Neural Computing and Applications (2025) 37:1973–1997
https://doi.org/10.1007/s00521-024-10827-6

REVIEW

Generalization potential of large language models

Mikhail Budnikov [1] · Anna Bykova [2] · Ivan P. Yamshchikov[1,3]

Received: 29 April 2024 / Accepted: 20 November 2024 / Published online: 17 December 2024

Abstract
The rise of deep learning techniques and especially the advent of large language models (LLMs) intensified the discussions around possibilities that artificial intelligence with higher generalization capability entails. The range of opinions on the capabilities of LLMs is extremely broad: from equating language models with stochastic parrots to stating that they are already conscious. This paper represents an attempt to review LLM landscape in the context of their generalization capacity as an information theoretic property of those complex systems. We discuss the suggested theoretical explanations for generalization in LLMs and highlight possible mechanisms responsible for these generalization properties. Through an examination of existing literature and theoretical frameworks, we endeavor to provide insights into the mechanisms driving the generalization capacity of LLMs, thus contributing to a deeper understanding of their capabilities and limitations in natural language processing tasks.

Keywords Large language models · Generalization · Semantic information

1 Introduction

Machine learning systems are experiencing a rapid surge in power and capability. These systems, driven by data-centric methodologies, exhibit the capacity to model increasingly intricate domains, whether it is an image classifier [1] or a language model [2], showcasing performance levels that surpass human abilities on designated benchmarks. Such advancements owe their existence to extensive scaling, encompassing both the augmentation of the model's parameters and the amplification of training data volumes.

However, the growth in data availability faces a critical bottleneck, particularly concerning unlabeled data, which is forecasted to become scarce by the end of this decade [3]. This scarcity poses a formidable challenge, particularly for tasks necessitating human supervision or those that are seldom encountered in web data sources. Mere aggregation of diverse datasets no longer suffices to propel machine learning systems to greater heights; rather, it necessitates the development of methodologies capable of efficient generalization. Furthermore, as researchers endeavor to scale AI to superhuman capacities safely, it becomes imperative to comprehend the precise mechanisms through which these systems generalize. This understanding is essential to mitigate the emergence of unforeseen capabilities or undesirable outcomes [4]. Hence, advancing beyond the era of brute-force data accumulation, the focus shifts toward refining the algorithms' ability to generalize across varied domains while ensuring their safety and reliability in real-world applications.

In this work we undertake a comprehensive examination of the phenomenon of generalization and explore strategies for its control. Our primary focus centers on natural language processing (NLP), with particular emphasis on large language models (LLMs). LLMs currently stand as the apex of universal machine learning systems, rendering them a fitting subject for analysis. Nonetheless, many of the concepts and methodologies discussed herein hold relevance across the broader spectrum of machine learning.

To provide a lucid exposition of the subject matter, we organize our discussion around the various stages of the machine learning pipeline. By structuring our analysis in this manner, we aim to offer a systematic understanding of the factors influencing generalization. At the conclusion of

✉ Ivan P. Yamshchikov
ivan.yamshchikov@thws.de

[1] Constructor University, 28759 Bremen, Germany

[2] LEYA, Higher School of Economics, St. Petersburg 28759, Russia

[3] THWS, CAIRO, 97082 Würzburg, Germany

each section, we list key insights from the literature under examination.

Our work encompasses a wide array of questions pertinent to the study of generalization:

- Data
  - What qualities of data are important for pre-training?
  - Which datasets lead to better out-of-distribution generalization?
  - How the data can be used to control the inductive bias of the model?
- Model architecture
  - What properties of a trained model are correlated with better generalization?
  - How can model architecture explain these properties?
  - How can model architecture be used to control the inductive bias?
- Training process
  - Which optimizers lead to better generalization and how are they related to each other?
  - Can one combine or mitigate inductive biases of different models by changing the training process?
- Inference
  - What kinds of learning and generalization can happen at inference time?
  - How can one amplify model's capabilities by augmenting it with tools, memory and prompting schemes?

2 Methodology

We have formed this review based on the different hypotheses about what influences generalization, especially in the NLP tasks. Our review is organized around the various stages of the machine learning pipeline that influences generalization results: data, model architecture, training process and inference. We tried to make the most detailed review within of each stages.

