Transformer models: an introduction and catalog

Xavier Amatriain `xavier@amatriain.net`
Ananth Sankar `ansankar@linkedin.com`
Jie Bing `jbing@linkedin.com`
Praveen Kumar Bodigutla `pbodigutla@linkedin.com`
Timothy J. Hazen `thazen@linkedin.com`
Michaeel Kazi `mkazi@linkedin.com`

May 26, 2023

Abstract

In the past few years we have seen the meteoric appearance of dozens of foundation models of the Transformer family, all of which have memorable and sometimes funny, but not self-explanatory, names. The goal of this paper is to offer a somewhat comprehensive but simple catalog and classification of the most popular Transformer models. The paper also includes an introduction to the most important aspects and innovations in Transformer models. Our catalog will include models that are trained using self-supervised learning (e.g., BERT or GPT3) as well as those that are further trained using a human-in-the-loop (e.g. the InstructGPT model used by ChatGPT).

Contents

1   Introduction: What are Transformers

Transformers are a class of deep learning models that are defined by some architectural traits. They were first introduced in the now famous "Attention is All you Need" paper (and associated blog post[1]) by Google researchers in 2017 (Vaswani et al., 2017). The paper has accumulated a whopping 38k citations in only 5 years.

---

[1] https://ai.googleblog.com/2017/08/transformer-novel-neural-network.html

3

The original Transformer architecture is a specific instance of the encoder-decoder models (Cho et al., 2014)[2] that had become popular just over the 2–3 years prior. Up until that point however, attention was just one of the mechanisms used by these models, which were mostly based on LSTM (Long Short Term Memory) (Hochreiter & Schmidhuber, 1997) and other RNN (Recurrent Neural Networks) (Mikolov et al., 2010) variations. The key insight of the Transformers paper, as the title implies, was that attention could be used as the only mechanism to derive dependencies between input and output.

The input to the Transformer is a sequence of tokens. The output of the encoder is a fixed-dimensional representation for each of the tokens along with a separate embedding for the sequence as a whole. The decoder takes the output of the encoder as input, and spits out a sequence of tokens as its output. In natural language processing (NLP), the tokens can be words or subwords. Subwords are used in all popular Transformer NLP models because they enable us to address the out-of-vocabulary (OOV) issue that is inherent in a word-based system. For simplicity, we will use the term "token" to refer to the items in the input and output sequences, understanding that these tokens are subwords for NLP systems. When Transformers are used for processing images or video, the tokens can represent sub-images or objects.

Since the publication of the paper, popular models like BERT and GPT have used only the encoder or decoder aspects of the original architecture. The core commonality of these models is, thus, not the encoder-decoder aspect, but, rather, the architecture of the individual layers in the encoders and decoders. The layer architecture of Transformers is based on a self-attention mechanism and a feed-forward layer, the core aspect of this being that each input token flows through the layers in its own path, while, at the same time, being directly dependent on every other token in the input sequence. This enables parallel and direct computation of contextual token representations which was previously not possible with sequential models like RNNs.

It is beyond the scope of this paper to go into all the details of the Transformer architecture. For that, we will refer you to the original paper (Vaswani et al., 2017) or to The Illustrated Transformer[3] post. That being said, we will briefly describe the most important aspects since we will be referring to them in the catalog below. Let's start with the basic architectural diagram from the original paper, and describe some of the components.

1.1 Encoder/Decoder architecture

A generic encoder/decoder architecture (see Figure 1) is composed of two models. The encoder takes the input and encodes it into a fixed-length vector. The decoder takes that vector and decodes it into the output sequence. The encoder and decoder are jointly trained to maximize the conditional log-likelihood of the output given the input. Once trained, the encoder/decoder can generate an output given an input sequence or can score a pair of input/output sequences.

---

[2]`https://machinelearningmastery.com/encoder-decoder-long-short-term-memory-networks/`
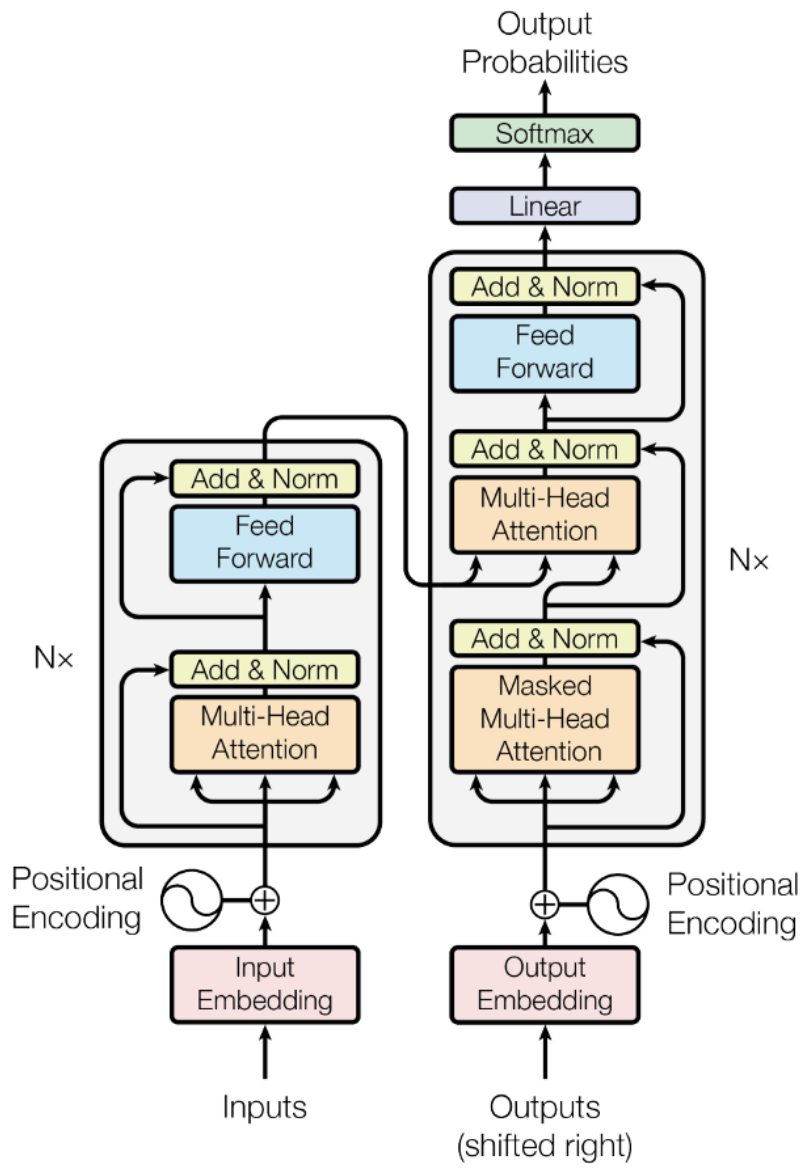[3]`https://jalammar.github.io/illustrated-transformer/`

Figure 1: Transformer Architecture from (Vaswani et al., 2017)

In the case of the original Transformer architecture, both encoder and decoder had 6 identical layers. In each of those 6 layers the Encoder had two sub layers: a multi-head self attention layer, and a simple feed forward network. The self attention layer computes the output representation of each of its input tokens based on *all the input tokens*. Each sublayer also has a residual connection and a layer normalization. The output representation size of the Encoder was 512. The multi-head self-attention layer in the decoder is slightly different than that in the encoder. It masks all tokens to the right of the token whose representation is being computed so as to ensure that the decoder can only attend to tokens that come before the token it is trying to predict. This is shown in Figure 1 as "masked multi-head attention." The Decoder also added a third sublayer, which is another multi-head attention layer over all the outputs of the Encoder. Note that all those specific details have since been modified in the many Transformer variations we will discuss. For example, as we noted before, models like BERT and GPT are based on only the encoder or decoder.

1.2 Attention

It is clear from the description above that the only "exotic" elements of the model architecture are the multi-head attention layers, but, as described above, that is where the whole power of the model lies! So, what is attention anyway? An attention function is a mapping between a query and a set of key-value pairs to an output. Each token in the input to the attention layer is converted to a query, key and value using three corresponding matrices. The output representation of each token is computed as a weighted sum of the values of all the tokens, where the weight assigned to each value is computed by a compatibility function of its associated key and the query of the token whose representation is being computed. The compatibility function used in Transformers is just a scaled dot product. A key aspect of this attention mechanism in Transformers is that each token flows through its own computation path, thus lending itself to parallel computation of the representation of all the tokens in the input sequence. Now that we understand how attention works, what is multi-head attention? Well, that is just multiple attention blocks independently computing representations for each token. All these representations are then aggregated to give the final representation of the token. We will refer you again to the The Illustrated Transformer[4] post for many more details on how the attention mechanism works, but will reproduce the diagram from the original paper in Figure 2 so you get the main idea.

There are several advantages of attention layers over recurrent and convolutional networks, the two most important being their lower computational complexity and their higher connectivity, especially useful for learning long-term dependencies in sequences.

---

[4]https://jalammar.github.io/illustrated-transformer/

Scaled Dot-Product Attention   Multi-Head Attention
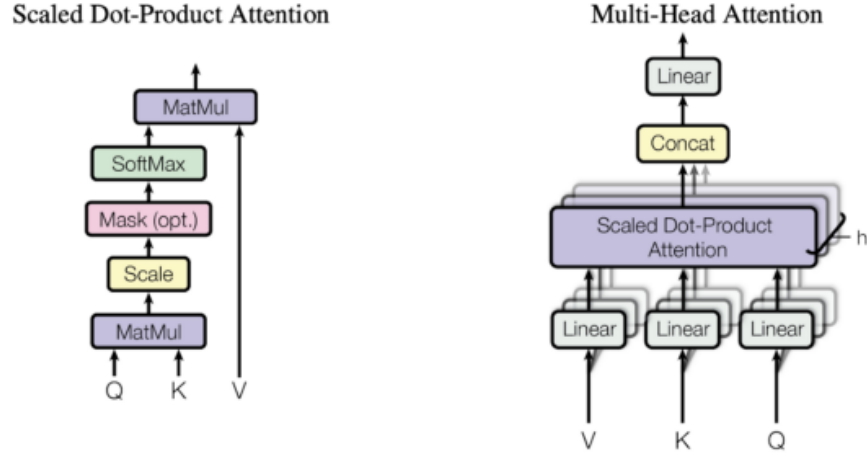
Figure 2: The Attention Mechanism from(Vaswani et al., 2017). (left) Scaled Dot-Product Attention, (right) Multi-Head Attention

1.3 Foundation vs Fine-tuned models

A foundation model is defined as "any model that is trained on broad data (generally using self-supervision at scale) that can be adapted (e.g., fine-tuned) to a wide range of downstream tasks" (Bommasani et al., 2021). When the foundation model is further trained on a small amount of target-specific data, it is called a fine-tuned model[5] because it has been fine-tuned to the specifics of the task at hand.

The BERT paper (Devlin et al., 2018) popularized this approach of pretraining and finetuning for natural language processing, resulting in many researchers using this approach for many different tasks. As a consequence, most of the leaderboards for any language-related machine leartning (ML) task became completely dominated by some version of the Transformer architecture (see for example the well known SQUAD leaderboard[6] for question answering or the GLUE leaderboard[7] for general language understanding, where all systems at the top employ Transformer-based models).

In its original usage, "fine-tuning" referred to tweaking a foundation model for a specific task, such as spam classification or question answering. Models, such as BERT, produce representations of the input tokens, but do not, by themselves, accomplish any task. Thus, it is necessary to fine-tune them by adding extra neural layers on top of the foundation model and training the model end to end.

With generative models like GPT, things are a little different. GPT is a

---

[5] https://huggingface.co/docs/transformers/training
[6] https://rajpurkar.github.io/SQuAD-explorer
[7] https://gluebenchmark.com/leaderboard

decoder language model trained to predict the next token of a sentence given all the previous tokens. By training on huge amounts of web corpora covering almost any topic one can think about, it was found that GPT could actually produce reasonable outputs to input queries or prompts. GPT accomplished this by simply predicting the next token given the input prompt sequence and the output sequence GPT had already predicted. This language generation actually did a somewhat reasonable job of tasks like answering questions about general web knowledge, writing poems etc. Notwithstanding, GPT's outputs were often untruthful or really not very helpful to the user. To address this, OpenAI researchers came up with the idea of training GPT to follow human instructions (Ouyang et al., 2022). The resulting models are called InstructGPT. The authors did this by using a small amount of human-labeled data from a large variety of tasks to further train GPT. As before, this is a "fine-tuning" process, but the resulting Instruct GPT model is capable of doing a wide range of tasks, and is, in fact, the class of models used by the popular ChatGPT engine. Since these models can accomplish a myriad of tasks, we refer to them as foundation models.

Such additional fine-tuning has been used to generate other general purpose model variants as well, specifically designed for uses cases beyond language modeling (predicting the next token in a sequence). For example, there is a subclass of models fined-tuned to learn text string embeddings optimized for semantic-relatedness, making them directly useful for higher-level semantic tasks (e.g. text classification, clustering, search retrieval, etc.). Examples include OpenAI's text embedding models[8], E5[9], and InstructOR[10]. Transformer encoders have also been successfully fined-tuned within multi-task learning frameworks to be able to perform multiple different semantic tasks using a single shared Transformer model (Liu et al., 2019; Aghajanyan et al., 2021).

