Ginter, F.; Pyysalo, S. Multilingual is not enough: BERT for Finnish. *arXiv* **2019**, arXiv:1912.07076.
130. Polignano, M.; Basile, P.; De Gemmis, M.; Semeraro, G.; Basile, V. Alberto: Italian BERT language understanding model for NLP challenging tasks based on tweets. In Proceedings of the 6th Italian Conference on Computational Linguistics, CLiC-it 2019, Bari, Italy, 13–15 November 2019; Volume 2481, pp. 1–6.
131. Souza, F.; Nogueira, R.; Lotufo, R. BERTimbau: Pretrained BERT models for Brazilian Portuguese. In *Intelligent Systems. BRACIS 2020*; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2020; Volume 12319, pp. 403–417._28. [CrossRef]
132. Kuratov, Y.; Arkhipov, M. Adaptation of deep bidirectional multilingual transformers for russian language. *arXiv* **2019**, arXiv:1905.07213.
133. Bhattacharjee, A.; Hasan, T.; Samin, K.; Rahman, M.S.; Iqbal, A.; Shahriyar, R. Banglabert: Combating embedding barrier for low-resource language understanding. *arXiv* **2021**, arXiv:2101.00204.
134. Abdul-Mageed, M.; Elmadany, A.; Nagoudi, E.M.B. ARBERT and MARBERT: Deep bidirectional transformers for Arabic. *arXiv* **2020**, arXiv:2101.01785.
135. Farahani, M.; Gharachorloo, M.; Farahani, M.; Manthouri, M. Parsbert: Transformer-based model for persian language understanding. *Neural Process. Lett.* **2021**, *53*, 3831–3847. [CrossRef]
136. Antoun, W.; Baly, F.; Hajj, H. Aragpt2: Pre-trained transformer for arabic language generation. *arXiv* **2020**, arXiv:2012.15520.
137. Roy, A.; Sharma, I.; Sarkar, S.; Goyal, P. Meta-ED: Cross-lingual Event Detection using Meta-learning for Indian Languages. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.* **2022**. [CrossRef]
138. Lowphansirikul, L.; Polpanumas, C.; Jantrakulchai, N.; Nutanong, S. Wangchanberta: Pretraining transformer-based thai language models. *arXiv* **2021**, arXiv:2101.09635.
139. Carmo, D.; Piau, M.; Campiotti, I.; Nogueira, R.; Lotufo, R. PTT5: Pretraining and validating the T5 model on Brazilian Portuguese data. *arXiv* **2020**, arXiv:2008.09144.
140. Wagner, J.; Wilkens, R.; Idiart, M.A.P.; Villavicencio, A. The brWaC Corpus: A New Open Resource for Brazilian Portuguese. In Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018), Miyazaki, Japan, 7–12 May 2018.
141. Antoun, W.; Baly, F.; Hajj, H. Araelectra: Pre-training text discriminators for arabic language understanding. *arXiv* **2020**, arXiv:2012.15516.
142. Cahyawijaya, S.; Winata, G.I.; Wilie, B.; Vincentio, K.; Li, X.; Kuncoro, A.; Ruder, S.; Lim, Z.Y.; Bahar, S.; Khodra, M.L.; et al. Indonlg: Benchmark and resources for evaluating indonesian natural language generation. *arXiv* **2021**, arXiv:2104.08200.
143. Lee, H.; Yoon, J.; Hwang, B.; Joe, S.; Min, S.; Gwon, Y. Korealbert: Pretraining a lite bert model for korean language understanding. In Proceedings of the IEEE 2020 25th International Conference on Pattern Recognition (ICPR), Milan, Italy, 10–15 January 2021; pp. 5551–5557.
144. Straka, M.; Naplava, J.; Strakova, J.; Samuel, D. RobeCzech: Czech RoBERTa, a monolingual contextualized language representation model. In Proceedings of the 24th International Conference on Text, Speech, and Dialogue (TSD 2021), Olomouc, Czech Republic, 6–9 September 2021; Springer: Cham, Switzerland, 2021; pp. 197–209.

145. Canete, J.; Chaperon, G.; Fuentes, R.; Ho, J.H.; Kang, H.; Perez, J. Spanish pre-trained bert model and evaluation data. In Proceedings of the Practical Machine Learning for Developing Countries Workshop (PML4DC) at the Eleventh International Conference on Learning Representations (ICLR), Addis Ababa, Ethiopia, 26 April 2020.
146. Zampieri, M.; Malmasi, S.; Nakov, P.; Rosenthal, S.; Farra, N.; Kumar, R. SemEval-2019 Task 6: Identifying and Categorizing Offensive Language in Social Media (OffensEval). In Proceedings of the 13th International Workshop on Semantic Evaluation, Minneapolis, MN, USA, 6–7 June 2019; pp. 75–86.
147. Caselli, T.; Basile, V.; Mitrovic, J.; Kartoziya, I.; Granitzer, M. I Feel Offended, Don't Be Abusive! Implicit/Explicit Messages in Offensive and Abusive Language. In Proceedings of the Twelfth Language Resources and Evaluation Conference (LREC), Marseille, France, 11–16 May 2020; pp. 6193–6202.
148. Basile, V.; Bosco, C.; Fersini, E.; Nozza, D.; Patti, V.; Pardo, F.M.R.; Rosso, P.; Sanguinetti, M. SemEval-2019 Task 5: Multilingual Detection of Hate Speech Against Immigrants and Women in Twitter. In Proceedings of the 13th International Workshop on Semantic Evaluation, Minneapolis, MN, USA, 6–7 June 2019; pp. 54–63.
149. Nguyen, D.Q.; Vu, T.; Nguyen, A.T. BERTweet: A pre-trained language model for English Tweets. *arXiv* **2020**, arXiv:2005.10200.
150. Rahali, A.; Akhloufi, M.A.; Therien-Daniel, A.M.; Brassard-Gourdeau, E. Automatic Misogyny Detection in Social Media Platforms using Attention-based Bidirectional-LSTM. In Proceedings of the 2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Melbourne, Australia, 17–20 October 2021; pp. 2706–2711.
151. Sawhney, R.; Neerkaje, A.T.; Gaur, M. A Risk-Averse Mechanism for Suicidality Assessment on Social Media. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), Dublin, Ireland, 22–27 May 2022; pp. 628–635.
152. Ta, H.T.; Rahman, A.B.S.; Najjar, L.; Gelbukh, A.F. Multi-Task Learning for Detection of Aggressive and Violent Incidents from Social Media. In Proceedings of the 2022 Iberian Languages Evaluation Forum, IberLEF 2022, A Coruna, Spain, 20 September 2022.
153. Sakhrani, H.; Parekh, S.; Ratadiya, P. Contextualized Embedding based Approaches for Social Media-specific Sentiment Analysis. In Proceedings of the 2021 International Conference on Data Mining Workshops (ICDMW), Auckland, New Zealand, 7–10 December 2021; pp. 186–193.
154. Ahmed, T.; Kabir, M.; Ivan, S.; Mahmud, H.; Hasan, K. Am I Being Bullied on Social Media? An Ensemble Approach to Categorize Cyberbullying. In Proceedings of the 2021 IEEE International Conference on Big Data (Big Data), Online, 15–18 December 2021; pp. 2442–2453.
155. Perez, J.M.; Furman, D.A.; Alemany, L.A.; Luque, F.M. RoBERTuito: A pre-trained language model for social media text in Spanish. *arXiv* **2021**, arXiv:2111.09453.
156. Wang, C.; Gou, J.; Fan, Z. News Recommendation Based On Multi-Feature Sequence Transformer. In Proceedings of the 2021 11th International Conference on Information Technology in Medicine and Education (ITME), Wuyishan, China, 19–21 November 2021; pp. 132–139.
157. Aljohani, A.A.; Rakrouki, M.A.; Alharbe, N.; Alluhaibi, R. A Self-Attention Mask Learning-Based Recommendation System. *IEEE Access* **2022**, *10*, 93017–93028. [CrossRef]
158. Bhumika; Das, D. MARRS: A Framework for multi-objective risk-aware route recommendation using Multitask-Transformer. In Proceedings of the 16th ACM Conference on Recommender Systems, Seattle, WA, USA, 18–23 September 2022.
159. Ghorbanpour, F.; Ramezani, M.; Fazli, M.A.; Rabiee, H.R. FNR: A Similarity and Transformer-Based Approach to Detect Multi-Modal Fake News in Social Media. *arXiv* **2021**, arXiv:2112.01131.
160. Chen, B.; Chen, B.; Gao, D.; Chen, Q.; Huo, C.; Meng, X.; Ren, W.; Zhou, Y. Transformer-based Language Model Fine-tuning Methods for COVID-19 Fake News Detection. *arXiv* **2021**, arXiv:2101.05509.
161. Mehta, D.; Dwivedi, A.; Patra, A.; Kumar, M.A. A transformer-based architecture for fake news classification. *Soc. Netw. Anal. Min.* **2021**, *11*, 39. [CrossRef]
162. Hande, A.; Puranik, K.; Priyadharshini, R.; Thavareesan, S.; Chakravarthi, B.R. Evaluating Pretrained Transformer-based Models for COVID-19 Fake News Detection. In Proceedings of the 2021 5th International Conference on Computing Methodologies and Communication (ICCMC), Erode, India, 8–10 April 2021; pp. 766–772.