3 Data

Machine learning algorithms are fundamentally designed to acquire knowledge from data but the precise nature of this learning process often remains not clear. In our work, we explore various methodologies employed to shape the learning process and the resultant impact on the acquired knowledge.

We delve into a multitude of approaches, including:

- Data Selection: High-quality datasets for a model with relevant and informative examples, thus steering its learning trajectory toward desired outcomes.
- Demonstrations and Instruction Following: Leveraging demonstrations and explicit instructions to guide the learning process, facilitating the acquisition of specific problem-solving behaviors or task-oriented strategies.
- Data Generation from Models or Algorithms: Using data generation techniques wherein data are synthesized from existing models or algorithms, thereby expanding the diversity and richness of the training dataset.
- Using Data from Different Tasks or Modalities: Data sourced from disparate tasks or modalities can foster a more comprehensive understanding of underlying patterns and relationships, leading to enhanced model generalization.
- Integration of Random Data: Introducing elements of randomness into the training dataset, thereby encouraging adaptability and resilience in the model's learning process.

By scrutinizing these diverse methodologies, we aim to elucidate the multifaceted ways in which the information used for training can shape the final model.

3.1 Irrelevant data

Pre-training language models entail transferring linguistic knowledge collected from a data-rich task to a downstream task [5]. This process serves as a foundational step in the development of robust and high-performing models. However, the ramifications of pre-training extend far beyond this initial transfer of knowledge, encompassing a myriad of nuanced effects that warrant closer examination.

Papadimitriou and Jurafsky [6] demonstrate an intriguing finding regarding pre-training long short-term memory (LSTM) networks [7] on structured, non-linguistic data such as MIDI music, Java code, or nested parentheses. Remarkably, this pre-training approach leads to a reduction in perplexity when the LSTM model is tested on Spanish text. This observation underscores the versatility of pre-training methodologies and highlights the potential for leveraging diverse datasets to enhance model performance across linguistic domains.

Moreover, the efficacy of pre-training extends beyond language-specific tasks. Lu et al. [8] showcase this phenomenon by achieving performance levels comparable to training from scratch across various modalities. Their approach involves fine-tuning only the input and output embeddings of a pre-trained Generative Pre-trained

Transformer 2 (GPT-2) model [9]. This finding not only underscores the transferability of pre-trained models but also underscores the importance of efficient fine-tuning strategies in optimizing model performance across diverse domains.

Sinha et al. [10] conducted a notable investigation wherein they explored the impact of removing all information pertaining to word order during the pre-training phase of a model. Their findings indicate that despite the absence of word order information during pre-training, the final performance of the model is minimally affected, provided that a fine-tuning phase has the correct word order.

Krishna et al. [11] conducted a study in which they sampled the training data from a completely artificial language, comprising random n-grams. Their investigation aimed to explore the impact of pre-training objectives that necessitate processing such information, such as tasks involving copying sentences in the correct order. Despite the synthetic nature of the training data, authors observed that pre-training objectives requiring the processing of this artificial language still led to performance improvements in the model on summarization tasks when compared to a randomly initialized version.

Maennel et al. [12] conducted a study wherein they trained neural networks with entirely random labels, revealing an unexpected phenomenon: in certain cases, this training approach still facilitated improved performance during subsequent fine-tuning stages. Investigating the underlying mechanisms of this intriguing effect, they discovered that the first layer of the network exhibited remarkable adaptability to the data. Specifically, authors found that the weights connecting the inputs to randomly selected neurons in the first hidden layer behaved as random variables during pre-training. Remarkably, they observed that, over the course of pre-training, the eigenvectors of the covariance matrix associated with these weights aligned with those of the covariance matrix derived from the data itself. This alignment indicated that the first layer of the network was effectively adapting to capture statistical properties of the input data, despite the random nature of the labels. Building upon this discovery, Chowers and Weiss [13] further elucidated the behavior of the first layer in neural networks. Their study revealed that, during pre-training, the first layer progressively converged toward a whitening transformation for the training dataset. This transformation involves decorrelating and normalizing the input features, effectively enhancing the network's ability to extract meaningful patterns from the data.

