



Large Language Model

Can J.A.R.V.I.S becomes reality?

Author: occupymars

Date: June. 23, 2025

Version: 0.1

Contents

Chapter 1 Survey	1
1.1 Large Language Models: A Survey	1
Chapter 2 History Language Models	4
2.1 Statistical Language Model	4

Chapter 1 Survey

1.1 Large Language Models: A Survey

Introduction

□ Early Pretrained Large Language Models

More details please refer to [Large Language Models: A Survey](#).

1.1.1 Introduction

Statistical language models are not good because of the data sparsity, early neural language models are task specific.

Large language models have emergent abilities: *in-context learning*, where LLMs learn a new task from a small set of examples presented in the prompt at inference time; *instruction following*, where LLMs, after instruction tuning, can follow the instructions for new types of tasks without using explicit examples; *multi-step reasoning*, where LLMs can solve a complex task by breaking down that task into intermediate reasoning steps.

Type	Model Name	#Parameters	Release	Base Models	Open Source	#Tokens	Training dataset
Encoder-Only	BERT	110M, 340M	2018	-	✓	137B	BooksCorpus, English Wikipedia
	RoBERTa	355M	2019	-	✓	2.2T	BooksCorpus, English Wikipedia, CC-NEWS, STORIES (a subset of Common Crawl), Reddit
	ALBERT	12M, 18M, 60M, 235M	2019	-	✓	137B	BooksCorpus, English Wikipedia
	DeBERTa	-	2020	-	✓	-	BooksCorpus, English Wikipedia, STORIES, Reddit content
	XLNet	110M, 340M	2019	-	✓	32.89B	BooksCorpus, English Wikipedia, Giga5, Common Crawl, ClueWeb 2012-B
Decoder-only	GPT-1	120M	2018	-	✓	1.3B	BooksCorpus
	GPT-2	1.5B	2019	-	✓	10B	Reddit outboud
Encoder-Decoder	T5 (Base)	223M	2019	-	✓	156B	Common Crawl
	MT5 (Base)	300M	2020	-	✓	-	New Common Crawl-based dataset in 101 languages (m Common Crawl)
	BART (Base)	139M	2019	-	✓	-	Corrupting text
GPT Family	GPT-3	125M, 350M, 760M, 1.3B, 2.7B, 6.7B, 13B, 175B	2020	-	×	300B	Common Crawl (filtered), WebText2, Books1, Books2, Wikipedia
	CODEX	12B	2021	GPT	✓	-	Public GitHub software repositories
	WebGPT	760M, 13B, 175B	2021	GPT-3	×	-	ELI5
	GPT-4	1.76T	2023	-	×	13T	-
	LLaMA1	7B, 13B, 33B, 65B	2023	-	✓	1T, 1.4T	Online sources
LLaMA Family	LLaMA2	7B, 13B, 34B, 70B	2023	-	✓	2T	Online sources
	Alpaca	7B	2023	LLaMA1	✓	-	GPT-3.5
	Vicuna-13B	13B	2023	LLaMA1	✓	-	GPT-3.5
	Koala	13B	2023	LLaMA	✓	-	Dialogue data
	Mistral-7B	7.3B	2023	-	✓	-	-
	Code Llama	34	2023	LLaMA2	✓	500B	Publicly available code
	LongLLaMA	3B, 7B	2023	OpenLLaMA	✓	1T	-
	LLaMA-Pro-8B	8.3B	2024	LLaMA2-7B	✓	80B	Code and math corpora
	TinyLlama-1.1B	1.1B	2024	LLaMA1.1B	✓	3T	SlimPajama, StarCoderdata
	PaLM	8B, 62B, 540B	2022	-	×	780B	Web documents, books, Wikipedia, conversations, GitHub code
PaLM Family	U-PaLM	8B, 62B, 540B	2022	-	×	1.3B	Web documents, books, Wikipedia, conversations, GitHub code
	PaLM-2	340B	2023	-	✓	3.6T	Web documents, books, code, mathematics, conversational data
	Med-PaLM	540B	2022	PaLM	×	780B	HealthSearchQA, MedicationQA, LiveQA
	Med-PaLM 2	-	2023	PaLM 2	×	-	MedQA, MedMCQA, HealthSearchQA, LiveQA, MedicationQA
	FLAN	137B	2021	LaMDA-PT	✓	-	Web documents, code, dialog data, Wikipedia
Other Popular LLMs	Gopher	280B	2021	-	×	300B	MassiveText
	ERNIE 4.0	10B	2023	-	×	4TB	Chinese text
	Retro	7.5B	2021	-	×	600B	MassiveText
	LaMDA	137B	2022	-	×	168B	public dialog data and web documents
	Chinchilla	70B	2022	-	×	1.4T	MassiveText
	Galactia-120B	120B	2022	-	-	450B	-
	CodeGen	16.1B	2022	-	✓	-	THE PILE, BIGQUERY, BIGPYTHON
	BLOOM	176B	2022	-	✓	366B	ROOTS
	Zephyr	7.24B	2023	Mistral-7B	✓	800B	Synthetic data
	Grok-0	33B	2023	-	×	-	Online source
	ORCA-2	13B	2023	LLaMA2	-	2001B	-
	StarCoder	15.5B	2023	-	✓	35B	GitHub
	MPT	7B	2023	-	✓	1T	RedPajama, m Common Crawl, S2ORC, Common Crawl
	Mistral-8x7B	46.7B	2023	-	✓	-	Instruction dataset
	Falcon 180B	180B	2023	-	✓	3.5T	RefinedWeb
	Gemini	1.8B, 3.25B	2023	-	✓	-	Web documents, books, and code, image data, audio data, video data
	DeepSeek-Coder	1.3B, 6.7B, 33B	2024	-	✓	2T	GitHub's Markdown and StackExchange
	DocLLM	1B, 7B	2024	-	×	2T	IT-CDIP Test Collection 1.0, DocBank

