Computer Science and Information Systems 21(3):1097–1145 https://doi.org/10.2298/CSIS230217031C

Natural Language Processing: An Overview of Models, Transformers and Applied Practices

Santiago Canchila[1], Carlos Meneses-Eraso[1], Javier Casanoves-Boix[2], Pascual Cortés-Pellicer[3], and Fernando Castelló-Sirvent[3]

[1] Universidad Sergio Arboleda
Calle 74 No. 14 - 14 Bogotá, Colombia
santiago.canchila01@usa.edu.co
carlos.meneses@usa.edu.co
[2] Universidad Alfonso X El Sabio
Avda. de la Universidad, 1. Villanueva de la Cañada, 28691, Madrid, Spain
jcasaboi@uax.es
[3] Department of Business Management Universitat Politècnica de València (UPV)
Camí de Vera, s/n - 46022 València, Spain
pascorpe@omp.upv.es
fernando.castello@upv.es

Abstract. The study of utilizing human language in computer systems referred to as NLP, is becoming increasingly significant in various aspects of life, including research, daily activities, commerce, and entrepreneurship endeavors. A multitude of tech companies are dedicating resources towards the development and improvement of NLP methods, models, and products. To add to that, open-source contributions to the field are on the rise. However, with so much progress being made, it may be challenging to understand the current state of NLP and what models are considered to be the most efficient. To help those grappling with the fast-paced and constantly evolving NLP landscape, we have put together a comprehensive overview of the latest NLP research and advancements.

Keywords: NLP, Deep Learning, Transformers.

1. Introduction

The area of Natural Language Processing (NLP) appears to have a great impact on today's research, day-to-day life, commercial, and entrepreneurial endeavors. Some of the biggest tech companies have a footing in the development of newer NLP techniques, models, architectures, and products. Some open-sourcing efforts have also started to materialize. With all those new creations it may be hard to understand what the state-of-the-art is and what are the optimal models for NLP. We conduct a survey of NLP research and novelties to help those groups that may be entangled in the accelerating and messy area of current NLP.

Many applications in the field of NLP have been studied, including question answering (QA), text classification, summarization, text generation, zero-shot or few-shot classification, translation, text similarity, sentiment analysis, named entity recognition (NER), relation extraction, information retrieval, natural language inference, dialogue, factual information detection, text simplification, text clustering among others.

The area of NLP has had a massive growth fig 5 over the past years. Different approaches, techniques, and models have been proposed by researchers, trying to improve upon an, or various, aspects of previous proposals. These aspects can be summarized with different metrics, some of the most relevant ones being task-specific performance, energy efficiency, memory used, and training complexity.

Different models have been proposed to solve NLP tasks, and trade-offs have to be made when choosing the best model to use. Previous approaches such as Bag of Words or TF-IDF still prove to be useful in the NLP landscape, and should be a viable consideration when solving NLP tasks, especially when computational resources and data may be scarce. With this into consideration, it is important to note that Transformer models have achieved state-of-the-art performance on different tasks. According to Ref [46] BERT, pre-trained on a financial news corpus, allows to detect the dominant themes in the news, as well as to evaluate the tone of the articles. For their part, Ref [12] argues that, in contrast to ordinary text classification, short text presents a unique problem, consisting of less vocabulary and scarcity of features, which generates a higher need for semantic feature representation. Therefore, in order to address this problem they propose a model based on BERT.

A way to identify Spanish language hate discourse on social media using pre-trained monolingual and multilingual models, including BERT, XLM [81] and BETO [4] was proposed [8].

Ref [91] studies the possibility that removing news tagged as harmful may also eliminate satiric news. Using DistilBERT [116] they show a way to distinguish between satiric and fake news. Social media generates a large amount of data that could be used to identify the sentiment of its users using XLNet [160], DistilBERT [116] or RoBERTa [90].

Client relationships can also be improved using NLP, detecting possible weak points in client-company interactions. RoBERTa is used to detect five weak points to improve this interaction [115].

Taking those applications into consideration, it is still difficult to contemplate the whole NLP landscape. With a growing research scene, more models with various benefits, and different levels of access, it can become a difficulty to those groups wanting to use what the NLP area has to offer. With this in mind, an overview of different models and techniques will be shown, emphasizing the state-of-the-art transformer architecture [143]. A small analysis of the impact of transformers, especially Bidirectional Encoder Representations from Transformers (BERT) [43], on the research community will be presented. An analysis of the abstracts of many NLP related papers was made using various approaches to better understand their trade-offs. Some of the most relevant models, architectures, and approaches can be a point of discussion, showing that there is a trend for bigger, decoder-only, generative models, and what that implies. Finally, some guidance on what the future of NLP may look like will be provided.

2. Historic Overview

The question of whether or not computers can process natural language has been an inherent topic in computer science. In the 1950s, Alan Turing proposed a way to identify if

[4] https://github.com/dccuchile/beto

machines can think, with the imitation game [140]. This test is essentially an NLP problem, as the computer would need to process the question given by the person, understand it, and answer. This paves the way for NLP research, as it would not only advance an important field but would, if successful, also solve one of the open questions in computer science.

The earliest wave of computational NLP research came in the 1950s, introducing the concept and starting to involve the rudimentary computational power of the time. Because of the limitations of computers at that time, both in memory and processing power, it was not feasible to test or implement the proposed techniques, but a theoretical origin of NLP was starting to develop. Some groundwork was created around natural language, establishing common problems in a computational representation of language. Natural language is complicated and has rules and exceptions that are the result of thousands of years of change, so it was obvious at the beginning that, for example, hard coding a chatbot with answers was not feasible, because of the sheer amount of possibilities. Some interpretation and variability were needed in order to obtain good results. Rule-based systems started to be implemented to solve those limitations. An example from this rule-based approach to NLP was one of the first applications of NLP in a practical field, being of help in a medical setting, used by physicians in training was proposed in Ref [124]. Some areas of research started to develop, including text-to-speech [4], syntactic analysis [31], part of speech tagging [18], machine translation [5] and human interaction [152].

NLP was described as a range of computational methods driven by theories to analyze and represent natural language text at various linguistic levels, aiming to imitate human language processing for various tasks and applications [23]. NLP is a branch of machine learning that deals with text and speech [2], that began in the 1940s as an intersection of artificial intelligence and linguistics [96]. Since its inception, contributions in the theory of universal grammar, have been definitive for the development of NLP to adequately address the study of language [30,10]. Other investigations, such as the contributions of Ref [58], in which they propose the use of an associative categorical dual model for the representation of the meaning of words and its possible applications in artificial intelligence and NLP.

Ref [86], shows that NLP had two distinct focuses: Natural Language Understanding, NLU (reader/listener) for the purpose of producing a meaningful representation, and Natural Language Generation (writer/speaker) that refers to the production of language from a representation, that, according to Ref [76], the first has five sections: Phonology, Morphology, Syntax, Semantics and Pragmatics, and the second refers to identifying the objectives, planning how the objectives can be achieved by evaluating the situation and the available communication sources and making the plans as a text. research that analyzes the concepts of NLP such as Information Retrieval (IR) and Information Extraction (IE), which is the process of finding in a large data repository, the material of an unstructured or semi-structured nature, while Information Extraction (IE), consisting of obtaining sections of interest in the text to transfer them to a structured format [56]

It is important to note that previous approaches to NLP are still used, especially in the preprocessing stage, and they also give good performance on certain tasks and certain situations.

[5] https://www.ibm.com/ibm/history/exhibits/701/701_translator.html

3. Previous Approaches

Traditional approaches have various advantages over the current state-of-the-art approaches, as they are less computationally expensive, they may be a good option especially when data is scarce, and they can still obtain competitive performances. Some of those approaches will be presented next.

3.1. Rule-based and symbolic

This approach may include regex, rule-based systems, morphological analysis, tokenization, syntax analysis, parsing, and text transformation and normalization. NLP systems that are interpretable can promote trust by allowing end-users, practitioners, and researchers to grasp the model's prediction mechanisms, thereby ensuring ethical NLP practices. In the past, conventional NLP systems, like rule-based approaches, have typically operated in this manner [69,155]. Some examples of rule-based collections, topics, systems, and their possible uses are:

- One of the first books on the subject is presented, documenting many of the existing expert systems, as well as some of the available software for building them. From this, they abstract some principles and general approaches for the development of expert systems. And, some time later, the principles and history of the method are explained, exposing that rule-based systems automate knowledge for problem-solving, provide a means to capture and refine human experience [118].
- The ISCREEN system is described, a prototype for text message detection. ISCREEN includes a high-level interface for users to define rules, a text message filtering component, and a conflict detection component that examines the rules for inconsistencies. An explanation component uses text generation to respond to user queries about the system's past or potential actions, based on Grice's conversational maxims [108,55].
- Research on natural language processing involves at least four different but closely related areas of study, which are: (1) research on the psychological processes involved in understanding human language, (2) the construction of computational systems for analyzing natural language input (and/or generating natural language output), (3) the development of theories about the structure of natural language, and (4) the determination of the mathematical properties of grammatical formalisms [119].
- Some recent approaches to fully rule-based systems can still be useful. A cloud-based rule-based computing system for evaluating requests for academic scholarships for postgraduate students is presented. Among the findings, it is demonstrated that the system allowed students to submit their documents online and view the score for each item entered. The final cumulative points were automatically calculated according to the standards. The system reduced the burden on students during their applications. As a result, the system could motivate students to engage in academic scholarship applications by providing transparent and user-friendly public rules [165].

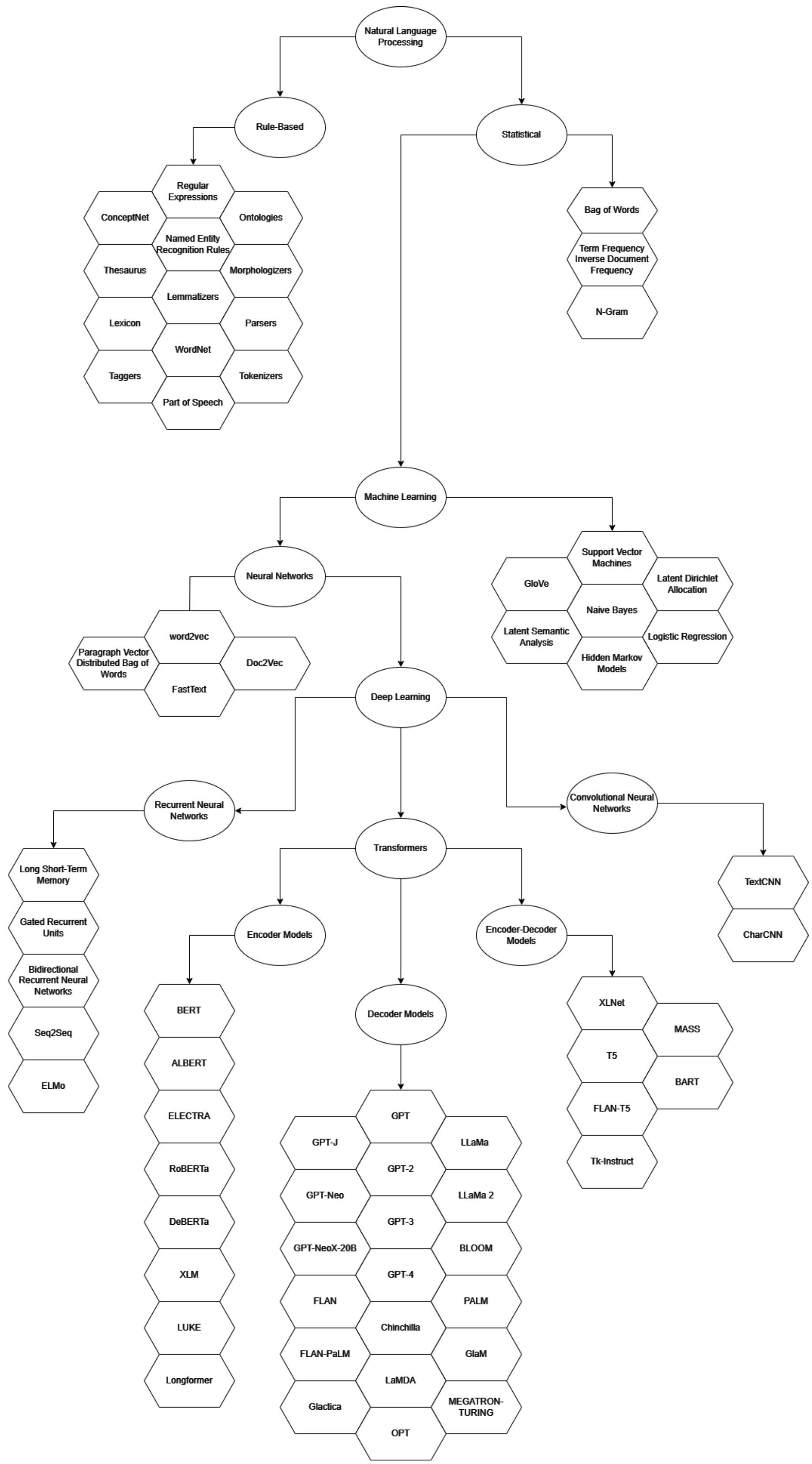

Fig. 1. Map of NLP techniques and models

3.2. Bag of Words

The Bag of Words (BoW) methodology is a simple yet effective way to represent textual data using word frequency vectors. It was initially proposed for text document analysis [59], but similar approaches have found a place in other areas such as image analysis. In this method, the number of occurrences of each "bag" generated for each type of instance or word is recorded, without considering their order or grammar. Different approaches to represent BoWs are used, which can include hashmaps or sparse matrices [6]. BoWs have been used extensively in NLP, and it has shown to be effective when paired with other algorithms, especially machine learning and statistical ones.

3.3. TF-IDF

TF-IDF works by determining the relative frequency of words in a specific document compared to the inverse proportion of that word in the entire document corpus. Intuitively, this calculation determines how relevant a particular word is in a specific document. Words that are common in a single document or a small group of documents tend to have higher TF-IDF values than common words like articles and prepositions. The formal procedure for implementing TF-IDF has some minor differences in its various applications, but the general approach works as follows. Given a collection of documents D, a term t, an individual document $d \in D$, a function $f(t, d)$ that returns the frequency of term t in document d, and T_d different terms in d, we calculate:

$$TF(t, d) = \frac{f(t, d)}{\sum_{i=0}^{T_d} f(t_i, d)} \tag{1}$$

$$IDF(t, D) = log(\frac{|D|}{|\{d \in D | t \in d, \forall d \in D\}|}) \tag{2}$$

$$TF\text{-}IDF(t, d, D) = TF(t, d) \cdot IDF(t, D) \tag{3}$$

TF-IDF also works best when paired with other algorithms, as the vector, mathematical representation of documents allows the use of different models and approaches.

3.4. N-gram

N-grams are sequences of characters or words extracted from a text. They can be categorized into two main types, character-based and, word-based. A character-based N-gram is a set of n consecutive characters extracted from a word. The primary motivation behind this approach is that similar words will have a high proportion of common N-grams. Typical values for n are 2 or 3, corresponding to the use of bigrams or trigrams, respectively.

Let S be a sequence of text data, and S_i represent the i-th item in the sequence, where $i \in [0..|S|]$. An n-gram can be defined as a contiguous sequence of n items from S, denoted as $S_i, S_{i+1}, ..., S_{i+n}$, where $0 \leq i \leq (|S| - n)$ [122]. N-grams are useful when capturing local patterns in text.

[6] https://docs.scipy.org/doc/scipy/reference/sparse.html

3.5. LSA

LSA is a technique in natural language processing and information retrieval that uses singular value decomposition (SVD) to analyze and represent the relationships between terms and documents in a high-dimensional space. It's widely used for tasks such as document retrieval, information retrieval, and text analysis to discover the latent semantic structure in a corpus of text [40]. LSA has found success in topic modeling, similarity, clustering, and classification among others.

A Document-Term Matrix must be constructed, representing the frequency of a term t in a document d for every $t \in T$ where T are all the terms in a corpus D, and every $d \in D$. A TF-IDF, or any other weighted, matrix can also be used. A decomposition into three matrices U, Σ and V^T is done, where

- U: The left singular vectors matrix represents the relationships between documents.
- Σ : A diagonal matrix with singular values, representing the strength of the latent semantic concepts.
- V^T: The right singular vectors matrix represents the relationships between terms.

Dimensionality reduction can then be applied by keeping the top k singular values and their vectors. After this reduction, a semantic space is achieved, where all documents and terms are described by a vector in the space, capturing a latent semantic relationship between them.

3.6. LDA

Latent Dirichlet Allocation represents a three-tiered hierarchical Bayesian model, where every item in a collection is depicted as a finite mixture of underlying topics. These topics are further represented as an infinite mixture of topic probabilities, which, in the context of text modeling, offer a clear way to represent a document [16].

In LDA, a document, denoted as d, is a collection of $|d|$ words, where $d = w_0, w_1, ..., w_{|d|}$. For all documents d in the corpus D and all topics k in the set of topics K, LDA models each document as a distribution over topics, creating a matrix of size $(|D|, |K|)$ to represent the different latent topic proportions for each document.

Additionally, a distribution β is created for every topic k, which represents the probability distribution over words for that topic. The goal of LDA is to learn the optimal θ distributions for every document d and the optimal β distributions for every topic k.

The matrix β is constructed, where $\beta_{ij} = p(w^j = 1|z^i = 1)$ represents the probability of word w^j occurring in topic z^i. The ultimate objective of LDA is to discover these latent topic proportions and word distributions to represent the underlying structure of the document collection.

LDA has proven to be a performant model for topic modeling in a variety of applications and domains. Its ability to uncover latent topics within a corpus of text data has made it a valuable tool for tasks such as document clustering, content recommendation, and sentiment analysis.

3.7. word2vec

Although vectorial and spatial representation of texts were proposed previously in other models and methodologies, word2vec [34] is a model that achieved major improvements in NLP with the use of embeddings.

The generation of a word embedding through Word2Vec can be done in two different ways: Skip-Gram and Continuous Bag of Words. Both approaches are based on a different handling of input and output variables but essentially use the same neural network structure. They have similar architectures but different learning objectives. The skip-gram model uses the current word to predict the surrounding words, while the CBOW model predicts the current word based on the surrounding words [44]. Skip-gram training tends to give a better semantic representation of words. The idea of a skip-gram is to be able to, given a probability distribution, predict the surrounding words in a window for every word W_n. This was one of the objectives proposed with the word2vec model[34].

$$P(W_{n+j}|W_n) \tag{4}$$

where the word W_t is the center word. Then an objective function to be minimized for N words can be formulated for a window of size m surrounding a word W_n:

$$J'(\theta) = \prod_{n=0}^{N} \prod_{\substack{-m \leq j \leq m \\ j \neq 0}} P(W_{n+j}|W_n;\theta) \tag{5}$$

Where the probability is also dependent on some parameters θ

A loss function to be optimized can then be proposed as follows. This loss function is generally optimized using gradient descent:

$$J(\theta) = \frac{1}{N} \sum_{n=1}^{N} \sum_{\substack{-m \leq j \leq m \\ j \neq 0}} logP(W_{n+j}|w_n) \tag{6}$$

3.8. GloVe

GloVe is an unsupervised learning algorithm used to obtain vector representations of words. It trains on global statistics derived from word-word co-occurrence in a corpus, and the resulting representations reveal interesting linear substructures within the word vector space.