Thus, as we see, while originally foundation models were fine-tuned for very specific target tasks for specific groups of users, today fine-tuning is used to also create further versions of foundation models that can be used by a huge number of users. The process used by ChatGPT and similar dialog agents, like BlenderBot3 or Sparrow, is fairly simple: Given a pretrained language model like GPT, we use it to generate different responses to input prompts (or instructions) and have humans rank the results. We then use those rankings (aka preferences or feedback) to train a reward model. The reward model attaches a score to each output for a given input instruction. After this, a reinforcement learning with human feedback (RLHF) process (Christiano et al., 2023) is used to train the model on more input instructions, but, rather than use a human to generate the feedback, the reward model is used to rank the outputs of the model. You can read much more in these two wonderful posts by Huggingface[11] and Ayush Thakur[12].

---

[8]https://platform.openai.com/docs/guides/embeddings/what-are-embeddings
[9]https://huggingface.co/intfloat/e5-large
[10]https://huggingface.co/hkunlp/instructor-xl
[11]https://huggingface.co/blog/rlhf
[12]https://wandb.ai/ayush-thakur/RLHF/reports/Understanding-Reinforcement-Learning-from-Human-Feedback-RLHF-Part
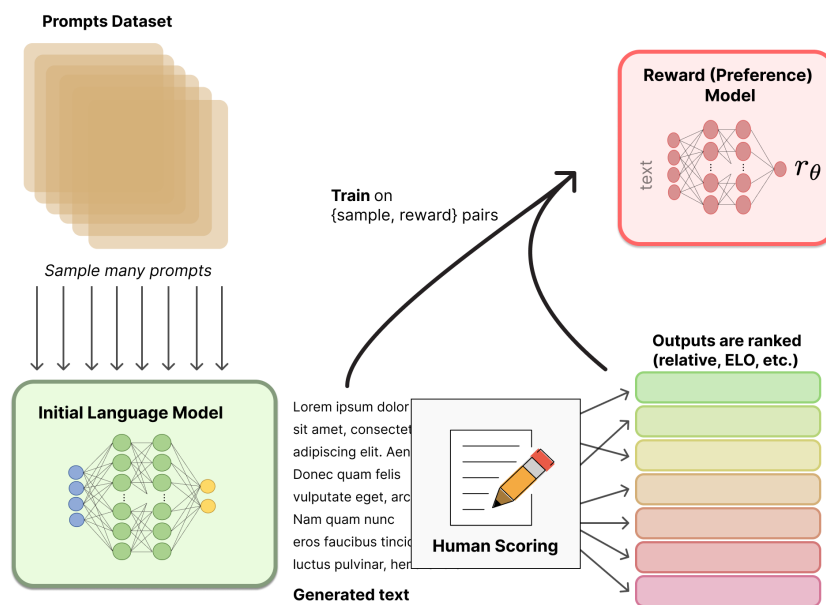
Figure 3: Reinforcement Learning with Human Feedback. From HuggingFace's RLHF blog post at `https://huggingface.co/blog/rlhf`

1.4 The Impact of Transformers

The application demonstrated in the original Transformer paper (Vaswani et al., 2017) was language translation. This seminal work also showed the architecture generalized well to other language tasks. Over the next several months, researchers figured out that Transformers could be used to capture a lot of inherent knowledge about language by pretraining them on a very large amount of unsupervised text. The knowledge captured in these models could then be transferred to target tasks by training on a small amount of labeled data.

While original Transformers were designed for language tasks, the same Transformer architecture has been applied to many other applications like the generation of images, audio, music, or even actions. Because of that, Transformers are considered a key, if not the key, component to the new wave of the so-called "Generative AI". Generative AI and its many applications are already revolutionizing many aspects of society (Stokel-Walker & Noorden, 2023; Baidoo-Anu & Owusu Ansah, 2023)

Of course all these applications would not have been possible but for the myriad of tools that made them readily available to anyone that could write a few lines of code. Not only were Transformers quickly integrated into the main AI frameworks (namely Pytorch[13] and TensorFlow (TF)[14]), but they even enabled the creation of an entire company around them. Huggingface[15], a startup that has raised over $ 60M to this day, is almost entirely built around the idea of commercializing their open source Transformers library[16].

Transformer model adoption is further accelerated as specialized hardware is developed by commercial players to improve model training and inference speed. NVIDIA's Hopper Tensor Cores[17] can apply mixed FP8 and FP16 precisions to dramatically accelerate AI calculations for Transformers.

Last but not least, we would be remiss if we did not mention the impact of ChatGPT on the popularization of Transformers. ChatGPT was released by OpenAI in November 2022, and became the fastest growing app in history, reaching 1 million users in less than a month, and 100 million in less than two (Dennean et al., 2023). ChatGPT was originally a chatbot application built on top of the Instruct-GPT model (Ouyang et al., 2022) also called GPT-3.5. Not much later, OpenAI announced the release of the more powerful GPT-4[18], which achieves human capabilities in tasks such as passing the USMLE exam for medical doctors or the bar exam for lawyers (OpenAI, 2023).

1.5 A Note on Diffusion models

Diffusion models have become the new state-of-the-art in image generation, clearly pushing aside the previous approaches such as GANs (Generative Adversarial

---

[13]https://pytorch.org/tutorials/beginner/transformer_tutorial.html
[14]https://www.tensorflow.org/text/tutorials/transformer
[15]https://huggingface.co/docs
[16]https://github.com/huggingface/transformers
[17]https://resources.nvidia.com/en-us-tensor-core/nvidia-tensor-core-gpu-datasheet
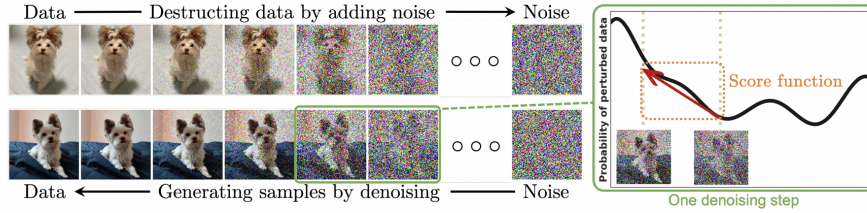[18]https://openai.com/research/gpt-4

Fig. 2. Diffusion models smoothly perturb data by adding noise, then reverse this process to generate new data from noise. Each denoising step in the reverse process typically requires estimating the score function (see the illustrative figure on the right), which is a gradient pointing to the directions of data with higher likelihood and less noise.

Figure 4: Probabilistic diffusion model architecture from "Diffusion Models: A Comprehensive Survey of Methods and Applications," Figure 2 (Yang et al., 2022)

Networks). It is important to note, though, that the diffusion mechanism is not dependent on the Transformer architecture. However, most modern diffusion approaches do include a Transformer backbone (Esser et al., 2021).

Diffusion models are a class of latent variable models trained through variational inference. What this means in practice is that we train a deep neural network to denoise images blurred with some sort of noise function. Networks that are trained this way are in fact learning the latent space of what those images represent (see Figure 4).

Diffusion models have relation to other generative models like Denoising Autoencoders and the famous Generative Adversarial Networks (GAN)[19], which they have mostly replaced in many applications. Some authors[20] will go as far as saying that Diffusion models are just a specific instance of autoencoders. However, they also admit that the small differences do transform their application, from the latent representation of autoencoders to the pure generative nature of Diffusion models.

2 The Transformers Catalog

2.1 Features of a Transformer

In this section we will introduce a catalog of the most important Transformer models that have been developed to this day. We will categorize each model according to the following properties: Family, Pretraining Architecture, Pretraining or Fine-tuning Task, Extension, Application, Date (of first known publication), Number of Parameters, Corpus, License, and Lab. Some are relative simple to understand: *Family* represents what original foundation model the specific model is extending, *extension* describes what the model is adding to the one it is

---

[19]https://en.wikipedia.org/wiki/Generative_adversarial_network
[20]https://benanne.github.io/2022/01/31/diffusion.html

deriving from, *Date* is when the model was firts published, *Number of parameters* of the pretrained model, *Corpus* is what data sources the model was pre-trained or fine-tuned on, *License* describes how the model can be legally used, and *Lab* lists the institution that published the model. The remaining propterties deserve a bit more explanation. We do that in the paregraphs that follow:

2.1.1 Pretraining Architecture

We described the Transformer architecture as being made up of an Encoder and a Decoder, and that is true for the original Transformer. However, since then, different advances have been made that have revealed that in some cases it is beneficial to use only the encoder, only the decoder, or both.

Encoder Pretraining   These models, which are also called bi-directional or auto-encoding, only use the encoder during pretraining, which is usually accomplished by masking tokens in the input sentence and training the model to reconstruct those tokens. At each stage during pretraining, self-attention layers can access all their input tokens. This family of models are most useful for tasks that require understanding complete sentences or passages, such as text classification, entailment, and extractive question answering.

Decoder Pretraining   Decoder models use only the decoder during a pretraining. They are also called auto-regressive language models because they are trained to predict the next token based on the previous sequence of tokens.

The self-attention layers can only access the tokens positioned before a given token in the sentence. They are best suited for tasks involving text generation.

Transformer (Encoder-Decoder) Pretraining   Encoder-decoder models, also called sequence-to-sequence, use both parts of the Transformer architecture.

Self-attention layers of the encoder can access all their input tokens, while the self-attention layers of the decoder can only access the tokens positioned before a given token. As explained before, the additional attention layer in the decoder enables access to all encoder token representations.

An encoder-decoder model can be pre-trained by optimizing denoising objectives (Lewis et al., 2019) or a combination of denoising and causal language modeling objectives (Soltan et al., 2022). These objective functions are complex in comparison to the ones used to pretrain encoder only or decoder only models. The encoder-decoder models are best suited for tasks revolving around generating new sentences depending on a given input, such as summarization, translation, or generative question answering.

2.1.2 Pretraining or Finetuning Task

When training a model we need to define an objective, or task, for the model to learn on. Some of the typical tasks, such as predicting the next token or learning to reconstruct masked tokens were already mentioned above. "Pre-trained Models

for Natural Language Processing: A Survey" (Qiu et al., 2020) includes a pretty comprehensive taxonomy of pretraining tasks, all of which can be considered self-supervised:

1. **Language Modeling (LM):** Predict the next token (in the case of unidirectional LM) or the previous and next token (in the case of bidirectional LM).

2. **Causal Language Modeling (Causality-masked LM):** Autoregressively (left-to-right, in general) predict a text sequence, similar to unidirectional LM.

3. **Prefix Language Modeling (Prefix LM):** In this task, a separate 'prefix' section is separated from the main sequence. Within the prefix, any token can attend to any other token (non-causal). Outside of the prefix, decoding proceeds autoregressively.

4. **Masked Language Modeling (MLM):** Mask out some tokens from the input sentences and then train the model to predict the masked tokens using the surrounding context.

5. **Permuted Language Modeling (PLM):** Same as LM, but on a random permutation of input sequences. A permutation is randomly sampled from all possible permutations. Then some of the tokens are chosen as the target, and the model is trained to predict these targets.

6. **Denoising Autoencoder (DAE):** Take a partially corrupted input and aim to recover the original, undistorted input. Examples of corrupted input include randomly sampling tokens from the input and replacing them with "[MASK]" elements, randomly deleting tokens from the input, or shuffling sentences in random order.

7. **Replaced Token Detection (RTD):** Using a "generator" model, randomly replace certain tokens in the text. The "discriminator" is tasked to predict whether a token comes from the original text, or the generator model.

8. **Next Sentence Prediction (NSP):** Train the model to distinguish whether two input sentences are continuous segments from the training corpus.

Note that in the case of fine-tuned models, this property is used to describe the task the model was fine-tuned to, not how it was pre-trained.

2.1.3 Application

Here we will note what are the main practical applications of the Transformer model. Most of these applications will be in the language domain (e.g. question answering, sentiment analysis, or entity recognition). However, as mentioned before, some Transformer models have also found applications well beyond NLP and are also included in the catalog.
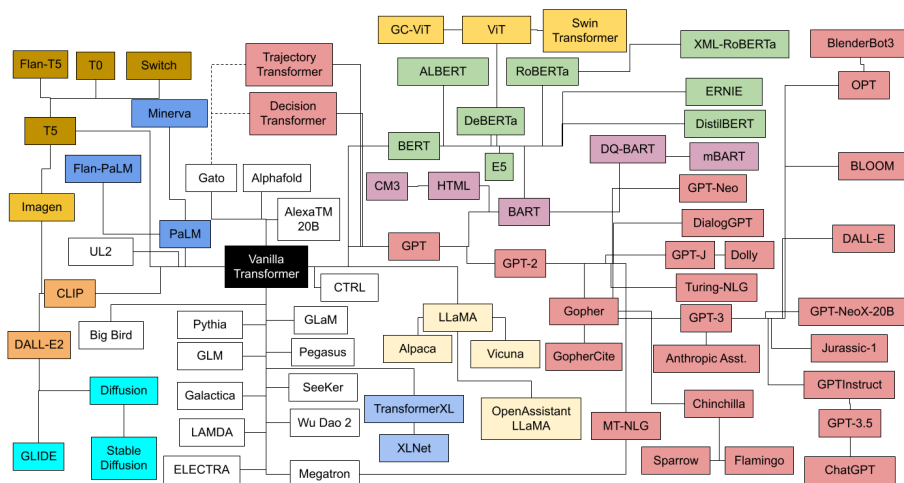
Figure 5: Transformers Family Tree

2.2 Catalog table

You can access a table format of the catalog at `http://bit.ly/3YFqRn9` for easier browsing across the different model features.