163. Lu, S.; Guo, D.; Ren, S.; Huang, J.; Svyatkovskiy, A.; Blanco, A.; Clement, C.; Drain, D.; Jiang, D.; Tang, D.; et al. Codexglue: A machine learning benchmark dataset for code understanding and generation. *arXiv* **2021**, arXiv:2102.04664.
164. Guo, D.; Ren, S.; Lu, S.; Feng, Z.; Tang, D.; Liu, S.; Zhou, L.; Duan, N.; Svyatkovskiy, A.; Fu, S.; et al. Graphcodebert: Pre-training code representations with data flow. *arXiv* **2020**, arXiv:2009.08366.
165. Phan, L.; Tran, H.; Le, D.; Nguyen, H.; Anibal, J.; Peltekian, A.; Ye, Y. Cotext: Multi-task learning with code-text transformer. *arXiv* **2021**, arXiv:2105.08645.
166. Feng, Z.; Guo, D.; Tang, D.; Duan, N.; Feng, X.; Gong, M.; Shou, L.; Qin, B.; Liu, T.; Jiang, D.; et al. Codebert: A pre-trained model for programming and natural languages. *arXiv* **2020**, arXiv:2002.08155.
167. Sherstinsky, A. Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network. *arXiv* **2018**, arXiv:1808.03314.
168. O'Shea, K.; Nash, R. An Introduction to Convolutional Neural Networks. *arXiv* **2015**, arXiv:1511.08458.
169. Kalyan, K.S.; Rajasekharan, A.; Sangeetha, S. AMMU—A Survey of Transformer-based Biomedical Pretrained Language Models. *J. Biomed. Inform.* **2022**, *126*, 103982. [CrossRef]

170. Journal, I. Transformer Health Monitoring System Using Internet of Things. In Proceedings of the 2018 2nd IEEE International Conference on Power Electronics, Intelligent Control and Energy Systems (ICPEICES), Delhi, India, 22–24 October 2018. [CrossRef]
171. Roitero, K.; Bozzato, C.; Mea, V.D.; Mizzaro, S.; Serra, G. Twitter goes to the Doctor: Detecting Medical Tweets using Machine Learning and BERT. In Proceedings of the Workshop on Semantic Indexing and Information Retrieval for Health from Heterogeneous Content Types and Languages Co-Located with 42nd European Conference on Information Retrieval, SIIRH@ECIR 2020, Lisbon, Portugal, 14 April 2020.
172. Li, Y.; Rao, S.; Solares, J.R.A.; Hassaine, A.; Ramakrishnan, R.; Canoy, D.; Zhu, Y.; Rahimi, K.; Salimi-Khorshidi, G. BEHRT: Transformer for Electronic Health Records. *Sci. Rep.* **2020**, *10*, 7155. [CrossRef]
173. Li, Y.; Mamouei, M.; Salimi-Khorshidi, G.; Rao, S.; Hassaine, A.; Canoy, D.; Lukasiewicz, T.; Rahimi, K. Hi-BEHRT: Hierarchical Transformer-based model for accurate prediction of clinical events using multimodal longitudinal electronic health records. *arXiv* **2021**, arXiv:2106.11360.
174. Taghizadeh, N.; Doostmohammadi, E.; Seifossadat, E.; Rabiee, H.R.; Tahaei, M.S. SINA-BERT: A pre-trained Language Model for Analysis of Medical Texts in Persian. *arXiv* **2021**, arXiv:2104.07613.
175. Balouchzahi, F.; Sidorov, G.; Shashirekha, H.L. ADOP FERT-Automatic Detection of Occupations and Profession in Medical Texts using Flair and BERT. In Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2021) Co-Located with the Conference of the Spanish Society for Natural Language Processing (SEPLN 2021), IberLEF@SEPLN 2021, Malaga, Spain, 21 September 2021.
176. Kim, Y.; Kim, J.H.; Lee, J.M.; Jang, M.J.; Yum, Y.J.; Kim, S.; Shin, U.; Kim, Y.M.; Joo, H.J.; Song, S. A pre-trained BERT for Korean medical natural language processing. *Sci. Rep.* **2022**, *12*, 13847. [CrossRef] [PubMed]
177. Wada, S.; Takeda, T.; Manabe, S.; Konishi, S.; Kamohara, J.; Matsumura, Y. A pre-training technique to localize medical BERT and enhance BioBERT. *arXiv* **2020**, arXiv:2005.07202.
178. Mutinda, F.W.; Nigo, S.; Wakamiya, S.; Aramaki, E. Detecting Redundancy in Electronic Medical Records Using Clinical BERT. In Proceedings of the 26th Annual Conference of the Association for Natural Language Processing (NLP2020), Online, 16–19 March 2020. Available online: https://www.anlp.jp/proceedings/annual_meeting/2020/pdf_dir/E3-3.pdf (accessed on 2 October 2022).
179. Davari, M.; Kosseim, L.; Bui, T.D. TIMBERT: Toponym Identifier For The Medical Domain Based on BERT. In Proceedings of the 28th International Conference on Computational Linguistics, Barcelona, Spain, 8–13 December 2020.
180. Wu, Z.L.; Ge, S.; Wu, X. A BERT-Based Framework for Chinese Medical Entity Type Inference. 2020. Available online: https://bj.bcebos.com/v1/conference/ccks2020/eval_paper/ccks2020_eval_paper_1_1_3.pdf (accessed on 2 October 2022).
181. Guo, Y.; Ge, Y.; Al-Garadi, M.A.; Sarker, A. Pre-trained Transformer-based Classification and Span Detection Models for Social Media Health Applications. In Proceedings of the Sixth Social Media Mining for Health (SMM4H) Workshop and Shared Task, Mexico City, Mexico, 10 June 2021; pp. 52–57.
182. Çelikten, A.; Bulut, H. Turkish Medical Text Classification Using BERT. In Proceedings of the 2021 29th Signal Processing and Communications Applications Conference (SIU), Istanbul, Turkey, 9–11 June 2021; pp. 1–4.
183. Wang, X.; Tao, M.; Wang, R.; Zhang, L. Reduce the medical burden: An automatic medical triage system using text classification BERT based on Transformer structure. In Proceedings of the 2021 2nd International Conference on Big Data and Artificial Intelligence and Software Engineering (ICBASE), Zhuhai, China, 24–26 September 2021; pp. 679–685.
184. Aji, A.F.; Nityasya, M.N.; Wibowo, H.A.; Prasojo, R.E.; Fatyanosa, T.N. BERT Goes Brrr: A Venture Towards the Lesser Error in Classifying Medical Self-Reporters on Twitter. In Proceedings of the Sixth Social Media Mining for Health (SMM4H) Workshop and Shared Task, Mexico City, Mexico, 10 June 2021; pp. 58–64.
185. Lahlou, C.; Crayton, A.; Trier, C.; Willett, E.J. Explainable Health Risk Predictor with Transformer-based Medicare Claim Encoder. *arXiv* **2021**, arXiv:2105.09428.
186. Qin, Q.; Zhao, S.; Liu, C. A BERT-BiGRU-CRF Model for Entity Recognition of Chinese Electronic Medical Records. *Complex.* **2021**, *2021*, 6631837:1–6631837:11. [CrossRef]
187. Li, Z.; Yun, H.; Guo, Z.; Qi, J. Medical Named Entity Recognition Based on Multi Feature Fusion of BERT. In Proceedings of the 2021 4th International Conference on Big Data Technologies, Zibo China, 24–26 September 2021.
188. Xue, K.; Zhou, Y.; Ma, Z.; Ruan, T.; Zhang, H.; He, P. Fine-tuning BERT for Joint Entity and Relation Extraction in Chinese Medical Text. In Proceedings of the 2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), San Diago, CA, USA, 18–21 November 2019; pp. 892–897.
189. He, Y.; Zhu, Z.; Zhang, Y.; Chen, Q.; Caverlee, J. Infusing Disease Knowledge into BERT for Health Question Answering, Medical Inference and Disease Name Recognition. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Online, 16–20 November 2020.
190. Kieuvongngam, V.; Tan, B.; Niu, Y. Automatic Text Summarization of COVID-19 Medical Research Articles using BERT and GPT-2. *arXiv* **2020**, arXiv:2006.01997.
191. Heo, T.S.; Yoo, Y.; Park, Y.; Jo, B.C. Medical Code Prediction from Discharge Summary: Document to Sequence BERT using Sequence Attention. In Proceedings of the 2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA), Pasadena, CA, USA, 13–16 December 2021; pp. 1239–1244.
192. Wang, J.; Zhang, G.; Wang, W.; Zhang, K.; Sheng, Y. Cloud-based intelligent self-diagnosis and department recommendation service using Chinese medical BERT. *J. Cloud Comput.* **2021**, *10*, 4. [CrossRef]
193. Roy, A.; Pan, S. Incorporating medical knowledge in BERT for clinical relation extraction. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, Punta Cana, Dominican Republic, 7–11 November 2021.