The training objective of GloVe is to learn word vectors in such a way that their dot product equals the logarithm of the co-occurrence probability of words. This objective associates (the logarithm of) probability ratios of co-occurrence with vector differences in the word vector space, thanks to the mathematical equivalence of logarithmic ratios and vector differences. This allows the encoding of some form of meaning, as these ratios can capture semantic relationships [106] [7].

[7] https://nlp.stanford.edu/projects/glove/

3.9. Neural Networks and Deep Learning

Given the advancement of computational power, the use of more demanding models became possible. In this order, neural language modeling aims to improve the performance of multiple related learning tasks [37] by taking advantage of useful information among them [28]. Task relationship is based on understanding how different tasks are related and includes learning tasks such as supervised tasks, which include classification and regression tasks, unsupervised tasks, including clustering tasks, semi-supervised tasks, active learning tasks, reinforcement learning tasks, online learning tasks, and multi-view learning tasks [166]. In the same way, the concept of word embedding is expanded on further, improving search tasks, and content recommendation [34]. Sequence-by-sequence learning models with neural networks were included in the development of NLP in [131] through a translation task using a multilayer LSTM to assign the input sequence to a vector of fixed dimensionality, and then another deep LSTM to decode the target sequence of the vector.

On the other hand, Ref [76], summarizes the use of convolutional neural networks in the fields of Sentence Classification, Sentiment Analysis, Text Classification, Text Summary, Machine Translation, and Response Relations, as well as the use of recurrent neural networks (RNN) which, according to Ref [133] is defined as a type of artificial neural network where the connections between units form a directed cycle creating an internal state of the network that allows you to exhibit dynamic behavior, usually applied to sequential data such as text, time series, financial data, voice, audio, video among, others [76]. CNNs have been used extensively for NLP, as Refs [75,51,77,128,147,117] have all used CNNs successfully to solve NLP tasks.

With the growth in computational power, RNNs started to become viable. Long-Short Term Memory (LSTM) [62] and Bidirectional LSTM [54], were used to solve NLP tasks [68,168]. The sequential-like nature of natural language made it perfect for LSTMs, as they specialize in that. LSTMS, however, also had some drawbacks [167], but they still performed exceptionally.

Currently, RNNs have almost been super-seeded by transformers, as they do not offer a better performance and are more computationally expensive to use and train, but they may still offer unique benefits compared to other approaches.

ELMo: Embeddings from Language Models (ELMo) is a model that achieved great performance on various tasks. It represents one of the early large, pre-trained language models that demonstrated a significant improvement in the quality of contextualized word representations. ELMo is a model that relies on LSTM and biLSTM units to create these contextual embeddings. It achieved great performance on a broad range of NLP tasks and benchmarks, including Q
A, Textual entailment, NER, Semantic role labeling, and Sentiment analysis among others [107].

4. Transformers

In 2017 a breakthrough was made in the area of NLP, the transformer architecture was proposed [143], creating a path for some of the current state-of-the-art models to be realized. The original architecture made use of attention mechanisms exclusively, being able to move away from recurrent and convolutional neural networks. Having such a simple, but scalable, architecture would make transformers the preferred architecture for solving NLP tasks and create state-of-the-art models. Transformers provided improvements in two important aspects without a trade-off: they were more computationally inexpensive, requiring less time to train, and being more parallelizable, while also obtaining better results on standard NLP tasks and NLP benchmarks.

Without the need to use an RNN, the inherent need for sequential computational steps was not needed, so a full token sequence could be computed without the need to precompute previous steps. This allows parallelization in sentence processing, making training more efficient on devices such as GPUs and TPUs. This would result in two of the most attractive aspects of transformers: they are highly scalable and can be easily trained on specialized hardware; they can be pre-trained on large unlabeled text corpus, in a self-supervised manner, to generate language understanding, and then fine-tuned to solve specific NLP tasks while achieving state-of-the-art performance [143]. Generally, the amount of data necessary to perform the fine-tuning on a transformer-based model that has been pre-trained is also considerably lower than if a model was trained from scratch.

The benefits of the transformer architecture made it so researchers started to use it to create more complex, bigger, or specific, derived architectures, while also utilizing different training or fine-tuning data. Currently, transformers represent the state-of-the-art architecture for NLP.

4.1. The Transformer Architecture

The original transformer architecture has different mechanisms that make it so effective. First, a preprocessing pipeline is used, in which a trainable tokenizer is used.

The tokenizer outputs are passed along a learned embedding layer to create a word embedding of the input tokens. This embedding layer generally works similarly to a lookup table, in which the input is an index, related to the tokens, and the output is the embedding for that token, which may differ in dimensionality and may include contextual information. The initialization matrix of this embedding layer can come from other trained word vectors like GloVe or word2vec, or with a random distribution. This embedding layer will then be trained during backpropagation.

A positional encoding is used to maintain information about the position of the tokens in a sequence. In the original transformer architecture a positional encoding layer with the same shape as the embedding layer was proposed so that they could be added. The following functions were used, for a position n, a constant k, and an input dimensionality d_model. The value of k was 10000 in the original architecture [143].

$$PE(n, 2i) = sin(\frac{n}{k^{2i/d_{model}}}) \tag{7}$$

$$PE(n, 2i + 1) = cos(\frac{n}{k^{2i/d_{model}}}) \tag{8}$$

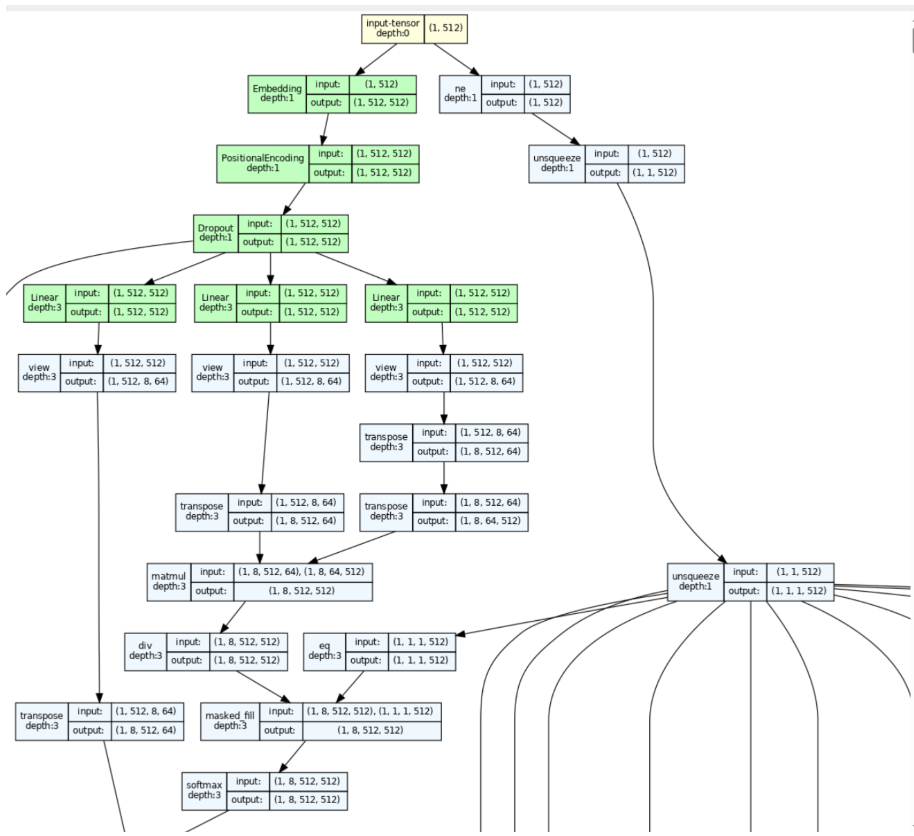

Fig. 2. Computation graph of the transformer architecture

Other models may use a trained positional encoding layer. The positional encoding is achieved with varying degrees of success when using a trained layer, as some models may not actually be learning an absolute or relative positional encoding of the sequence. This may depend on the training objective used when training the model, with better positional representation achieved on models with an autoregressive nature, in comparison to models that use masked modeling [148].

Afterwards, one of the most important points in the transformer architecture is used, the attention heads. Attention was a mechanism proposed some years prior to the transformer architecture [11], and it was used in conjunction with RNNs. The use of an attention head in a transformer expenses it from needing to use a kind of recurrent or convolutional architecture. An attention head takes 3 inputs, a query Q, a key K, and a value V, where [143]

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{9}$$

A version of attention that can be run in parallel, called Multi-Head Attention, was also proposed. The function takes three arguments: Q, K, and V, which are matrices that correspond to a sequence of query vectors, key vectors, and value vectors, respectively. W_i^Q, W_i^K, and W_i^V are weight matrices for the i-th attention head. They project the input Q, K, and V matrices into a different subspace suitable for the i-th head's attention operation. By having separate weight matrices, each head can potentially learn to focus on different parts of the input or capture different types of dependencies in the data.

$$MultiHead(Q, K, V) = [A_0; ...; A_h]W^O \tag{10}$$

$$where A_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$$

The intention of using attention heads is that the attention mechanism allows transformers to capture the relations between the tokens in a sequence by providing a spatial similarity of the value related to the query, while also assigning different levels of importance to different parts of the input sequence dynamically. Multi-Head Attention is able to attend to information that may be encoded in different spaces, and may attend to, for example, relationships with different length dependencies[143].

While a transformer model that uses both the encoder block and the decoder block uses attention mechanisms, they can be further categorized into self-attention and attention.

Encoder Block : The encoder block employs self-attention. Self-attention allows each token in the input sequence to attend to all other tokens within the same input sequence. This mechanism captures relationships and dependencies among tokens in the input. In this case, $Q = V = K$, where Q, V, and K are the query, value, and key vectors. The encoder block will then be composed of the following components:

- Self-Attention Layer: As mentioned, the self-attention mechanism allows each token in the input sequence to attend to all other tokens. Since $Q = K = V$, this is specifically known as self-attention because each token is effectively attending to itself and other tokens within the same sequence. This helps the model understand the context and inter-token relationships.
- Feed-Forward Neural Network: After self-attention, the output goes through a Feed-Forward Neural Network (FFNN) that is applied position-wise to each token's representation. The FFNN typically consists of two linear transformations with an activation function in between.
- Residual Connections: Each of the two layers (self-attention and FFNN) has a residual connection that adds the input of the layer to its output, followed by layer normalization. This skip-connection helps with gradient flow and allows the network to learn more effectively by providing an alternative pathway for gradient propagation.
- Layer Normalization: Layer normalization is applied after the residual connection, which normalizes the outputs across the features for each data instance, helping in stabilizing the learning process.

Decoder Block : The decoder block also uses self-attention, but it serves a slightly different purpose. Here, self-attention enables each token in the target sequence to attend to other tokens within the same target sequence. This is crucial for modeling dependencies within the output sequence.

In addition to self-attention, the decoder block utilizes attention between the encoder and decored blocks. This type of attention allows the decoder to focus on information from the encoder's output. This attention enables the decoder to consider relevant information from the input sequence while generating the output sequence. The decoder block will then have the following components:

– Masked Self-Attention Layer: The early self-attention layer in the decoder is modified to prevent each token from attending to subsequent tokens in the sequence. This is known as masked self-attention and is necessary because, during training, the decoder should only be allowed to consider tokens it has already generated.
– Encoder-Decoder Attention Layer: This layer is responsible for attending to the encoder's output. Here, the Q vectors come from the previous masked self-attention layer's outputs, whereas the K and V vectors are the outputs from the encoder stack. This mechanism helps the decoder focus on different parts of the encoder's output which is important for tasks such as translation where the alignment between input and output sequence elements is not known in advance.
– FFNN: Like the encoder block, the decoder also contains a position-wise FFNN.
– Residual Connections and Layer Normalization: Each sub-layer within the decoder (the masked self-attention, encoder-decoder attention layer, and FFNN) includes a residual connection followed by layer normalization.

Attention mechanisms are a fundamental component of transformers, and they encompass both self-attention, for capturing relationships within the same sequence, and attention, for connecting information between different sequences, such as the encoder's output and the decoder's input.

It is important to note that all of these layers are trained with backpropagation. With current frameworks, such as Pytorch [104,103] or JAX[17], automatic differentiation can be used to obtain the needed gradients for training. The autograd those frameworks use, along with graph execution storage, has also been a key point for the efficient training of different DL architectures.

The first steps of the transformer architecture computation may look like fig 2, using PyTorch and torchview [8], while the transformer architecture may look as follows:

```
Transformer(
  (encoder_embedding): Embedding(10000, 512)
  (decoder_embedding): Embedding(10000, 512)
  (positional_encoding): PositionalEncoding()
  (encoder_layers): ModuleList(
    (0-5): 6 x EncoderLayer(
      (self_attn): MultiHeadAttention(
        (W_q): Linear(in_features=512, out_features=512, bias=True)
        (W_k): Linear(in_features=512, out_features=512, bias=True)
        (W_v): Linear(in_features=512, out_features=512, bias=True)
        (W_o): Linear(in_features=512, out_features=512, bias=True)
      )
      (feed_forward): PositionWiseFeedForward(
        (fc1): Linear(in_features=512, out_features=2048, bias=True)
        (fc2): Linear(in_features=2048, out_features=512, bias=True)
        (relu): ReLU()
      )
      (norm1): LayerNorm((512,), eps=1e-05, elementwise_affine=True)
```

[8] https://github.com/mert-kurttutan/torchview

```
      (norm2): LayerNorm((512,), eps=1e-05, elementwise_affine=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
  )
  (decoder_layers): ModuleList(
    (0-5): 6 x DecoderLayer(
      (self_attn): MultiHeadAttention(
        (W_q): Linear(in_features=512, out_features=512, bias=True)
        (W_k): Linear(in_features=512, out_features=512, bias=True)
        (W_v): Linear(in_features=512, out_features=512, bias=True)
        (W_o): Linear(in_features=512, out_features=512, bias=True)
      )
      (cross_attn): MultiHeadAttention(
        (W_q): Linear(in_features=512, out_features=512, bias=True)
        (W_k): Linear(in_features=512, out_features=512, bias=True)
        (W_v): Linear(in_features=512, out_features=512, bias=True)
        (W_o): Linear(in_features=512, out_features=512, bias=True)
      )
      (feed_forward): PositionWiseFeedForward(
        (fc1): Linear(in_features=512, out_features=2048, bias=True)
        (fc2): Linear(in_features=2048, out_features=512, bias=True)
        (relu): ReLU()
      )
      (norm1): LayerNorm((512,), eps=1e-05, elementwise_affine=True)
      (norm2): LayerNorm((512,), eps=1e-05, elementwise_affine=True)
      (norm3): LayerNorm((512,), eps=1e-05, elementwise_affine=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
  )
  (fc): Linear(in_features=512, out_features=5000, bias=True)
  (dropout): Dropout(p=0.1, inplace=False)
)
```

4.2. Usage Growth and Abstract Analysis

The main problems of the Information Explosion registered in the bibliometric context during the last decades is that Scholarly Visibility and efficient searches are largely determined by the keyword selection criteria used by the authors. The success, precision, and adequacy of the choice of these descriptors with respect to the content of published scientific articles represent a matter of great importance for researchers [50].

In order to extend the analytical implications beyond keyword processing, previous studies have addressed a detailed analysis of the title or abstract [41], however, traditional methods applied this analysis manually, which makes it difficult to process large numbers of articles and, consequently, to understand the internal structure of the global academic debate on topics of broad or transversal scope.

Abstract analysis is similar to bibliometric keyword analysis. Unlike the keywords, the abstract offers more information about the evaluated research [114]. By their nature, keywords are limited, both for authors (author keywords) and for journals (keyword plus®). As a consequence, bibliometric analyzes based solely on keywords can incorporate biases that can cause Type II Errors.

At times, abstract analysis has been an unassisted prior conceptualization mechanism [102,121] used by researchers to facilitate their decision-making to help guide the subsequent bibliometric analysis carried out with specific text mining tools [36,142]. Abstract analysis has been used as a content evaluation mechanism in areas such as business and management [121] or project management [93], among others, and has been applied as an inductive technique [121], registering important limitations on the possibilities of automation and the implications of the findings obtained.

This study uses custom software that allows the automated analysis of the abstracts of the evaluated articles (N=75527), offering an important evolution that improves the process and the actual analysis of the content, overcoming the reported limitations and allowing a robust and rigorous analysis of the academic literature. The dataset was obtained from Web of Science, indexed sources.

A Bag of Words (BOW) was created on all the abstracts from the 75527 articles analyzed. A dataset with the most common English words [9], from the Google Web Trillion Word Corpus, was used to eliminate the first 500 words from the BOW, as they provided little value to the analysis since they were not directly related to NLP. Tab 1 was generated with some words directly related to particular models, architectures, or approaches.

This dataset will be made publicly available, with a caveat on the distribution of the abstracts. A SHA256 hash of the abstract will be provided along with other data, such as authors, DOI, title, and publication date. The raw abstracts will not be shared due to concerns of licensing and copyright holding, as not all of the papers have open access [25]. The BERT and SetFit models are also available [24].

Data from Tab 1 comes from all the NLP related articles that had an abstract available, spanning 6 decades of research. If we select only the papers published in the last 5 years, a different picture is obtained. As expected the usage of previous techniques falls down as time progresses, but even then, some of the usages of more recent techniques, like LSTM, have not changed as fast as transformers based models. BERT, transformers, pre-trained, encoder, embedding, bidirectional, and deep are some of the terms that saw a significant change in usage.