Research suggests that pre-training guides optimization toward a flatter basin within the loss landscape, a phenomenon with significant implications for model performance and adaptability. Mehta et al. [14] demonstrated this finding through empirical analysis and proposed it as a key factor contributing to the reduced susceptibility of pre-trained models to catastrophic forgetting during fine-tuning.

Furthermore, Neyshabur et al. [15] observed a similar phenomenon and provided additional insights into its implications. They demonstrated that models fine-tuned from the same pre-trained checkpoint tend to remain within the same basin of the loss landscape.

3.2 Internal representations

Beyond their proficiency in language processing, contemporary language models have demonstrated a remarkable capacity to learn and encode higher-level information spanning diverse domains. Gurnee and Tegmark [16] recently showed the emergence of detailed world models within pre-trained language models. These world models are presented in terms of both time and space, allowing for the extraction of nuanced information regarding the timing and location of events. Remarkably, through linear projection from activations in layer 50 corresponding to specific tokens, it becomes feasible to discern the temporal and spatial contexts associated with various events encoded within the model. Moreover, Li et al. [17] conducted a study wherein they trained a language model to predict moves in a simple board game, Othello. They discovered that the model inherently represented the state of the Othello board internally. This finding underscores the model's capacity to capture complex relational structures and dynamic states within diverse problem domains, transcending its original training objectives in natural language processing. Furthermore, Jin and Rinard [18] provide compelling evidence that language models trained on code exhibit sophisticated representations of program states. Their study reveals a strong correlation between the quality of these representations and the model's proficiency in generating correct programs.

Another discovery in the field of language models is their innate tendency to learn linear representations, even in the absence of explicit direction to do so. This phenomenon mirrors the insights offered by Word2vec [19], which demonstrated that simple vector arithmetic applied to word embeddings yields semantically meaningful results. Building upon this foundational work, recent studies have further illuminated the linear nature of representations within advanced language models. For instance, Turner et al. [20] revealed that the vector arithmetic manipulation also gives insightful results when applied to activations within the GPT-2 model. Moreover, Nanda [21] delved into game-based language models, demonstrating that Othello-GPT exhibits a linear world model. Further corroborating these findings, Jin and Rinard

[18] provided compelling evidence that representations within language models trained on code also exhibit linearity.

3.3 High-quality data

State-of-the-art language models represent a pinnacle of natural language understanding, fueled by training on vast corpora of textual data. However, the sheer scale of data required for training these models comes with significant costs and challenges. As the volume of available data approaches its finite limits, the feasibility and scalability of training such models become increasingly constrained, potentially posing a bottleneck to further advancements in NLP research and development.

In the context of applications, a critical inquiry revolves around determining the optimal composition and volume of data necessary for endowing language models with specific capabilities. For instance, understanding the minimum data requirements for achieving competencies such as coherent English generation or zero-shot reasoning is pivotal for streamlining model training and deployment pipelines. This entails delving into questions regarding the diversity, quality, and relevance of training data, as well as exploring strategies for dataset curation and augmentation.

Eldan and Li [22] conducted a study that underscores the potential of training language models on stories with highly restricted vocabularies. Their findings demonstrate that even with stringent constraints on vocabulary size, it is feasible to develop language models with less than 10 million parameters, capable of generating coherent and grammatically correct stories. This research clarifies on the adaptability of language models to operate effectively within constrained linguistic environments.

Building upon prior research, Gunasekar et al. [23] have extended the exploration into dataset selection strategies, particularly within the programming domain. Their study reveals that through data with maximal educational value, it is feasible to achieve substantial reductions in the size of language models for code-related tasks. Similarly, Li et al. [24] have investigated the impact of dataset selection strategies on models designed for commonsense reasoning. Recently, Surkov and Yamshchikov [25] introduced "Vygotsky distance", a tool for benchmark similarity assessment that could potentially structure further choices of training data and benchmarks to maximize generalization potential of the models.