194. Adrian Schiegl, D.T. Disease-Symptom Relation Extraction from Medical Text Corpora with BERT. Available online: https://web.archive.org/web/20210629045352/https://repositum.tuwien.at/bitstream/20.500.12708/17874/1/Schiegl%20Adrian%20-%202021%20-%20Disease-Symptom%20relation%20extraction%20from%20medical%20text...pdf (accessed on 2 October 2022).
195. Gao, S.; Du, J.; Zhang, X. Research on Relation Extraction Method of Chinese Electronic Medical Records Based on BERT. In Proceedings of the 2020 6th International Conference on Computing and Artificial Intelligence, Tianjin, China, 23–26 April 2020.
196. Peng, S.; Yuan, K.; Gao, L.; Tang, Z. Mathbert: A pre-trained model for mathematical formula understanding. *arXiv* **2021**, arXiv:2105.00377.
197. Liu, X.; Yin, D.; Zhang, X.; Su, K.; Wu, K.; Yang, H.; Tang, J. Oag-bert: Pre-train heterogeneous entity-augmented academic language models. *arXiv* **2021**, arXiv:2103.02410.
198. Liu, H.; Qiu, Q.; Wu, L.; Li, W.; Wang, B.; Zhou, Y. Few-shot learning for name entity recognition in geological text based on GeoBERT. *Earth Sci. Inform.* **2022**, *15*, 979–991. [CrossRef]
199. Xu, Z.; Li, J.; Yang, Z.; Li, S.; Li, H. SwinOCSR: End-to-end optical chemical structure recognition using a Swin Transformer. *J. Cheminform.* **2022**, *14*, 41. [CrossRef] [PubMed]
200. Quatra, M.L.; Cagliero, L. Transformer-based highlights extraction from scientific papers. *Knowl. Based Syst.* **2022**, *252*, 109382. [CrossRef]
201. Glazkova, A.; Glazkov, M. Detecting Generated Scientific Papers using an Ensemble of Transformer Models. *arXiv* **2022**, arXiv:2209.08283.
202. Balabin, H.; Hoyt, C.T.; Birkenbihl, C.; Gyori, B.M.; Bachman, J.A.; Kodamullil, A.T.; Ploger, P.G.; Hofmann-Apitius, M.; Domingo-Fernandez, D. STonKGs: A sophisticated transformer trained on biomedical text and knowledge graphs. *Bioinformatics* **2021**, *38*, 1648–1656. [CrossRef] [PubMed]
203. Phan, L.; Anibal, J.T.; Tran, H.; Chanana, S.; Bahadroglu, E.; Peltekian, A.; Altan-Bonnet, G. SciFive: A text-to-text transformer model for biomedical literature. *arXiv* **2021**, arXiv:2106.03598.
204. Parrilla-Gutierrez, J.M. Predicting Real-time Scientific Experiments Using Transformer models and Reinforcement Learning. In Proceedings of the 2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA), Pasadena, CA, USA, 13–16 December 2021; pp. 502–506.
205. Ghosh, S.; Chopra, A. Using Transformer based Ensemble Learning to classify Scientific Articles. In *Trends and Applications in Knowledge Discovery and Data Mining. PAKDD 2021*; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2021; Volume 12705. [CrossRef]
206. Zaratiana, U.; Holat, P.; Tomeh, N.; Charnois, T. Hierarchical Transformer Model for Scientific Named Entity Recognition. *arXiv* **2022**, arXiv:2203.14710.
207. Santosh, T.Y.S.; Chakraborty, P.; Dutta, S.; Sanyal, D.K.; Das, P.P. Joint Entity and Relation Extraction from Scientific Documents: Role of Linguistic Information and Entity Types. In Proceedings of the 2nd Workshop on Extraction and Evaluation of Knowledge Entities from Scientific Documents (JCDL 2021), Online, IL, USA, 30 September 2021.
208. Kubal, D.R.; Nagvenkar, A. Effective Ensembling of Transformer based Language Models for Acronyms Identification. In Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI-21), Online, 9 February 2021.
209. Tian, X.; Wang, J. Retrieval of Scientific Documents Based on HFS and BERT. *IEEE Access* **2021**, *9*, 8708–8717. [CrossRef]
210. Grail, Q. Globalizing BERT-based Transformer Architectures for Long Document Summarization. In Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, Online, 19–23 April 2021; pp. 1792–1810.
211. Leivaditi, S.; Rossi, J.; Kanoulas, E. A benchmark for lease contract review. *arXiv* **2020**, arXiv:2010.10386.
212. Chalkidis, I.; Fergadiotis, M.; Malakasiotis, P.; Aletras, N.; Androutsopoulos, I. LEGAL-BERT: The muppets straight out of law school. *arXiv* **2020**, arXiv:2010.02559.
213. Paul, S.; Mandal, A.; Goyal, P.; Ghosh, S. Pre-training Transformers on Indian Legal Text. *arXiv* **2022**, arXiv:2209.06049.
214. Thanh, N.H.; Nguyen, L.M. Logical Structure-based Pretrained Models for Legal Text Processing. Available online: https://www.scitepress.org/Papers/2022/108520/108520.pdf (accessed on 2 October 2022).
215. Savelka, J.; Ashley, K.D. Discovering Explanatory Sentences in Legal Case Decisions Using Pre-trained Language Models. *arXiv* **2021**, arXiv:2112.07165.
216. Shaheen, Z.; Wohlgenannt, G.; Muromtsev, D. Zero-Shot Cross-Lingual Transfer in Legal Domain Using Transformer Models. In Proceedings of the 2021 International Conference on Computational Science and Computational Intelligence (CSCI), Las Vegas, NV, USA, 26–29 July 2021; pp. 450–456.
217. Garneau, N.; Gaumond, E.; Lamontagne, L.; Deziel, P.L. CriminelBART: A French Canadian legal language model specialized in criminal law. In Proceedings of the Eighteenth International Conference on Artificial Intelligence and Law, São Paulo, Brazil, 21–25 June 2021.
218. Peric, L.; Mijic, S.; Stammbach, D.; Ash, E. Legal Language Modeling with Transformers. In Proceedings of the Automated Semantic Analysis of Information in Legal Text at 33rd International Conference on Legal Knowledge and Information Systems (ASAIL@JURIX), Online Event, Brno, Czech Republic, 9–11 December 2020.
219. Cemri, M.; Çukur, T.; Koç, A. Unsupervised Simplification of Legal Texts. *arXiv* **2022**, arXiv:2209.00557.

220. Klaus, S.; Hecke, R.V.; Naini, K.D.; Altingovde, I.S.; Bernabe-Moreno, J.; Herrera-Viedma, E.E. Summarizing Legal Regulatory Documents using Transformers. In Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, 11–15 July 2022; pp. 2426–2430.
221. Yoon, J.; Junaid, M.; Ali, S.; Lee, J. Abstractive Summarization of Korean Legal Cases using Pre-trained Language Models. In Proceedings of the 2022 16th International Conference on Ubiquitous Information Management and Communication (IMCOM), Seoul, Korea, 3–5 January 2022; pp. 1–7.
222. Aumiller, D.; Almasian, S.; Lackner, S.; Gertz, M. Structural text segmentation of legal documents. In Proceedings of the Eighteenth International Conference on Artificial Intelligence and Law, São Paulo, Brazil, 21–25 June 2021; pp. 2–11.
223. Mullick, A.; Nandy, A.; Kapadnis, M.N.; Patnaik, S.; Raghav, R. Fine-grained Intent Classification in the Legal Domain. *arXiv* **2022**, arXiv:2205.03509.
224. Prasad, N.; Boughanem, M.; Dkaki, T. Effect of Hierarchical Domain-specific Language Models and Attention in the Classification of Decisions for Legal Cases. In Proceedings of the CIRCLE (Joint Conference of the Information Retrieval Communities in Europe), Samatan, Gers, France, 4–7 July 2022.
225. Nghiem, M.Q.; Baylis, P.; Freitas, A.; Ananiadou, S. Text Classification and Prediction in the Legal Domain. In Proceedings of the 13th Conference on Language Resources and Evaluation (LREC 2022), Marseille, France, 20–25 June 2022; pp. 4717–4722.
226. Braun, D.; Matthes, F. Clause Topic Classification in German and English Standard Form Contracts. In Proceedings of the Fifth Workshop on e-Commerce and NLP (ECNLP 5), Online, 26 May 2022.
227. Papaloukas, C.; Chalkidis, I.; Athinaios, K.; Pantazi, D.A.; Koubarakis, M. Multi-granular Legal Topic Classification on Greek Legislation. *arXiv* **2021**, arXiv:2109.15298.
228. Bambroo, P.; Awasthi, A. LegalDB: Long DistilBERT for Legal Document Classification. In Proceedings of the 2021 International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT), Bhilai, India, 19–20 February 2021; pp. 1–4.
229. Shaheen, Z.; Wohlgenannt, G.; Filtz, E. Large Scale Legal Text Classification Using Transformer Models. *arXiv* **2020**, arXiv:2010.12871.