[9] Dataset taken from https://www.kaggle.com/datasets/rtatman/english-word-frequency
[10] Since BERT was introduced in 2018 it had 0 mentions

Table 1. Models

Word	Usage	Word	Usage	Word	Usage
All time usage					
neural	16324	random	3052	transformer	2047
deep	12581	pretrained	3034	rulebased	1951
networks	9568	recurrent	3011	fuzzy	1904
attention	8274	latent	3004	linear	1901
vector	6563	cnn	2967	regression	1894
statistical	5569	lstm	2846	encoder	1808
rules	5546	ner	2746	wordnet	1744
embeddings	5404	means	2554	endtoend	1693
graph	5365	inference	2497	naive	1601
graphs	5365	bidirectional	2417	word2vec	1480
embedding	5031	multimodal	2280	clusters	1476
clustering	3971	svm	2224	bayes	1468
convolutional	3518	pos	2201	markov	1436
bert	3195	sequences	2107		
Usage before 2018					
neural	5522	pos	1397	embedding	903
graphs	4397	wordnet	1397	clusters	884
statistical	4147	svm	1384	bayes	874
rules	4049	fuzzy	1352	lstm	833
networks	3820	linear	1247	cnn	819
vector	3578	inference	1234	regression	727
deep	3103	markov	1173	bidirectional	464
clustering	2454	recurrent	1169	word2vec	431
attention	2114	rulebased	1156	endtoend	419
graph	1803	sequences	1126	pretrained	184
embeddings	1694	ner	1078	encoder	145
means	1669	multimodal	962	transformer	21
latent	1572	naive	960	bert	0
random	1523	convolutional	916		
Usage after 2018					
neural	10802	bidirectional	1953	sequences	981
deep	9478	recurrent	1842	graphs	968
attention	6160	ner	1668	means	885
networks	5748	encoder	1663	svm	840
embedding	4128	random	1529	pos	804
embeddings	3710	clustering	1517	rulebased	795
graph	3562	rules	1497	linear	654
bert	3195	latent	1432	naive	641
vector	2985	statistical	1422	bayes	594
pretrained	2850	multimodal	1318	clusters	592
convolutional	2602	endtoend	1274	fuzzy	552
cnn	2148	inference	1263	wordnet	347
transformer	2026	regression	1167	markov	263
lstm	2013	word2vec	1049		
Change in usage after 2018					
bert	NA[10]	graph	1.976	clusters	0.67
transformer	96.476	neural	1.956	naive	0.668
pretrained	15.489	regression	1.605	clustering	0.618
encoder	11.469	recurrent	1.576	svm	0.607
embedding	4.571	ner	1.547	pos	0.576
bidirectional	4.209	networks	1.505	means	0.53
deep	3.054	multimodal	1.37	linear	0.524
endtoend	3.041	inference	1.024	fuzzy	0.408
attention	2.914	random	1.004	rules	0.37
convolutional	2.841	latent	0.911	statistical	0.343
cnn	2.623	sequences	0.871	wordnet	0.248
word2vec	2.434	vector	0.834	markov	0.224
lstm	2.417	rulebased	0.688	graphs	0.22
embeddings	2.19	bayes	0.68		

Some of the terms selected are very similar, like "embedding" and "embeddings", or refer to similar architectures like "recurrent" and "lstm". We can group and lemmatize those terms and obtain a clearer picture. 4 groups were created based on the similarity of the words included in them, The groups will include the following terms:

- Group 1: bert, transformer, pretrained, encoder, decoder, attention, gpt
- Group 2: lstm, cnn, recurrent, convolutional, siamese
- Group 3: svm, means, bayes, random, graph, mean, regression, logistic, linear, vector, markov
- Group 4: fuzzy, rules, rulebased, valence, vader, pattern, matching, statistical, feature

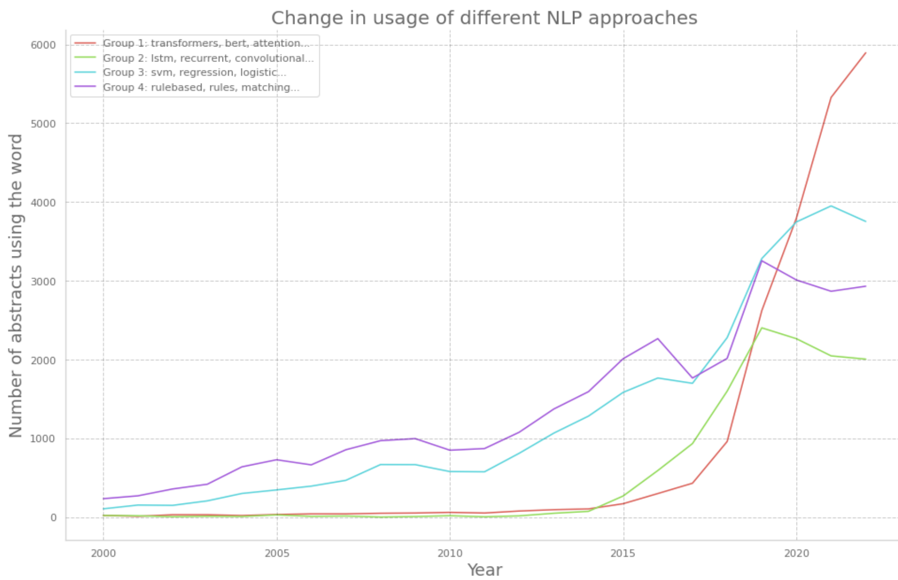

Fig. 3. Change in usage of different NLP approaches

It can be seen in fig 3 that the group related to transformers has had a massive growth in the past years, while the usage of other approaches seems to be slowing down. Further work can be done to take into account all the words and possible clusters from different NLP approaches, and there are limitations to this section, but this serves as another point to understand the impact that transformers, attention models, and BERT have had in the NLP research space.

We then propose the year 2018 as a turning point in NLP research. Even though transformers were proposed a year prior, in 2018 a paper was published, in which BERT was introduced [43]. BERT is a model, based on the transformer architecture that uses attention mechanisms to learn contextual relations between tokens to generate language understanding. Following the benefits of transformers, BERT is trained on large datasets of unlabeled text, using self-supervised training, and is then fine-tuned for specific NLP tasks. It introduced some novelties to transformers, such as being able to extract deep bidirectional representations, meaning that tokens before and after would affect the meaning of a token being paid attention to. BERT had 2 original variations, $\text{BERT}_{\text{BASE}}$ and $\text{BERT}_{\text{LARGE}}$,

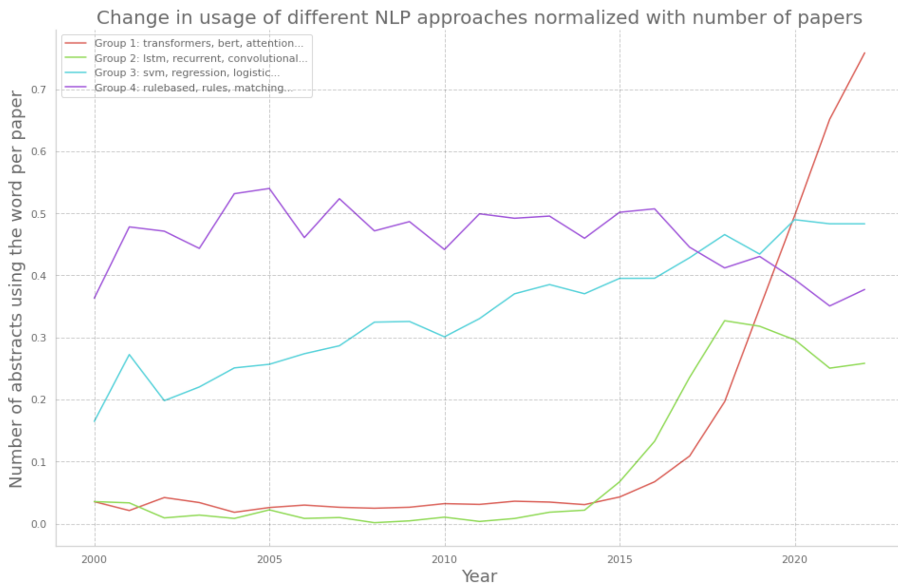

Fig. 4. Change in usage of different NLP approaches normalized with papers per year

where the main difference is the size of the model. $\text{BERT}_{\text{BASE}}$ had 12 layers, 12 attention heads, a total of 110M parameters, and a hidden layer size of 768. $\text{BERT}_{\text{LARGE}}$ had 24 layers, 16 attention heads, a total of 340M parameters, and a hidden layer size of 1024. The output of $\text{BERT}_{\text{BASE}}$ contains an embedding for every token in the input, with an embedding size of 768, and the output of $\text{BERT}_{\text{LARGE}}$ also contains an embedding for every token, with an embedding size of 1024. A trade-off has to be made when choosing what models to use for solving tasks, and in the case of the 2 BERT variations the trade-off is performance for computational expense. Even with transformers being so relatively efficient, bigger models appear to achieve better results, while also being computationally more expensive. In general, $\text{BERT}_{\text{LARGE}}$ is expected to achieve better performance than $\text{BERT}_{\text{BASE}}$ on tasks that require more fine-grained modeling of the input data, but at the cost of requiring more energy and memory to use.

he BERT model, in conjunction with the transformer architecture, has seemingly created a new wave of research within the field of NLP. As illustrated in Table 1, BERT has been cited 3195 times since its introduction in 2018, demonstrating greater relevance than other models like LSTMs or CNNs, which were proposed several years earlier. It is noteworthy that the data analyzed predominantly originates from recent publications, as depicted in Figure 5. Yet, this does not detract from the conclusion that BERT has significantly impacted NLP usage and research. BERT, alongside transformer and attention-based models, have established themselves as pivotal architectures. The trend in the adoption of these terms, as shown in Figures 6, 7, and 8, indicates that their usage has either increased or remained steady since their inception. A positive value in the graph indicates that the usage of the word increased, while a negative value indicates that the usage of the word decreased. A value of 0 in the graph indicates that the change remained the same for that period. It is clear that research papers implementing the BERT model inherently use transformer and attention mechanisms, as BERT itself is built upon these architectures.

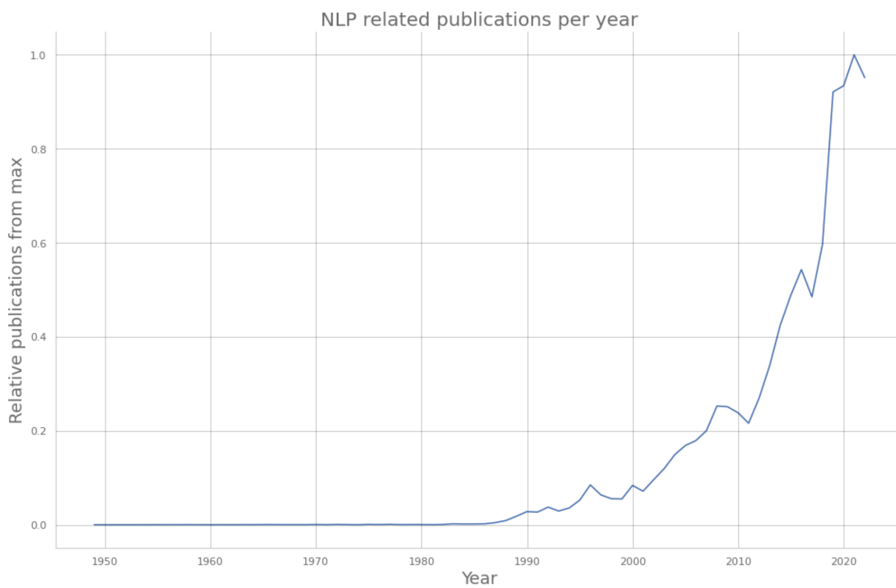

Fig. 5. NLP related publications per year

Analyzing the times BERT was mentioned in the articles since its introduction in 2018 in Table 3 furthers the understanding of its impact on the research community. It can be observed that the usage, or at least the research related to BERT, is at its highest point. These mentions include variations on the original BERT model, such as DistilBERT [116], BioBERT [83] or SciBERT [13]. BERT's versatility and its utilization of the transformer architecture and pre-training, fine-tuning approach make it accessible to a wide range of teams. This accessibility is facilitated by the availability of high-performance computing hardware or cloud-based computing services. These factors contribute to the ability of researchers to achieve state-of-the-art performance on a variety of natural language processing tasks, without the need for an excessive amount of labeled data or access to specialized computing resources.

Table 2. Number of articles mentioning BERT per year

Year	Papers mentioning BERT
2019	150
2020	434
2021	778
2022	802

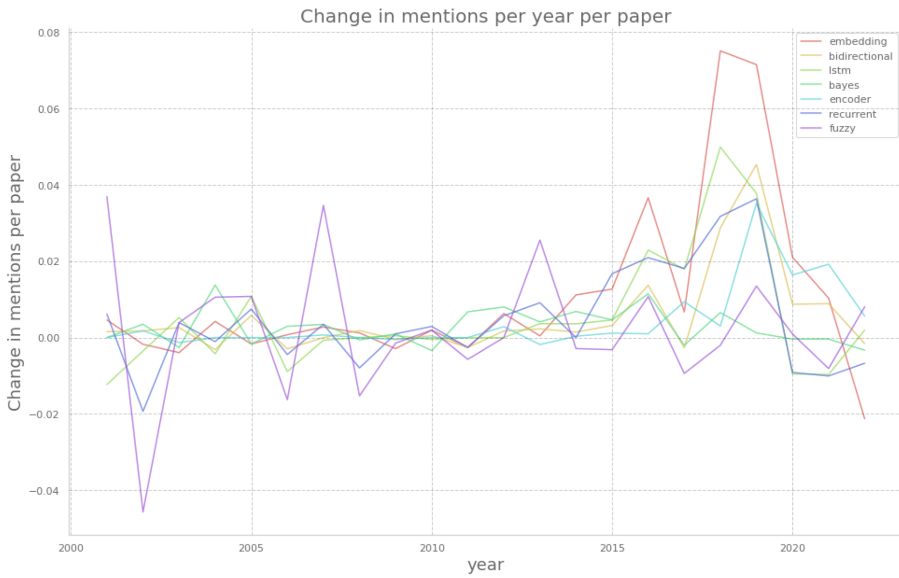

Fig. 6. Change in word usage per paper per year. A positive change means the word had more usage in the year prior by y per paper

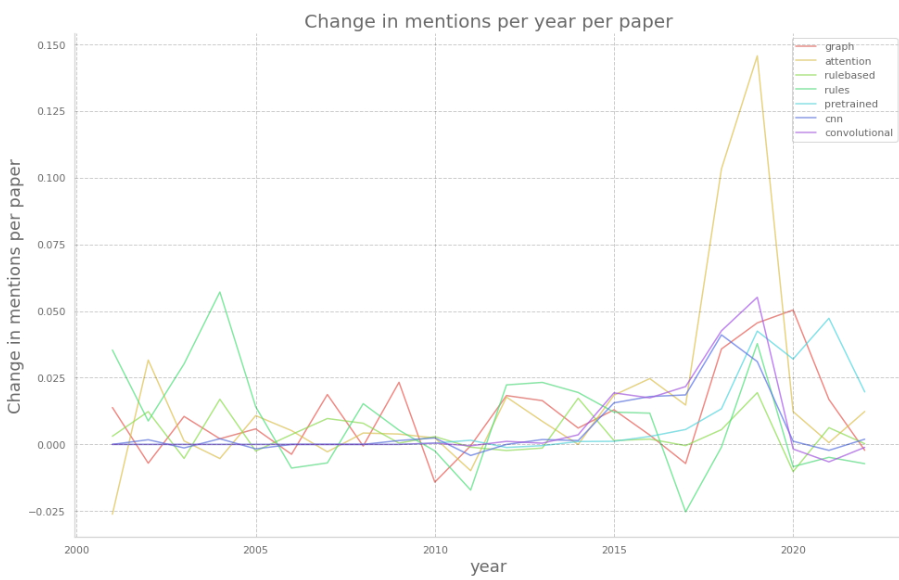

Fig. 7. Change in word usage per paper per year

A combination of these factors appears to have made BERT a good option for researchers. It provided state-of-the-art results in most NLP tasks, while only requiring

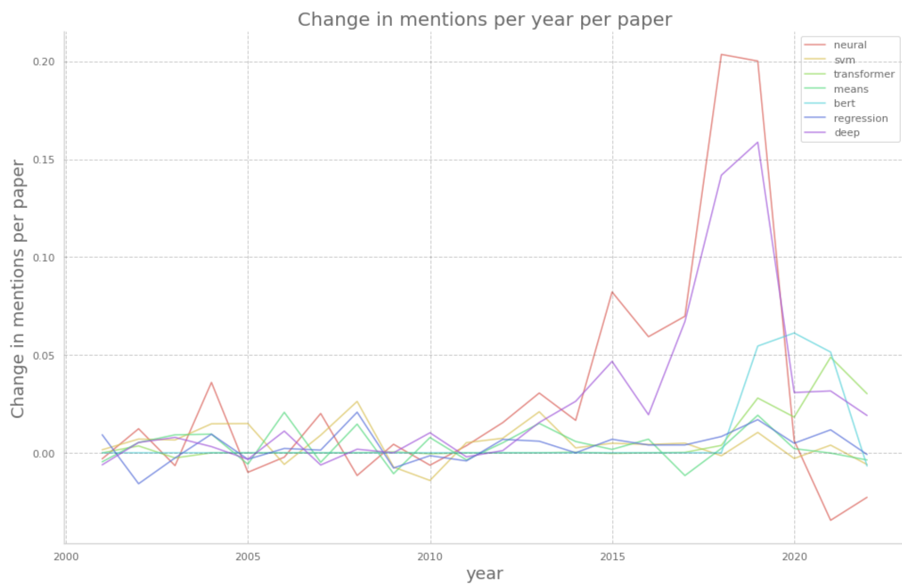

Fig. 8. Change in word usage per paper per year

comparatively small datasets for fine-tuning while being computationally cheap enough to be able to be run on smaller specialized hardware or on cloud computing services. The open-sourcing of the whole pre-trained model was also a likely factor for its widespread usage. Summarizing the reasons why researchers have been attracted to BERT:

- State-of-the-art performance: BERT has achieved state-of-the-art performance on a wide range of NLP tasks, outperforming previous models on benchmarks such as the GLUE and SQuAD datasets.
- Pre-training and fine-tuning: BERT's architecture is designed to be pre-trained on large amounts of unlabeled text data and then fine-tuned on smaller labeled datasets for specific tasks. This approach allows researchers to leverage the information contained in the pre-trained model, reducing the amount of labeled data needed for fine-tuning, and the computational cost of training a model from scratch.
- Versatility: BERT can be fine-tuned for a wide range of NLP tasks, including text classification, NER, and question answering, by simply changing the architecture of the final layer of the model.
- Bidirectionality: BERT's architecture uses a bidirectional transformer, which means that the model takes into account the context from both the left and the right side of a given word when generating embeddings. This allows improved performance in tasks where understanding the context is crucial, such as question answering.
- Large model size: BERT models are large, with hundreds of millions of parameters, which gives them the ability to model complex patterns and relationships within the text. This makes them suitable for handling large and complex datasets.
- Accessibility: BERT models are widely available, and pre-trained models, along with the fine-tuning code, can be easily accessed and used in many NLP tasks.

Even though BERT has had a big impact on the research community, other models with other architectures, pre-training datasets, training objectives, sizes, and overall design changes have been proposed. Those models achieve better performances in different tasks and may be more suitable for different situations and objectives.

4.3. Further Analysis of Abstracts

An experiment to show the capabilities of different models and approaches was conducted. A task was proposed to showcase how one might start to solve an NLP task, and what benefits and trade-offs the different models might provide in terms of performance, cost, and training data required. The training, and following labeling of abstracts, can also be used as further proof of the impacts transformers have had in the NLP field. An increase in papers using DNNs, especially the transformer architecture, is evident when looking at the labeled abstracts through time.

A more in-depth analysis of the abstracts was conducted to further understand the change in usage of different NLP techniques. All the abstracts were categorized into 5 different categories, based on which model or technique they used in the study. Four approaches were proposed for the categorization. The categories are:

- 0) The abstract does not mention a particular model or technique. Some papers analyzing frameworks, surveys, papers centered on the computer vision component of NLP, and dataset proposals among others fall into this category.
- 1) A model based on rules or symbolic analysis is used.
- 2) An approach using statistical methods is used. This includes BoWs, N-Grams, and TF-IDF, along with other machine learning techniques like SVMs, Logistic Regression, LDA, and others. Shallow neural network models like word2vec also belong in this category.
- 3) Approaches that use Deep Learning and other Deep Neural Network architectures such as RNNs, CNNs, and LSTM are included in this category.
- 4) The approach proposed uses transformer based models, like BERT, GPT, T5, and others. Although transformers are a type of DNN, they will be differentiated from other architectures included in category 3.