2.3 Family Tree

The diagram in Figure 5 is a simple view that highlights the different families of Transformers and how they relate to each other.

2.4 Chronological timeline

Another interesting perspective of the catalog is to see it as a chronological timeline. In Figure 6 you will find all the Transformers in the catalog sorted by their date of publication. In this first visualization, the Y-axis is only used to cluster Transformers of related heritage/family.

In Figure 7, the Y-axis represents model size in millions of parameters. You won't be able to see all the models in the catalog since many fall right on the same time and size, so please refer to the previous image for that.

Since the introduction of chatGPT, the LLM open-source community has experienced a significant surge in activity. With each passing week, we have observed a proliferation of refined models fine-tuned using the latest technologies. As a result, these models are continuously improving, growing more robust and powerful. Figure 8 demonstrates the recent emerged models since Feb, 2023.
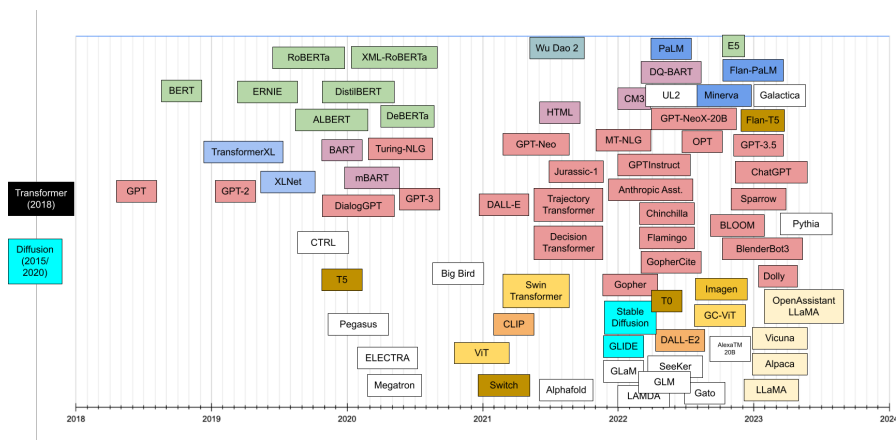
Figure 6: Transformer timeline. Colors describe Transformer family.
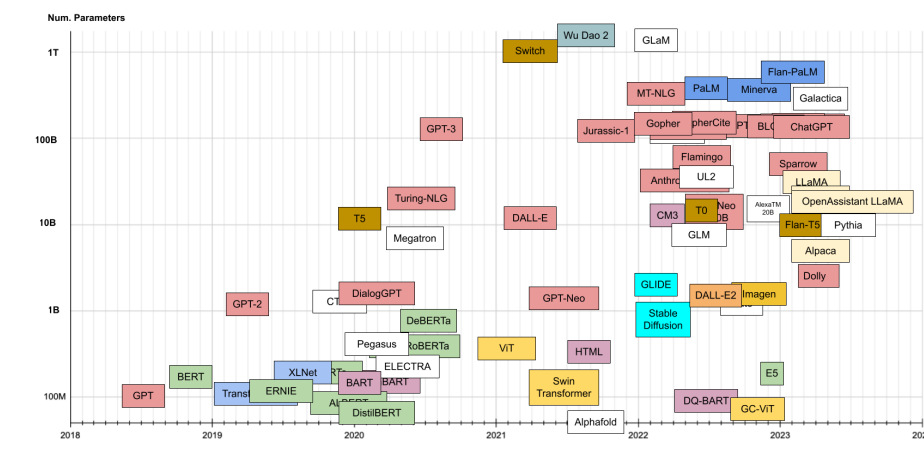
Figure 7: Transformer timeline. On the vertical axis, number of parameters. Colors describe Transformer family.

15
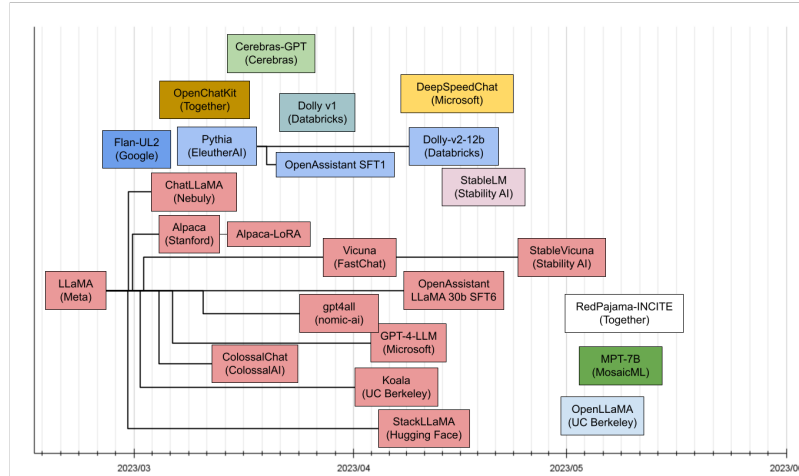
Figure 8: Recently published LLMs

A Catalog List

Finally, here is the full list view that might be easier to follow along in some cases:

A.1 ALBERT

- **Reference:** (Lan et al., 2019)

- **Link:** https://huggingface.co/docs/transformers/model_doc/albert

- **Family:** BERT

- **Pretraining Architecture:** Encoder

- **Pretraining Task:** MLM/NSP

- **Extension:** Compressed version of BERT using parameter sharing, which is much more efficient given the same number of parameters

- **Application:** Same as BERT

- **Date (of first known publication):** 09/2019

- **Num. Params:** Base = 12M, Large = 18M, XLarge = 60M*

- **Corpus:** Same as BERT

- **License:** Open, Apache-2.0

- **Lab:** Google

16

A.2 AlexaTM 20B

- **Reference:** (Soltan et al., 2022)

- **Link:** `https://github.com/amazon-science/alexa-teacher-models`

- **Family:** Transformer

- **Pretraining Architecture:** Encoder/Decoder

- **Pretraining Task:** Optimizes denoising (80%) and Prefix LM (20%)

- **Extension:** Derived from BART and layernorms located exactly at the beginning of each layer. Encoder initialized with internal 10B pre-trained encoder.

- **Application:** Summarization, multi-lingual machine translation and NLU tasks

- **Date (of first known publication):** 08/2022

- **Num. Params:** 20B

- **Corpus:** Wikipedia and mC4 datasets in 12 languages.

- **License:** Limited, non-commercial

- **Lab:** Amazon

A.3 Alpaca

- **Reference:** (Taori et al., 2023)

- **Link:** `https://github.com/tatsu-lab/stanford_alpaca`

- **Family:** LLaMA

- **Pretraining Architecture:** Decoder

- **Fine-tuning Task:** human instructions

- **Extension:** Alpaca is fine-tuned from a 7B LLaMA model.

- **Application:** Evaluated on a variety of text generation and classification tasks.

- **Date (of first known publication):** 03/2023

- **Num. Params:** 7B

- **Corpus:** 52K instruction-following data generated using self-instruct mechanism, from 175 human-written instruction-output pairs.

- **License:** Limited, Non-commercial bespoke license

- **Lab:** Stanford

A.4 AlphaFold

- **Reference:** (Jumper et al., 2021)

- **Link:** https://github.com/deepmind/alphafold

- **Family:** SE(3) Transformer (Fuchs et al., 2020)

- **Pretraining Architecture:** Encoder

- **Pretraining Task:** Protein folding prediction of BERT using parameter sharing, which is much more efficient given the same number of parameters

- **Extension:** The original Alphafold used a BERT-style Transformer. The details of Alphafold's Transformer are not known, but it is believed it is an extension of the SE(3)-Tranformer, a 3-D equivariant Transformer (see this blog post[21])

- **Application:** Protein folding

- **Date (of first known publication):** 09/2019

- **Num. Params:**b12M, Large = 18M, XLarge = 60M*

- **Corpus:** Same as BERT

- **License:** the code is open sourced, with Apache-2.0

- **Lab:** Deepmind

A.5 Anthropic Assistant

- **Reference:** (Bai et al., 2022a; Askell et al., 2021; Bai et al., 2022b)

- **Link:** N/A

- **Family:** Transformer

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** LM

- **Extension:** These models do not introduce novelties at the architecture/pretraining level and they are similar to GPT-3, but they focus on how to improve alignment through fine-tuning and prompting. Note that the Anthropic Assistant includes several models optimized for different tasks. The work often focus on the benefits of RLHF. Latest versions of this work study using an LLM to critique the model output for harmlessness, and provide feedback data for RL this way (RLHF -> RLAIF).

---

[21]https://fabianfuchsml.github.io/alphafold2/

- **Application:** Different models with different applications from general dialog to code assistant.

- **Date (of first known publication):** 12/2021

- **Num. Params:**10M to 52B

- **Corpus:** 400B tokens from filtered Common Crawl and Books, and 10% python code. They also create several Dialogue Preference datasets for the RLHF training.

- **License:** N/A

- **Lab:** Anthropic

A.6 BART

- **Reference:** (Lewis et al., 2019)

- **Link:** https://huggingface.co/docs/transformers/model_doc/bart

- **Family:** BERT for encoder, GPT for Decoder

- **Pretraining Architecture:** Encoder/Decoder

- **Pretraining Task:** DAE

- **Extension:** It can be seen as a generalization of BERT and GPT in that it combines ideas from both in the encoder and decoder

- **Application:** Mostly text generation but also some text understanding tasks*

- **Date (of first known publication):** 10/2019*

- **Num. Params:** Base = 140M, Large = 400M. In general, roughly 10% larger than BART for equivalent architectures.

- **Corpus:**Same as RoBERTa (160Gb of news, books, stories)

- **License:** Open, Apache-2.0

- **Lab:**Facebook

A.7 BERT

- **Reference:** (Devlin et al., 2018)

- **Link:** https://huggingface.co/docs/transformers/model_doc/bert

- **Family:** BERT

- **Pretraining Architecture:** Encoder

- **Pretraining Task:** MLM/NSP

- **Extension:It can be seen as a generalization of BERT and GPT in that it combines ideas from both in the encoder and decoder**

- **Application:**General Language Understanding and Question Answering. Many other language applications followed

- **Date (of first known publication):** 10/2018

- **Num. Params:**Base = 110M, Large = 340MT

- **Corpus:**Toronto Book Corpus and Wikipedia (3.3B Tokens)

- **License:** Open, Apache-2.0

- **Lab:**Google

A.8   Big Bird

- **Reference:** (Zaheer et al., 2020)

- **Link:** `https://huggingface.co/docs/transformers/model_doc/big_bird`

- **Family:** BERT

- **Pretraining Architecture:** Encoder

- **Pretraining Task:** MLM

- **Extension:** Big Bird can extend other architectures such as BERT, Pegasus, or RoBERTa by using a sparse attention mechanism that elminates the quadratic dependency thus making it more suitable for longer sequences

- **Application:**Particularly well suited for longer sequences, not only in text but also e.g. in genomics

- **Date (of first known publication):** 07/2020

- **Num. Params:**Depends on the overall architecture

- **Corpus:**Books, CC-News, Stories and Wikipedia)

- **License:** Open, Apache-2.0

- **Lab:**Google

A.9 BlenderBot3

- **Reference:** (Shuster et al., 2022b)

- **Link:** `https://parl.ai/projects/bb3/`

- **Family:** GPT

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** LM

- **Extension:** BlenderBot 3 is based on a pre-trained OPT. It adds features needed for a dialog agent such as long-term memory or the ability to search the internet. It is also fine-tuned for some specific tasks given human feedback on them.

- **Application:** Same as GPT-3

- **Date (of first known publication):** 08/2022

- **Num. Params:** 3B, 30B, and 175B

- **Corpus:** 180B tokens = RoBERTa + the Pile + PushShift.io Reddit item **License:** Limited, non-commercial, research only

- **Lab:** Facebook

A.10 BLOOM

- **Reference:** See blog post[22]

- **Link:** `https://huggingface.co/docs/transformers/model_doc/bloom`

- **Family:** GPT

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** LM

- **Extension:** Main difference to GPT-3 is that it uses full attention instead of sparse attention

- **Application:** Same as GPT-3

- **Date (of first known publication):** 07/2022

- **Num. Params:** 176B

- **Corpus:** 366B tokens (1.5 TB of text data) multilingual dataset

- **Lab:** Big Science/Huggingface

- **License:** Open, but need to follow restrictions in Attachment A, BigScience RAIL License v1.0

---

[22]`https://huggingface.co/blog/bloom-inference-optimization`

A.11 ChatGPT

- **Reference:** See blog post[23]

- **Link:** https://chat.openai.com

- **Family:** GPT

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** LM

- **Extension:** ChatGPT takes a GPT3.5 (aka GPT3 Davinci-003) pretrained model and uses RLHF to finetune the model mostly like described in InstructGPT but with slight differences in the data collection. ChatGPT is also more than a model since it includes extensions for Memory Store and retrieval similar to BlenderBot3

- **Application:** Dialog agents

- **Date (of first known publication):** 10/2022

- **Num. Params:** Same as GPT3

- **Corpus:** Same as GPT3 + datasets generated for RLHF

- **License:** Closed source, accessible through API

- **Lab:** OpenAI

A.12 Chinchilla

- **Reference:** (Hoffmann et al., 2022)

- **Link:** N/A

- **Family:** GPT

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** LM

- **Extension:** Same as Gopher but with optimizations to reduce model size and therefore training/inference time with equal or superior performance

- **Application:** Same as Gopher/GPT3

- **Date (of first known publication):** 03/2022

- **Num. Params:**70B

- **Corpus:** Massive Text

- **License:** Closed source.