Figure 1.1: LLM Overview

1.1.2 Large Language Models

First let us walk through early pre-trained neural language models.

- The first category is encoder-only models, represented by BERT.
 1. BERT, pre-trained by two tasks: *masked language modeling* (MLM) and *next sentence prediction* (NSP)
 2. RoBERTa, remove NSP, train with much larger mini-batches and learning rates.
 3. ALBERT, from $V \times H$ to $V \times E \rightarrow E \times H$ by adding a linear layer, so the embedding layer will have a smaller param size. 24 layers are shared (but the computation speed is same). And new NSP, instead of randomly choose next sequence, inverse the true sequence pair order.
 4. DeBERTa (Decoding-enhanced BERT with disentangled attention), use *disentangled attention* compute the content and position separately; At fine-tune stage, add absolute position into decoding layer to predict masked tokens, enhance the MLM; Introduce a novel pre-training task, ELECTRA, known as *replace token detection* (RTD), instead of mask the word, we replace it with plausible alternatives sampled from a small generated model. And pre-training will try to distinguish whether tokens are replaced or not. RTD is more sample-efficient than MLM because the former is defined over all input tokens rather than just the small subset being masked out.
 5. XLMs, 3 pre-training methods, casual language model (CLM) for warmup and basic ability; MLM, shared BPE between different languages; Translation Language Modeling (TLM), [CLS] ENGLISH [SEP] FRANCH [SEP] with mask.
 6. XLNet, based on Transformer-XL, utilize auto-regressive (decoder), use PLM (permutation language model) to permute the tokens of a sequence, use AR to predict it (only predict last K tokens) so it can get bidirectional information, then use a two stream attention because PLM makes the position a problem to handle.
 7. UNILM, shared Transformer but three pre-training methods, AR, AE (encoder only), seq2seq (encoder-decoder), using different masks to control the context the prediction is conditioned on.
- Second category is decoder-only models, represented by GPT.
 1. GPT-1, next token prediction
 2. GPT-2, Layer normalization is moved to the input of each sub-block, additional layer normalization is added after the final self-attention block.
- Last category is encoder-decoder PLMs, represented by T5.
 1. T5 and mT5 (multilingual variance of T5).
 2. MASS (MAsked Sequence to Sequence pre-training), mask several consecutive tokens as input, decoder predicts the masked fragment, so it jointly trains language embedding and generation.
 3. BART, it is pre-trained by corrupting text with an arbitrary noising function, and then learning to reconstruct the original text.

Then we come to the LLM family.