Since an abstract can contain conflicting categories, meaning that it could, for example, include both categories 3 and 4, the highest category is to be selected. Most abstracts would fall into category 1, as most of them include some form of rule based preprocessing. All of the abstracts in Category 4 would also belong in Category 3, as transformers are a subset of DNNs, and all DNNs would also fall into Category 2 as they use statistical methods in training and in architecture design. The classification of the abstracts was not done in a definitive manner, it was done to revise the capabilities of different models and their ability to adapt to the classification criteria that we proposed.

A small dataset of 384 labeled abstracts was created to train and test all the models. A split of 250 abstracts was dedicated to training, while the other 134 abstracts were used for testing. The distributions of the training dataset can be seen in Fig 9:

TF-IDF approach. A vector representation of the abstracts was created using TF-IDF. A K-nearest neighbors model was trained using 250 of the labeled abstracts. Afterwards, all the 75527 abstracts were categorized into one of the 5 categories and organized by year.

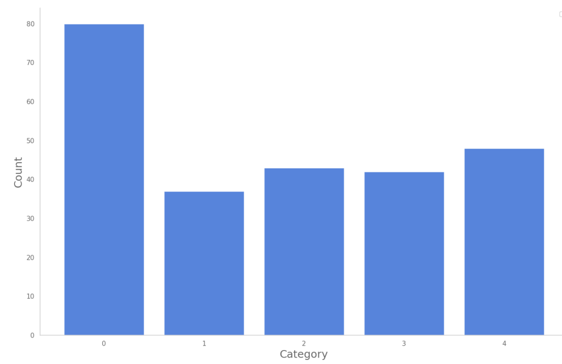

Fig. 9. Distribution of training dataset

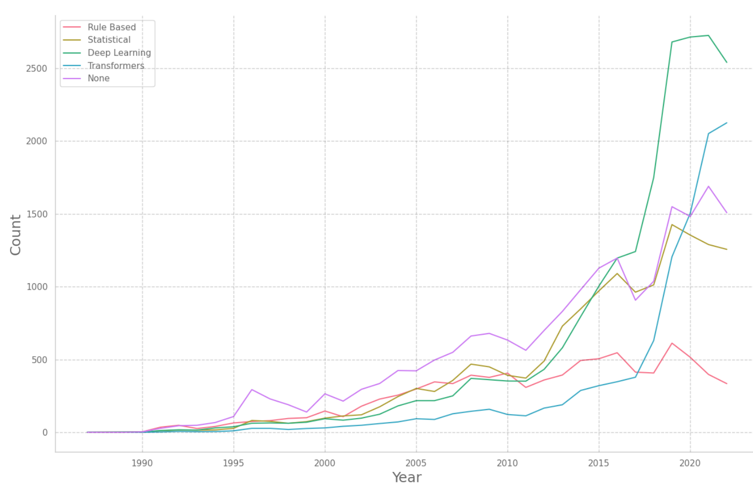

Fig. 10. TF-IDF KNN Predictions per Year

BERT model approach. A BERT based model was fine-tuned using 250 labeled abstracts. The fine-tuning used a loss to categorize each abstract into one of the 5 categories. Afterwards, all the 75527 abstracts were categorized into one of the 5 categories using the fine-tuned model.

The results immediately give BERT an edge, as any paper categorized in Category 4 before 2017 would have been miscategorized. BERT also obtained a significantly higher accuracy on the testing dataset. Another model like RoBERTa, or even some encoder-decoder models like FLAN-T5 could achieve better results with the same training param-

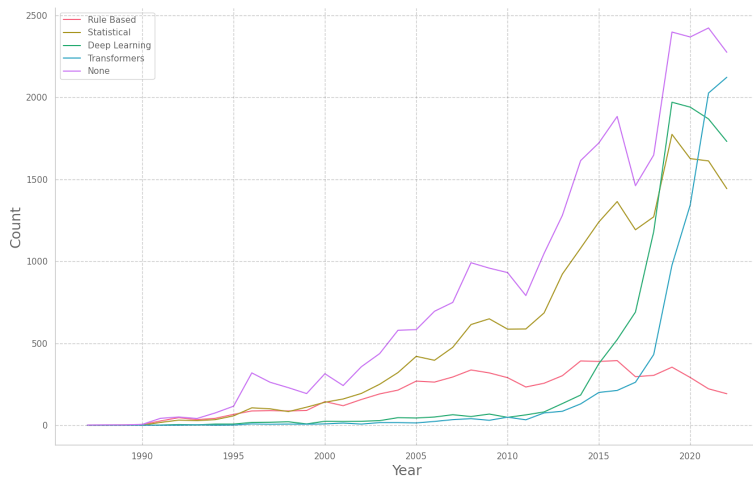

Fig. 11. BERT Predictions per Year

eters and resources, other than computer power required, but BERT was used as a baseline for a fine-tuning approach.

SetFit few-shot approach. Sentence Transformer Fine-tuning (SetFit) is a framework for few-shot fine-tuning, using contrastive learning, Pattern Exploiting Training (PET), and Parameter Efficient Fine-Tuning (PEFT) training [139]. Using Sentence [111] MP-Net [127] [11] as the base model, the training was done on only 15 labeled examples per category, for a total of 75 labeled examples, in comparison to the 250 labeled examples of the previous approaches.

LLM few-shot approach. A Llama-2-70b model was used to categorize all the testing dataset abstracts. Not all the 75527 abstracts were categorized, as using this model is by far the most computationally expensive approach. A prompt was created using only one abstract per category and then passed to the LLama2-70b model along with the new abstract to categorize.

This approach was the least time consuming in human hours, as only 5 abstracts in total were categorized for the few-shot prompting. This approach also obtained similar accuracy on the test dataset compared to BERT.

Analysis Results. The results of this analysis of the abstracts reinforces the argument for the transformers architecture being an inflection point in NLP. The tradeoffs of the three different are more evident in this analysis. The TF-IDF approach was the less performant,

[11] https://huggingface.co/sentence-transformers/paraphrase-mpnet-base-v2

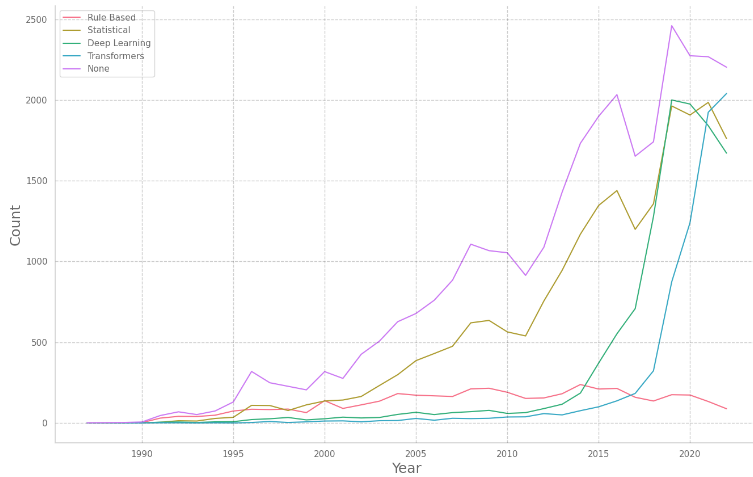

Fig. 12. SetFit Predictions per Year

but also the less computationally expensive. The BERT approach was a good balance between computer expense and performance. The few-shot Llama-2-70b approach was the most computationally expensive, but also obtained a good performance, similar to the one of BERT, and did not require extensive labeling of abstracts. On a classification task that has categories that are so loosely defined, it makes sense that models that are good at reasoning, like Llama-2, will have a good performance by default. Some examples of the answers Llama-2 gave are included 9. The dataset included [25] has the classification given by the models tested. Note that the classification should be used as a starting point for further use, as it may not be highly accurate.

Table 3. Different approaches on text classification

Approach	Accuracy	Labeled Samples	Computer power - Device Used
TF-IDF / KNN	0.4701	250	low - cpu
BERT	0.7164	250	medium - NVIDIA V100
SetFit	0.5970	75	medium - NVIDIA V100
Llama 2	0.7089	5	high - 2x NVIDIA A100

With the results into consideration, the following advantages and disadvantages for the models can be outlined:

TF-IDF with KNN: This model, employing classical NLP techniques, served as a baseline to gauge performance against more traditional methods. It was the least computationally demanding and least performant, highlighting the advancements in NLP techniques over time.

BERT-based model: Representing advanced methods, BERT utilized fine-tuning to categorize abstracts achieving higher accuracy. BERT's results underscore the superiority of current state-of-the-art models and their balance of computational cost and performance.

SetFit few-shot approach: This approach demonstrated that with few-shot learning, it is possible to train models efficiently with a minimal amount of labeled data. While not as performant as BERT, it offers a middle ground for projects with limited data availability.

LLM few-shot approach: Using Llama-2-70b, this method explored the capabilities of large language models, which require significant computational resources but minimize human labor in terms of data labeling, setup, and training logistics. Despite its expense, this method competes in accuracy with BERT, showing promise for high-efficiency applications.

Each model provided unique insights, with BERT emerging as a strong all-around option for balancing cost, data requirements, and performance. Llama-2-70b showcased the potential for achieving similar accuracy with less labeled data but at a greater computational cost. The findings illustrate that the choice between models depends on the specific needs and constraints of the project at hand. Models with lower computational costs may suffice for certain tasks, while advanced models may be necessary for applications demanding high accuracy and efficiency. This experiment serves as a foundation for selecting appropriate models in NLP tasks and for understanding the ongoing advancements within the field.

All the models could improve their performance using different techniques and higher quality data, but this serves as a good baseline to contrast the advantages and disadvantages of the different approaches.

It is important to note that the technique used with Llama-2-70b, based on prompting rather than training, appears to be gaining traction, and with models that allow a greater context size and better understanding, it might show even better results in the near future.

4.4. Extra considerations for transformers

While the architecture of Transformer models is a critical determinant of their performance, other factors also play significant roles in achieving optimal results. Understanding and fine-tuning these aspects can significantly enhance a Transformer's ability to generalize and excel on a wide range of tasks

Regularization Techniques Overfitting is a common challenge with complex models, including Transformers. Regularization techniques such as dropout, label smoothing, and weight decay can be implemented to encourage the model to develop a more general representation and prevent over reliance on specific training examples.

Attention Mechanism Adjustments The attention mechanism is the heart of Transformer models. Adjusting the attention heads, exploring different types of attention (such as local, global, or sparse attention), and experimenting with the attention flow can lead to improvements in model performance and efficiency.

Tokenizers As mentioned earlier, DL models, including BERT, typically process input data that has been tokenized and represented as numerical vectors, rather than raw text. This is a common practice in NLP as it allows the model to more easily manipulate and analyze the input data. The tokenizer used for BERT was originally Wordpiece [157], a subword tokenizer, meaning that it breaks words up into smaller units called subwords or word pieces, rather than using fixed-size word tokens like many other tokenizers. Wordpiece breaks down words into smaller units based on the most frequent character sequences in the training data. It then inserts two special tokens, [CLS] and [SEP], into the input sequence. [CLS] is placed at the beginning of every input, while [SEP] is inserted between every pair of input sequences. These tokens are used by BERT to mark the start and end of sentences and to distinguish between different sentences within an input. Computational word representations and tokenizers are still a point of improvement and difference between models. Some of the more commonly used tokenizers are [157], [78] and byte pair encoding (BPE) [120]

A good tokenizer is crucial for the performance of a model, as it will dictate what are the minimum elements that the language will hold. A balance of simple objects, such as "a", "I", "am", or "is", and more complicated objects like "individual", "account" or "program" is imperative for both efficient and performant generation of tokens.

The tokens generated by the tokenizer will then be transformed into embeddings. Initially, each token gets mapped to an integer, to then be mapped as a vector representing the token on an embedding layer of the transformer. These embeddings hold the spatial information of the token, so they can be processed by the model.

The paragraphs at 9 were generated using GPT-3, and it will be used to showcase how the tokenizers of different models would split text. Each of the words separated by a space represents a token.

Datasets Datasets are a fundamental part of the pre-training and fine-tuning of models. A good, refined and rich set of training data is essential for the performance of complex models such as transformers. There is an extensive list of datasets aimed at different tasks and a variety of datasets that compile massive amounts of text from different sources. GLUE [146], a collection that includes data from other datasets, such as [149,153,126], QQP[12], STS-B[13] among others, has been popular for its variety. The Pile, an 825GiB dataset constructed on 22 subsets [47] has been used to train models like [15]. BookCorpus [169], a dataset containing a large number of data from books, used to train BERT, ALBERT, and XLNet among others. SQuAD 2.0 [110] has also been used to fine-tune models like BERT. Other relevant datasets include RedPajama [38] or RefinedWeb[105].

Benchmarks Benchmarks are among the most reliable methods for assessing the improvements a model brings. Many datasets are commonly used as benchmarks to evaluate a model's performance. Some examples are RACE [80], SQuAD 2.0, GLUE, XTREME [65], or BIG-bench [129].

It is important to note that no perfect benchmark has been developed yet, and there are questions over their reliability and usability as a tool to measure the actual performance of a model [94,49].

[12] data.quora.com/First-Quora-Dataset-Release-Question-Pairs
[13] http://ixa2.si.ehu.eus/stswiki/index.php/STSbenchmark

5. Transformers in Practice: Model Variants

5.1. Some Variations on BERT

All of the next models improve on some aspects of BERT, All of them being encoder-only models.

ALBERT: A Lite BERT achieved state-of-the-art performance on the GLUE, RACE, and SQuAD benchmarks, being better than BERT while also having fewer parameters than $\text{BERT}_{\text{LARGE}}$, making training faster and requiring less memory usage. It uses three main techniques that differ from BERT, being factorized embedding parameterization, cross-layer parameter sharing, and sentence-order prediction as an objective [82]

DistilBERT: It is a variation on the BERT model, focusing on reducing the computational power and memory required to run it while being faster and still maintaining many of BERT's capabilities. It uses a model compression technique called knowledge distillation [21], in which a model, the "student" is trained on a model, the "teacher", that will be giving a distillation loss to the "student". [116]

ELECTRA: A new method for pre-training, replacing token detection, is proposed as a more efficient alternative to previous ones, including MLM. The model proposed with this approach, ELECTRA, outperforms other models, like BERT or XLNet, using a similar model size. By predicting if a token was or not replaced, these efficiency gains can be obtained. [35]

DeBERTa: It improves on the BERT and RoBERTa architectures, by introducing an enhanced masked decoder to improve on positional data, and a disentangled attention mechanism, making it so each word is represented by two separated vectors encoding their content and positions. [60]

RoBERTa: Under the finding that BERT was undertrained, a newer model that improves on it is developed. RoBERTa was trained using more data, more training time, longer batches, removing the Next Sentence Prediction (NSP) objective, using longer sequences, and changing the masking objective dynamically. [90]

5.2. Decoder models

Decoder models, known for their auto-regressive nature, play a pivotal role in various generative tasks, thanks to their unique architecture that enables sequential generation. This auto-regressive property means that these models generate output tokens one by one, conditioned on the previously generated tokens, allowing them to capture intricate dependencies and context in the data they produce. While decoder models boast impressive capabilities, they are not without their limitations. The auto-regressive generation process can be slow, as each token's generation is dependent on previous tokens, leading to increased inference times, especially for long sequences. Additionally, this sequential generation approach can make it challenging to parallelize the inference process efficiently. Despite their limitations [87], they have demonstrated their capability to create the current state-of-the-art models. Some of these challenges are being currently researched.

Some relevant models are Chinchilla [63], which works on the scaling of Large Language Models (LLMs), and how to optimize pre-training on them, making it more efficient; GPT-3 [19], which achieves a few-shot performance comparable to those models that are fine-tuned; OPT [163], an open-source model based on GPT-3, having similar performance while being more energy efficient on the development stage; Galactica [134], a model focused on scientific tasks, achieving state-of-the-art performance on scientific benchmarks; BLOOM [156], an open access multilingual model; PaLM [32], a dense model, having a massive size of 540B parameters, to study the impact of scaling on language models; GLaM [45], that uses sparsity to scale LLMs, reaching an impressive 1.2T parameters, while still being less computationally expensive to train than GPT-3.

Most of the current state-of-the-art models fall into the decoder only architecture. It includes models like GPT4, LLama 2, Mistral 7b [72], Zephyr 7b, PaLM 2 or Claude 2 [14]

6. Results: General guidance

6.1. Performance

With models like Chinchilla [63] and Galactica [134] it can be observed that the quality and quantity of data used in pre-training plays an important role in the performance of the model. This makes it so that the size of the model, including parameters, number of attention heads, layers, and other hyper-parameters, are not the only important components when deciding on what model to use in regard to performance. A model like Galactica could be more suited to working with scientific data. Some domain-specific versions of some models have been published, including :

- SciBERT, a model based on BERT, pre-trained on a large multi-domain corpus of scientific publications to label scientific data [13].
- BioBERT, a model with largely the same architecture as BERT, pre-trained on large-scale biomedical corpora to perform text mining on biomedical text [83].
- FinBERT, a model based on BERT, trained to solve NLP tasks related to finances [7].
- LEGAL-BERT, states that the guidelines for pre-training BERT-like models may not be as suitable for the legal domain, proposing an investigation on available strategies for specialized domains. It was pre-trained on legislation, court cases, and contracts [27].
- ClinicalBERT, pre-trained on clinical data and fine-tuned to predict hospital re-admissions [66].
- DSGPT, proposing a domain-specific generative pre-training method, and then creating a model based on it, applied to E-commerce title and review summarization [164].

[14] https://www.anthropic.com/index/claude-2

The languages on which LLMs are pre-trained also have an impact on their performance. English seems to be the most common language for model releases, but some language-specific variations of some models have been created:

- Japanese: Refs [95,70,123,141] create variations on the BERT model to be able to handle different NLP tasks on the Japanese language. A variation on GPT-2 for story co-creation in Japanese was made in Ref [100]
- Spanish: Refs [3,26,145,48] create variations on BERT for the Spanish language. A model to solve the Reading Comprehension and Reasoning Explanation for Spanish was created [22] in Ref [113], using a variation on GPT-J.
- Chinese: Variations on the BERT architecture have been implemented in Ref [71,39,88].
- Multilingual: Some multilingual models have been proposed in Refs [159,161,156]

6.2. Running the Models

The idea of pre-training and then fine-tuning a model in NLP is to leverage the possible language understanding created by training the model on large datasets, and then fine-tune it to solve specific tasks. This is an approach that enables different groups that may not possess the computational power to train LLMs to still be able to use them in production or research. Although some computational power and memory will still be needed, cloud computing services provide a way to, without purchasing dedicated computing equipment and servers, be able to run the LLMs. There are some important characteristics that may indicate how much computational resources will be needed to run the models. The quantity of parameters a dense model has is generally a good way to estimate the computational resources needed to run the model: bigger models, like OPT, or BLOOM, will require more resources than GPT-Neo or GPT2.

In some cases using a smaller model may be needed, but techniques such as distillation [61,73,116], or module substitution [158] among others [53,57,130], can provide similar performance while keeping the model small, like in the case on DistilBERT and BERT. Although the size of the state-of-the-art models seems to growing [109], most likely faster than the growth in computational power, even considering High Performance Computing (HPC) and distributed computing [144,67,97], smaller models still can achieve good results on most NLP tasks.