3.4 Learning inductive bias from data

Inductive biases play a crucial role in shaping the learning behavior and generalization capabilities of machine learning models. These biases can be effectively instilled into models through pre-training on datasets that exemplify specific properties or tasks. Several recent studies have showcased the effectiveness of leveraging pre-training models with inductive biases across various domains. For instance, McCoy et al. [26] adopted a model-agnostic meta-learning approach [27] to identify and incorporate inductive biases essential for rapid language acquisition. Similarly, Wu et al. [28] devised synthetic datasets that necessitate deductive, inductive, and abductive reasoning skills, reflecting fundamental principles of mathematical reasoning. Meanwhile, Lindemann et al. [29] explored the utility of pre-training models to simulate finite state transducers (FSTs) based on their descriptions. By pre-training models on tasks involving FST emulation, they injected specific inductive biases into the models, enabling them to better generalize to NLP tasks characterized by similar structural properties.

Mukherjee et al. [30] have introduced an approach to knowledge distillation adapted specifically for language models. In this methodology, a smaller model is trained on the detailed explanations generated by a larger, more complex model. By leveraging these rich training data derived from the explanations of the larger model, the smaller model is equipped to achieve enhanced performance, leading to improved generalization across various tasks. This transfer of knowledge can be compared with transferring a superior inductive bias from the teacher model to the student model.

3.5 Demonstrations of instruction following and problem solving

A predominant trend within the domain of language models centers around the paradigm of pre-training followed by fine-tuning, particularly adapted toward instruction following tasks [31]. For instance, Wei et al. [32] showed that fine-tuning language models on instruction following demonstrations gives better zero-shot performance than models of comparable size trained solely through zero-shot or few-shot learning approaches. Moreover, Ouyang et al. [33] introduced reinforcement learning techniques alongside supervised fine-tuning, showcasing that models trained using this hybrid approach exhibit improved generalization to previously unseen instructions. Furthermore, Ye et al. [34] explored a training methodology wherein models are trained to infer instructions based on task inputs and correct answers. This instruction guessing approach has been shown to significantly enhance generalization performance, particularly for tasks featuring unseen answer formats.

Chen et al. [35] highlight a connection between agents and fine-tuning within the field of language models. Their research clarifies on the potential benefits of fine-tuning the

underlying language model using demonstrations of agent behavior across diverse tasks and various prompting methods. By fine-tuning language models on such agent-based demonstrations, authors observed notable improvements in both performance and generalization of the resulting agents. In Sect. 6, we will discuss language agents in more details, exploring their role and capabilities.

Wang et al. [36] conducted a comprehensive study on the performance of pre-trained models trained on instruction-following datasets, revealing insights into the relationship between model performance, task diversity, model size, and data quantity. Their findings indicate that the performance of pre-trained models exhibits a log-linear increase with the number of tasks and the size of the model architecture. The authors observed that model performance does not significantly depend on the quantity of data available for each individual task. Moreover, Chan et al. [37] highlighted additional crucial features of data distribution that impact in-context learning, particularly in scenarios where models are required to comprehend and generate text within specific contexts.

3.6 Summary

Simple structural patterns, such as matching tokens in bracket sequences, word co-occurrence statistics are important elements that significantly influence the performance of pre-trained language models. However, beyond merely capturing superficial statistics from data, large language models also develop internal representations that reflect a broader understanding of the world, often characterized by linear structures.

Efforts to optimize the efficiency and capabilities of language models often involve training on datasets that offer greater interpretability and educational value. Furthermore, the distribution of model capabilities can be adjusted by training on datasets that align with desired inductive biases. For instance, datasets featuring exercises on reasoning primitives can promote the development of models with enhanced reasoning abilities.

To enhance the effectiveness of language models in few-shot and zero-shot learning scenarios, incorporating training or fine-tuning with demonstrations of instruction following is crucial. Similarly, to bolster the model's proficiency in complex reasoning tasks, datasets should encompass applications of problem-solving strategies, enabling models to acquire nuanced reasoning skills.

Another noteworthy observation is that in-context learning tends to accelerate when it significantly facilitates task-solving efficiency. This phenomenon becomes more pronounced with an increasing number of tasks, as the learning process becomes more conducive to meta-learning. With a greater diversity of tasks, the model learns not only how to perform individual tasks but also how to effectively generalize across them, thereby enhancing its overall learning efficiency and adaptability.