- GPT Family
 1. GPT-3, maybe the first LLM, with 175B parameters, the first to show in-context learning ability (one of the emergent abilities).
 2. CODEX, based on GPT-3, fine-tuned on GitHub, powers Copilot.
 3. WebGPT, based on GPT-3, 3 steps fine-tuned to answer open-ended questions using a text-based web browser (mimic the human, reward, reinforce).
 4. InstructGPT, SFT on labeler-written prompt and demonstrations, then further with human-ranked model outputs build a reinforcement learning flow.
 5. ChatGPT, sibling (brother) model to InstructGPT, the **Milestone**.
 6. GPT-4, larger, multimodal input.
- LLaMA
 1. LLaMA-1, use the transformer architecture of GPT-3, with modifications: use SwiGLU instead of ReLU; use rotary positional embeddings instead of absolute one; use RMS layer normalization instead of standard one; 13B are better than GPT-3 175B.

2. LLaMA-2, pre-train -> SFT -> RLFH, last step use accumulation of human feedback for revising the reward model (instead of a big change at short time) to save the ability of the model.
3. Alpaca, first human create 175 instructions, then use pre-trained model to generate other instructions, then generate outputs, clean up, use as the SFT data, so this is efficient.
4. Vicuna-13B, fine-tuned on user-shared conversations from ShareGPT.
5. Guanaco, use *QLoRA* to fine-tune, back-propagates gradients through a frozen, 4-bit quantized pre-trained language model into Low Rank Adapters.
6. Koala fine-tuned with data from ChatGPT.
7. Mistral-7B leverages grouped-query attention for faster inference, sliding window attention to efficiently handle sequences of arbitrary length. Better than most 13B and 34B model at that time.
- PaLM (Pathways [a distributed system] Language Model)
 1. PaLM is a 540B model.
 2. U-PaLM trained with UL2R (Unified Language Learner with Residual Learning), first learn simple task (80% are simple tasks), then learn complicated one, use mixture-of-denoiser objective.
 3. Flan-PaLM is fine-tuned U-PaLM with instruction. Large SFT datasets, 473 datasets with 146 categories.
 4. PaLM-2 is efficient and multilingual model.
 5. Med-PaLM, use *instruction prompt tuning*, add a soft prompt at the embedding, and learn it (the model is frozen). So the method is more efficient than adapter method like LoRA, and its performance is better than hard prompting.
- Other LLMs
 1. FLAN, shows that *instruction fine-tune* is good for zero-shot and multi-shot performance, inspires ChatGPT instruction align tech and prompt engineer.
 2. Gopher, analysis of model scales on model performance.
 3. T0, extend of FLAN, multi-prompt training.
 4. ERNIE 3.0, combine AR and AE. Injection of knowledge graph.
 5. RETRO, milestone of *Retrieval-Augmented Generation*, with a similarity based chunk cross attention.
 6. GLaM, *Mixture-of-experts* (MOE), replace the FFN layer with Gated FFN, which will choose different sub-layer using the gate, less training computation and inference computation.
 7. LaMDA, trained on dialog data, focus on safety.
 8. OPT
 9. Chinchilla, limit the compute budget, train different size of model and dataset, find that these two size should be scaled equally.
 10. Galactica, trained on scientific corpus.
 11. CodeGen, conversational programming (multi-step programming).
 12. AlexaTM, mixture of denoising and CLM tasks, more efficient few-shot learners than decoder-only models on various tasks. Multilingual model.
 13. Sparrow, use RLFH to create harmless, safety dialogue. Two modeling of outputs to help human labeler.
 14. Minerva, math enhanced.
 15. MoD, mixture-of-denoisers as the objectives.
 16. BLOOM, which promotes transparent, collaborative AI research.
 17. GLM, english-chinese model.
 18. Pythia, trained on public data in a consistent sequence, with 154 checkpoints per model. It enables research on LLM training dynamics, memorization, term frequency effects, and gender bias reduction.
 19. Orca, imitation learning, teacher assistance from ChatGPT.
 20. StarCoder, 80+ programming languages.
 21. KOSMOS, advancing research in cross-modal AI applications.