230. Ni, Z. Key Information Extraction of Food Environmental Safety Criminal Judgment Documents Based on Deep Learning. *J. Environ. Public Health* **2022**, *2022*, 4661166. [CrossRef]
231. Kim, M.Y.; Rabelo, J.; Okeke, K.; Goebel, R. Legal Information Retrieval and Entailment Based on BM25, Transformer and Semantic Thesaurus Methods. *Rev. Socionetw. Strateg.* **2022**, *16*, 157–174. [CrossRef]
232. Trias, F.; Wang, H.; Jaume, S.; Idreos, S. Named Entity Recognition in Historic Legal Text: A Transformer and State Machine Ensemble Method. In Proceedings of the Natural Legal Language Processing Workshop 2021, Punta Cana, Dominican Republic, 7–11 November 2021.
233. Thanh, N.H.; Nguyen, P.M.; Vuong, T.H.Y.; Bui, M.Q.; Nguyen, M.C.; Dang, B.; Tran, V.D.; Nguyen, L.M.; Satoh, K. Transformer-Based Approaches for Legal Text Processing. *Rev. Socionetw. Strateg.* **2022**, *16*, 135–155. [CrossRef]
234. Sun, M.; Guo, Z.; Deng, X. Intelligent BERT-BiLSTM-CRF Based Legal Case Entity Recognition Method. In Proceedings of the ACM Turing Award Celebration Conference; China (ACM TURC 2021), Hefei, China, 30 July–1 August 2021; pp. 186–191. [CrossRef]
235. Caballero, E.Q.; Rahman, M.S.; Cerny, T.; Rivas, P.; Bejarano, G. Study of Question Answering on Legal Software Document using BERT based models. In Proceedings of the LatinX in Natural Language Processing Research Workshop, Seattle, WA, USA, 10 July 2022.
236. Khazaeli, S.; Punuru, J.; Morris, C.; Sharma, S.; Staub, B.; Cole, M.; Chiu-Webster, S.; Sakalley, D. A Free Format Legal Question Answering System. In Proceedings of the Natural Legal Language Processing Workshop 2021, Punta Cana, Dominican Republic, 7–11 November 2021.
237. Vold, A.; Conrad, J.G. Using transformers to improve answer retrieval for legal questions. In Proceedings of the Eighteenth International Conference on Artificial Intelligence and Law, Online, 21–25 June 2021.
238. Huang, Y.; Shen, X.; Li, C.; Ge, J.; Luo, B. Dependency Learning for Legal Judgment Prediction with a Unified Text-to-Text Transformer. *arXiv* **2021**, arXiv:2112.06370.
239. Dong, Q.; Niu, S. Legal Judgment Prediction via Relational Learning. In Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, 11–15 July 2021.
240. Sukanya, G.; Priyadarshini, J. A Meta Analysis of Attention Models on Legal Judgment Prediction System. *Int. J. Adv. Comput. Sci. Appl. (IJACSA)* **2021**, *12*, 531–538. [CrossRef]
241. Masala, M.; Iacob, R.C.A.; Uban, A.S.; Cidotã, M.A.; Velicu, H.; Rebedea, T.; Popescu, M.C. jurBERT: A Romanian BERT Model for Legal Judgement Prediction. In Proceedings of the Natural Legal Language Processing Workshop 2021, Punta Cana, Dominican Republic, 7–11 November 2021.
242. Salaun, O.; Langlais, P.; Benyekhlef, K. Exploiting Domain-Specific Knowledge for Judgment Prediction Is No Panacea. In Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021), Online, 1–3 September 2021.
243. Zhu, K.; Guo, R.; Hu, W.; Li, Z.; Li, Y. Legal Judgment Prediction Based on Multiclass Information Fusion. *Complexity* **2020**, *2020*, 3089189:1–3089189:12. [CrossRef]

244. Lian, M.; Li, J. Financial product recommendation system based on transformer. In Proceedings of the 2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), Chongqing, China, 12–14 June 2020; Volume 1, pp. 2547–2551.
245. Goel, T.; Chauhan, V.; Verma, I.; Dasgupta, T.; Dey, L. TCS WITM 2021 @FinSim-2: Transformer based Models for Automatic Classification of Financial Terms. In Proceedings of the WWW '21: Companion Proceedings of the Web Conference 2021, Ljubljana, Slovenia, 19–23 April 2021; pp. 311–315 . [CrossRef]
246. Yang, L.; Li, J.; Dong, R.; Zhang, Y.; Smyth, B. NumHTML: Numeric-Oriented Hierarchical Transformer Model for Multi-task Financial Forecasting. In Proceedings of the 36th AAAI Conference on Artificial Intelligence, Online, 22 February–1 March 2022.
247. Ding, Q.; Wu, S.; Sun, H.; Guo, J.; Guo, J. Hierarchical Multi-Scale Gaussian Transformer for Stock Movement Prediction. In Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI-20), Special Track on AI in FinTech, Yokohama, Japan, 7–15 January 2021.
248. Yoo, J.; Soun, Y.; Park, Y.; Kang, U. Accurate Multivariate Stock Movement Prediction via Data-Axis Transformer with Multi-Level Contexts. In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, Virtual Event, Singapore, 14–18 August 2021.
249. Hu, J. Local-constraint transformer network for stock movement prediction. *Int. J. Comput. Sci. Eng.* **2021**, *24*, 429–437. [CrossRef]
250. Daiya, D.; Lin, C. Stock Movement Prediction and Portfolio Management via Multimodal Learning with Transformer. In Proceedings of the ICASSP 2021—2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Toronto, ON, Canada, 6–11 June 2021; pp. 3305–3309.
251. Caron, M.; Müller, O. Hardening Soft Information: A Transformer-Based Approach to Forecasting Stock Return Volatility. In Proceedings of the 2020 IEEE International Conference on Big Data (Big Data), Atlanta, GA, USA, 10–13 December 2020; pp. 4383–4391.
252. Chen, Q. Stock Movement Prediction with Financial News using Contextualized Embedding from BERT. *arXiv* **2021**, arXiv:2107.08721.
253. Kim, A.S.; Yoon, S. Corporate Bankruptcy Prediction with BERT Model. In Proceedings of the Third Workshop on Economics and Natural Language Processing, Punta Cana, Dominican Republic, 11 November 2021; pp. 26–36.
254. Wan, C.X.; Li, B. Financial causal sentence recognition based on BERT-CNN text classification. *J. Supercomput.* **2022**, *78*, 6503–6527. [CrossRef]
255. Arslan, Y.; Allix, K.; Veiber, L.; Lothritz, C.; Bissyande, T.F.; Klein, J.; Goujon, A. A Comparison of Pre-Trained Language Models for Multi-Class Text Classification in the Financial Domain. In Proceedings of the WWW'21: Companion Proceedings of the Web Conference 2021, Ljubljana, Slovenia, 19–23 April 2021; pp. 260–268;. [CrossRef]
256. Zhong, A.; Han, Q. Automated Investor Sentiment Classification using Financial Social Media. In Proceedings of the CONF-CDS 2021: The 2nd International Conference on Computing and Data Science, Stanford, CA, USA, 28–30 January 2021; pp. 356–361.
257. Chapman, C.; Hillebrand, L.P.; Stenzel, M.R.; Deusser, T.; Biesner, D.; Bauckhage, C.; Sifa, R. Towards Generating Financial Reports from Tabular Data Using Transformers. In *Machine Learning and Knowledge Extraction. CD-MAKE 2022*; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2022; Volume 13480._14. [CrossRef]
258. Agrawal, Y.; Anand, V.; Gupta, M.; Arunachalam, S.; Varma, V. Goal-Directed Extractive Summarization of Financial Reports. In Proceedings of the 30th ACM International Conference on Information & Knowledge Management, Gold Coast, QLD, Australia, 1–5 November 2021.
259. Singh, A.K. PoinT-5: Pointer Network and T-5 based Financial Narrative Summarisation. *arXiv* **2020**, arXiv:2010.04191.
260. Li, Q.; Zhang, Q. A Unified Model for Financial Event Classification, Detection and Summarization. In Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence Special Track on AI in FinTech, Yokohama, Japan, 11–17 July 2020; pp. 4668–4674. [CrossRef]
261. Kamal, S.; Sharma, S. A Comprehensive Review on Summarizing Financial News Using Deep Learning. *arXiv* **2021**, arXiv:2109.10118.
262. Zhao, L.; Li, L.; Zheng, X. A BERT based Sentiment Analysis and Key Entity Detection Approach for Online Financial Texts. In Proceedings of the 2021 IEEE 24th International Conference on Computer Supported Cooperative Work in Design (CSCWD), Dalian, China, 5–7 May 2021; pp. 1233–1238.
263. Hiew, J.Z.G.; Huang, X.; Mou, H.; Li, D.; Wu, Q.; Xu, Y. BERT-based Financial Sentiment Index and LSTM-based Stock Return Predictability. *arXiv* **2019**, arXiv:1906.09024.
264. Salunkhe, A.; Mhaske, S. Aspect Based Sentiment Analysis on Financial Data using Transferred Learning Approach using Pre-Trained BERT and Regressor Model. *Int. Res. J. Eng. Technol. (IRJET)* **2019**, *6*, 1097–1101.