- **Lab:** Deepmind

---

[23]https://openai.com/blog/chatgpt/

A.13 CLIP

- **Reference:** (Radford et al., 2021)

- **Link:** https://huggingface.co/docs/transformers/model_doc/clip

- **Family:** CLIP (Also using Resnet, ViT, and vanilla Transformer for text)

- **Pretraining Architecture:** Encoder

- **Pretraining Task:** predict which of the N × N possible (image, text) pairings across a batch actually occurred

- **Extension:** Combines Resnet and ViT for the visual encoding with Transformer for the Textual encoder

- **Application:** Image/object classification

- **Date (of first known publication):** 02/2021

- **Num. Params:** N/A

- **Corpus:** WIT (WebImageText) - 400 million text,image pairs

- **License:** Open, MIT license

- **Lab:** OpenAI

A.14 CM3

- **Reference:** (Aghajanyan et al., 2022)

- **Link:** N/A

- **Family:** HTML

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** Causality-masked LM

- **Extension:** This is somewhat similar to HTML in its use of structured training data. However, it is a different architecture and uses causal masking, which makes the model predict, at the end of the sequence, an entire missing span of text. It also includes image input via Vector Quantized Variational Autoencoding (VQ-VAE) tokens.

- **Application:** Multimodal language model with the ability to do structured prompting, zero-shot captioning, image generation, and entity linking (via target text prediction of hyperlinks)

- **Date (of first known publication):** 01/2022

- **Num. Params:**13B (largest)

- **Corpus:** CC-News, English Wikipedia

- **License:** N/A

- **Lab:** Facebook

A.15 CTRL

- **Reference:** (Keskar et al., 2019)

- **Link:** https://huggingface.co/docs/transformers/model_doc/ctrl

- **Family:**

- **Pretraining Architecture:** Decoder

- **Pretraining Task:**

- **Extension:** model can generate text conditioned on control codes that specify domain, style, topics, dates, entities, relationships between entities, plot points, and task-related behavior

- **Application:** Controllable text generation

- **Date (of first known publication):** 09/2019

- **Num. Params:**1.63B

- **Corpus:** 140 GB of text including: Wikipedia (En, De, Es, Fr), Project Gutenberg, 45 subreddits, OpenWebText2, Amazon Reviews, Europarl and UN data from WMT, question-answer pairs from ELI5, and the MRQA shared task3, which includes the Stanford Question Answering Dataset, NewsQA, TriviaQA, SearchQA, HotpotQA , and Natural Questions

- **License:** Open, BSD-3-Clause license

- **Lab:** Salesforce

A.16 DALL-E

- **Reference:** (Ramesh et al., 2021)

- **Link:** https://openai.com/blog/dall-e

- **Family:** GPT

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** Caption prediction

- **Extension:** A differential variational auto-encoder is used to learn the visual codebook. The Transformer is a variation of GPT-3

- **Application:** Text to image

- **Date (of first known publication):** 01/2021

- **Num. Params:**12B

- **Corpus:** 250 million text-images pairs from the internet

- **License:** N/A

- **Lab:** OpenAI

A.17   DALL-E 2

- **Reference:** (Ramesh et al., 2022)

- **Link:** `https://openai.com/dall-e-2`

- **Family:** CLIP, GLIDE

- **Pretraining Architecture:** Encoder/Decoder

- **Pretraining Task:** Caption prediction

- **Extension:** Combines CLIP encoder and Diffusion decoder similar to GLIDE

- **Application:** Text to image

- **Date (of first known publication):** 04/2022

- **Num. Params:**3.5B

- **Corpus:** Combination of the DALL-E and CLIP datasets

- **License:** Closed source, accessible through API

- **Lab:** OpenAI

A.18   DeBERTa

- **Reference:** (He et al., 2021)

- **Link:** `https://huggingface.co/microsoft/deberta-large`

- **Family:** BERT

- **Pretraining Architecture:** Encoder

- **Pretraining Task:** MLM

- **Extension:** Separate positional embedding vector independent from the content embedding using disentangled attention matrices for contents and relative positions

- **Application:** Same as BERT

- **Date (of first known publication):** 06/2020

- **Num. Params:** 134M (base), 384M (large), 750M (xlarge)

- **Corpus:** English Wikipedia, BookCorpus, OPENWEBTEXT and STO-RIES

- **License:** Open, MIT license

- **Lab:** Microsoft

A.19 Decision Transformers

- **Reference:** (Chen et al., 2021)

- **Link:** `https://github.com/kzl/decision-transformer`

- **Family:** GPT, Control Transformers" (not per se a family, but grouping here those Transformers that try to model more general control, RL-like, tasks)

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** Next action prediction

- **Extension:** Decision Transformers use a GPT architecture and extend it by encoding trajectories in a way that they can be learned by an auto-regressive task

- **Application:** General RL (reinforcement learning tasks)

- **Date (of first known publication):** 06/2021

- **Num. Params:**Same as GPT

- **Corpus:** Different corpus for different experiments

- **License:** Open, MIT license

- **Lab:** Google/UC Berkeley/Facebook

A.20 DialoGPT

- **Reference:** (Zhang et al., 2019a)

- **Link:** `https://huggingface.co/docs/transformers/model_doc/dialogpt`

- **Family:** GPT

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** LM

- **Extension:** GPT-2 architecture trained on dialog data

- **Application:** Text generation in dialog settings

- **Date (of first known publication):** 10/2019

- **Num. Params:**1.5B

- **Corpus:** 140M Reddit conversations

- **License:** Open, MIT license

- **Lab:** Microsoft

A.21   DistilBERT

- **Reference:** (Sanh et al., 2019)

- **Link:** `https://huggingface.co/docs/transformers/model_doc/distilbert`

- **Family:** BERT

- **Pretraining Architecture:** Encoder

- **Pretraining Task:** MLM/NSP

- **Extension:** Compressed version of BERT using distillation, which is much more efficient given the same number of parameters

- **Application:** Same as BERT

- **Date (of first known publication):** 10/2019

- **Num. Params:**66M

- **Corpus:** Same as BERT

- **License:** Open, Apache-2.0

- **Lab:** Huggingface

A.22   DQ-BART

- **Reference:** (Li et al., 2022)

- **Link:** `https://github.com/amazon-science/dq-bart`

- **Family:** BART

- **Pretraining Architecture:** Encoder/Decoder

- **Pretraining Task:** DAE

- **Extension:** Adds quantization and distillation to a BART model to improve performance and model size

- **Application:** Text generation and understanding

- **Date (of first known publication):** 03/2022

- **Num. Params:** Up to 30x reduction in parameters compared to standard BART

- **Corpus:** CNN/DM, XSUM, ELI5, WMT16 En-Ro (1M tokens)

- **License:** Open, Apache-2.0

- **Lab:** Amazon

A.23   Dolly

- **Reference:** See blog post[24]

- **Link:** https://huggingface.co/databricks/dolly-v1-6b

- **Family:** GPT

- **Pretraining Architecture:** Decoder

- **Fine-tuning Task:** human instructions

- **Extension:** fine-tuned based on the GPT-J-6B (V1) and Pythia model (V2)

- **Application:** Similar to Alpaca

- **Date (of first known publication):** 03/2023

- **Num. Params:** V1: 6B, V2: 12B

- **Corpus:** V1: Instruction corpus same as Alpaca, V2: databricks own dataset.

- **License:** Open

- **Lab:** Databricks, Inc

---

[24]https://www.databricks.com/blog/2023/04/12/dolly-first-open-commercially-viable-instruction-tuned-llm

A.24   E5

- **Reference:** (Wang et al., 2022)
- **Link:** https://huggingface.co/intfloat/e5-large
- **Family:** BERT
- **Pretraining Architecture:** Encoder
- **Fine-tuning Task:** Semantic similarity using contrastive loss
- **Extension:** Fine-tunes BERT-based models to create text string embeddings optimized for semantic relatedness.
- **Application:** Text embeddings for semantic relatedness tasks such as text clustering or search retrieval.
- **Date (of first known publication):** 12/2022
- **Num. Params:** 300M (large version)
- **Corpus:** MS-MARCO, NQ, NLI
- **License:** Open, MIT license
- **Lab:** Microsoft

A.25   ELECTRA

- **Reference:** (Clark et al., 2020)
- **Link:** https://huggingface.co/docs/transformers/model_doc/electra
- **Family:** BERT
- **Pretraining Architecture:** Encoder
- **Pretraining Task:** RTD
- **Extension:** Applied new training techniques including Replaced Token Detection
- **Application:** 03/2020
- **Date (of first known publication):** 2020
- **Num. Params:** Base = 110M, Large = 330M
- **Corpus:** Same as BERT except for Large which is same as XLNet
- **License:** Open, Apache-2.0
- **Lab:** Stanford/Google

A.26   ERNIE

- **Reference:**(Zhang et al., 2019b)

- **Link:** N/A

- **Family:** BERT

- **Pretraining Architecture:** Encoder

- **Pretraining Task:** MLM

- **Extension:** Uses BERT for Encoder architecture, but stacks and aggregates two of them for text and entities. This architecture could be understood as BERT for text + knowledge graphs

- **Application:** Knowledge intensive related tasks that might benefit from knowledge graphs or entities such as entity recognition

- **Date (of first known publication):** 05/2019

- **Num. Params:** Ernie-ViLG 2.0 = 10B, Ernie 3.0 Titan = 260B

- **Corpus:** English Wikipedia + Wikidata for entitites (note that they initialize model to original BERT parameter values

- **License:** Closed source

- **Lab:** Baidu, Pengcheng Lab

A.27   Flamingo

- **Reference:** (Alayrac et al., 2022)

- **Link:** N/A

- **Family:** Chinchilla

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** Log likelihood of text given some visual input

- **Extension:** It uses a frozen textual language model (like Chinchilla) conditioned on the visual representation, which is encoded from a Normalizer-Free ResNet

- **Application:** Text to image

- **Date (of first known publication):** 04/2022

- **Num. Params:**80B (largest)

- **Corpus:** MultiModal MassiveWeb (M3W): 185 million images and 182 GB text + a number of text paired with image datasets: ALIGN + LTIP (Long Text & Image Pairs) = 312 million images, and VTP (Video & Text Pairs) = 27 million short videos (approximately 22 seconds on average)

- **License:** Closed source

- **Lab:** Deepmind

A.28 Flan-T5

- **Reference:** (Chung et al., 2022)

- **Link:** https://huggingface.co/docs/transformers/model_doc/flan-t5

- **Family:** T5

- **Pretraining Architecture:** Encoder/Decoder

- **Fine-tuning Task:** Instructions for zero-shot and few-shot tasks

- **Extension:** Flan-T5 is generated by "Flan Finetuning" the T5 models: (1) scaling the number of tasks to 1,836, (2) scaling the model size, and (3) finetuning on chain-of-thought data.

- **Application:** The primary use is to underestand how to improve large language models with the right kind of instruction fine-tuning. The focus is research on zero-shot and in-context few-shot learning NLP tasks, such as reasoning, and question answering; advancing fairness and safety research, and understanding limitations of current large language models

- **Date (of first known publication):** 11/2022

- **Num. Params:** 80M(small), 250M(base), 780M(large), 3B(xl), 11B(xxl)

- **Corpus:** Flan finetuned with tasks in Muffin, T0-SF, NIV2, and CoT.

- **License:** Open, Apache-2.0

- **Lab:** Google

A.29 Flan-PaLM

- **Reference:** (Chung et al., 2022)

- **Link:** N/A

- **Family:** PaLM

- **Pretraining Architecture:** Decoder

- **Fine-tuning Task:** Instructions for zero-shot and few-shot tasks

- **Extension:** Flan-PaLM is generated by "Flan Finetuning" the PaLM models: (1) scaling the number of tasks to 1,836, (2) scaling the model size, and (3) finetuning on chain-of-thought data.

- **Application:** Same as Flan-T5. The goal is to show Flan finetuning can even improve on the largest Google LMs (+9.4% improvement average across tasks), with improvements to chain of thought, self consistency, multilingual tasks, arithmetic reasoning

- **Date (of first known publication):** 11/2022

- **Num. Params:** 8B, 62B, 540B

- **Corpus:** Flan finetuned with tasks in Muffin, T0-SF, NIV2, and CoT.

- **License:** Closed source

- **Lab:** Google

A.30   Galactica

- **Reference:** (Taylor et al., 2022)

- **Link:** https://galactica.org

- **Family:** Transformer

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** LM for scientific domain

- **Extension:** Transformer based architecture in a decoder-only setup with a few modifications. Data extensions include special tokens for working memory, citations, genetic data, and a few other biology related tasks.

- **Application:** The models are designed to perform scientific tasks, including but not limited to citation prediction, scientific QA, mathematical reasoning, summarization, document generation, molecular property prediction and entity extraction.