1.1.3 How LLMs are built

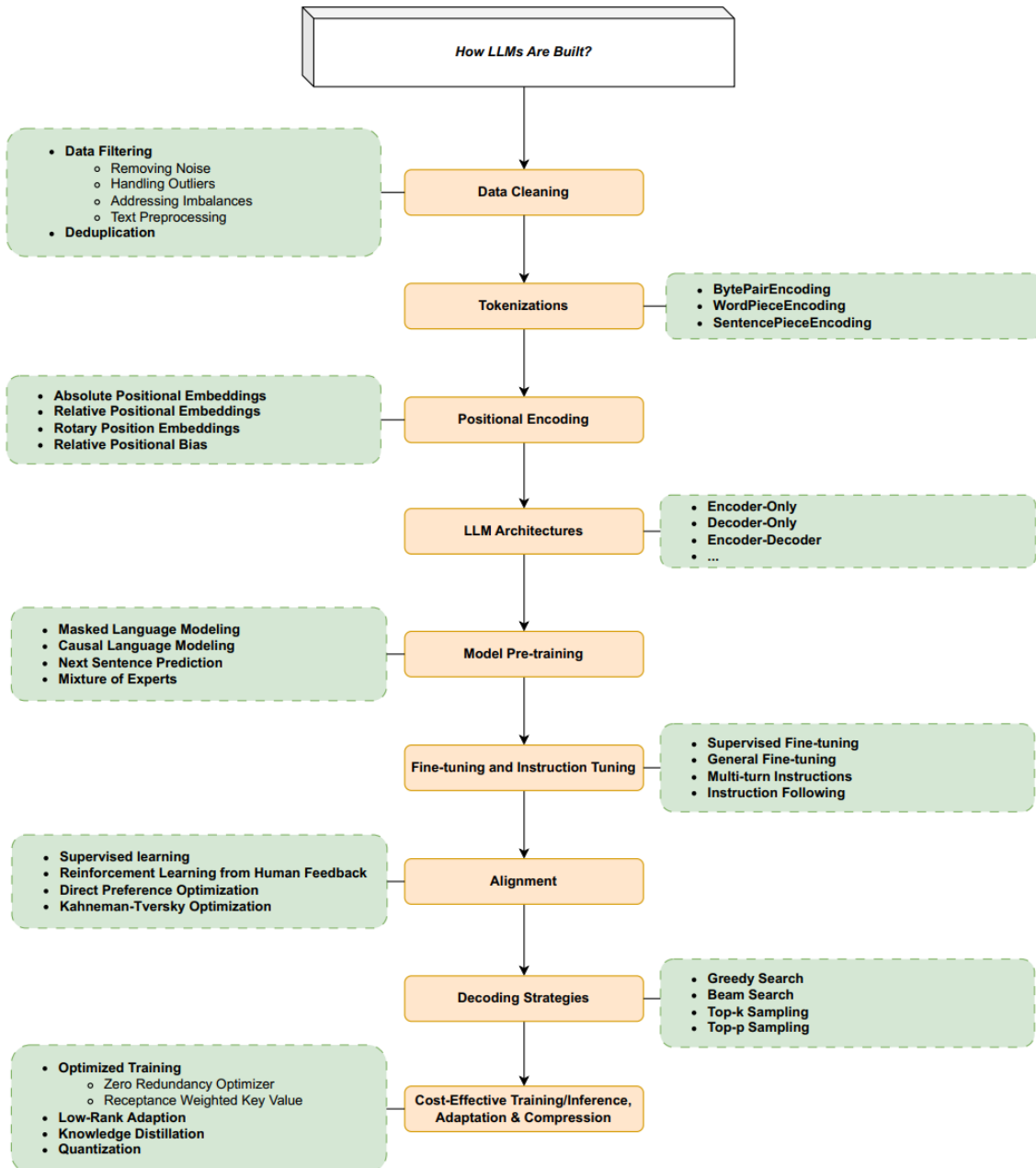


Figure 1.2: Components of LLMs

Chapter 2 History Language Models

2.1 Statistical Language Model

N-Gram model is the most dominating form of SLMs, which is a Markov chain model, using the statistic from previous $n - 1$ words and current word.

$$P(w_n \mid w_{n-k}, \dots, w_{n-1}) = \frac{\text{count}(w_{n-k}, \dots, w_n)}{\text{count}(w_{n-k}, \dots, w_{n-1})}$$

in n-gram, k is always $n - 1$.

In order to deal with unseen words and small probability of larger n-pairs, smooth method are often used, like *Laplace-smooth*.