6.3. Finding newer models

As mentioned prior, newer models have some limitations as to who can train and use them. As it seems, big groups, be it open-source communities, corporations, or institutions, are the ones creating the new state-of-the-art models. Most of these organizations have communication channels and blogs [15], along with other regular publications from AI research centers or experts.

[15] https://huggingface.co/blog, https://ai.googleblog.com/, https://www.eleuther.ai/, https://ai.facebook.com/blog/, https://bigscience.huggingface.co/blog, https://www.deepmind.com/, https://news.microsoft.com/source/topics/ai/

Model (Including variations)	Parameters	Hidden State Size	Layers	Attention Heads	Main association	Availability[16]	Paper publication date [17]
Encoder Architecture							
BERT	110M - 340M	768 - 1024	12 - 24	12 - 16	Google	Open - Model versions available	2018-10-11
ALBERT	11M - 223M	768 - 4096	12 - 24	12 - 64	Google	Open - Model versions available	2019-09-26
DistilBERT	66M	768	6	12	HuggingFace	Open - Model versions available	2019-10-02
ELECTRA	110M - 335M	256 - 1024	12 - 24	4 - 16	Google	Open - Model versions available	2020-03-23
RoBERTa	125M - 355M	768 - 1024	12 - 16	12 - 16	Meta	Open - Model versions available	2019-07-26
LUKE	483M	1024	24	16	Studio Ousia	Open - Model versions available	2020-10-02
Longformer	149M-435M	768-1024	12-24	12-16	Allen Institute	Open - Model versions available	2020-04-10
Decoder Architecture							
LLaMa	6.7B - 65.2B	4096 - 8192	32 - 80	32 - 64	Meta	Available under non commercial, research license	2023-02-27
LLaMa 2 [18]	7B - 70B	4096 - 8192	32 - 80	32 - 64	Meta	Open - Model Versions Available	2023-07-18
GPT	110M	768	12	12	OpenAI	Open - Model versions available	2018-06-10
GPT2	117M - 1542M	768 - 1600	12 - 48	12 - 25	OpenAI	Open - Model versions available	2019-07-28
GPT3	125M - 175B	768 - 12288	12 - 96	12 - 96	OpenAI	Closed - Access via API	2020-05-28
GPT-J	6B	4096	28	16	EleutherAI	Open - Model versions available	2021-07-07
GPT-Neo	125M - 2.7B	2048	24	16	EleutherAI	Open - Model versions available	2021-05-21
GPT-NeoX-20B	20B	6144	44	64	EleutherAI	Open - Model versions available	2022-04-14
OPT	125M - 175B	768 - 12288	12 - 96	12 - 96	Meta	Open - Model versions available	2022-05-02
Chinchilla	70M	8192	80	64	DeepMind	Closed	2022-05-29
LaMDA	2B - 137B	2560 - 8192	10 - 64	40 - 128	Google	Closed-Architecture implementation available	2022-06-20
Galactica	125M - 120B	768 – 10240	12 - 96	12 - 80	Meta	Open - Model versions available	2022-11-16
BLOOM	550M - 176B	1024 - 14336	24 - 70	16 - 112	BigScience	Open - Model versions available	2022-11-09
PaLM	8B - 540B	4096 - 18432	32 - 118	16 - 48	Google	Closed-Architecture implementation available	2022-04-05
GlaM	1.9B - 1.2T[19]	3072 - 65536	12 - 64	12 - 128	Google	Closed	2021-12-13
MEGATRON-TURING	530B	20480	105	128	NVIDIA-Microsoft	Closed - Codebase available	2022-01-28
GLM-130B	130B	12288	70	96	Tsinghua U	Open – Model versions available	2022-10-05
FLAN	137B	8192	64	128	Google	Closed	2021-09-03
FLAN-PaLM	8B-540B	4096-18432	32-118	16-48	Google	Closed	2022-04-05
Falcon	180B	14848	80	64	TII [20]	Open – Model versions available	2023-10-12
Encoder-Decoder Architecture							
XLNet	110M - 340M	768 - 1024	12 - 24	12 - 16	CMU, Google	Open - Model versions available	2019-06-19
T5	60M - 11B	512 - 1024	6 - 24	8 - 128	Google	Open - Model versions available	2019-10-23
Tk-Instruct	60M - 11B	512 - 1024	6 - 24	8 - 128	Various Institutions [21]	Open - Model versions available	2022-04-16
FLAN	60M - 11B	512 - 1024	6 - 24	8 - 128	Google	Open - Model versions available	2021-09-03

7. Future

In the past five years, significant advancements have been made in the field of NLP. There are different directions that newer models seem to be taking to improve performance and computational expense. Making bigger models appears to be a good way to improve performance, but the computational cost makes it so those models use more energy and take longer to train. For these massive models, specialized methods and hardware may be needed to train them [125,97,67].

Using bigger, cleaner datasets can be another way to improve models, without the need to make them bigger. Processing larger datasets can still signify an increase in the energy needed to pre-train the models, but the impact is smaller than increasing the size of the model [134,63].

Decoder-only, auto-regressive, generative models seem to be the new approach to achieving state-of-the-art results on many tasks, even those that were previously dominated by BERT-like, fine-tuned models. GPT-3, Galactica, and others prove that auto-regressive models appear to be few-shot learners [19,134], while also showing some emergent capabilities [150]. The drawback of these generative models is that they have been considerably larger than the encoder-only models, taking more energy and memory to utilize. Parameter growth can be seen in fig 13.

---

[16] Most open models are available at Huggingface [1] and the transformers library [154]

[17] Publications on developer blogs or other communications could have been made earlier. The date of the largest model is considered

[18] LLaMa 2 has been made open [138]

[19] GLaM has ~145M-96.6B activated parameters

[20] Technology Innovation Institute

[21] Allen Institute for AI, Univ. of Washington, Arizona State Univ., Sharif Univ. of Tech., Tehran Polytechnic, PSG College of Tech., IIT Kharagpur, Univ. of Amsterdam, UC Berkeley, Columbia Univ. , Factored AI, Govt. Polytechnic Rajkot, Microsoft Research, Stanford Univ., Zycus Infotech, Univ. of Massachusetts Amherst, National Inst. of Tech. Karnataka, TCS Research, IIT Madras, National Univ. of Singapore, Johns Hopkins Univ.

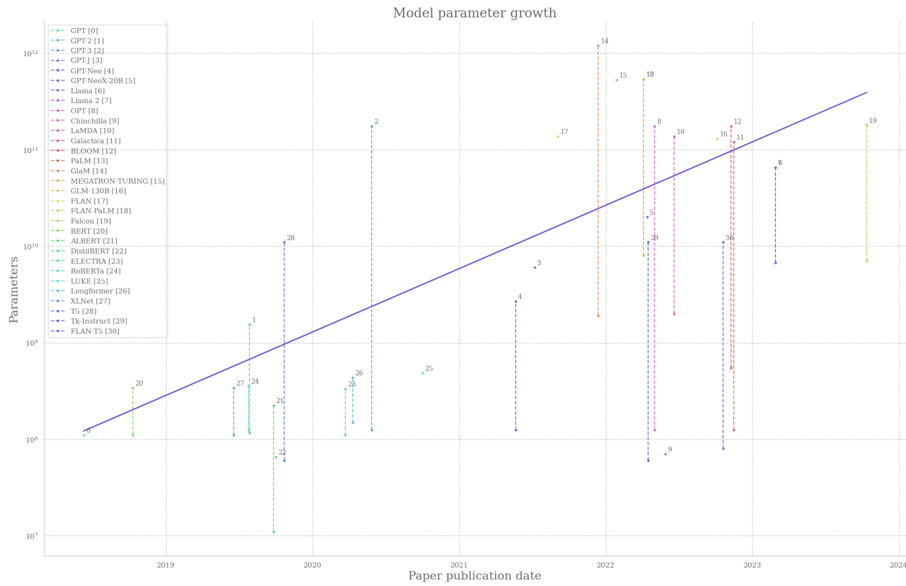

Fig. 13. Model size growth

A wave of research on model to human, consumer, interaction has been developing. Concerns over the safety and abuse of models have been studied [20,14]. LaMDA is a model that improves on this human interaction, achieving more safety and factual grounding [136]. Models like InstructGPT [101], which most likely paved the way to OpenAI's ChatGPT, have used techniques involving Reinforcement Learning (RL), specifically reinforcement learning from human feedback (RLFH) [33] to try to solve alignment issues.

Further advancements in solving the limitations of LLM usage have been made. Foundational models like LLaMa [137], which achieve good performance even with fewer parameters, that can be run on consumer hardware, have been created. Open, permissive versions of LLaMa have been replicated [52]. Improved models that use RLHF, building on LLaMa and other models have also been created [138,132,29,79,92].

Even with the efforts from open-source communities, with models like BLOOM, having direct access to massive models with billions or even trillions of parameters is difficult due to their size. Artificial Intelligence as a Service (AIaaS) may be a way to put high-performance models on production without the need to worry about multi-accelerator setups or expensive equipment. Nevertheless, researchers are conducting investigations into smaller models and compression techniques, yielding promising results and the potential to deploy LLMs on low power devices.

In recent months, there has been remarkable progress in the development of newer models that have achieved state-of-the-art performance. One such model is GPT-4 [99], which has demonstrated human-level performance on numerous benchmarks, while outperforming most other models. GPT-4 has the advantage of being multimodal, giving it extended capabilities. This model, currently available under an API, will most likely be the new standard to beat for new coming models. Not many details about its architecture have been given under official channels, so it is hard to make statements about it, which seems to be the new trend for new non open-source models, as product competitivity starts to grow.

Another notable achievement is attributed to PaLM 2 [6], representing a significant advancement in language modeling, incorporating techniques that enhance its understanding and generation of text

The open-source space of LLMs has also gained traction. Newer models like LLama 2 [138], Falcon [5], Mistral 7B [72], INCITE [22], MPT [135] have achieved promising results.

Improvements in the efficiency of training and usage of LLMs have been made. PEFT techniques such as LoRA [64], QLoRA [42], Prefix-Tuning [85], P-Tuning [89] and AdaLoRA [162] are some methods to efficiently fine-tune LLMs.

New frameworks, projects and tools have been created in order to help those that are building with NLP. LangChain [23] provides different tools to reach a production-ready app faster. LlamaIndex [24] provides ways to connect, structure and retrieve data. Haystack [25] also provides different tools to integrate LLMs in NLP apps. Vector databases and vector stores have also gained popularity to structure NLP projects and applications. Some examples include FAISS [74], pgvector for Postgres [26], Chroma [27] and Vespa [28]

Some challenging areas that require improvement in LLMs are enhancing their reasoning and factual capabilities, despite RLHF and Chain-of-Tought [151] providing a partial pathway to address these issues. Long term memory and context length are also areas in which improvements can be made. Retrieval-Augmented Generation (RAG) [84] has also been used to provide a form of memory to LLMS, and to generate text that is more informative, factually correct, and contextually relevant.

There has also been extensive, yet not conclusive, research on the different implications of these new NLP models and approaches. Scholars can develop new lines of research around ethics in AI [98], the design and supervision of algorithms [112] , according to a particular emphasis on business practices [9]. A gap that researchers must close in the future is the exploration of ethical challenges of LLMs such as GPT-4 through experiments and robust diagnostic models [170].

NLP has emerged as a significant field of research, and it continues to expand. Many organizations have already adopted, and will continue to integrate, the techniques and models discussed in this study. Consequently, it is essential to anticipate the future landscape of NLP, recognizing current trends and the insights they offer. Bearing this in mind, it is expected that NLP will be increasingly applied across various industries, with its effects becoming more evident.

[22] https://huggingface.co/togethercomputer/RedPajama-INCITE-7B-Base
[23] https://github.com/langchain-ai/langchain
[24] https://github.com/run-llama/llama_index
[25] https://github.com/deepset-ai/haystack
[26] https://github.com/pgvector/pgvector
[27] https://github.com/chroma-core/chroma
[28] https://github.com/vespa-engine/vespa

8. Conclusion

With this work, we have given some insight into the area of NLP. With this information, researchers, organizations, and individuals will be able to delve into NLP with current models, especially LLMs, and their different approaches. Further research can also be conducted in a guided way, having more context of where current NLP trends came from, and where they may be going.

It is important to note that some of the information in this paper will most likely become outdated in a short time, because, as previously discussed, the field of NLP is moving at an ever increasing speed. The general recommendations will, however, remain useful for researchers, individuals, and institutions that want to approach the field of NLP.

References

1. Hugging face - the ai community building the future., `https://huggingface.co/`
2. Agarwal, M.: An overview of natural language processing. International Journal for Research in Applied Science and Engineering Technology (IJRASET) 7(5), 2811–2813 (2019)
3. Akhtyamova, L.: Named entity recognition in spanish biomedical literature: Short review and bert model. In: 2020 26th Conference of Open Innovations Association (FRUCT). pp. 1–7. IEEE (2020)
4. Allen, J., Hunnicutt, M.S., Klatt, D.H., Armstrong, R.C., Pisoni, D.B.: From text to speech: The MITalk system. Cambridge University Press (1987)
5. Almazrouei, E., Alobeidli, H., Alshamsi, A., Cappelli, A., Cojocaru, R., Debbah, M., Goffinet, E., Heslow, D., Launay, J., Malartic, Q., Noune, B., Pannier, B., Penedo, G.: Falcon-40B: an open large language model with state-of-the-art performance (2023)
6. Anil, R., Dai, A.M., Firat, O., Johnson, M., Lepikhin, D., Passos, A., Shakeri, S., Taropa, E., Bailey, P., Chen, Z., Chu, E., Clark, J.H., Shafey, L.E., Huang, Y., Meier-Hellstern, K., Mishra, G., Moreira, E., Omernick, M., Robinson, K., Ruder, S., Tay, Y., Xiao, K., Xu, Y., Zhang, Y., Abrego, G.H., Ahn, J., Austin, J., Barham, P., Botha, J., Bradbury, J., Brahma, S., Brooks, K., Catasta, M., Cheng, Y., Cherry, C., Choquette-Choo, C.A., Chowdhery, A., Crepy, C., Dave, S., Dehghani, M., Dev, S., Devlin, J., Díaz, M., Du, N., Dyer, E., Feinberg, V., Feng, F., Fienber, V., Freitag, M., Garcia, X., Gehrmann, S., Gonzalez, L., Gur-Ari, G., Hand, S., Hashemi, H., Hou, L., Howland, J., Hu, A., Hui, J., Hurwitz, J., Isard, M., Ittycheriah, A., Jagielski, M., Jia, W., Kenealy, K., Krikun, M., Kudugunta, S., Lan, C., Lee, K., Lee, B., Li, E., Li, M., Li, W., Li, Y., Li, J., Lim, H., Lin, H., Liu, Z., Liu, F., Maggioni, M., Mahendru, A., Maynez, J., Misra, V., Moussalem, M., Nado, Z., Nham, J., Ni, E., Nystrom, A., Parrish, A., Pellat, M., Polacek, M., Polozov, A., Pope, R., Qiao, S., Reif, E., Richter, B., Riley, P., Ros, A.C., Roy, A., Saeta, B., Samuel, R., Shelby, R., Slone, A., Smilkov, D., So, D.R., Sohn, D., Tokumine, S., Valter, D., Vasudevan, V., Vodrahalli, K., Wang, X., Wang, P., Wang, Z., Wang, T., Wieting, J., Wu, Y., Xu, K., Xu, Y., Xue, L., Yin, P., Yu, J., Zhang, Q., Zheng, S., Zheng, C., Zhou, W., Zhou, D., Petrov, S., Wu, Y.: Palm 2 technical report (2023)
7. Araci, D.: Finbert: Financial sentiment analysis with pre-trained language models (2019), `https://arxiv.org/abs/1908.10063`
8. Plaza-del Arco, F.M., Molina-González, M.D., Urena-López, L.A., Martín-Valdivia, M.T.: Comparing pre-trained language models for spanish hate speech detection. Expert Systems with Applications 166, 114120 (2021)
9. Attard-Frost, B., De los Ríos, A., Walters, D.R.: The ethics of ai business practices: a review of 47 ai ethics guidelines. AI and Ethics 3(2), 389–406 (2023)

10. Backus, J.W., Bauer, F.L., Green, J., Katz, C., McCarthy, J., Naur, P., Perlis, A.J., Rutishauser, H., Samelson, K., Vauquois, B., et al.: Revised report on the algorithmic language algol 60. The Computer Journal 5(4), 349–367 (1963)
11. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate (2016)
12. Bao, T., Ren, N., Luo, R., Wang, B., Shen, G., Guo, T.: A bert-based hybrid short text classification model incorporating cnn and attention-based bigru. Journal of Organizational and End User Computing (JOEUC) 33(6), 1–21 (2021)
13. Beltagy, I., Cohan, A., Lo, K.: Scibert: Pretrained contextualized embeddings for scientific text. CoRR abs/1903.10676 (2019), http://arxiv.org/abs/1903.10676
14. Bender, E.M., Gebru, T., McMillan-Major, A., Shmitchell, S.: On the dangers of stochastic parrots: Can language models be too big? In: Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency. p. 610–623. FAccT '21, Association for Computing Machinery, New York, NY, USA (2021), https://doi.org/10.1145/3442188.3445922
15. Black, S., Leo, G., Wang, P., Leahy, C., Biderman, S.: Gpt-neo: Large scale autoregressive language modeling with mesh-tensorflow (Mar 2021), https://doi.org/10.5281/zenodo.5297715
16. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. Journal of machine Learning research 3(Jan), 993–1022 (2003)
17. Bradbury, J., Frostig, R., Hawkins, P., Johnson, M.J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., Zhang, Q.: JAX: composable transformations of Python+NumPy programs (2018), http://github.com/google/jax
18. Brown, R.W.: Linguistic determinism and the part of speech. The Journal of Abnormal and Social Psychology 55(1), 1 (1957)
19. Brown, T.B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D.M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., Amodei, D.: Language models are few-shot learners (2020), https://arxiv.org/abs/2005.14165
20. Buchanan, B., Lohn, A., Musser, M., Sedova, K.: Truth, lies, and automation. Center for Security and Emerging Technology 1(1), 2 (2021)
21. Bucilua, C., Caruana, R., Niculescu-Mizil, A.: Model compression. In: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 535–541 (2006)
22. Cabezudo, M.A.S., Diestra, D., López, R., Gómez, E., Oncevay, A., Alva-Manchego, F.: Overview of recores at iberlef 2022: Reading comprehension and reasoning explanation for spanish. Procesamiento del Lenguaje Natural 69, 281–287 (2022)
23. Cambria, E., White, B.: Jumping nlp curves: A review of natural language processing research [review article]. IEEE Computational Intelligence Magazine 9(2), 48–57 (2014)
24. Canchila, S., Meneses-Eraso, C., Casanoves-Boix, J., Cortés-Pellicer, P., Castelló-Sirvent, F.: Indexed NLP article classifier models (2023), https://doi.org/10.7910/DVN/X0FTLZ
25. Canchila, S., Meneses-Eraso, C., Casanoves-Boix, J., Cortés-Pellicer, P., Castelló-Sirvent, F.: Indexed NLP Article Metadata Dataset (2023), https://doi.org/10.7910/DVN/5YIGNG
26. Cañete, J., Chaperon, G., Fuentes, R., Ho, J.H., Kang, H., Pérez, J.: Spanish pre-trained bert model and evaluation data. Pml4dc at iclr 2020(2020), 1–10 (2020)
27. Chalkidis, I., Fergadiotis, M., Malakasiotis, P., Aletras, N., Androutsopoulos, I.: Legal-bert: The muppets straight out of law school (2020), https://arxiv.org/abs/2010.02559