265. Qian, T.; Xie, A.; Bruckmann, C. Sensitivity Analysis on Transferred Neural Architectures of BERT and GPT-2 for Financial Sentiment Analysis. *arXiv* **2022**, arXiv:2207.03037.
266. Ghosh, S.; Naskar, S.K. FiNCAT: Financial Numeral Claim Analysis Tool. In Proceedings of the Companion Proceedings of the Web Conference 2022, Virtual Event, Lyon, France, 25–29 April 2022; pp. 583–585. [CrossRef]
267. Soong, G.H.; Tan, C.C. Sentiment Analysis on 10-K Financial Reports using Machine Learning Approaches. In Proceedings of the 2021 IEEE 11th International Conference on System Engineering and Technology (ICSET), Shah Alam, Malaysia, 6 November 2021; pp. 124–129.
268. Guti'errez-Fandino, A.; i Alonso, M.N.; Kolm, P.N.; Armengol-Estap'e, J. FinEAS: Financial Embedding Analysis of Sentiment. *J. Financ. Data Sci.* **2022**. [CrossRef]

269. Mansar, Y.; Kang, J.; Maarouf, I.E. The FinSim-2 2021 Shared Task: Learning Semantic Similarities for the Financial Domain. In Proceedings of the Companion Proceedings of the Web Conference 2021, Ljubljana, Slovenia, 19–23 April 2021.
270. Li, M.; Chen, L.; Zhao, J.; Li, Q. Sentiment analysis of Chinese stock reviews based on BERT model. *Appl. Intell.* **2021**, *51*, 5016–5024. [CrossRef]
271. Li, M.; Chen, L.; Zhao, J.; Li, Q. A Chinese Stock Reviews Sentiment Analysis Based on BERT Model. 2020. Available online: https://www.researchsquare.com/article/rs-69958/latest (accessed on 2 October 2022).
272. Hillebrand, L.P.; Deusser, T.; Khameneh, T.D.; Kliem, B.; Loitz, R.; Bauckhage, C.; Sifa, R. KPI-BERT: A Joint Named Entity Recognition and Relation Extraction Model for Financial Reports. *arXiv* **2022**, arXiv:2208.02140.
273. Liao, L.; Yang, C. Enterprise risk information extraction based on BERT. In Proceedings of the 2022 7th International Conference on Intelligent Computing and Signal Processing (ICSP), Xi'an, China, 15–17 April 2022; pp. 1454–1457.
274. Cao, L.; Zhang, S.; Chen, J. CBCP: A Method of Causality Extraction from Unstructured Financial Text. In Proceedings of the 2021 5th International Conference on Natural Language Processing and Information Retrieval (NLPIR), Sanya, China, 17–20 December 2021.
275. Zhang, Y.; Zhang, H. FinBERT-MRC: Financial named entity recognition using BERT under the machine reading comprehension paradigm. *arXiv* **2022**, arXiv:2205.15485.
276. Loukas, L.; Fergadiotis, M.; Chalkidis, I.; Spyropoulou, E.; Malakasiotis, P.; Androutsopoulos, I.; Paliouras, G. FiNER: Financial Numeric Entity Recognition for XBRL Tagging. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Dublin, Ireland, 22–27 May 2022.
277. Reyes, D.; Barcelos, A.; Vieira, R.; Manssour, I.H. Related Named Entities Classification in the Economic-Financial Context. In Proceedings of the EACL Hackashop on News Media Content Analysis and Automated Report Generation, Online, 19 April 2021; pp. 8–15.
278. Liang, Y.C.; Chen, M.; Yeh, W.C.; Chang, Y.C. Numerical Relation Detection in Financial Tweets using Dependency-aware Deep Neural Network. In Proceedings of the 33rd Conference on Computational Linguistics and Speech Processing (ROCLING 2021), Taoyuan, Taiwan, 15–16 October 2021; pp. 218–225.
279. Sangaraju, V.R.; Bolla, B.K.; Nayak, D.; Kh, J. Topic Modelling on Consumer Financial Protection Bureau Data: An Approach Using BERT Based Embeddings. In Proceedings of the 2022 IEEE 7th International conference for Convergence in Technology (I2CT), Mumbai, India, 7–9 April 2022; pp. 1–6.
280. Wang, Z.; Liu, Z.; Luo, L.; Chen, X. A Multi-Neural Network Fusion Based Method for Financial Event Subject Extraction. In Proceedings of the 2020 3rd International Conference on Advanced Electronic Materials, Computers and Software Engineering (AEMCSE), Shenzhen, China, 24–26 April 2020; pp. 360–365.
281. Lin, H.; Wu, J.S.; Huang, Y.S.; Tsai, M.F.; Wang, C.J. NFinBERT: A Number-Aware Language Model for Financial Disclosures (short paper). In Proceedings of the Swiss Text Analytics Conference 2021, Online, Winterthur, Switzerland, 14–16 June 2021.
282. Liu, Z.; Huang, D.; Huang, K.; Li, Z.; Zhao, J. FinBERT: A Pre-trained Financial Language Representation Model for Financial Text Mining. In Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, Special Track on AI in FinTech, Yokohama, Japan, 7–15 January 2021; pp. 4513–4519.
283. Lu, Q.; Zhang, H.; Kinawi, H.; Niu, D. Self-Attentive Models for Real-Time Malware Classification. *IEEE Access* **2022**, *10*, 95970–95985. [CrossRef]
284. Ameri, K.; Hempel, M.; Sharif, H.R.; Lopez, J.; Perumalla, K.S. CyBERT: Cybersecurity Claim Classification by Fine-Tuning the BERT Language Model. *J. Cybersecur. Priv.* **2021**, *1*, 615–637. [CrossRef]
285. Ampel, B.; Samtani, S.; Ullman, S.; Chen, H. Linking Common Vulnerabilities and Exposures to the MITRE ATT&CK Framework: A Self-Distillation Approach. *arXiv* **2021**, arXiv:2108.01696.
286. Rahali, A.; Akhloufi, M.A. MalBERT: Malware Detection using Bidirectional Encoder Representations from Transformers. In Proceedings of the 2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Melbourne, Australia, 17–20 October 2021; pp. 3226–3231.
287. Kale, A.S.; Pandya, V.; Troia, F.D.; Stamp, M. Malware classification with Word2Vec, HMM2Vec, BERT, and ELMo. *J. Comput. Virol. Hacking Tech.* **2022**. [CrossRef]
288. Yesir, S.; Sogukpinar, I. Malware Detection and Classification Using fastText and BERT. In Proceedings of the 2021 9th International Symposium on Digital Forensics and Security (ISDFS), Elazig, Turkey, 28–29 June 2021; pp. 1–6.
289. Jahromi, M.Z.; Jahromi, A.A.; Kundur, D.; Sanner, S.; Kassouf, M. Data analytics for cybersecurity enhancement of transformer protection. *ACM Sigenergy Energy Inform. Rev.* **2021**, *1*, 12–19. [CrossRef]
290. Jahromi, M.Z.; Jahromi, A.A.; Sanner, S.; Kundur, D.; Kassouf, M. Cybersecurity Enhancement of Transformer Differential Protection Using Machine Learning. In Proceedings of the 2020 IEEE Power and Energy Society General Meeting (PESGM), Virtual Event, 3–6 August 2020; pp. 1–5.
291. Liu, Y.; Pan, S.; Wang, Y.G.; Xiong, F.; Wang, L.; Lee, V.C.S. Anomaly Detection in Dynamic Graphs via Transformer. *arXiv* **2021**, arXiv:2106.09876.
292. Lin, L.H.; Hsiao, S.W. Attack Tactic Identification by Transfer Learning of Language Model. *arXiv* **2022**, arXiv:2209.00263.
293. Ghourabi, A. A Security Model Based on LightGBM and Transformer to Protect Healthcare Systems From Cyberattacks. *IEEE Access* **2022**, *10*, 48890–48903. [CrossRef]

294. Hemalatha, J.; Roseline, S.A.; Geetha, S.; Kadry, S.N.; Damavsevicius, R. An Efficient DenseNet-Based Deep Learning Model for Malware Detection. *Entropy* **2021**, *23*, 344. [CrossRef] [PubMed]
295. Ranade, P.; Piplai, A.; Mittal, S.; Joshi, A.; Finin, T. Generating Fake Cyber Threat Intelligence Using Transformer-Based Models. In Proceedings of the 2021 International Joint Conference on Neural Networks (IJCNN), Shenzhen, China, 18–22 July 2021; pp. 1–9.
296. Alam, M.T.; Bhusal, D.; Park, Y.; Rastogi, N. CyNER: A Python Library for Cybersecurity Named Entity Recognition. *arXiv* **2022**, arXiv:2204.05754.
297. Evangelatos, P.; Iliou, C.; Mavropoulos, T.; Apostolou, K.; Tsikrika, T.; Vrochidis, S.; Kompatsiaris, Y. Named Entity Recognition in Cyber Threat Intelligence Using Transformer-based Models. In Proceedings of the 2021 IEEE International Conference on Cyber Security and Resilience (CSR), Rhodes, Greece, 26–28 July 2021; pp. 348–353.