- **Date (of first known publication):** 11/2022

- **Num. Params:** mini: 125M, base: 1.3B, standard: 6.7B, large: 30B, huge: 120B

- **Corpus:** Trained on 106 billion tokens of open-access scientific text and data. This includes papers, textbooks, scientific websites, encyclopedias, reference material, knowledge bases, and more

- **License:** Limited, non-commerical CC BY-NC 4.0 license

- **Lab:** Meta

A.31 Gato

- **Reference:** (Reed et al., 2022)

- **Link:** https://www.deepmind.com/blog/a-generalist-agent

- **Family:** "Control Transformers" (not per se a family, but grouping here those Transformers that try to model more general control, RL-like, tasks)

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** MLM (where tokens are either text or agent actions)

- **Extension:** The standard decoder-only Transformer architecture is preceded by an embedding layer that can embed text and images, plus add position encodings to add spatial information when applicable.

- **Application:** Gato presents a generalizable agent that can be used beyond text to tasks such as playing Atari or controlling a robot arm.

- **Date (of first known publication):** 05/2022

- **Num. Params:** 1.2B

- **Corpus:** 1.5T tokens including standard text (e.g. MassiveText), vision (e.g. ALIGN), and simulation environments (e.g. ALE Atari, or RGB Stacking Real Robot)

- **License:** Closed source

- **Lab:** Deepmind

A.32 GLaM

- **Reference:** (Du et al., 2022a)

- **Link:** See blog post[25]

- **Family:** Transformer

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** LM

- **Extension:** GLaM introduces a Mixture of 64 Experts to increase parameter count and generalization properties in a somewhat standard decoder-only. Transformer architecture. Only two experts get activated at a time per token, which makes the model also more efficient in training and inference.

---

[25]https://ai.googleblog.com/2021/12/more-efficient-in-context-learning-with.html

- **Application:** General language modeling

- **Date (of first known publication):** 12/2021

- **Num. Params:**1.2T across 64 experts, but only 96B get activated for inference

- **Corpus:** 1.6T tokens including web pages filtered by Wikipedia and books for quality

- **License:** Closed source

- **Lab:** Google

A.33   GLIDE

- **Reference:** (Nichol et al., 2021)

- **Link:** `https://github.com/openai/glide-text2im`

- **Family:** Diffusion models

- **Pretraining Architecture:** Encoder

- **Pretraining Task:** Caption prediction

- **Extension:** GLIDE can be seen as an extension of the ADM (Ablated Diffusion Model) by the same authors. However, ADM is not per se a Transformer architecture although it does resemble one in some of the configurations the authors use. Given that ADM is by the same authors and was quickly followed up by GLIDE, we think it is fair to consider GLIDE as the first of its kind.

- **Application:** Text to image

- **Date (of first known publication):** 12/2021

- **Num. Params:**3.5B diffusion model (2.3B for visual encoding, 1.2B for textual) + 1.5B for model for upsampling

- **Corpus:** Same as DALL-E

- **License:** Open, MIT license

- **Lab:** OpenAI

A.34 GLM

- **Reference:** (Du et al., 2022b)

- **Link:** `https://github.com/THUDM/GLM-130B`

- **Family:** GLM (General Language Model)

- **Pretraining Architecture:** Encoder and decoder

- **Pretraining Task:** Auto regressive blank infilling

- **Extension:** GLM has a bidirectional encoder and a unidirectional decoder in a unified model.

- **Application:** a General Language Model pretrained with an autoregressive blank-filling objective and can be finetuned on various natural language understanding and generation tasks.

- **Date (of first known publication):** 03/2022

- **Num. Params:** Base = 110M, Large = 335M, and also 2B, 10B, 130B

- **Corpus:** Pile, GLM-130B Chinese corpora, P3, DeepStruct finetuning dataset

- **License:** Open, MIT license

- **Lab:** Tsinghua

A.35 Global Context ViT

- **Reference:** (Hatamizadeh et al., 2022)

- **Link:** `https://github.com/NVlabs/GCVit`

- **Family:** ViT

- **Pretraining Architecture:** Encoder

- **Pretraining Task:** Image classification

- **Extension:** hierarchical ViT architecture consisting of local and global self-attention modules

- **Application:** Image generation

- **Date (of first known publication):** 06/2022

- **Num. Params:** 90M

- **Corpus:** Imagenet-1K and other task dependent dataasets

- **License:** Limited, non-commercial license CC-BY-NC-SA-4.0

- **Lab:** NVidia

A.36   Gopher

- **Reference:** (Rae et al., 2021)
- **Link:** See blog post[26]
- **Family:** GPT
- **Pretraining Architecture:** Decoder
- **Pretraining Task:** LM
- **Extension:** Same as GPT-2 but use RSNorm instead of LayerNorm and relative positional encoding rather than absolute
- **Application:** Mostly Language Modeling and NLU, but also extensible like GPT
- **Date (of first known publication):** 12/2021
- **Num. Params:**280B
- **Corpus:** Massive Text (2.35 billion documents, or about 10.5 TB of text including Massive Web, Books, Github, News, C4, and Wikipedia.
- **License:** Closed source
- **Lab:** Deepmind

A.37   GopherCite

- **Reference:** (Menick et al., 2022)
- **Link:** See blog post[27]
- **Family:** GPT
- **Pretraining Architecture:** Decoder
- **Pretraining Task:** LM
- **Extension:** GopherCite is based on Gopher but adds a step using RLHP (Reinforcement Learning from Human Preferences) to learn whether not only a response is plausible but also supported
- **Application:** Dialog systems, Q&A, general language generation tasks
- **Date (of first known publication):** 03/2022
- **Num. Params:**280B

---

[26]https://www.deepmind.com/blog/language-modelling-at-scale-gopher-ethical-considerations-and-retrieval
[27]https://www.deepmind.com/blog/gophercite-teaching-language-models-to-support-answers-with-verified-quotes

- **Corpus:** Same as Gopher plus specific dataset generated in the RLHP process

- **License:** Closed source

- **Lab:** Deepmind

A.38 GPT

- **Reference:** (Radford et al., 2018)

- **Link:** https://huggingface.co/docs/transformers/model_doc/openai-gpt

- **Family:** GPT

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** LM

- **Extension:**

- **Application:** Text generation, but adaptable to many other NLP tasks when fine tuned.

- **Date (of first known publication):** 06/2018

- **Num. Params:**117M

- **Corpus:** Unsupervised Pretraining on BookCorpus dataset. Supervised Finetuning on several task-specific datasets including SNLI, RACE, Quora...

- **License:** N/A

- **Lab:** OpenAI

A.39 GPT-2

- **Reference:** (Radford et al., 2019)

- **Link:** https://huggingface.co/docs/transformers/model_doc/gpt2

- **Family:** GPT

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** LM

- **Extension:** Minor extensions to the GPT architecture (e.g. layer normalization moved to the input of each sub-layer, or increased context size from 512 to 1024)

- **Application:** Text generation, but adaptable to many other NLP tasks when fine tuned.

- **Date (of first known publication):** 02/2019

- **Num. Params:** 124M, 355M, 774M, 1.5B

- **Corpus:** 8 million web pages (40 GB). 10X GPT . WebText dataset is created by crawling all links at Reddit with at least 3 Karma points.

- **License:** Open, Modified MIT license

- **Lab:** OpenAI

A.40   GPT-3

- **Reference:** (Brown et al., 2020)

- **Link:** `https://github.com/openai/gpt-3`

- **Family:** GPT

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** LM

- **Extension:** Same as GPT-2 with the only addition of alternating dense and locally banded sparse attention patterns, inspired by the Sparse Transformer

- **Application:** Initially text generation, but has over time been used for a large range of applications in areas such as code generation, but also image and audio generation

- **Date (of first known publication):** 05/2020

- **Num. Params:**175 B

- **Corpus:** 500B tokens including CommonCrawl (410B), WebText2 (19B), Books1 (12B), Books2 (55B), and Wikipedia (3B)

- **License:** Closed source

- **Lab:** OpenAI

A.41 GPT-3.5

- **Reference:** N/A

- **Link:** `https://platform.openai.com/docs/model-index-for-researchers/models-referred-to-as-gpt-3-5`

- **Family:** GPT

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** LM

- **Extension:** The GPT3.5 series includes a number of models like Davinci-003. They are basically versions of the InstructGPT model. See blog post[28] for details on the comparison of the performance to older GPT3 models.

- **Application:** Dialog and general language, but there is a code specific model - codex

- **Date (of first known publication):** 10/2022

- **Num. Params:**175B

- **Corpus:** Same as InstructGPT

- **License:** Closed source, accessible through API

- **Lab:** OpenAI

A.42 GPT-J

- **Reference:** (Wang & Komatsuzaki, 2021)

- **Link:** `https://huggingface.co/EleutherAI/gpt-j-6B`

- **Family:** GPT

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** LM

- **Extension:** GPT-J 6B is a Transformer model trained using Mesh Transformer JAX and same tokenizer as GPT2/3

- **Application:** Same as GPT-3

- **Date (of first known publication):** 05/2021

- **Num. Params:** 6B

---

[28]`https://scale.com/blog/gpt-3-davinci-003-comparison`

- **Corpus:** Pile corpus, a large-scale curated dataset created by EleutherAI.

- **License:** Open, Apache-2.0

- **Lab:** EleutherAI

A.43 GPT-Neo

- **Reference:** `https://huggingface.co/docs/transformers/model_doc/gpt_neo`

- **Link:** `https://github.com/EleutherAI/gpt-neo`

- **Family:** GPT

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** LM

- **Extension:** Similar to GPT-2 but uses local attention in every other layer with a window size of 256 tokens

- **Application:** Text generation, but adaptable to many other NLP tasks when fine tuned

- **Date (of first known publication):** 03/2021

- **Num. Params:** 5B, 2.7B (XL)

- **Corpus:** Pile — 840 GB open source text dataset that combines 22 pre existing datasets

- **License:** Open, MIT license

- **Lab:** EleutherAI

A.44 GPT-NeoX-20B

- **Reference:** (Black et al., 2022)

- **Link:** `https://huggingface.co/EleutherAI/gpt-neox-20b`

- **Family:** GPT

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** LM

- **Extension:** Similar to GPT-3 with rotary encoders instead of positional, parallel attention and feed forward layers, different initialization, and all dense layers instead of alternate dense/sparse

- **Application:** same as GPT-3

- **Date (of first known publication):** 04/2022

- **Num. Params:**20B

- **Corpus:** Pile — 840 GB open source text dataset that combines 22 pre existing datasets

- **License:** Open, Apache-2.0

- **Lab:** EleutherAI

A.45   HTLM

- **Reference:** (Aghajanyan et al., 2021)

- **Link:** N/A

- **Family:** BART

- **Pretraining Architecture:** Encoder/Decoder

- **Pretraining Task:** DAE

- **Extension:** As opposed to BART, they don't do sentence shuffling

- **Application:** General purpose language model that allows structured HTML prompting

- **Date (of first known publication):** 07/2021

- **Num. Params:**400M

- **Corpus:** 23TB of simplified HTML extracted from CommonCrawl

- **License:** N/A

- **Lab:** Facebook

A.46   Imagen

- **Reference:** (Saharia et al., 2022)

- **Link:** `https://imagen.research.google`

- **Family:** T5, CLIP, Diffusion models

- **Pretraining Architecture:** T5 (or CLIP or BERT) for frozen text encoder + U-net architecture for cascaded diffusion models for text to image

- **Pretraining Task:** image/text pair prediction

- **Extension:** Imagen adds a few extensions to the U-net diffusion architecture (pooled embedding vector, cross attention over text embeddings, and Layer Normalizations)

- **Application:** Text to image

- **Date (of first known publication):** 06/2022

- **Num. Params:**2B

- **Corpus:** a combination of internal datasets, with 460M image-text pairs, and the publicly available Laion dataset, with 400M image-text pairs

- **License:** Closed source

- **Lab:** Google

A.47   InstructGPT

- **Reference:** (Ouyang et al., 2022)

- **Link:** https://github.com/openai/following-instructions-human-feedback

- **Family:** GPT

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** LM

- **Extension:** GPTInstruct starts off with a pretrained GPT3 model and adds reward modeling through reinforcement learning after a supervised finetuning

- **Application:** Knowledge-intensive dialog or language tasks

- **Date (of first known publication):** 01/2022

- **Num. Params:** Same as GPT3

- **Corpus:** Same as GPT3 for pretraining, but finetuned and optimized using labeler data and prompts

- **License:** Closed source, Accessible through API

- **Lab:** OpenAI

A.48 InstructOR

- **Reference:** (Su et al., 2022)
- **Link:** https://huggingface.co/hkunlp/instructor-xl
- **Family:** T5
- **Pretraining Architecture:** Encoder/Decoder
- **Fine-tuning Tasks:** Wide variety of instruction based text-to-text tasks
- **Extension:** Fine-tunes T5 explicitly to optimize encoder to produce a general purpose text string embedding useful for many NLU tasks.
- **Application:** Any NLU task requiring a single text string embedding. As of April 2023 InstructOR is the top-ranked system on the Massive Text Embedding Benchmark (MTEB).[29]
- **Date (of first known publication):** 12/2022
- **Num. Params:** 330M
- **Corpus:** Finetuned on MEDI
- **License:** Open, Apache-2.0
- **Lab:** University of Hong Kong, University of Washington, META AI

A.49 Jurassic-1

- **Reference:** (Lieber et al., 2021)
- **Link:** https://github.com/ai21labs/lm-evaluation
- **Family:** GPT
- **Pretraining Architecture:** Decoder
- **Pretraining Task:** LM
- **Extension:** Very similar to GPT-3, but far more parameters and improved training efficiency mostly because of the improved tokenizer. Also, different ratio of depth to breadth
- **Application:** Similar to GPT-3
- **Date (of first known publication):** 09/2021
- **Num. Params:** 178B (Jumbo), 17B (Grande), 7.5B (Large)
- **Corpus:** 300B tokens (same as GPT-3)
- **License:** Closed source, accessible through API
- **Lab:** AI21

---

[29]https://huggingface.co/spaces/mteb/leaderboard

A.50   LAMDA

- **Reference:** (Thoppilan et al., 2022)

- **Link:** See blog post[30]

- **Family:** Transformer

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** LM

- **Extension:** LAMDA focuses on how to improve safety, quality, and groundeness using different fine-tuning strategies

- **Application:** General language modeling, such as translation, summarization, question and answers.