28. Chen, S., Zhang, Y., Yang, Q.: Multi-task learning in natural language processing: An overview. arXiv preprint arXiv:2109.09138 (2021)
29. Chiang, W.L., Li, Z., Lin, Z., Sheng, Y., Wu, Z., Zhang, H., Zheng, L., Zhuang, S., Zhuang, Y., Gonzalez, J.E., Stoica, I., Xing, E.P.: Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality (March 2023), https://lmsys.org/blog/2023-03-30-vicuna/
30. Chomsky, N.: Three models for the description of language. IRE Transactions on Information Theory 2(3), 113–124 (1956)
31. Chomsky, N.: Systems of syntactic analysis. The Journal of Symbolic Logic 18(3), 242–256 (1953)
32. Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H.W., Sutton, C., Gehrmann, S., Schuh, P., Shi, K., Tsvyashchenko, S., Maynez, J., Rao, A., Barnes, P., Tay, Y., Shazeer, N., Prabhakaran, V., Reif, E., Du, N., Hutchinson, B., Pope, R., Bradbury, J., Austin, J., Isard, M., Gur-Ari, G., Yin, P., Duke, T., Levskaya, A., Ghemawat, S., Dev, S., Michalewski, H., Garcia, X., Misra, V., Robinson, K., Fedus, L., Zhou, D., Ippolito, D., Luan, D., Lim, H., Zoph, B., Spiridonov, A., Sepassi, R., Dohan, D., Agrawal, S., Omernick, M., Dai, A.M., Pillai, T.S., Pellat, M., Lewkowycz, A., Moreira, E., Child, R., Polozov, O., Lee, K., Zhou, Z., Wang, X., Saeta, B., Diaz, M., Firat, O., Catasta, M., Wei, J., Meier-Hellstern, K., Eck, D., Dean, J., Petrov, S., Fiedel, N.: Palm: Scaling language modeling with pathways (2022), https://arxiv.org/abs/2204.02311
33. Christiano, P.F., Leike, J., Brown, T., Martic, M., Legg, S., Amodei, D.: Deep reinforcement learning from human preferences. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 30. Curran Associates, Inc. (2017), https://proceedings.neurips.cc/paper/2017/file/d5e2c0adad503c91f91df240d0cd4e49-Paper.pdf
34. Church, K.W.: Word2vec. Natural Language Engineering 23(1), 155–162 (2017)
35. Clark, K., Luong, M., Le, Q.V., Manning, C.D.: ELECTRA: pre-training text encoders as discriminators rather than generators. CoRR abs/2003.10555 (2020), https://arxiv.org/abs/2003.10555
36. Cobo, M.J., López-Herrera, A.G., Herrera-Viedma, E., Herrera, F.: Scimat: A new science mapping analysis software tool. Journal of the American Society for Information Science and Technology 63(8), 1609–1630 (2012)
37. Collobert, R., Weston, J.: A unified architecture for natural language processing: Deep neural networks with multitask learning. In: Proceedings of the 25th International Conference on Machine Learning. p. 160–167. ICML '08, Association for Computing Machinery, New York, NY, USA (2008), https://doi.org/10.1145/1390156.1390177
38. Computer, T.: Redpajama: An open source recipe to reproduce llama training dataset (April 2023), https://github.com/togethercomputer/RedPajama-Data
39. Cui, Y., Che, W., Liu, T., Qin, B., Yang, Z.: Pre-training with whole word masking for chinese bert. IEEE/ACM Transactions on Audio, Speech, and Language Processing 29, 3504–3514 (2021)
40. Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R.: Indexing by latent semantic analysis. Journal of the American society for information science 41(6), 391–407 (1990)
41. Derrick, G., Jonkers, K., Lewison, G.: Characteristics of bibliometrics articles in library and information sciences (lis) and other journals. In: Proceedings, 17th international conference on science and technology indicators. STI. pp. 449–551 (2012)
42. Dettmers, T., Pagnoni, A., Holtzman, A., Zettlemoyer, L.: Qlora: Efficient finetuning of quantized llms (2023)
43. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. CoRR abs/1810.04805 (2018), http://arxiv.org/abs/1810.04805

44. Di Gennaro, G., Buonanno, A., Palmieri, F.A.: Considerations about learning word2vec. The Journal of Supercomputing pp. 1–16 (2021)
45. Du, N., Huang, Y., Dai, A.M., Tong, S., Lepikhin, D., Xu, Y., Krikun, M., Zhou, Y., Yu, A.W., Firat, O., Zoph, B., Fedus, L., Bosma, M., Zhou, Z., Wang, T., Wang, Y.E., Webster, K., Pellat, M., Robinson, K., Meier-Hellstern, K., Duke, T., Dixon, L., Zhang, K., Le, Q.V., Wu, Y., Chen, Z., Cui, C.: Glam: Efficient scaling of language models with mixture-of-experts (2021), https://arxiv.org/abs/2112.06905
46. Fedorova, E., Druchok, S., Drogovoz, P.: Impact of news sentiment and topics on ipo underpricing: Us evidence. International Journal of Accounting & Information Management 30(1), 73–94 (2022)
47. Gao, L., Biderman, S., Black, S., Golding, L., Hoppe, T., Foster, C., Phang, J., He, H., Thite, A., Nabeshima, N., et al.: The pile: An 800gb dataset of diverse text for language modeling. arXiv preprint arXiv:2101.00027 (2020)
48. García-Pablos, A., Perez, N., Cuadros, M.: Sensitive data detection and classification in spanish clinical text: Experiments with bert. arXiv preprint arXiv:2003.03106 (2020)
49. Gardner, M., Merrill, W., Dodge, J., Peters, M., Ross, A., Singh, S., Smith, N.A.: Competency problems: On finding and removing artifacts in language data. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. pp. 1801–1813. Association for Computational Linguistics, Online and Punta Cana, Dominican Republic (Nov 2021), https://aclanthology.org/2021.emnlp-main.135
50. Gbur Jr, E.E., Trumbo, B.E.: Key words and phrases—the key to scholarly visibility and efficiency in an information explosion. The American Statistician 49(1), 29–33 (1995)
51. Gehring, J., Auli, M., Grangier, D., Yarats, D., Dauphin, Y.N.: Convolutional sequence to sequence learning. In: Precup, D., Teh, Y.W. (eds.) Proceedings of the 34th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 70, pp. 1243–1252. PMLR (06–11 Aug 2017), https://proceedings.mlr.press/v70/gehring17a.html
52. Geng, X., Liu, H.: Openllama: An open reproduction of llama (May 2023), https://github.com/openlm-research/open_llama
53. Gordon, M.A., Duh, K., Andrews, N.: Compressing bert: Studying the effects of weight pruning on transfer learning (2020), https://arxiv.org/abs/2002.08307
54. Graves, A., Fernández, S., Schmidhuber, J.: Bidirectional lstm networks for improved phoneme classification and recognition. In: Duch, W., Kacprzyk, J., Oja, E., Zadrożny, S. (eds.) Artificial Neural Networks: Formal Models and Their Applications – ICANN 2005. pp. 799–804. Springer Berlin Heidelberg, Berlin, Heidelberg (2005)
55. Grice, H.P.: Logic and conversation. In: Speech acts, pp. 41–58. Brill (1975)
56. Grishman, R.: Information extraction: Techniques and challenges. In: Pazienza, M.T. (ed.) Information Extraction A Multidisciplinary Approach to an Emerging Information Technology. pp. 10–27. Springer Berlin Heidelberg, Berlin, Heidelberg (1997)
57. Guo, F.M., Liu, S., Mungall, F.S., Lin, X., Wang, Y.: Reweighted proximal pruning for large-scale language representation (2019), https://arxiv.org/abs/1909.12486
58. Haralick, R.M., Ripken, K.: An associative-categorical model of word meaning. Artificial Intelligence 6(1), 75–99 (1975)
59. Harris, Z.S.: Distributional structure. WORD 10(2-3), 146–162 (1954), https://doi.org/10.1080/00437956.1954.11659520
60. He, P., Liu, X., Gao, J., Chen, W.: Deberta: Decoding-enhanced bert with disentangled attention (2020), https://arxiv.org/abs/2006.03654
61. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network (2015), https://arxiv.org/abs/1503.02531
62. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation 9, 1735–80 (12 1997)

63. Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., Casas, D.d.L., Hendricks, L.A., Welbl, J., Clark, A., Hennigan, T., Noland, E., Millican, K., Driessche, G.v.d., Damoc, B., Guy, A., Osindero, S., Simonyan, K., Elsen, E., Rae, J.W., Vinyals, O., Sifre, L.: Training compute-optimal large language models (2022), https://arxiv.org/abs/2203.15556
64. Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W.: Lora: Low-rank adaptation of large language models (2021)
65. Hu, J., Ruder, S., Siddhant, A., Neubig, G., Firat, O., Johnson, M.: Xtreme: A massively multilingual multi-task benchmark for evaluating cross-lingual generalization (2020), https://arxiv.org/abs/2003.11080
66. Huang, K., Altosaar, J., Ranganath, R.: Clinicalbert: Modeling clinical notes and predicting hospital readmission (2019), https://arxiv.org/abs/1904.05342
67. Huang, Y., Cheng, Y., Bapna, A., Firat, O., Chen, M.X., Chen, D., Lee, H., Ngiam, J., Le, Q.V., Wu, Y., Chen, Z.: Gpipe: Efficient training of giant neural networks using pipeline parallelism (2018), https://arxiv.org/abs/1811.06965
68. Huang, Z., Xu, W., Yu, K.: Bidirectional lstm-crf models for sequence tagging (2015), https://arxiv.org/abs/1508.01991
69. Ignat, O., Jin, Z., Abzaliev, A., Biester, L., Castro, S., Deng, N., Gao, X., Gunal, A., He, J., Kazemi, A., Khalifa, M., Koh, N., Lee, A., Liu, S., Min, D.J., Mori, S., Nwatu, J., Perez-Rosas, V., Shen, S., Wang, Z., Wu, W., Mihalcea, R.: A phd student's perspective on research in nlp in the era of very large language models (2023)
70. Iwasaki, Y., Yamashita, A., Konno, Y., Matsubayashi, K.: Japanese abstractive text summarization using bert. In: 2019 International Conference on Technologies and Applications of Artificial Intelligence (TAAI). pp. 1–5 (2019)
71. Jia, C., Shi, Y., Yang, Q., Zhang, Y.: Entity enhanced BERT pre-training for Chinese NER. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 6384–6396. Association for Computational Linguistics, Online (Nov 2020), https://aclanthology.org/2020.emnlp-main.518
72. Jiang, A.Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D.S., de las Casas, D., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., Lavaud, L.R., Lachaux, M.A., Stock, P., Scao, T.L., Lavril, T., Wang, T., Lacroix, T., Sayed, W.E.: Mistral 7b (2023)
73. Jiao, X., Yin, Y., Shang, L., Jiang, X., Chen, X., Li, L., Wang, F., Liu, Q.: Tinybert: Distilling bert for natural language understanding (2019), https://arxiv.org/abs/1909.10351
74. Johnson, J., Douze, M., Jégou, H.: Billion-scale similarity search with GPUs. IEEE Transactions on Big Data 7(3), 535–547 (2019)
75. Kalchbrenner, N., Grefenstette, E., Blunsom, P.: A convolutional neural network for modelling sentences. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 655–665. Association for Computational Linguistics, Baltimore, Maryland (Jun 2014), https://aclanthology.org/P14-1062
76. Khurana, D., Koli, A., Khatter, K., Singh, S.: Natural language processing: State of the art, current trends and challenges. Multimedia tools and applications pp. 1–32 (2022)
77. Kim, Y.: Convolutional neural networks for sentence classification. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 1746–1751. Association for Computational Linguistics, Doha, Qatar (Oct 2014), https://aclanthology.org/D14-1181
78. Kudo, T., Richardson, J.: Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. CoRR abs/1808.06226 (2018), http://arxiv.org/abs/1808.06226
79. Köpf, A., Kilcher, Y., von Rütte, D., Anagnostidis, S., Tam, Z.R., Stevens, K., Barhoum, A., Duc, N.M., Stanley, O., Nagyfi, R., ES, S., Suri, S., Glushkov, D., Dantuluri, A., Maguire, A.,

Schuhmann, C., Nguyen, H., Mattick, A.: Openassistant conversations – democratizing large language model alignment (2023)
80. Lai, G., Xie, Q., Liu, H., Yang, Y., Hovy, E.: Race: Large-scale reading comprehension dataset from examinations (2017), https://arxiv.org/abs/1704.04683
81. Lample, G., Conneau, A.: Cross-lingual language model pretraining (2019), https://arxiv.org/abs/1901.07291
82. Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., Soricut, R.: Albert: A lite bert for self-supervised learning of language representations (2019), https://arxiv.org/abs/1909.11942
83. 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. CoRR abs/1901.08746 (2019), http://arxiv.org/abs/1901.08746
84. Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., tau Yih, W., Rocktäschel, T., Riedel, S., Kiela, D.: Retrieval-augmented generation for knowledge-intensive nlp tasks (2021)
85. Li, X.L., Liang, P.: Prefix-tuning: Optimizing continuous prompts for generation. 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). pp. 4582–4597. Association for Computational Linguistics, Online (Aug 2021), https://aclanthology.org/2021.acl-long.353
86. Liddy, E.D.: Natural language processing.(2001). Encyclopedia of Library and Information Science, 2nd Ed. NY. Marcel Decker, Inc (2001)
87. Lin, C.C., Jaech, A., Li, X., Gormley, M.R., Eisner, J.: Limitations of autoregressive models and their alternatives. arXiv preprint arXiv:2010.11939 (2020)
88. Liu, W., Fu, X., Zhang, Y., Xiao, W.: Lexicon enhanced chinese sequence labeling using bert adapter. arXiv preprint arXiv:2105.07148 (2021)
89. Liu, X., Zheng, Y., Du, Z., Ding, M., Qian, Y., Yang, Z., Tang, J.: Gpt understands, too (2021)
90. 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. CoRR abs/1907.11692 (2019), http://arxiv.org/abs/1907.11692
91. Low, J.F., Fung, B.C., Iqbal, F., Huang, S.C.: Distinguishing between fake news and satire with transformers. Expert Systems with Applications 187, 115824 (2022)
92. Luo, H., Chuang, Y.S., Gong, Y., Zhang, T., Kim, Y., Wu, X., Fox, D., Meng, H., Glass, J.: Sail: Search-augmented instruction learning (2023)
93. Machado, F., Martes, C.D.: Project management success: A bibliometric analisys. Revista de Gestão e Projetos-GeP 6(1), 28–44 (2015)
94. Mishra, S., Arunkumar, A., Bryan, C., Baral, C.: A survey of parameters associated with the quality of benchmarks in nlp (2022)
95. Mutinda, F.W., Yada, S., Wakamiya, S., Aramaki, E.: Semantic textual similarity in japanese clinical domain texts using bert. Methods of Information in Medicine 60(S 01), e56–e64 (2021)
96. Nadkarni, P.M., Ohno-Machado, L., Chapman, W.W.: Natural language processing: an introduction. J Am Med Inform Assoc 18(5), 544–551 (Sep 2011)
97. Narayanan, D., Harlap, A., Phanishayee, A., Seshadri, V., Devanur, N., Ganger, G., Gibbons, P., Zaharia, M.: Pipedream: generalized pipeline parallelism for dnn training. pp. 1–15 (10 2019)
98. Nasim, S.F., Ali, M.R., Kulsoom, U.: Artificial intelligence incidents & ethics a narrative review. International Journal of Technology, Innovation and Management (IJTIM) 2(2), 52–64 (2022)
99. OpenAI: Gpt-4 technical report (2023)

100. Osone, H., Lu, J.L., Ochiai, Y.: Buncho: Ai supported story co-creation via unsupervised multitask learning to increase writers' creativity in japanese. In: Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems. CHI EA '21, Association for Computing Machinery, New York, NY, USA (2021), https://doi.org/10.1145/3411763.3450391
101. Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C.L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P., Leike, J., Lowe, R.: Training language models to follow instructions with human feedback (2022), https://arxiv.org/abs/2203.02155
102. Park, J., Shin, H., Choi, H.J., Heo, J., Kim, W.H.: A rapid review of the use of appropriate technology in global health. Journal of Appropriate Technology 7(2), 235–247 (2021)
103. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in pytorch. In: NIPS-W (2017)
104. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library
105. Penedo, G., Malartic, Q., Hesslow, D., Cojocaru, R., Cappelli, A., Alobeidli, H., Pannier, B., Almazrouei, E., Launay, J.: The refinedweb dataset for falcon llm: Outperforming curated corpora with web data, and web data only (2023)
106. Pennington, J., Socher, R., Manning, C.: GloVe: Global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 1532–1543. Association for Computational Linguistics, Doha, Qatar (Oct 2014), https://aclanthology.org/D14-1162
107. Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L.: Deep contextualized word representations (2018), https://arxiv.org/abs/1802.05365
108. Pollock, S.: A rule-based message filtering system. ACM Trans. Inf. Syst. 6(3), 232–254 (jul 1988), https://doi.org/10.1145/45945.214327
109. Qiu, X., Sun, T., Xu, Y., Shao, Y., Dai, N., Huang, X.: Pre-trained models for natural language processing: A survey. Science China Technological Sciences 63(10), 1872–1897 (2020)
110. Rajpurkar, P., Jia, R., Liang, P.: Know what you don't know: Unanswerable questions for squad (2018), https://arxiv.org/abs/1806.03822
111. Reimers, N., Gurevych, I.: Sentence-bert: Sentence embeddings using siamese bert-networks. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics (11 2019), http://arxiv.org/abs/1908.10084
112. Rodgers, W., Nguyen, T.: Advertising benefits from ethical artificial intelligence algorithmic purchase decision pathways. Journal of business ethics 178(4), 1043–1061 (2022)
113. De la Rosa, J., Fernández, A.: Zero-shot reading comprehension and reasoning for spanish with bertin gpt-j-6b (2022)
114. Saglam, M.: Key themes in brand reputation research: a bibliometric analysis with vosviewer software. Research Journal of Business and Management 9(1), 1–12 (2022)
115. Salminen, J., Mustak, M., Corporan, J., Jung, S.g., Jansen, B.J.: Detecting pain points from user-generated social media posts using machine learning. Journal of Interactive Marketing 57(3), 517–539 (2022)
116. Sanh, V., Debut, L., Chaumond, J., Wolf, T.: Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. CoRR abs/1910.01108 (2019), http://arxiv.org/abs/1910.01108
117. Santos, I., Nedjah, N., de Macedo Mourelle, L.: Sentiment analysis using convolutional neural network with fasttext embeddings. In: 2017 IEEE Latin American Conference on Computational Intelligence (LA-CCI). pp. 1–5 (2017)