298. Lai, G.; Xie, Q.; Liu, H.; Yang, Y.; Hovy, E. Race: Large-scale reading comprehension dataset from examinations. *arXiv* **2017**, arXiv:1704.04683.
299. Socher, R.; Perelygin, A.; Wu, J.; Chuang, J.; Manning, C.D.; Ng, A.; Potts, C. Recursive deep models for semantic compositionality over a sentiment treebank. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, Seattle, WA, USA, 18–21 October 2013; pp. 1631–1642.
300. Warstadt, A.; Singh, A.; Bowman, S.R. Neural Network Acceptability Judgments. *arXiv* **2018**, arXiv:1805.12471.
301. Dolan, W.B.; Brockett, C. Automatically Constructing a Corpus of Sentential Paraphrases. In Proceedings of the Third International Workshop on Paraphrasing (IWP2005), Jeju Island, Korea, 14 October 2005.
302. Iyer, S.; Dandekar, N.; Csernai, K. First Quora Dataset Release: Question Pairs. 18 May 2017. Available online: https://karthikrevanuru.github.io/assets/documents/projects/Quora_Pairs.pdf (accessed on 2 October 2022).
303. Williams, A.; Nangia, N.; Bowman, S.R. A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), New Orleans, LA, USA, 1–6 June 2018; pp. 1112–1122.
304. Bowman, S.R.; Angeli, G.; Potts, C.; Manning, C.D. A large annotated corpus for learning natural language inference. *arXiv* **2015**, arXiv:1508.05326.
305. Levesque, H.; Davis, E.; Morgenstern, L. The winograd schema challenge. In Proceedings of the Thirteenth International Conference on the Principles of Knowledge Representation and Reasoning, Rome, Italy, 10–14 June 2012.
306. Dagan, I.; Glickman, O.; Magnini, B. The PASCAL recognising textual entailment challenge. In *Machine Learning Challenges Workshop*; Springer: Berlin/Heidelberg, Germany, 2005; pp. 177–190. [CrossRef]
307. Cer, D.; Diab, M.; Agirre, E.; Lopez-Gazpio, I.; Specia, L. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv* **2017**, arXiv:1708.00055.
308. Wang, A.; Pruksachatkun, Y.; Nangia, N.; Singh, A.; Michael, J.; Hill, F.; Levy, O.; Bowman, S. Superglue: A stickier benchmark for general-purpose language understanding systems. In Proceedings of the NIPS'19: Proceedings of the 33rd International Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; Article No.: 294, pp. 3266–3280.
309. Diebold, F.X.; Mariano, R.S. Comparing Predictive Accuracy. *J. Bus. Econ. Stat.* **2020**, *20*, 134–144. [CrossRef]
310. Powers, D.M. Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation. *arXiv* **2020**, arXiv:2010.16061.
311. Woods, K.S.; Bowyer, K. Generating ROC curves for artificial neural networks. *IEEE Trans. Med. Imaging* **1997**, *16*, 329–337. [CrossRef]
312. Papineni, K.; Roukos, S.; Ward, T.; Zhu, W.J. Bleu: A Method for Automatic Evaluation of Machine Translation. In Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, Philadelphia, PA, USA, 7–12 July 2002; pp. 311–318. [CrossRef]
313. Lin, C.Y. ROUGE: A Package for Automatic Evaluation of Summaries. In Proceedings of the Text Summarization Branches Out, Barcelona, Spain, 25–26 July 2004; pp. 74–81.
314. Jelinek, F.; Mercer, R.L.; Bahl, L.R.; Baker, J. Perplexity—A measure of the difficulty of speech recognition tasks. *J. Acoust. Soc. Am.* **1977**, *62*. [CrossRef]
315. Li, J.; Galley, M.; Brockett, C.; Gao, J.; Dolan, W.B. A Diversity-Promoting Objective Function for Neural Conversation Models. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego, CA, USA, 12–17 June 2016.
316. Kusner, M.J.; Sun, Y.; Kolkin, N.I.; Weinberger, K.Q. From Word Embeddings To Document Distances. In Proceedings of the 32nd International Conference on International Conference on Machine Learning (ICML), Lille, France, 6–11 July 2015; Volume 37, pp. 957–966.
317. Lo, C. MEANT 2.0: Accurate semantic MT evaluation for any output language. In Proceedings of the Second Conference on Machine Translation, Copenhagen, Denmark, 7 September 2017; pp. 589–597.
318. Yujian, L.; Bo, L. A Normalized Levenshtein Distance Metric. *IEEE Trans. Pattern Anal. Mach. Intell.* **2007**, *29*, 1091–1095. [CrossRef] [PubMed]
319. Liu, C.; Dahlmeier, D.; Ng, H.T. TESLA: Translation Evaluation of Sentences with Linear-Programming-Based Analysis. In Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and Metrics (MATR), Uppsala, Sweden, 15–16 July 2010; pp. 354–359.

320. Agirre, E.; Gonzalez-Agirre, A.; Lopez-Gazpio, I.; Maritxalar, M.; Rigau, G.; Uria, L. SemEval-2016 Task 2: Interpretable Semantic Textual Similarity. In Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016), San Diego, CA, USA, 16–17 June 2016; pp. 512–524.
321. Huang, P.S.; He, X.; Gao, J.; Deng, L.; Acero, A.; Heck, L. Learning deep structured semantic models for web search using clickthrough data. In Proceedings of the 22nd ACM international conference on Information and Knowledge Management, San Francisco, CA, USA, 27 October–1 November 2013.
322. Agarwal, A.; Lavie, A. Meteor, M-BLEU and M-TER: Evaluation Metrics for High-Correlation with Human Rankings of Machine Translation Output. In *Natural Language Processing and Information Systems. NLDB 2009*; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2008; Volume 5723. ._20. [CrossRef]
323. Popovic, M. chrF: Character n-gram F-score for automatic MT evaluation. In Proceedings of the Tenth Workshop on Statistical Machine Translation, Lisbon, Portugal, 17–18 September 2015; pp. 392–395.
324. Lo, C. Extended Study on Using Pretrained Language Models and YiSi-1 for Machine Translation Evaluation. In Proceedings of the Fifth Conference on Machine Translation, Online, 19–20 November 2020; pp. 895–902.
325. Lo, C.; Larkin, S. Machine Translation Reference-less Evaluation using YiSi-2 with Bilingual Mappings of Massive Multilingual Language Model. In Proceedings of the Fifth Conference on Machine Translation, Online, 19–20 November 2020; pp. 903–910.
326. Chen, Q.; Zhu, X.D.; Ling, Z.; Wei, S.; Jiang, H.; Inkpen, D. Enhanced LSTM for Natural Language Inference. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Vancouver, BC, Canada, 30 July–4 August 2017; pp. 1657–1668.
327. Och, F.J. Minimum Error Rate Training in Statistical Machine Translation. In Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics, Sapporo, Japan, 7–12 July 2003; pp. 160–167.
328. Levenshtein, V.I. Binary codes capable of correcting deletions, insertions, and reversals. *Sov. Phys. Dokl.* **1965**, *10*, 707–710.
329. Doddington, G.R.; Przybocki, M.A.; Martin, A.F.; Reynolds, D.A. The NIST speaker recognition evaluation—Overview, methodology, systems, results, perspective. *Speech Commun.* **2000**, *31*, 225–254. [CrossRef]
330. Banerjee, S.; Lavie, A. METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. In Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization, Ann Arbor, MI, USA, 29 June 2005; pp. 65–72.
331. Mauser, A.; Hasan, S.; Ney, H. Automatic Evaluation Measures for Statistical Machine Translation System Optimization. In Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08), Marrakech, Morocco, 28–30 May 2008.
332. Mathur, N.; Baldwin, T.; Cohn, T. Tangled up in BLEU: Reevaluating the Evaluation of Automatic Machine Translation Evaluation Metrics. *arXiv* **2020**, arXiv:2006.06264.
333. Velankar, A.; Patil, H.; Joshi, R. Mono vs Multilingual BERT for Hate Speech Detection and Text Classification: A Case Study in Marathi. *arXiv* **2022**, arXiv:2204.08669.
334. Alammary, A.S. BERT Models for Arabic Text Classification: A Systematic Review. *Appl. Sci.* **2022**, *12*, 5720. [CrossRef]
335. Dai, X.; Chalkidis, I.; Darkner, S.; Elliott, D. Revisiting Transformer-based Models for Long Document Classification. *arXiv* **2022**, arXiv:2204.06683.
336. Hamid, A.; Ahmad, R.M.T.R.L.; Hassan, W.A.W.; Majudi, S.; Fahmy, S. Text Classification on Social Media using Bidirectional Encoder Representations from Transformers (BERT) for Zakat Sentiment Analysis. *Int. J. Synerg. Eng. Technol.* **2022**, *3*, 79–87.
337. Li, Z.; Si, S.; Wang, J.; Xiao, J. Federated Split BERT for Heterogeneous Text Classification. *arXiv* **2022**, arXiv:2205.13299.
338. Rahali, A.; Akhloufi, M.A. MalBERT: Using Transformers for Cybersecurity and Malicious Software Detection. *arXiv* **2021**, arXiv:2103.03806.