- **Date (of first known publication):** 01/2022

- **Num. Params:**137B

- **Corpus:** 1.56T words from public dialog data and other public web documents

- **License:** Closed source

- **Lab:** Google

A.51   LLaMA

- **Reference:** (Touvron et al., 2023)

- **Link:** https://huggingface.co/docs/transformers/main/model_doc/llama

- **Family:** Transformer

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** LM

- **Extension:** LLaMA uses a Transformer architecture, and with extensions: Pre-normalization, SwiGLU activations, RoPE embeddings, reduced memory usage and runtime through efficient implementation of the causal multi-head attention, checkpointing to reduce the amount of activations that are recomputed during the backward pass, model and sequence parallelism to reduce memory usage of the model, and uses 1.4T BPE tokens after tokenization.

---

[30]https://ai.googleblog.com/2022/01/lamda-towards-safe-grounded-and-high.html

- **Application:** Zero and few shot Commonsense reasoning, Question answering, Code generation and Reading comprehension.

- **Date (of first known publication):** 02/2023

- **Num. Params:** 7B, 13B, 33B and 65B

- **Corpus:** English CommonCrawl + C4 + Github + Wikipedia + Gutenberg and Books3 + ArXiv + Stack Exchange

- **License:** Limited, Non-commercial bespoke license

- **Lab:** Meta

A.52   mBART

- **Reference:** (Liu et al., 2020)

- **Link:** `https://huggingface.co/docs/transformers/model_doc/mbart`

- **Family:** BART

- **Pretraining Architecture:** Encoder/Decoder

- **Pretraining Task:** DAE

- **Extension:** Extends BART to multilingual capability

- **Application:** Translation

- **Date (of first known publication):** 01/2020

- **Num. Params:** Same as BART

- **Corpus:** CC25 Corpus includes 25 monolingual corpuses in different languages. Largest corpuses are English (300 GB) and Russian (280GB)

- **License:** Open, MIT license

- **Lab:** facebook

A.53   Megatron

- **Reference:** (Shoeybi et al., 2019)

- **Link:** `https://github.com/NVIDIA/Megatron-LM`

- **Family:** GPT/BERT/T5

- **Pretraining Architecture:** Encoder or Decorder, depending on the base model

- **Pretraining Task:** Same as base model

- **Extension:** Megatron is a family of models that extend previously known architectures (namely GPT-2 and BERT originally, but also T5 more recently) by introducing model parallelism primitives. In the case of BERT, the authors also replace the next sentence prediction head with sentence order prediction and use whole word n-gram masking.

- **Application:** Same as base model

- **Date (of first known publication):** 03/2020

- **Num. Params:** 8.3B (GPT-like), 3.9B (BERT-like)

- **Corpus:** Original paper uses an aggregate dataset consisting of Wikipedia), CC-Stories), RealNews, and OpenWebtext

- **License:** Limited, Non-commercial usage

- **Lab:** NVidia

A.54   Minerva

- **Reference:** (Lewkowycz et al., 2022)

- **Link:** See blog post[31]

- **Family:** PaLM

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** LM

- **Extension:** Extends PaLM by fine-tuning on the mathematical dataset

- **Application:** Mathematical reasoning

- **Date (of first known publication):** 06/2022

- **Num. Params:**540B

- **Corpus:** Same as PaLM + 118GB dataset of scientific papers from the arXiv preprint server and web pages that contain mathematical expressions using LaTeX, MathJax, or other mathematical typesetting formats

- **License:** Closed source

- **Lab:** Google

---

[31]https://ai.googleblog.com/2022/06/minerva-solving-quantitative-reasoning.html

A.55   MT-NLG (Megatron TuringNLG)

- **Reference:** (Smith et al., 2022)

- **Link:** See blog post[32]

- **Family:** GPT

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** LM

- **Extension:** Uses parallelization similar to Megatron to train a LM double the size of GPT-3

- **Application:** Language generation and others (similar to GPT-3)

- **Date (of first known publication):** 10/2021

- **Num. Params:**530B

- **Corpus:** The Pile[33] (800GB dataset) + 2 Common Crawl snapshots

- **License:** Limited, Non-commercial usage

- **Lab:** NVidia

A.56   OpenAssistant LLaMa

- **Reference:** N/A

- **Link:** https://open-assistant.io/

- **Family:** LLaMA

- **Pretraining Architecture:** Decoder

- **Extension:** Supervised fine-tuning on crowd sourced conversation/assistant data.

- **Application:** Same as ChatGPT, but open source. Compared to alternatives, it uses human generated conversation data

- **Date (of first known publication):** 04/2023

- **Num. Params:** 30B for LLaMa

- **Corpus:** Conversations collected by volunteers (Köpf et al., 2023) available at https://huggingface.co/datasets/OpenAssistant/oasst1

- **License:** Limited, Non-commercial bespoke license. There is also a version based on Pythia which is Apache licensed.

- **Lab:** Various open source contributors

---

[32]https://developer.nvidia.com/blog/using-deepspeed-and-megatron-to-train-megatron-turing-nlg-530b-the-worlds-
[33]https://arxiv.org/abs/2101.00027

A.57 OPT

- **Reference:** (Zhang et al., 2022)

- **Link:** See blog post[34]

- **Family:** GPT

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** LM

- **Extension:** Basically same architecture as GPT-3 but with some training improvements introduced in Megatron-LM

- **Application:** Same as GPT-3

- **Date (of first known publication):** 05/2022

- **Num. Params:** 175B (and other smaller versions)

- **Corpus:** 180B tokens = RoBERTa + the Pile + PushShift.io Reddit

- **License:** Limited, non-commercial license

- **Lab:** Facebook

A.58 PalM

- **Reference:** (Chowdhery et al., 2022)

- **Link:** See blog post[35]

- **Family:** Transformer

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** LM

- **Extension:** Palm uses a typical decoder-only Transformer architecture, but adds quite a few extensions: SwiGLU activations, parallel layers, multi-query attention, RoPE embeddings, Shared Input-Output Embeddings, no biases, and a 256k SentencePiece vocabulary generated from the training data.

- **Application:** PalM is designed as a general purpose language model with applicability to hundreds of different language tasks

- **Date (of first known publication):** 04/2022

---

[34]https://ai.facebook.com/blog/democratizing-access-to-large-scale-language-models-with-opt-175b/
[35]https://ai.googleblog.com/2022/04/pathways-language-model-palm-scaling-to.html

- **Num. Params:** 540B

- **Corpus:** 780B tokens from filtered webpages, books, Wikipedia, news articles, source code, and social media conversations. Code includes 24 programming languages.

- **License:** Closed source, Accessible through API

- **Lab:** Google

A.59 Pegasus

- **Reference:** (Zhang et al., 2020)

- **Link:** https://huggingface.co/docs/transformers/model_doc/pegasus

- **Family:** Transformer

- **Pretraining Architecture:** Encoder/Decoder

- **Pretraining Task:** DAE (more concretely GSG) and MLM

- **Extension:** Extends vanilla Transformer by using a different pretraining task (GSG: Gap Sentence Generation) that is better suited for summarization

- **Application:** Summarization

- **Date (of first known publication):** 12/2019

- **Num. Params:** Base = 223M, Large = 568M

- **Corpus:** C4 (750GB) + HugeNews (3.8 TB)

- **License:** N/A

- **Lab:** UCL/Google

A.60 Pythia

- **Reference:** (Biderman et al., 2023)

- **Link:** https://github.com/EleutherAI/pythia

- **Family:** Pythia

- **Pretraining Architecture:** Decoder

- **Extension:** Trained with the library GPT-NeoX

- **Application:** Research on language model's behavior, functionality, and limitations.

- **Date (of first known publication):** 04/2023

- **Num. Params:** 70M, 160M, 410M, 1B, 1.4B, 2.8B, 6.9B, 12B

- **Corpus:** Pile

- **License:** Open, Apache-2.0

- **Lab:** Eleuther AI

A.61   RoBERTa

- **Reference:** (Liu et al., 2019)

- **Link:** https://huggingface.co/docs/transformers/model_doc/roberta

- **Family:** BERT

- **Pretraining Architecture:** Encoder

- **Pretraining Task:** MLM (Dynamic)

- **Extension:** Extension of BERT with optimized training procedure and more data

- **Application:** Same as BERT

- **Date (of first known publication):** 07/2019

- **Num. Params:** 356M

- **Corpus:** Same as BERT + CC News + OpenWebText + Stories (33B Tokens)

- **License:** N/A

- **Lab:** UW/Google

A.62   SeeKer

- **Reference:** (Shuster et al., 2022a)

- **Link:** https://parl.ai/projects/seeker

- **Family:** GPT (but can extend any family)

- **Pretraining Architecture:** Encoder/decoder or decoder only, depending on the base model it's extending

- **Pretraining Task:** LM training, Dialogue training

- **Extension:** SeeKer is an extension that can be applied to any Transformer architecture by introducing "search", "knowledge", and "response" modules that are introduced during pretraining

- **Application:** Same as base models

- **Date (of first known publication):** 03/2022

- **Num. Params:** SeeKeR Dialogue: 400M, 3B; SeeKeR LM: 365M, 762M, 1.5B, R2C2 BlenderBot: 400M, 3B

- **Corpus:** Wizard of the Internet/Wikipedia, PersonaChat, Blended Skill Talk, Empatheic Dialogues, Multi-Session Chat, MS MARCO, Natural questions, SQuAD, TriviaQA

- **License:** the code is open sourced.

- **Lab:** Facebook

A.63 Sparrow

- **Reference:** (Glaese et al., 2022)

- **Link:** N/A

- **Family:** GPT

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** LM

- **Extension:** Starts from the Chinchilla 70B model but adds RLHF (Reinforcement Learning with Human Feedback). It also adds inline evidence a la GopherCite

- **Application:** Dialog agents and general language generation applications like Q&A

- **Date (of first known publication):** 09/2022

- **Num. Params:** 70B

- **Corpus:** Same as Chinchilla + interactive data gathering with human annotators during the RLHF process

- **License:** Closed source

- **Lab:** Deepmind

A.64 StableDiffusion

- **Reference:** (Rombach et al., 2022)

- **Link:** https://huggingface.co/CompVis/stable-diffusion

- **Family:** Diffusion

- **Pretraining Architecture:** Encoder/Decoder

- **Pretraining Task:** Caption prediction

- **Extension:** Stable diffusion is basically the Latent Diffusion model developed by LMU Munich researchers + some learnings on conditional diffusion from DALL-e and Imagen

- **Application:** Text to image

- **Date (of first known publication):** 12/2021

- **Num. Params:** 890M (although there are different, smaller, variants)

- **Corpus:** LAION-5B, a publicly available dataset derived from Common Crawl

- **License:** open, CreativeML Open RAIL++-M License

- **Lab:** LMU Munich + Stability.ai + Eleuther.ai

A.65 Swin Transformer

- **Reference:** (Liu et al., 2021)

- **Link:** https://github.com/microsoft/Swin-Transformer

- **Family:** ViT

- **Pretraining Architecture:** Encoder

- **Pretraining Task:** Same as ViT

- **Extension:** Extends ViT by replacing the standard multi-head self attention (MSA) module by a module based on shifted windows (Swin) allowing ViT-like architectures to generalize to higher resolution images

- **Application:** Image (object detection, image classification..)

- **Date (of first known publication):** 03/2021

- **Num. Params:** 29M-197M

- **Corpus:** Imagenet and Imagenet-22k

- **License:** the code is open sourced, with MIT-license

- **Lab:** Microsoft

A.66 Switch

- **Reference:** (Fedus et al., 2021)

- **Link:** `https://github.com/google-research/t5x`

- **Family:** T5

- **Pretraining Architecture:** Encoder/Decoder

- **Pretraining Task:** DAE

- **Extension:** Goal to increase parameter count while keeping FLOP operations constant by using efficient routing of MoE (Mixture of Experts)

- **Application:** General language tasks (e.g. question answering)

- **Date (of first known publication):** 01/2021

- **Num. Params:** 1T

- **Corpus:** Colossal Clean Crawled Corpus

- **License:** Open, Apache-2.0

- **Lab:** Google

A.67 T0

- **Reference:** (Sanh et al., 2021)

- **Link:** `https://huggingface.co/bigscience/T0`

- **Family:** T5

- **Pretraining Architecture:** Encoder/Decoder

- **Fine-tuning Task:** Natural language prompts

- **Extension:** T0 stands for "T5 for Zero Shot", obtained by fine-tuning the T5 model on multitask mixture covering many different NLP tasks. Compared with T0, T0p and T0pp were fine-tuned with more datasets. T0pp is recommended as it leads (on average) to the best performances on a variety of NLP tasks.