118. Self, J.: Building expert systems, edited by f. hayes-roth d.a. waterman and d.b. lenat addison-wesley, london, 1983 (£27.50). Robotica 2(2), 119–119 (1984)
119. Sells, P., Shieber, S.M., Wasow, T.: Foundational issues in natural language processing: Introduction. Mit Press (1991)
120. Sennrich, R., Haddow, B., Birch, A.: Neural machine translation of rare words with subword units (2015), `https://arxiv.org/abs/1508.07909`
121. Shah, Z.A., Khalidi, L.D.M.A.: Content analysis of abstracts of business administration and management sciences theses. Multicultural Education 7(12) (2021)
122. Shannon, C.E.: A mathematical theory of communication. ACM SIGMOBILE mobile computing and communications review 5(1), 3–55 (2001)
123. Shibayama, N., Shinnou, H.: Construction and evaluation of japanese sentence-bert models. In: Proceedings of the 35th Pacific Asia Conference on Language, Information and Computation. pp. 249–256 (2021)
124. Shortliffe, E.: Mycin: A rule-based computer program for advising physicians regarding antimicrobial therapy selection pp. 739–739 (01 1976)
125. Smith, S., Patwary, M., Norick, B., LeGresley, P., Rajbhandari, S., Casper, J., Liu, Z., Prabhumoye, S., Zerveas, G., Korthikanti, V., Zhang, E., Child, R., Aminabadi, R.Y., Bernauer, J., Song, X., Shoeybi, M., He, Y., Houston, M., Tiwary, S., Catanzaro, B.: Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model (2022), `https://arxiv.org/abs/2201.11990`
126. 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. pp. 1631–1642. Association for Computational Linguistics, Seattle, Washington, USA (Oct 2013), `https://aclanthology.org/D13-1170`
127. Song, K., Tan, X., Qin, T., Lu, J., Liu, T.Y.: Mpnet: Masked and permuted pre-training for language understanding (2020)
128. Song, P., Geng, C., Li, Z.: Research on text classification based on convolutional neural network. In: 2019 International Conference on Computer Network, Electronic and Automation (ICCNEA). pp. 229–232 (2019)
129. Srivastava, A., Rastogi, A., Rao, A., Shoeb, A.A.M., Abid, A., Fisch, A., Brown, A.R., Santoro, A., Gupta, A., Garriga-Alonso, A., Kluska, A., Lewkowycz, A., Agarwal, A., Power, A., Ray, A., Warstadt, A., Kocurek, A.W., Safaya, A., Tazarv, A., Xiang, A., Parrish, A., Nie, A., Hussain, A., Askell, A., Dsouza, A., Slone, A., Rahane, A., Iyer, A.S., Andreassen, A., Madotto, A., Santilli, A., Stuhlmüller, A., Dai, A., La, A., Lampinen, A., Zou, A., Jiang, A., Chen, A., Vuong, A., Gupta, A., Gottardi, A., Norelli, A., Venkatesh, A., Gholamidavoodi, A., Tabassum, A., Menezes, A., Kirubarajan, A., Mullokandov, A., Sabharwal, A., Herrick, A., Efrat, A., Erdem, A., Karakaş, A., Roberts, B.R., Loe, B.S., Zoph, B., Bojanowski, B., Özyurt, B., Hedayatnia, B., Neyshabur, B., Inden, B., Stein, B., Ekmekci, B., Lin, B.Y., Howald, B., Diao, C., Dour, C., Stinson, C., Argueta, C., Ramírez, C.F., Singh, C., Rathkopf, C., Meng, C., Baral, C., Wu, C., Callison-Burch, C., Waites, C., Voigt, C., Manning, C.D., Potts, C., Ramirez, C., Rivera, C.E., Siro, C., Raffel, C., Ashcraft, C., Garbacea, C., Sileo, D., Garrette, D., Hendrycks, D., Kilman, D., Roth, D., Freeman, D., Khashabi, D., Levy, D., González, D.M., Perszyk, D., Hernandez, D., Chen, D., Ippolito, D., Gilboa, D., Dohan, D., Drakard, D., Jurgens, D., Datta, D., Ganguli, D., Emelin, D., Kleyko, D., Yuret, D., Chen, D., Tam, D., Hupkes, D., Misra, D., Buzan, D., Mollo, D.C., Yang, D., Lee, D.H., Shutova, E., Cubuk, E.D., Segal, E., Hagerman, E., Barnes, E., Donoway, E., Pavlick, E., Rodola, E., Lam, E., Chu, E., Tang, E., Erdem, E., Chang, E., Chi, E.A., Dyer, E., Jerzak, E., Kim, E., Manyasi, E.E., Zheltonozhskii, E., Xia, F., Siar, F., Martínez-Plumed, F., Happé, F., Chollet, F., Rong, F., Mishra, G., Winata, G.I., de Melo, G., Kruszewski, G., Parascandolo, G., Mariani, G., Wang, G., Jaimovitch-López, G., Betz, G., Gur-Ari, G., Galijasevic, H., Kim, H., Rashkin, H., Hajishirzi, H., Mehta, H., Bogar, H., Shevlin, H., Schütze, H., Yakura, H., Zhang, H., Wong,

H.M., Ng, I., Noble, I., Jumelet, J., Geissinger, J., Kernion, J., Hilton, J., Lee, J., Fisac, J.F., Simon, J.B., Koppel, J., Zheng, J., Zou, J., Kocoń, J., Thompson, J., Kaplan, J., Radom, J., Sohl-Dickstein, J., Phang, J., Wei, J., Yosinski, J., Novikova, J., Bosscher, J., Marsh, J., Kim, J., Taal, J., Engel, J., Alabi, J., Xu, J., Song, J., Tang, J., Waweru, J., Burden, J., Miller, J., Balis, J.U., Berant, J., Frohberg, J., Rozen, J., Hernandez-Orallo, J., Boudeman, J., Jones, J., Tenenbaum, J.B., Rule, J.S., Chua, J., Kanclerz, K., Livescu, K., Krauth, K., Gopalakrishnan, K., Ignatyeva, K., Markert, K., Dhole, K.D., Gimpel, K., Omondi, K., Mathewson, K., Chiafullo, K., Shkaruta, K., Shridhar, K., McDonell, K., Richardson, K., Reynolds, L., Gao, L., Zhang, L., Dugan, L., Qin, L., Contreras-Ochando, L., Morency, L.P., Moschella, L., Lam, L., Noble, L., Schmidt, L., He, L., Colón, L.O., Metz, L., Şenel, L.K., Bosma, M., Sap, M., ter Hoeve, M., Farooqi, M., Faruqui, M., Mazeika, M., Baturan, M., Marelli, M., Maru, M., Quintana, M.J.R., Tolkiehn, M., Giulianelli, M., Lewis, M., Potthast, M., Leavitt, M.L., Hagen, M., Schubert, M., Baitemirova, M.O., Arnaud, M., McElrath, M., Yee, M.A., Cohen, M., Gu, M., Ivanitskiy, M., Starritt, M., Strube, M., Swędrowski, M., Bevilacqua, M., Yasunaga, M., Kale, M., Cain, M., Xu, M., Suzgun, M., Tiwari, M., Bansal, M., Aminnaseri, M., Geva, M., Gheini, M., T, M.V., Peng, N., Chi, N., Lee, N., Krakover, N.G.A., Cameron, N., Roberts, N., Doiron, N., Nangia, N., Deckers, N., Muennighoff, N., Keskar, N.S., Iyer, N.S., Constant, N., Fiedel, N., Wen, N., Zhang, O., Agha, O., Elbaghdadi, O., Levy, O., Evans, O., Casares, P.A.M., Doshi, P., Fung, P., Liang, P.P., Vicol, P., Alipoormolabashi, P., Liao, P., Liang, P., Chang, P., Eckersley, P., Htut, P.M., Hwang, P., Milkowski, P., Patil, P., Pezeshkpour, P., Oli, P., Mei, Q., Lyu, Q., Chen, Q., Banjade, R., Rudolph, R.E., Gabriel, R., Habacker, R., Delgado, R.R., Milliere, R., Garg, R., Barnes, R., Saurous, R.A., Arakawa, R., Raymaekers, R., Frank, R., Sikand, R., Novak, R., Sitelew, R., LeBras, R., Liu, R., Jacobs, R., Zhang, R., Salakhutdinov, R., Chi, R., Lee, R., Stovall, R., Teehan, R., Yang, R., Singh, S., Mohammad, S.M., Anand, S., Dillavou, S., Shleifer, S., Wiseman, S., Gruetter, S., Bowman, S.R., Schoenholz, S.S., Han, S., Kwatra, S., Rous, S.A., Ghazarian, S., Ghosh, S., Casey, S., Bischoff, S., Gehrmann, S., Schuster, S., Sadeghi, S., Hamdan, S., Zhou, S., Srivastava, S., Shi, S., Singh, S., Asaadi, S., Gu, S.S., Pachchigar, S., Toshniwal, S., Upadhyay, S., Shyamolima, Debnath, Shakeri, S., Thormeyer, S., Melzi, S., Reddy, S., Makini, S.P., Lee, S.H., Torene, S., Hatwar, S., Dehaene, S., Divic, S., Ermon, S., Biderman, S., Lin, S., Prasad, S., Piantadosi, S.T., Shieber, S.M., Misherghi, S., Kiritchenko, S., Mishra, S., Linzen, T., Schuster, T., Li, T., Yu, T., Ali, T., Hashimoto, T., Wu, T.L., Desbordes, T., Rothschild, T., Phan, T., Wang, T., Nkinyili, T., Schick, T., Kornev, T., Telleen-Lawton, T., Tunduny, T., Gerstenberg, T., Chang, T., Neeraj, T., Khot, T., Shultz, T., Shaham, U., Misra, V., Demberg, V., Nyamai, V., Raunak, V., Ramasesh, V., Prabhu, V.U., Padmakumar, V., Srikumar, V., Fedus, W., Saunders, W., Zhang, W., Vossen, W., Ren, X., Tong, X., Zhao, X., Wu, X., Shen, X., Yaghoobzadeh, Y., Lakretz, Y., Song, Y., Bahri, Y., Choi, Y., Yang, Y., Hao, Y., Chen, Y., Belinkov, Y., Hou, Y., Hou, Y., Bai, Y., Seid, Z., Zhao, Z., Wang, Z., Wang, Z.J., Wang, Z., Wu, Z.: Beyond the imitation game: Quantifying and extrapolating the capabilities of language models (2022), https://arxiv.org/abs/2206.04615

130. Sun, S., Cheng, Y., Gan, Z., Liu, J.: Patient knowledge distillation for bert model compression (2019), https://arxiv.org/abs/1908.09355
131. 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 - Volume 2. p. 3104–3112. NIPS'14, MIT Press, Cambridge, MA, USA (2014)
132. Taori, R., Gulrajani, I., Zhang, T., Dubois, Y., Li, X., Guestrin, C., Liang, P., Hashimoto, T.B.: Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca (2023)
133. Tarwani, K.M., Edem, S.: Survey on recurrent neural network in natural language processing. Int. J. Eng. Trends Technol 48(6), 301–304 (2017)

134. Taylor, R., Kardas, M., Cucurull, G., Scialom, T., Hartshorn, A., Saravia, E., Poulton, A., Kerkez, V., Stojnic, R.: Galactica: A large language model for science (2022)
135. Team, M.N.: Introducing mpt-7b: A new standard for open-source, commercially usable llms (2023), www.mosaicml.com/blog/mpt-7b, accessed: 2023-05-05
136. Thoppilan, R., De Freitas, D., Hall, J., Shazeer, N., Kulshreshtha, A., Cheng, H.T., Jin, A., Bos, T., Baker, L., Du, Y., Li, Y., Lee, H., Zheng, H.S., Ghafouri, A., Menegali, M., Huang, Y., Krikun, M., Lepikhin, D., Qin, J., Chen, D., Xu, Y., Chen, Z., Roberts, A., Bosma, M., Zhao, V., Zhou, Y., Chang, C.C., Krivokon, I., Rusch, W., Pickett, M., Srinivasan, P., Man, L., Meier-Hellstern, K., Morris, M.R., Doshi, T., Santos, R.D., Duke, T., Soraker, J., Zevenbergen, B., Prabhakaran, V., Diaz, M., Hutchinson, B., Olson, K., Molina, A., Hoffman-John, E., Lee, J., Aroyo, L., Rajakumar, R., Butryna, A., Lamm, M., Kuzmina, V., Fenton, J., Cohen, A., Bernstein, R., Kurzweil, R., Aguera-Arcas, B., Cui, C., Croak, M., Chi, E., Le, Q.: Lamda: Language models for dialog applications (2022), https://arxiv.org/abs/2201.08239
137. Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., Lample, G.: Llama: Open and efficient foundation language models (2023)
138. Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D., Blecher, L., Ferrer, C.C., Chen, M., Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., Fuller, B., Gao, C., Goswami, V., Goyal, N., Hartshorn, A., Hosseini, S., Hou, R., Inan, H., Kardas, M., Kerkez, V., Khabsa, M., Kloumann, I., Korenev, A., Koura, P.S., Lachaux, M.A., Lavril, T., Lee, J., Liskovich, D., Lu, Y., Mao, Y., Martinet, X., Mihaylov, T., Mishra, P., Molybog, I., Nie, Y., Poulton, A., Reizenstein, J., Rungta, R., Saladi, K., Schelten, A., Silva, R., Smith, E.M., Subramanian, R., Tan, X.E., Tang, B., Taylor, R., Williams, A., Kuan, J.X., Xu, P., Yan, Z., Zarov, I., Zhang, Y., Fan, A., Kambadur, M., Narang, S., Rodriguez, A., Stojnic, R., Edunov, S., Scialom, T.: Llama 2: Open foundation and fine-tuned chat models (2023)
139. Tunstall, L., Reimers, N., Jo, U.E.S., Bates, L., Korat, D., Wasserblat, M., Pereg, O.: Efficient few-shot learning without prompts (2022)
140. Turing, A.M.: Computing machinery and intelligence. Springer (2009)
141. Ueda, N., Kawahara, D., Kurohashi, S.: Bert-based cohesion analysis of japanese texts. In: Proceedings of the 28th International Conference on Computational Linguistics. pp. 1323–1333 (2020)
142. Van Eck, N.J., Waltman, L.: Citation-based clustering of publications using citnetexplorer and vosviewer. Scientometrics 111, 1053–1070 (2017)
143. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need (2017), https://arxiv.org/abs/1706.03762
144. Verbraeken, J., Wolting, M., Katzy, J., Kloppenburg, J., Verbelen, T., Rellermeyer, J.S.: A survey on distributed machine learning. ACM Comput. Surv. 53(2) (mar 2020), https://doi.org/10.1145/3377454
145. Vunikili, R., Supriya, H., Marica, V.G., Farri, O.: Clinical ner using spanish bert embeddings. In: IberLEF@ SEPLN. pp. 505–511 (2020)
146. Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., Bowman, S.R.: Glue: A multi-task benchmark and analysis platform for natural language understanding (2018), https://arxiv.org/abs/1804.07461
147. Wang, W., Gang, J.: Application of convolutional neural network in natural language processing. In: 2018 International Conference on Information Systems and Computer Aided Education (ICISCAE). pp. 64–70 (2018)
148. Wang, Y.A., Chen, Y.N.: What do position embeddings learn? an empirical study of pre-trained language model positional encoding (2020)
149. Warstadt, A., Singh, A., Bowman, S.R.: Neural network acceptability judgments (2018), https://arxiv.org/abs/1805.12471

150. Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., Chi, E.H., Hashimoto, T., Vinyals, O., Liang, P., Dean, J., Fedus, W.: Emergent abilities of large language models (2022)
151. Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., Zhou, D.: Chain-of-thought prompting elicits reasoning in large language models (2023)
152. Weizenbaum, J.: Eliza—a computer program for the study of natural language communication between man and machine. Commun. ACM 9(1), 36–45 (jan 1966), https://doi.org/10.1145/365153.365168
153. Williams, A., Nangia, N., Bowman, S.: 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). pp. 1112–1122. Association for Computational Linguistics, New Orleans, Louisiana (Jun 2018), https://aclanthology.org/N18-1101
154. Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Le Scao, T., Gugger, S., Drame, M., Lhoest, Q., Rush, A.M.: Transformers: State-of-the-Art Natural Language Processing. pp. 38–45. Association for Computational Linguistics (10 2020), https://www.aclweb.org/anthology/2020.emnlp-demos.6
155. Woods, W.A.: Progress in natural language understanding: An application to lunar geology. In: Proceedings of the June 4-8, 1973, National Computer Conference and Exposition. p. 441–450. AFIPS '73, Association for Computing Machinery, New York, NY, USA (1973), https://doi.org/10.1145/1499586.1499695
156. Workshop, B., :, Scao, T.L., Fan, A., Akiki, C., Pavlick, E., Ilić, S., Hesslow, D., Castagné, R., Luccioni, A.S., Yvon, F., Gallé, M., Tow, J., Rush, A.M., Biderman, S., Webson, A., Ammanamanchi, P.S., Wang, T., Sagot, B., Muennighoff, N., del Moral, A.V., Ruwase, O., Bawden, R., Bekman, S., McMillan-Major, A., Beltagy, I., Nguyen, H., Saulnier, L., Tan, S., Suarez, P.O., Sanh, V., Laurençon, H., Jernite, Y., Launay, J., Mitchell, M., Raffel, C., Gokaslan, A., Simhi, A., Soroa, A., Aji, A.F., Alfassy, A., Rogers, A., Nitzav, A.K., Xu, C., Mou, C., Emezue, C., Klamm, C., Leong, C., van Strien, D., Adelani, D.I., Radev, D., Ponferrada, E.G., Levkovizh, E., Kim, E., Natan, E.B., De Toni, F., Dupont, G., Kruszewski, G., Pistilli, G., Elsahar, H., Benyamina, H., Tran, H., Yu, I., Abdulmumin, I., Johnson, I., Gonzalez-Dios, I., de la Rosa, J., Chim, J., Dodge, J., Zhu, J., Chang, J., Frohberg, J., Tobing, J., Bhattacharjee, J., Almubarak, K., Chen, K., Lo, K., Von Werra, L., Weber, L., Phan, L., allal, L.B., Tanguy, L., Dey, M., Muñoz, M.R., Masoud, M., Grandury, M., Šaško, M., Huang, M., Coavoux, M., Singh, M., Jiang, M.T.J., Vu, M.C., Jauhar, M.A., Ghaleb, M., Subramani, N., Kassner, N., Khamis, N., Nguyen, O., Espejel, O., de Gibert, O., Villegas, P., Henderson, P., Colombo, P., Amuok, P., Lhoest, Q., Harliman, R., Bommasani, R., López, R.L., Ribeiro, R., Osei, S., Pyysalo, S., Nagel, S., Bose, S., Muhammad, S.H., Sharma, S., Longpre, S., Nikpoor, S., Silberberg, S., Pai, S., Zink, S., Torrent, T.T., Schick, T., Thrush, T., Danchev, V., Nikoulina, V., Laippala, V., Lepercq, V., Prabhu, V., Alyafeai, Z., Talat, Z., Raja, A., Heinzerling, B., Si, C., Taşar, D.E., Salesky, E., Mielke, S.J., Lee, W.Y., Sharma, A., Santilli, A., Chaffin, A., Stiegler, A., Datta, D., Szczechla, E., Chhablani, G., Wang, H., Pandey, H., Strobelt, H., Fries, J.A., Rozen, J., Gao, L., Sutawika, L., Bari, M.S., Al-shaibani, M.S., Manica, M., Nayak, N., Teehan, R., Albanie, S., Shen, S., Ben-David, S., Bach, S.H., Kim, T., Bers, T., Fevry, T., Neeraj, T., Thakker, U., Raunak, V., Tang, X., Yong, Z.X., Sun, Z., Brody, S., Uri, Y., Tojarieh, H., Roberts, A., Chung, H.W., Tae, J., Phang, J., Press, O., Li, C., Narayanan, D., Bourfoune, H., Casper, J., Rasley, J., Ryabinin, M., Mishra, M., Zhang, M., Shoeybi, M., Peyrounette, M., Patry, N., Tazi, N., Sanseviero, O., von Platen, P., Cornette, P., Lavallée, P.F., Lacroix, R., Rajbhandari, S., Gandhi, S., Smith, S., Requena, S., Patil, S., Dettmers, T., Baruwa, A., Singh, A., Cheveleva, A., Ligozat, A.L., Subramonian, A., Névéol, A., Lovering, C., Garrette, D., Tunuguntla, D., Reiter, E., Taktasheva, E., Voloshina, E., Bogdanov, E.,