339. Tezgider, M.; Yildiz, B.; Aydin, G. Text classification using improved bidirectional transformer. *Concurr. Comput. Pract. Exp.* **2022**, *34*, e6486. [CrossRef]
340. Zhang, Y.; Sun, S.; Galley, M.; Chen, Y.C.; Brockett, C.; Gao, X.; Gao, J.; Liu, J.; Dolan, B. Dialogpt: Large-scale generative pre-training for conversational response generation. *arXiv* **2019**, arXiv:1911.00536.
341. Peng, B.; Li, C.; Li, J.; Shayandeh, S.; Liden, L.; Gao, J. Soloist: Few-shot task-oriented dialog with a single pre-trained auto-regressive model. *arXiv* **2020**, arXiv:2005.05298.
342. Lamsiyah, S.; Mahdaouy, A.E.; Ouatik, S.E.A.; Espinasse, B. Unsupervised extractive multi-document summarization method based on transfer learning from BERT multi-task fine-tuning. *J. Inf. Sci.* **2021**, 0165551521990616. [CrossRef]
343. Khandelwal, U.; Clark, K.; Jurafsky, D.; Kaiser, L. Sample efficient text summarization using a single pre-trained transformer. *arXiv* **2019**, arXiv:1905.08836.
344. Liu, Y.; Lapata, M. Text summarization with pretrained encoders. *arXiv* **2019**, arXiv:1908.08345.
345. Zhang, H.; Xu, J.; Wang, J. Pretraining-based natural language generation for text summarization. *arXiv* **2019**, arXiv:1902.09243.
346. Reda, A.; Salah, N.; Adel, J.; Ehab, M.; Ahmed, I.; Magdy, M.; Khoriba, G.; Mohamed, E.H. A Hybrid Arabic Text Summarization Approach based on Transformers. In Proceedings of the IEEE 2022 2nd International Mobile, Intelligent, and Ubiquitous Computing Conference (MIUCC), Cairo, Egypt, 8–9 May 2022; pp. 56–62.
347. Dathathri, S.; Madotto, A.; Lan, J.; Hung, J.; Frank, E.; Molino, P.; Yosinski, J.; Liu, R. Plug and Play Language Models: A Simple Approach to Controlled Text Generation. *arXiv* **2020**, arXiv:1912.02164.

348. Wang, T.; Wan, X.; Jin, H. AMR-To-Text Generation with Graph Transformer. *Trans. Assoc. Comput. Linguist.* **2020**, *8*, 19–33. [CrossRef]
349. Zhao, K.; Ding, H.; Ye, K.; Cui, X. A Transformer-Based Hierarchical Variational AutoEncoder Combined Hidden Markov Model for Long Text Generation. *Entropy* **2021**, *23*, 1277. [CrossRef] [PubMed]
350. Diao, S.; Shen, X.; Shum, K.; Song, Y.; Zhang, T. TILGAN: Transformer-based Implicit Latent GAN for Diverse and Coherent Text Generation. In Proceedings of the Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, Online Event, 1–6 August 2021; pp. 4844–4858.
351. Chan, A.; Ong, Y.; Pung, B.T.W.; Zhang, A.; Fu, J. CoCon: A Self-Supervised Approach for Controlled Text Generation. *arXiv* **2021**, arXiv:2006.03535.
352. Wang, Y.; Xu, C.; Hu, H.; Tao, C.; Wan, S.; Dras, M.; Johnson, M.; Jiang, D. Neural Rule-Execution Tracking Machine For Transformer-Based Text Generation. In Proceedings of the 35th Conference on Neural Information Processing Systems (NeurIPS 2021), Online, 6–14 December 2021.
353. Kulkarni, A.; Shivananda, A.; Kulkarni, A. Named-Entity Recognition Using CRF and BERT. In *Natural Language Processing Projects*; Apress: Berkeley, CA, USA, 2021. [CrossRef]
354. Li, X.; Yan, H.; Qiu, X.; Huang, X. Flat: Chinese ner using flat-lattice transformer. *arXiv* **2020**, arXiv:2004.11795.
355. Ma, L.; Jian, X.; Li, X. PAI at SemEval-2022 Task 11: Name Entity Recognition with Contextualized Entity Representations and Robust Loss Functions. In Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022), Seattle, WA, USA, 14–15 July 2022; pp. 1665–1670. [CrossRef]
356. Jarrar, M.; Khalilia, M.; Ghanem, S. Wojood: Nested arabic named entity corpus and recognition using bert. In Proceedings of the International Conference on Language Resources and Evaluation (LREC 2022), Marseille, France, 20–25 June 2022.
357. Yu, Y.; Wang, Y.; Mu, J.; Li, W.; Jiao, S.; Wang, Z.; Lv, P.; Zhu, Y. Chinese mineral named entity recognition based on BERT model. J. Expert Syst. Appl. **2022**, *206*, 117727. [CrossRef]
358. Wu, S.; Song, X.; Feng, Z. Mect: Multi-metadata embedding based cross-transformer for chinese named entity recognition. *arXiv* **2021**, arXiv:2107.05418.
359. Xuan, Z.; Bao, R.; Jiang, S. FGN: Fusion glyph network for Chinese named entity recognition. In *China Conference on Knowledge Graph and Semantic Computing*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 28–40.
360. Sehanobish, A.; Song, C.H. Using chinese glyphs for named entity recognition. *arXiv* **2019**, arXiv:1909.09922.
361. Chekol Jibril, E.; Cuneyd Tantg, A. ANEC: An Amharic Named Entity Corpus and Transformer Based Recognizer. *arXiv* **2022**, arXiv:2207.00785.
362. Schneider, E.; Rivera-Zavala, R.M.; Martinez, P.; Moro, C.; Paraiso, E.C. UC3M-PUCPR at SemEval-2022 Task 11: An Ensemble Method of Transformer-based Models for Complex Named Entity Recognition. In Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022), Seattle, WA, USA, 14–15 July 2022; pp. 1448–1456.
363. He, J.; Uppal, A.; Mamatha, N.; Vignesh, S.; Kumar, D.; Sarda, A.K. Infrrd. ai at SemEval-2022 Task 11: A system for named entity recognition using data augmentation, transformer-based sequence labeling model, and EnsembleCRF. In Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022), Seattle, WA, USA, 14–15 July 2022; pp. 1501–1510.
364. Ren, K.; Li, H.; Zeng, Y.; Zhang, Y. Named Entity Recognition with CRF Based on ALBERT: A Natural Language Processing Model. In *China Conference on Command and Control*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 498–507.
365. Basmatkar, P.; Maurya, M. An Overview of Contextual Topic Modeling Using Bidirectional Encoder Representations from Transformers. In *Proceedings of the Third International Conference on Communication, Computing and Electronics Systems*; Lecture Notes in Electrical Engineering; Springer: Singapore, 2022; Volume 844. [CrossRef]
366. Alcoforado, A.; Ferraz, T.P.; Gerber, R.; Bustos, E.; Oliveira, A.S.; Veloso, B.M.; Siqueira, F.L.; Costa, A.H.R. ZeroBERTo: Leveraging Zero-Shot Text Classification by Topic Modeling. In *Computational Processing of the Portuguese Language. PROPOR 2022*; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2022; Volume 13208. [CrossRef]
367. Baird, A.; Xia, Y.; Cheng, Y. Consumer perceptions of telehealth for mental health or substance abuse: A Twitter-based topic modeling analysis. *JAMIA Open* **2022**, *5*, ooac028. [CrossRef]
368. Elaffendi, M.; Alrajhi, K. Beyond the Transformer: A Novel Polynomial Inherent Attention (PIA) Model and Its Great Impact on Neural Machine Translation. *Comput. Intell. Neurosci.* **2022**. [CrossRef]
369. Li, D.; Luo, Z. An Improved Transformer-Based Neural Machine Translation Strategy: Interacting-Head Attention. *Comput. Intell. Neurosci.* **2022**, *2022*, 2998242. [CrossRef]
370. Dione, C.M.B.; Lo, A.; Nguer, E.M.; Oumar, S. Low-resource Neural Machine Translation: Benchmarking State-of-the-art Transformer for Wolof French. In Proceedings of the 13th Conference on Language Resources and Evaluation (LREC 2022), Marseille, France, 21–23 June 2022; pp. 6654–6661.
371. Tho, C.; Heryadi, Y.; Kartowisastro, I.H.; Budiharto, W. A Comparison of Lexicon-based and Transformer-based Sentiment Analysis on Code-mixed of Low-Resource Languages. In Proceedings of the 2021 1st International Conference on Computer Science and Artificial Intelligence (ICCSAI), Jakarta, Indonesia, 28 October 2021; Volume 1, pp. 81–85.
372. Fu, Q.; Teng, Z.; White, J.; Schmidt, D.C. A Transformer-based Approach for Translating Natural Language to Bash Commands. In Proceedings of the 2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA), Pasadena, CA, USA, 13–16 December 2021; pp. 1245–1248.