- **Application:** Perform zero-shot inference tasks by specifying the query in natural language, and the models will generate a prediction.

- **Date (of first known publication):** 03/2022

- **Num. Params:** T0-3B: 3 billion, T0, T0p, T0pp: 11 billion

- **Corpus:** T0 (Multiple-choice QA, Extractive QA, Closed-Book QA, Structure-To-Text, Sentiment, Summarization, Topic Classification, Paraphrase Identification. T0p (same as T0, with additional datasets from GPT-3's evaluation suite). T0pp (same as T0p, with additional datasets from SuperGLUE, excluding NLI sets)

- **License:** Open, Apache-2.0

- **Lab:** BigScience

A.68 T5

- **Reference:** (Raffel et al., 2020)

- **Link:** https://huggingface.co/docs/transformers/model_doc/t5

- **Family:** Transformer

- **Pretraining Architecture:** Encoder/Decoder

- **Pretraining Task:** DAE

- **Extension:** Same as original Transformer with some additions such as relative positional embeddings like Transformer XL

- **Application:** General language tasks including machine translation, question answering, abstractive summarization, and text classification

- **Date (of first known publication):** 10/2019

- **Num. Params:** 11 B (up to)

- **Corpus:** Colossal Clean Crawled Corpus (C4) — Cleaned up version of the Common Crawl dataset — 750 GB

- **License:** Open, Apache-2.0

- **Lab:** Google

A.69 Trajectory Transformers

- **Reference:** (Janner et al., 2021)

- **Link:** https://trajectory-transformer.github.io

- **Family:** GPT, Control Transformers" (not per se a family, but grouping here those Transformers that try to model more general control, RL-like, tasks)

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** predict most likely sequence

- **Extension:** Similarly to the Decision Transformers, the main extension introduced by Trajectory Transformers is a way to encode a trajectory (state, actions, rewards)

- **Application:** General RL (reinforcement learning tasks)

- **Date (of first known publication):** 06/2021

- **Num. Params:** Smaller architecture than GPT

- **Corpus:** D4RL dataset and other RL datasets depending on the task at hand

- **License:** Open, MIT license

- **Lab:** UC Berkeley

A.70   Transformer XL

- **Reference:** (Dai et al., 2019)

- **Link:** `https://huggingface.co/docs/transformers/model_doc/transfo-xl`

- **Family:**

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** LM

- **Extension:** Relative positioned embeddings enable longer-context attention when compared to vanilla Transformer model

- **Application:** General language tasks

- **Date (of first known publication):** 01/2019

- **Num. Params:** 151M

- **Corpus:** Different training datasets depending on experiments, but baseline is Wikitext-103

- **License:** N/A

- **Lab:** CMU/Google

A.71   Turing-NLG

- **Reference:** (Rosset, 2020)

- **Link:** N/A

- **Family:** GPT

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** LM

- **Extension:** Optimized version of GPT2 with optimal hyperparameters and software/hardware platform to improve training

- **Application:** Same as GPT-2/3

- **Date (of first known publication):** 02/2020

- **Num. Params:** 17B originally, up to 530B more recently

- **Corpus:** Highest quality subset from The Pile + 2 CC snapshots (339B tokens)

- **License:** N/A

- **Lab:** Microsoft

A.72   UL2

- **Reference:** (Tay et al., 2022)

- **Link:** `https://github.com/google-research/google-research/tree/master/ul2`

- **Family:** Transformer

- **Pretraining Architecture:** Encoder/Decoder

- **Pretraining Task:** Mixture-of-Denoisers, which combines diverse pre-training paradigms together

- **Extension:** UL2-20B (Unifying Language Learning) can be interpreted as a model that is quite similar to T5 but trained with a different objective and slightly different scaling knobs.

- **Application:** A unified framework for pre-training models that are universally effective across datasets and setups.

- **Date (of first known publication):** 05/2022

- **Num. Params:** 20B

- **Corpus:** 1 trillion tokens on C4

- **License:** Open, Apache-2.0

- **Lab:** Google

A.73   Vicuna

- **Reference:** N/A

- **Link:** `https://vicuna.lmsys.org`

- **Family:** LLaMA

- **Pretraining Architecture:** Decoder

- **Fine-tuning Task:** human instructions

- **Extension:** LLaMA fine-tuned on user-shared conversations collected from ShareGPT.

- **Application:** Same as ChatGPT

- **Date (of first known publication):** 03/2023

- **Num. Params:** 13B

- **Corpus:** Conversations collected from ShareGPT

- **License:** Limited, Non-commercial bespoke license

- **Lab:** UC Berkeley, CMU, Stanford, UC San Diego, and MBZUAI

A.74   ViT

- **Reference:** (Dosovitskiy et al., 2020)

- **Link:** `https://huggingface.co/docs/transformers/model_doc/vit`

- **Family:** BERT

- **Pretraining Architecture:** Encoder

- **Pretraining Task:** Image classification

- **Extension:** Extension of BERT architecture to train on patches of images

- **Application:** Image classification

- **Date (of first known publication):** 10/2020

- **Num. Params:** 86M(Base) to 632M (Huge)

- **Corpus:** From standard Imagenet to JFT-300M (large inhouse dataset)

- **License:** N/A

- **Lab:** Google

A.75 Wu Dao 2.0

- **Reference:** See Wikipedia page[36]

- **Link:** See blog post[37]

- **Family:** GLM (General Language Model)

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** Autoregressive blank infilling

- **Extension:** Similar to GPT in that it uses a Decoder/autoregressive architecture but applies a different pretraining task proposed in the GLM family of models. Besides, Wu Dao uses a Fast Mixture of Experts (see `https://github.com/laekov/fastmoe`) approach to scale training to trillions of parameters

- **Application:** Language and multimodal (particularly image)

- **Date (of first known publication):** 06/2021

- **Num. Params:** 1.75T

- **Corpus:** 4.9 TB of high quality images and texts in both English and Chinese

- **License:** Closed source

- **Lab:** Beijing Academy of Artificial Intelligence

A.76 XLM-RoBERTa

- **Reference:** (Conneau et al., 2019)

- **Link:** `https://huggingface.co/docs/transformers/model_doc/xlm-roberta`

- **Family:** RoBERTa

- **Pretraining Architecture:** Encoder

- **Pretraining Task:** MLM (Dynamic)

- **Extension:** An extension of RoBERTa that introduces parameter tuning insights in the context of multilingual applications

- **Application:** Translation and other cross-lingual language tasks

- **Date (of first known publication):** 10/2019

- **Num. Params:** Base = 270M, Large = 550M

---

[36]`https://en.wikipedia.org/wiki/Wu_Dao`
[37]`https://mp.weixin.qq.com/s/BUQWZ5EdR19i4OGuFofpBg`

- **Corpus:** Cleaned Common Crawl in 100 languages

- **License:** Open, MIT license

- **Lab:** Facebook

A.77 XLNet

- **Reference:** (Yang et al., 2019)

- **Link:** `https://huggingface.co/docs/transformers/model_doc/xlnet`

- **Family:** Transformer XL

- **Pretraining Architecture:** Decoder

- **Pretraining Task:** PLM

- **Extension:** This model basically adapts Transformer XL architecture to permutation-based LM

- **Application:** General language tasks

- **Date (of first known publication):** 05/2019

- **Num. Params:** Base=117M, Large=360M

- **Corpus:** Same as BERT + Giga5 (16GB text) + and aggressively filtered ClueWeb 2012-B (19GB), Common Crawl (110 GB)

- **License:** Open, MIT license

- **Lab:** CMU/Google

References

Aghajanyan, A., Gupta, A., Shrivastava, A., Chen, X., Zettlemoyer, L., & Gupta, S. (2021). Muppet: Massive Multi-task Representations with Pre-Finetuning. `https://arxiv.org/abs/2101.11038`.

Aghajanyan, A., Huang, B., Ross, C., Karpukhin, V., Xu, H., Goyal, N., Okhonko, D., Joshi, M., Ghosh, G., Lewis, M., et al. (2022). Cm3: A causal masked multimodal model of the internet. `https://arxiv.org/abs/2201.07520`.

Aghajanyan, A., Okhonko, D., Lewis, M., Joshi, M., Xu, H., Ghosh, G., & Zettlemoyer, L. (2021). Htlm: Hyper-text pre-training and prompting of language models. `https://arxiv.org/abs/2107.06955`.

Alayrac, J.-B., Donahue, J., Luc, P., Miech, A., Barr, I., Hasson, Y., Lenc, K., Mensch, A., Millican, K., Reynolds, M., et al. (2022). Flamingo: a visual language model for few-shot learning. `https://arxiv.org/abs/2204.14198`.

Askell, A., Bai, Y., Chen, A., Drain, D., Ganguli, D., Henighan, T., Jones, A., Joseph, N., Mann, B., DasSarma, N., et al. (2021). A general language assistant as a laboratory for alignment. `https://arxiv.org/abs/2112.00861`.

Bai, Y., Jones, A., Ndousse, K., Askell, A., Chen, A., DasSarma, N., Drain, D., Fort, S., Ganguli, D., Henighan, T., Joseph, N., Kadavath, S., Kernion, J., Conerly, T., El-Showk, S., Elhage, N., Hatfield-Dodds, Z., Hernandez, D., Hume, T., Johnston, S., Kravec, S., Lovitt, L., Nanda, N., Olsson, C., Amodei, D., Brown, T., Clark, J., McCandlish, S., Olah, C., Mann, B., & Kaplan, J. (2022a). Training a helpful and harmless assistant with reinforcement learning from human feedback. `https://arxiv.org/abs/2204.05862`.

Bai, Y., Kadavath, S., Kundu, S., Askell, A., Kernion, J., Jones, A., Chen, A., Goldie, A., Mirhoseini, A., McKinnon, C., Chen, C., Olsson, C., Olah, C., Hernandez, D., Drain, D., Ganguli, D., Li, D., Tran-Johnson, E., Perez, E., Kerr, J., Mueller, J., Ladish, J., Landau, J., Ndousse, K., Lukosuite, K., Lovitt, L., Sellitto, M., Elhage, N., Schiefer, N., Mercado, N., DasSarma, N., Lasenby, R., Larson, R., Ringer, S., Johnston, S., Kravec, S., Showk, S. E., Fort, S., Lanham, T., Telleen-Lawton, T., Conerly, T., Henighan, T., Hume, T., Bowman, S. R., Hatfield-Dodds, Z., Mann, B., Amodei, D., Joseph, N., McCandlish, S., Brown, T., & Kaplan, J. (2022b). Constitutional AI: Harmlessness from AI Feedback. `https://arxiv.org/abs/2212.08073`.

Baidoo-Anu, D., & Owusu Ansah, L. (2023). Education in the era of generative artificial intelligence (AI): Understanding the potential benefits of ChatGPT in promoting teaching and learning. `https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4337484`.

Biderman, S., Schoelkopf, H., & and, Q. A. (2023). Pythia: A Suite for Analyzing Large Language Models Across Training and Scaling. `https://arxiv.org/abs/2304.01373`.

Black, S., Biderman, S., Hallahan, E., Anthony, Q., Gao, L., Golding, L., He, H., Leahy, C., McDonell, K., Phang, J., et al. (2022). Gpt-neox-20b: An open-source autoregressive language model. `https://arxiv.org/abs/2204.06745`.

Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M. S., Bohg, J., Bosselut, A., Brunskill, E., et al. (2021). On the opportunities and risks of foundation models. `https://arxiv.org/abs/2108.07258`.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. *Advances in Neural Information Processing Systems*, *33*, 1877–1901.

Chen, L., Lu, K., Rajeswaran, A., Lee, K., Grover, A., Laskin, M., Abbeel, P., Srinivas, A., & Mordatch, I. (2021). Decision transformer: Reinforcement learning via sequence modeling. *Advances in Neural Information Processing Systems*, *34*, 15084–15097.

Cho, K., Van Merriënboer, B., Bahdanau, D., & Bengio, Y. (2014). On the properties of neural machine translation: Encoder-decoder approaches. `https://arxiv.org/abs/1409.1259`.

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., et al. (2022). Palm: Scaling language modeling with pathways. `https://arxiv.org/abs/2204.02311`.

Christiano, P., Leike, J., Brown, T. B., Martic, M., Legg, S., & Amodei, D. (2023). Deep reinforcement learning from human preferences. `https://arxiv.org/abs/1706.03741`.

Chung, H. W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., Li, Y., Wang, X., Dehghani, M., Brahma, S., Webson, A., Gu, S. S., Dai, Z., Suzgun, M., Chen, X., Chowdhery, A., Castro-Ros, A., Pellat, M., Robinson, K., Valter, D., Narang, S., Mishra, G., Yu, A., Zhao, V., Huang, Y., Dai, A., Yu, H., Petrov, S., Chi, E. H., Dean, J., Devlin, J., Roberts, A., Zhou, D., Le, Q. V., & Wei, J. (2022). Scaling Instruction-Finetuned Language Models. `https://arxiv.org/abs/2210.11416`.