Winata, G.I., Schoelkopf, H., Kalo, J.C., Novikova, J., Forde, J.Z., Clive, J., Kasai, J., Kawamura, K., Hazan, L., Carpuat, M., Clinciu, M., Kim, N., Cheng, N., Serikov, O., Antverg, O., van der Wal, O., Zhang, R., Zhang, R., Gehrmann, S., Mirkin, S., Pais, S., Shavrina, T., Scialom, T., Yun, T., Limisiewicz, T., Rieser, V., Protasov, V., Mikhailov, V., Pruksachatkun, Y., Belinkov, Y., Bamberger, Z., Kasner, Z., Rueda, A., Pestana, A., Feizpour, A., Khan, A., Faranak, A., Santos, A., Hevia, A., Unldreaj, A., Aghagol, A., Abdollahi, A., Tammour, A., HajiHosseini, A., Behroozi, B., Ajibade, B., Saxena, B., Ferrandis, C.M., Contractor, D., Lansky, D., David, D., Kiela, D., Nguyen, D.A., Tan, E., Baylor, E., Ozoani, E., Mirza, F., Ononiwu, F., Rezanejad, H., Jones, H., Bhattacharya, I., Solaiman, I., Sedenko, I., Nejadgholi, I., Passmore, J., Seltzer, J., Sanz, J.B., Dutra, L., Samagaio, M., Elbadri, M., Mieskes, M., Gerchick, M., Akinlolu, M., McKenna, M., Qiu, M., Ghauri, M., Burynok, M., Abrar, N., Rajani, N., Elkott, N., Fahmy, N., Samuel, O., An, R., Kromann, R., Hao, R., Alizadeh, S., Shubber, S., Wang, S., Roy, S., Viguier, S., Le, T., Oyebade, T., Le, T., Yang, Y., Nguyen, Z., Kashyap, A.R., Palasciano, A., Callahan, A., Shukla, A., Miranda-Escalada, A., Singh, A., Beilharz, B., Wang, B., Brito, C., Zhou, C., Jain, C., Xu, C., Fourrier, C., Periñán, D.L., Molano, D., Yu, D., Manjavacas, E., Barth, F., Fuhrimann, F., Altay, G., Bayrak, G., Burns, G., Vrabec, H.U., Bello, I., Dash, I., Kang, J., Giorgi, J., Golde, J., Posada, J.D., Sivaraman, K.R., Bulchandani, L., Liu, L., Shinzato, L., de Bykhovetz, M.H., Takeuchi, M., Pàmies, M., Castillo, M.A., Nezhurina, M., Sänger, M., Samwald, M., Cullan, M., Weinberg, M., De Wolf, M., Mihaljcic, M., Liu, M., Freidank, M., Kang, M., Seelam, N., Dahlberg, N., Broad, N.M., Muellner, N., Fung, P., Haller, P., Chandrasekhar, R., Eisenberg, R., Martin, R., Canalli, R., Su, R., Su, R., Cahyawijaya, S., Garda, S., Deshmukh, S.S., Mishra, S., Kiblawi, S., Ott, S., Sang-aroonsiri, S., Kumar, S., Schweter, S., Bharati, S., Laud, T., Gigant, T., Kainuma, T., Kusa, W., Labrak, Y., Bajaj, Y.S., Venkatraman, Y., Xu, Y., Xu, Y., Xu, Y., Tan, Z., Xie, Z., Ye, Z., Bras, M., Belkada, Y., Wolf, T.: Bloom: A 176b-parameter open-access multilingual language model (2022), https://arxiv.org/abs/2211.05100
157. Wu, Y., Schuster, M., Chen, Z., Le, Q.V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., Klingner, J., Shah, A., Johnson, M., Liu, X., Kaiser, L., Gouws, S., Kato, Y., Kudo, T., Kazawa, H., Stevens, K., Kurian, G., Patil, N., Wang, W., Young, C., Smith, J., Riesa, J., Rudnick, A., Vinyals, O., Corrado, G., Hughes, M., Dean, J.: Google's neural machine translation system: Bridging the gap between human and machine translation. CoRR abs/1609.08144 (2016), http://arxiv.org/abs/1609.08144
158. Xu, C., Zhou, W., Ge, T., Wei, F., Zhou, M.: Bert-of-theseus: Compressing bert by progressive module replacing (2020), https://arxiv.org/abs/2002.02925
159. 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 (2020), https://arxiv.org/abs/2010.11934
160. Yang, Z., Dai, Z., Yang, Y., Carbonell, J.G., Salakhutdinov, R., Le, Q.V.: Xlnet: Generalized autoregressive pretraining for language understanding. CoRR abs/1906.08237 (2019), http://arxiv.org/abs/1906.08237
161. Zeng, A., Liu, X., Du, Z., Wang, Z., Lai, H., Ding, M., Yang, Z., Xu, Y., Zheng, W., Xia, X., Tam, W.L., Ma, Z., Xue, Y., Zhai, J., Chen, W., Zhang, P., Dong, Y., Tang, J.: Glm-130b: An open bilingual pre-trained model (2022), https://arxiv.org/abs/2210.02414
162. Zhang, Q., Chen, M., Bukharin, A., He, P., Cheng, Y., Chen, W., Zhao, T.: Adaptive budget allocation for parameter-efficient fine-tuning (2023)
163. Zhang, S., Roller, S., Goyal, N., Artetxe, M., Chen, M., Chen, S., Dewan, C., Diab, M., Li, X., Lin, X.V., Mihaylov, T., Ott, M., Shleifer, S., Shuster, K., Simig, D., Koura, P.S., Sridhar, A., Wang, T., Zettlemoyer, L.: Opt: Open pre-trained transformer language models (2022), https://arxiv.org/abs/2205.01068
164. Zhang, X., Jiang, Y., Shang, Y., Cheng, Z., Zhang, C., Fan, X., Xiao, Y., Long, B.: Dsgpt: Domain-specific generative pre-training of transformers for text generation in e-commerce

title and review summarization. In: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM (jul 2021), https://doi.org/10.1145%2F3404835.3463037

165. Zhang, Y., Liang, R., Qi, Y., Fu, X., Zheng, Y.: Assessing graduate academic scholarship applications with a rule-based cloud system. In: Cheng, E.C.K., Wang, T., Schlippe, T., Beligiannis, G.N. (eds.) Artificial Intelligence in Education Technologies: New Development and Innovative Practices. pp. 102–110. Springer Nature Singapore, Singapore (2023)

166. Zhang, Y., Yang, Q.: An overview of multi-task learning. National Science Review 5(1), 30–43 (2018)

167. Zhao, J., Huang, F., Lv, J., Duan, Y., Qin, Z., Li, G., Tian, G.: Do RNN and LSTM have long memory? In: III, H.D., Singh, A. (eds.) Proceedings of the 37th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 119, pp. 11365–11375. PMLR (13–18 Jul 2020), https://proceedings.mlr.press/v119/zhao20c.html

168. Zhou, C., Sun, C., Liu, Z., Lau, F.C.M.: A c-lstm neural network for text classification (2015), https://arxiv.org/abs/1511.08630

169. Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R., Torralba, A., Fidler, S.: Aligning books and movies: Towards story-like visual explanations by watching movies and reading books (2015), https://arxiv.org/abs/1506.06724

170. Zhuo, T.Y., Huang, Y., Chen, C., Xing, Z.: Exploring ai ethics of chatgpt: A diagnostic analysis. arXiv preprint arXiv:2301.12867 (2023)

9. Appendix

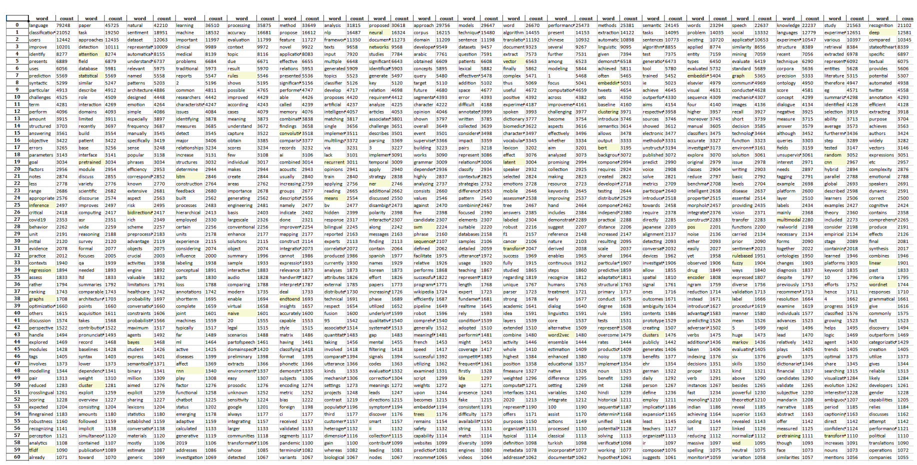

Fig. 14. Word count in abstracts

Fig. 15. Word cloud of abstracts

– BERT:

```
[CLS] a token ##izer is a program that takes a given sentence or phrase and splits it into its component
parts , or token ##s . these token ##s are typically words or sub ##words , such as individual
characters or strings of characters . for example , " ducks " might be token ##ized into " duck " and
" s . " token ##izer ##s also take into account plural forms of words , so that a sentence like " the
boys are ducks " might be token ##ized into " the " , " boy " , " s " , " are " , and " duck " ( with
the " s " representing the plural form of " boy " ) . token ##izer ##s are used in a variety of
applications , such as text processing and natural language processing . [SEP] [PAD] [PAD] [PAD] [PAD]
[PAD] [PAD] [PAD]}
```

– BLOOM:

```
<pad> <pad> <pad> ... A token izer is a program that takes a given sentence or phrase and
splits it into its component parts , or tokens . These tokens are typically words or sub
words , such as individual characters or strings of characters . For example , " duc ks "
might be token ized into " duc k" and "s . " Token izers also take into account plural
forms of words , so that a sentence like "The boys are duc ks " might be token ized
into "The " , "b oy " , " s" , " are " , and " duc k" ( with the " s" representing the
plural form of "b oy " ). Token izers are used in a variety of applications , such as
text processing and natural language processing .
```

– GPT-NeoX:

```
A token izer is a program that takes a given sentence or phrase and splits it into its
component parts , or tokens . These tokens are typically words or sub words , such as
individual characters or strings of characters . For example , " du cks " might be token
ized into " du ck " and " s ." Token izers also take into account plural forms of words ,
so that a sentence like " The boys are ducks " might be token ized into " The ", " boy
", " s ", " are ", and " du ck " ( with the " s " representing the plural form of " boy
"). Token izers are used in a variety of applications , such as text processing and
natural language processing . [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD
] [PAD] [PAD] [PAD] [PAD] [PAD]
```

– LUKE:

```
<s> A token izer is a program that takes a given sentence or phrase and splits it into its
component parts , or tokens . These tokens are typically words or sub words , such as
individual characters or strings of characters . For example , " du cks " might be token
ized into " du ck " and " s ." Token izers also take into account plural forms of words ,
so that a sentence like " The boys are ducks " might be token ized into " The ", " boy
", " s ", " are ", and " du ck " ( with the " s " representing the plural form of " boy
"). Token izers are used in a variety of applications , such as text processing and
natural language processing . </s> [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD]
[PAD] [PAD] [PAD] [PAD]
```

– XLNet:

```
A token izer is a program that takes a given sentence or phrase and split s it into its component parts , or
token s . These token s are typically words or sub word s , such as individual characters or strings
of characters . For example , " d uck s " might be token ized into " d uck " and " s . " To ken
izer s also take into account plural forms of words , so that a sentence like " The boys are duck s "
might be token ized into " The " , " boy " , " s " , " are " , and " d uck " ( with the " s "
representing the plural form of " boy " ) . To ken izer s are used in <sep> <cls>
```

9.1. Llama-2 experimenmts

The prompt used on LLama-2-70b to categorize the abstracts

```
You are a research scientist that specializes in Natural Language Processing. You are tasked with classifying
        different abstracts into one of 5 categories, depending on the newest technology or model the researchers used.
         The categories are:

0) The researchers did not explicitly mention a model or technique used, or it is not inmediately clear what they used
        .
1) Rule Based
2) Statistical
3) Neural Networks
4) Transformers
You are to use the highest category that fits the abstract (if the abstract fits in 2 and 3 you would choose 3). You
        can only choose one category.
The language used in these abstracts may be highly technical, so read every word very carefully.

Here is an example for category 3
{example abstract}

Here is an example for category 4
{example abstract}

Here is an example of category 2
{example abstract}

Here is an example for category 0
{example abstract}

Here is an example of category 1
{example abstract}

Here is an example of category 3
{example abstract}

The authors must be explicit about the model they are using, otherwise you should categorize with category 0

Here is how you should answer:
This abstract belongs to category []. Replace the [] with the category. Afterwards you should rationalize and recheck
        your answer. Also recheck if the researchers were really obvious about what they were using, or if you were
        making assumptions. finally, the last line should be what category you would put the abstract again, with the
        same format as before

Please categorize the next abstract
```

True category: 1. Llama-2 started with the correct category (1), and then miscategorized it as category 0. However, an argument could be made that Llama-2 was correct with this reasoning and the abstract was actually mislabeled.

```
This abstract belongs to category 1 (Rule-Based). The authors mention using HMM Toolkit (HTK) to construct the system,
         which suggests a rule-based approach to speech recognition. Additionally, they mention building a Mongolian
        acoustic model and collecting a large size corpus to construct the language model, which further supports a
        rule-based approach.

However, upon closer examination, it seems that the authors may have used a combination of techniques, including
        statistical modeling and machine learning, to develop their speech recognition system. Therefore, I would
        reevaluate my initial categorization and place this abstract in category 0 (The researchers did not explicitly
        mention a model or technique used), as it is unclear whether the approach used was purely rule-based or
        combined with other techniques.

Therefore, the final answer would be:
This abstract belongs to category 0.
```

True category: 3. Here Llama-2 appears to reach an incorrect conclusion based on the use of other techniques not related to the transformer architecture, classifying the abstract incorrectly.

```
Category: 3 (Neural Networks)

Rationalization: The abstract mentions the use of a neural network, specifically a reliability-based automatic repeat
        request (RB-ARQ), to improve the performance of a brain-computer interface (BCI) for communication.
        Additionally, the authors aim to combine RB-ARQ with error correction based on error-related potentials (ErrPs)
         to further enhance performance. The use of neural networks and error correction techniques suggests that the
        research falls under category 3.

Recheck: Upon rereading the abstract, I noticed that the authors actually used a combination of techniques, including
        signal averaging, RB-ARQ, and ErrP-based error correction. While the use of neural networks is still evident,
        the inclusion of other techniques means that the research may not exclusively fall under category 3. Therefore,
         I would revisit my initial answer and consider categorizing it as 4 (Transformers (BERT, GPT...)), as the
        authors seem to be combining different techniques to achieve their goals.

Final answer: After reevaluating the abstract, I believe it belongs to category 4
```

Santiago Canchila is Computer Science and Artificial Intelligence student with honorific recognition at Universidad Sergio Arboleda. He has imparted classes in Machine Learning and Blockchain Technologies. He has also built many projects and platforms with Artificial Intelligence as the main focus, with ample expertise in data processing and ML model building.

Carlos Eduardo Meneses is PhD student in Innovation and Business, Economist from the Nueva Granada Military University, Specialist in Urban and Regional Economics and Master's in Business Administration MBA from Prime Business School of the Sergio Arboleda University, candidate for a Master's in Urban and Regional Economics, PhD student in Innovation and Business and currently serves as Vice Dean of the School of Economics at the Sergio Arboleda University. Winner of the Sergio Arboleda Excellence Award in the Knowledge Management modality, passionate about the digital world, new economies and decentralized value systems, he has dedicated his research on issues related to the Collaborative Economy that leads with satisfaction to structuring and directing the International Collaborative Economy program of the Sergio Arboleda University in Colombia and Madrid Spain, as well as the opening of elective subjects such as Blockchain and Collaborative Economy at the same University and the creation of the new undergraduate program in Innovation and Economic Technology at the School of Economy. He currently teaches classes in the following fields: Professor of Business Economics Master of Business Administration, Professor of Economic Environment Analysis Specialization in Marketing Management, Professor of Strategic Analysis of Socioeconomic Environments (Bogotá, Barranquilla and Santa Marta), Collaborative Economy Teacher, Professor of Economics of Technological Platforms and Blockchain Economics teacher.

Javier Casanoves-Boix is lecturer at the Department of Marketing of the Faculty of Economics of from the Alfonso X El Sabio University (Madrid, Spain). His research attention has focused on the areas of Marketing and Branding. He has published papers in several refereed journals and editorial books. highlighting "Revista de Investigación Educativa", "International Visual Culture Review" and " UCJC Business & Society Review".

Pascual Cortés-Pellicer is an Associate Professor in the Department of Business Management of the Polytechnic University of Valencia (UPV), Spain. He has published several academic papers at internationally renowned journals and conference proceedings. His teaching and research interests include reverse logistics, sustainability, circular economy and supply chain management. In the 2021 year, He obtained the University Cel award to the best PhD thesis in the logistics field.

Fernando Castelló-Sirvent is an Economist and Associate Professor in the Department of Business Management at the Polytechnic University of Valencia (UPV), Spain. He holds a Ph.D. in Business Management from the University of Valencia. His research interests encompass artificial intelligence and its applications, sustainable transitions, strategy, crisis management, think tanks and public policy, entrepreneurship, and teaching innovation. He has extensive teaching experience in strategy, business information systems, international trade, international economics, and Spanish economics. For over two decades, he served as a senior strategy advisor to family businesses and has led numerous research and knowledge transfer projects.

Received: February 17, 2023; Accepted: January 06, 2024.