373. Zhao, L.; Gao, W.; Fang, J. High-Performance English–Chinese Machine Translation Based on GPU-Enabled Deep Neural Networks with Domain Corpus. *Appl. Sci.* **2021**, *11*, 10915. [CrossRef]
374. Ali, Z. Research Chinese-Urdu Machine Translation Based on Deep Learning. *J. Auton. Intell.* **2021**, *3*, 34–44. [CrossRef]
375. Jing, H.; Yang, C. Chinese text sentiment analysis based on transformer model. In Proceedings of the 2022 3rd International Conference on Electronic Communication and Artificial Intelligence (IWECAI), Zhuhai, China, 14–16 January 2022; pp. 185–189.
376. Tiwari, D.; Nagpal, B. KEAHT: A Knowledge-Enriched Attention-Based Hybrid Transformer Model for Social Sentiment Analysis. *New Gener. Comput.* **2022**, *40*, 1165–1202. [CrossRef] [PubMed]
377. Potamias, R.A.; Siolas, G.; Stafylopatis, A. A transformer-based approach to irony and sarcasm detection. *Neural Comput. Appl.* **2020**, *32*, 17309–17320. [CrossRef]
378. Mandal, R.; Chen, J.; Becken, S.; Stantic, B. Tweets Topic Classification and Sentiment Analysis based on Transformer-based Language Models. *Vietnam. J. Comput. Sci.* **2022**. [CrossRef]
379. Zhao, T.; Du, J.; Xue, Z.; Li, A.; Guan, Z. Aspect-Based Sentiment Analysis using Local Context Focus Mechanism with DeBERTa. *arXiv* **2022**, arXiv:2207.02424.
380. Kokab, S.T.; Asghar, S.; Naz, S. Transformer-based deep learning models for the sentiment analysis of social media data. *Array* **2022**, *14*, 100157. [CrossRef]
381. J, A.K.C.; Cambria, E.; Trueman, T.E. Transformer-Based Bidirectional Encoder Representations for Emotion Detection from Text. In Proceedings of the 2021 IEEE Symposium Series on Computational Intelligence (SSCI), Orlando, FL, USA, 5–7 December 2021; pp. 1–6.
382. Yue, T.; Jing, M. Sentiment Analysis Based on Bert and Transformer. In *Springer Proceedings in Business and Economics*; Springer: Cham, Switzerland, 2022._23. [CrossRef]
383. Fedus, W.; Zoph, B.; Shazeer, N.M. Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity. *arXiv* **2021**, arXiv:2101.03961.
384. Ontanon, S.; Ainslie, J.; Cvicek, V.; Fisher, Z. Making transformers solve compositional tasks. *arXiv* **2021**, arXiv:2108.04378.
385. Li, Z.; Wallace, E.; Shen, S.; Lin, K.; Keutzer, K.; Klein, D.; Gonzalez, J. Train Large, Then Compress: Rethinking Model Size for Efficient Training and Inference of Transformers. *arXiv* **2020**, arXiv:2002.11794.
386. Child, R.; Gray, S.; Radford, A.; Sutskever, I. Generating long sequences with sparse transformers. *arXiv* **2019**, arXiv:1904.10509.
387. Ye, Z.; Guo, Q.; Gan, Q.; Qiu, X.; Zhang, Z. Bp-transformer: Modelling long-range context via binary partitioning. *arXiv* **2019**, arXiv:1911.04070.
388. Kitaev, N.; Kaiser, L.; Levskaya, A. Reformer: The efficient transformer. *arXiv* **2020**, arXiv:2001.04451.
389. Choromanski, K.; Likhosherstov, V.; Dohan, D.; Song, X.; Gane, A.; Sarlos, T.; Hawkins, P.; Davis, J.; Mohiuddin, A.; Kaiser, L.; et al. Rethinking Attention with Performers. *arXiv* **2020**, arXiv:2009.14794.
390. Katharopoulos, A.; Vyas, A.; Pappas, N.; Fleuret, F. Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention. In Proceedings of the ICML'20: Proceedings of the 37th International Conference on Machine Learning (ICML), Virtual, 13–18 July 2020; Article No.: 478, pp. 5156–5165.
391. Su, J.; Lu, Y.; Pan, S.; Wen, B.; Liu, Y. Roformer: Enhanced transformer with rotary position embedding. *arXiv* **2021**, arXiv:2104.09864.
392. Zhang, T.; Wu, F.; Katiyar, A.; Weinberger, K.Q.; Artzi, Y. Revisiting few-sample BERT fine-tuning. *arXiv* **2020**, arXiv:2006.05987.
393. Chang, P. Advanced Techniques for Fine-Tuning Transformers. 2021. Available online: https://towardsdatascience.com/advanced-techniques-for-fine-tuning-transformers-82e4e61e16e (accessed on 2 October 2022).
394. Singh, T.; Giovanardi, D. How much does pre-trained information help? Partially re-initializing BERT during fine-tuning to analyze the contribution of layers. In Stanford CS224N Natural Language Processing with Deep Learning. 2020. Available online: https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1204/reports/custom/report13.pdf (accessed on 2 October 2022).
395. Li, Y.; Lin, Y.; Xiao, T.; Zhu, J. An Efficient Transformer Decoder with Compressed Sub-layers. In Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI-21), Virtual, 2–9 February 2021.
396. Song, Y.; Wang, J.; Liang, Z.; Liu, Z.; Jiang, T. Utilizing BERT Intermediate Layers for Aspect Based Sentiment Analysis and Natural Language Inference. *arXiv* **2020**, arXiv:2002.04815.
397. Zou, W.; Ding, J.; Wang, C. Utilizing BERT Intermediate Layers for Multimodal Sentiment Analysis. In Proceedings of the 2022 IEEE International Conference on Multimedia and Expo (ICME), Taipei, Taiwan, 18–22 July 2022; pp. 1–6.
398. Evci, U.; Dumoulin, V.; Larochelle, H.; Mozer, M.C. Head2Toe: Utilizing Intermediate Representations for Better Transfer Learning. In Proceedings of the 39th International Conference on Machine Learning, PMLR (2022), Baltimore, MD, USA, 17–23 July 2022.
399. Lewkowycz, A. How to decay your learning rate. *arXiv* **2021**, arXiv:2103.12682.
400. Lee, C.; Cho, K.; Kang, W. Mixout: Effective regularization to finetune large-scale pretrained language models. *arXiv* **2019**, arXiv:1909.11299.
401. Baldi, P.; Sadowski, P.J. Understanding dropout. In Proceedings of the NIPS'13: Proceedings of the 26th International Conference on Neural Information Processing Systems-Volume 2, Lake Tahoe, NV, USA, 5–10 December 2013; pp. 2814–2822.
402. Wan, L.; Zeiler, M.; Zhang, S.; Le Cun, Y.; Fergus, R. Regularization of neural networks using dropconnect. In Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16–21 June 2013; pp. 1058–1066.
403. Hua, H.; Li, X.; Dou, D.; Xu, C.; Luo, J. Fine-tuning Pre-trained Language Models with Noise Stability Regularization. *arXiv* **2022**, arXiv:2206.05658.

404. Ishii, M.; Sato, A. Layer-wise weight decay for deep neural networks. In *Pacific-Rim Symposium on Image and Video Technology*; Springer: Berlin/Heidelberg, Germany, 2017; pp. 276–289.
405. Yu, H.; Cao, Y.; Cheng, G.; Xie, P.; Yang, Y.; Yu, P. Relation Extraction with BERT-based Pre-trained Model. In Proceedings of the 2020 International Wireless Communications and Mobile Computing (IWCMC), Limassol, Cyprus, 15–19 June 2020; pp. 1382–1387.
406. Cao, Q.; Trivedi, H.; Balasubramanian, A.; Balasubramanian, N. DeFormer: Decomposing Pre-trained Transformers for Faster Question Answering. *arXiv* **2020**, arXiv:2005.00697.
407. Izmailov, P.; Podoprikhin, D.; Garipov, T.; Vetrov, D.; Wilson, A.G. Averaging weights leads to wider optima and better generalization. *arXiv* **2018**, arXiv:1803.05407.
408. Khurana, U.; Nalisnick, E.T.; Fokkens, A. How Emotionally Stable is ALBERT? Testing Robustness with Stochastic Weight Averaging on a Sentiment Analysis Task. *arXiv* **2021**, arXiv:2111.09612.
409. Smith, S.L.; Kindermans, P.J.; Ying, C.; Le, Q.V. Don't decay the learning rate, increase the batch size. *arXiv* **2017**, arXiv:1711.00489.
410. Dong, C.; Wang, G.; Xu, H.; Peng, J.; Ren, X.; Liang, X. EfficientBERT: Progressively Searching Multilayer Perceptron via Warm-up Knowledge Distillation. In Proceedings of the Findings of the Association for Computational Linguistics: EMNLP, Online, Punta Cana, Dominican Republic, 7–11 November 2021.
411. Liu, C.L.; Hsu, T.Y.; Chuang, Y.S.; Lee, H.Y. A Study of Cross-Lingual Ability and Language-specific Information in Multilingual BERT. *arXiv* **2020**, arXiv:2004.09205.
412. Lauscher, A.; Ravishankar, V.; Vulic, I.; Glavas, G. From Zero to Hero: On the Limitations of Zero-Shot Language Transfer with Multilingual Transformers. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Online, 16–20 November 2020.