Clark, K., Luong, M.-T., Le, Q. V., & Manning, C. D. (2020). Electra: Pre-training text encoders as discriminators rather than generators. `https://arxiv.org/abs/2003.10555`.

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., & Stoyanov, V. (2019). Unsupervised cross-lingual representation learning at scale. `https://arxiv.org/abs/1911.02116`.

Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le, Q. V., & Salakhutdinov, R. (2019). Transformer-xl: Attentive language models beyond a fixed-length context. `https://arxiv.org/abs/1901.02860`.

Dennean, K., Gantori, S., Limas, D. K., & Allen Pu, a. R. G. (2023). Let's chat about ChatGPT. `https://www.ubs.com/global/en/wealth-management/our-approach/marketnews/article.1585717.html`.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. `https://arxiv.org/abs/1810.04805`.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. https://arxiv.org/abs/2010.11929.

Du, N., Huang, Y., Dai, A. M., Tong, S., Lepikhin, D., Xu, Y., Krikun, M., Zhou, Y., Yu, A. W., Firat, O., et al. (2022a). Glam: Efficient scaling of language models with mixture-of-experts. In *International Conference on Machine Learning*, pp. 5547–5569. PMLR.

Du, Z., Qian, Y., Liu, X., Ding, M., Qiu, J., Yang, Z., & Tang, J. (2022b). GLM: General Language Model Pretraining with Autoregressive Blank Infilling. https://arxiv.org/abs/2103.10360.

Esser, P., Rombach, R., & Ommer, B. (2021). Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12873–12883.

Fedus, W., Zoph, B., & Shazeer, N. (2021). Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, *23*, 1–40.

Fuchs, F. B., Worrall, D. E., Fischer, V., & Welling, M. (2020). SE(3)-Transformers: 3D roto-translation equivariant attention networks. https://arxiv.org/abs/2006.10503.

Glaese, A., McAleese, N., Trębacz, M., Aslanides, J., Firoiu, V., Ewalds, T., Rauh, M., Weidinger, L., Chadwick, M., Thacker, P., et al. (2022). Improving alignment of dialogue agents via targeted human judgements. https://arxiv.org/abs/2209.14375.

Hatamizadeh, A., Yin, H., Kautz, J., & Molchanov, P. (2022). Global context vision transformers. https://arxiv.org/abs/2206.09959.

He, P., Liu, X., Gao, J., & Chen, W. (2021). DeBERTa: Decoding-enhanced BERT with disentangled attention. In *International Conference on Learning Representations*.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735–1780.

Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., Casas, D. d. L., Hendricks, L. A., Welbl, J., Clark, A., et al. (2022). Training compute-optimal large language models. https://arxiv.org/abs/2203.15556.

Janner, M., Li, Q., & Levine, S. (2021). Offline reinforcement learning as one big sequence modeling problem. *Advances in Neural Information Processing Systems*, *34*, 1273–1286.

Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Zidek, A., Potapenko, A., et al. (2021). Highly accurate protein structure prediction with AlphaFold. *Nature*, *596*(7873), 583–589.

Keskar, N. S., McCann, B., Varshney, L. R., Xiong, C., & Socher, R. (2019). Ctrl: A conditional transformer language model for controllable generation. `https://arxiv.org/abs/1909.05858`.

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. (2023). OpenAssistant Conversations – Democratizing Large Language Model Alignment. `https://arxiv.org/abs/2304.07327`.

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., & Soricut, R. (2019). ALBERT: A lite BERT for self-supervised learning of language representations. `https://arxiv.org/abs/1909.11942`.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., & Zettlemoyer, L. (2019). Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. `https://arxiv.org/abs/1910.13461`.

Lewkowycz, A., Andreassen, A., Dohan, D., Dyer, E., Michalewski, H., Ramasesh, V., Slone, A., Anil, C., Schlag, I., Gutman-Solo, T., et al. (2022). Solving quantitative reasoning problems with language models. `https://arxiv.org/abs/2206.14858`.

Li, Z., Wang, Z., Tan, M., Nallapati, R., Bhatia, P., Arnold, A., Xiang, B., & Roth, D. (2022). DQ-BART: Efficient Sequence-to-Sequence Model via Joint Distillation and Quantization. `https://arxiv.org/abs/2203.11239`.

Lieber, O., Sharir, O., Lenz, B., & Shoham, Y. (2021). Jurassic-1: Technical details and evaluation. `https://uploads-ssl.webflow.com/60fd4503684b466578c0d307/61138924626a6981ee09caf6_jurassic_tech_paper.pdf`.

Liu, X., He, P., Chen, W., & Gao, J. (2019). Multi-Task Deep Neural Networks for Natural Language Understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4487–4496 Florence, Italy. Association for Computational Linguistics.

Liu, Y., Gu, J., Goyal, N., Li, X., Edunov, S., Ghazvininejad, M., Lewis, M., & Zettlemoyer, L. (2020). Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, *8*, 726–742.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. `https://arxiv.org/abs/1907.11692`.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., & Guo, B. (2021). Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10012–10022.

Menick, J., Trebacz, M., Mikulik, V., Aslanides, J., Song, F., Chadwick, M., Glaese, M., Young, S., Campbell-Gillingham, L., Irving, G., et al. (2022). Teaching language models to support answers with verified quotes. `https://arxiv.org/abs/2203.11147`.

Mikolov, T., Karafiat, M., Burget, L., Cernocky, J., & Khudanpur, S. (2010). Recurrent neural network based language model. In *Interspeech*, Vol. 2, pp. 1045–1048.

Nichol, A., Dhariwal, P., Ramesh, A., Shyam, P., Mishkin, P., McGrew, B., Sutskever, I., & Chen, M. (2021). Glide: Towards photorealistic image generation and editing with text-guided diffusion models. `https://arxiv.org/abs/2112.10741`.

OpenAI (2023). GPT-4 Technical Report. `https://arxiv.org/abs/2303.08774`.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. (2022). Training language models to follow instructions with human feedback. `https://arxiv.org/abs/2203.02155`.

Qiu, X., Sun, T., Xu, Y., Shao, Y., Dai, N., & Huang, X. (2020). Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, *63*(10), 1872–1897.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. (2021). Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR.

Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., et al. (2018). Improving language understanding by generative pre-training. `https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf`.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. (2019). Language models are unsupervised multitask learners. `https://paperswithcode.com/paper/language-models-are-unsupervised-multitask`.

Rae, J. W., Borgeaud, S., Cai, T., Millican, K., Hoffmann, J., Song, F., Aslanides, J., Henderson, S., Ring, R., Young, S., et al. (2021). Scaling language models: Methods, analysis & insights from training gopher. `https://arxiv.org/abs/2112.11446`.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., & Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, *21*(1), 5485–5551.

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., & Chen, M. (2022). Hierarchical text-conditional image generation with clip latents. `https://arxiv.org/abs/2204.06125`.

Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., & Sutskever, I. (2021). Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pp. 8821–8831. PMLR.

Reed, S., Zolna, K., Parisotto, E., Colmenarejo, S. G., Novikov, A., Barth-Maron, G., Gimenez, M., Sulsky, Y., Kay, J., Springenberg, J. T., et al. (2022). A generalist agent. `https://arxiv.org/abs/2205.06175`.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., & Ommer, B. (2022). High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695.

Rosset, C. (2020). Turing-NLG: A 17-billion-parameter language model by Microsoft. `https://www.microsoft.com/en-us/research/blog/turing-nlg-a-17-billion-parameter-language-model-by-microsoft/`.

Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E., Ghasemipour, S. K. S., Ayan, B. K., Mahdavi, S. S., Lopes, R. G., et al. (2022). Photorealistic text-to-image diffusion models with deep language understanding. `https://arxiv.org/abs/2205.11487`.

Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. `https://arxiv.org/abs/1910.01108`.

Sanh, V., Webson, A., Raffel, C., Bach, S. H., & Lintang Sutawika, e. a. (2021). Multitask Prompted Training Enables Zero-Shot Task Generalization. `https://arxiv.org/abs/2110.08207`.

Shoeybi, M., Patwary, M., Puri, R., LeGresley, P., Casper, J., & Catanzaro, B. (2019). Megatron-lm: Training multi-billion parameter language models using model parallelism. `https://arxiv.org/abs/1909.08053`.

Shuster, K., Komeili, M., Adolphs, L., Roller, S., Szlam, A., & Weston, J. (2022a). Language models that seek for knowledge: Modular search & generation for dialogue and prompt completion. `https://arxiv.org/abs/2203.13224`.

Shuster, K., Xu, J., Komeili, M., Ju, D., Smith, E. M., Roller, S., Ung, M., Chen, M., Arora, K., Lane, J., et al. (2022b). Blenderbot 3: a deployed conversational agent that continually learns to responsibly engage. `https://arxiv.org/abs/2208.03188`.

Smith, S., Patwary, M., Norick, B., LeGresley, P., Rajbhandari, S., Casper, J., Liu, Z., Prabhumoye, S., Zerveas, G., Korthikanti, V., et al. (2022). Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model. `https://arxiv.org/abs/2201.11990`.

Soltan, S., Ananthakrishnan, S., FitzGerald, J., Gupta, R., Hamza, W., Khan, H., Peris, C., Rawls, S., Rosenbaum, A., Rumshisky, A., Prakash, C. S., Sridhar, M., Triefenbach, F., Verma, A., Tur, G., & Natarajan, P. (2022). AlexaTM 20B: Few-Shot Learning Using a Large-Scale Multilingual Seq2Seq Model. `https://arxiv.org/abs/2208.01448`.

Stokel-Walker, C., & Noorden, R. V. (2023). What ChatGPT and generative AI mean for science. `https://www.nature.com/articles/d41586-023-00340-6`.

Su, H. S., Shi, W. S., Kasai, J., Wang, Y., Hu, Y., Ostendorf, M., Yih, W.-t., Smith, N. A., Zettlemoyer, L., & Yu, T. (2022). One Embedder, Any Task: Instruction-Finetuned Text Embeddings. `https://arxiv.org/abs/2212.09741`.

Taori, R., Gulrajani, I., Zhang, T., Dubois, Y., Li, X., Guestrin, C., Liang, P., & Hashimoto, T. B. (2023). Stanford Alpaca: An Instruction-following LLaMA model. `https://github.com/tatsu-lab/stanford_alpaca`.

Tay, Y., Dehghani, M., Tran, V. Q., Garcia, X., Bahri, D., Schuster, T., Zheng, H. S., Houlsby, N., & Metzler, D. (2022). Unifying Language Learning Paradigms. `https://arxiv.org/abs/2205.05131`.

Taylor, R., Kardas, M., Cucurull, G., Scialom, T., Hartshorn, A., Saravia, E., Poulton, A., Kerkez, V., & Stojnic, R. (2022). GALACTICA: A Large Language Model for Science. `https://arxiv.org/abs/2211.09085`.

Thoppilan, R., De Freitas, D., Hall, J., Shazeer, N., Kulshreshtha, A., Cheng, H.-T., Jin, A., Bos, T., Baker, L., Du, Y., et al. (2022). Lamda: Language models for dialog applications. `https://arxiv.org/abs/2201.08239`.

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. (2023). LLaMA: Open and Efficient Foundation Language Models. `https://arxiv.org/abs/2302.13971`.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, *30*.

Wang, B., & Komatsuzaki, A. (2021). GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. `https://github.com/kingoflolz/mesh-transformer-jax`.

Wang, L., Yang, N., Huang, X., Jiao, B., Yang, L., Jiang, D., Majumder, R., & Wei, F. (2022). Text Embeddings by Weakly-Supervised Contrastive Pre-training. `https://arxiv.org/abs/2212.03533`.

Yang, L., Zhang, Z., Song, Y., Hong, S., Xu, R., Zhao, Y., Shao, Y., Zhang, W., Cui, B., & Yang, M.-H. (2022). Diffusion models: A comprehensive survey of methods and applications. `https://arxiv.org/abs/2209.00796`.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., & Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in Neural Information Processing Systems*, *32*.

Zaheer, M., Guruganesh, G., Dubey, K. A., Ainslie, J., Alberti, C., Ontanon, S., Pham, P., Ravula, A., Wang, Q., Yang, L., et al. (2020). Big bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems*, *33*, 17283–17297.

Zhang, J., Zhao, Y., Saleh, M., & Liu, P. (2020). Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pp. 11328–11339. PMLR.

Zhang, S., Roller, S., Goyal, N., Artetxe, M., Chen, M., Chen, S., Dewan, C., Diab, M., Li, X., Lin, X. V., et al. (2022). Opt: Open pre-trained transformer language models. `https://arxiv.org/abs/2205.01068`.

Zhang, Y., Sun, S., Galley, M., Chen, Y.-C., Brockett, C., Gao, X., Gao, J., Liu, J., & Dolan, B. (2019a). Dialogpt: Large-scale generative pre-training for conversational response generation. `https://arxiv.org/abs/1911.00536`.

Zhang, Z., Han, X., Liu, Z., Jiang, X., Sun, M., & Liu, Q. (2019b). ERNIE: Enhanced language representation with informative entities. `https://arxiv.org/abs/1905.07129`.