4 Model architecture

To understand how model architecture affects generalization we first review some properties that are usually found in models that demonstrate comparatively high generalization potential or that can provide lower bounds for it, and then we discuss explanation proposed to explain why models have those properties and achieve superior generalization. Finally, we review architecture modifications, mostly specific to language models, which help model to generalize better in certain directions.

4.1 Factors related to generalization

One common setting for studying generalization is the i.i.d. (independent and identically distributed) case, where both training and testing samples are drawn from the same distribution and are independent of each other. This setup enables the quantification of generalization and the establishment of lower bounds for it. Numerous properties have been identified to provide such lower bounds, with many of them focusing on describing model complexity in various ways:

- Compressibility: Arora et al. [38], Lotfi et al. [39]
- Weight norm: Bartlett [40], Wei and Ma [41], Kawaguchi et al. [42]
- Flatness: Hochreiter and Schmidhuber [43], Bahri et al. [44], Orvieto et al. [45]
- Algorithmic stability: Chatterjee and Zielinski [46], Bousquet and Elisseeff [47]

When deploying a model in real-world scenarios, encountering a distribution shift is almost inevitable. In practice, training data can only fully represent the actual distribution of inputs under the most straightforward circumstances. Thus, a more pertinent, albeit challenging, consideration is out-of-distribution generalization. As elucidated by Wolpert [48] in no free lunch theorems, without additional assumptions, all learning algorithms perform equally poorly. Even widely employed techniques such as cross-validation will lose to anti-cross-validation in half of the cases. This underscores the significance of addressing out-of-distribution scenarios and highlights the limitations of relying solely on traditional methodologies in complex real-world contexts.

If one assumes that simpler hypotheses are preferable to complex ones, it lays the foundation for deriving a general method of inductive inference known as Solomonoff

induction [49]. This method involves averaging predictions from all possible models, weighted by their Kolmogorov complexity, resulting in a finite number of errors while predicting any computable sequence. Goldblum et al. [50] highlight that many real-world data sources exhibit a bias toward simplicity, meaning they can be effectively compressed compared to the uniform distribution suggested by the no free lunch theorems. An interesting observation is that modern deep learning models, including large language models, whether trained or randomly initialized, also demonstrate a propensity for simpler solutions [50, 51]. This alignment with simplicity suggests a fundamental principle guiding both human and artificial intelligence in learning and problem-solving tasks.

4.2 Role of depth

One advantage of deep models lies in their enhanced expressiveness. According to Cover's theorem [52], simply projecting data into a higher-dimensional space with non-linear transformations is likely to render it linearly separable. If this transformation occurs before the dimensionality surpasses the number of samples, the resultant linear separation gives a simple hypothesis with promising potential for generalization. From a theoretical standpoint, Eldan and Shamir [53] demonstrate that adding an extra layer to a neural network can exponentially reduce the number of parameters for certain functions. Empirically, deep randomized neural networks [54] confirm this notion, indicating that the addition of more non-linear layers, even without training, can offer utility in improving model performance.

Moreover, Hinton et al. [55] argues that neural networks' ability to learn distributed representations enables them to capture relevant information in a more efficient and organic manner. As the network learns various concepts in one layer, it can then use arbitrary linear combinations of these concepts in subsequent layers, thereby representing compositions of the original concepts. Consequently, in datasets exhibiting hierarchical structures, where higher-level concepts are formed by combinations of concepts from lower levels, deep models are better suited to represent such data.

Some model architectures possess the capability to internally execute algorithms, with each non-linear layer serving as a computation step. For instance, Von Oswald et al. [56] demonstrate that in-context learning in Transformers bears resemblance to gradient descent. Xu et al. [57] explore the alignment of certain neural architectures with common classes of algorithms and illustrate that better alignment correlates with improved sample efficiency and generalization. For instance, graph neural networks (GNNs) [58] excel at learning algorithms resembling dynamic programming [59], they struggle with problems requiring exhaustive search. These latter problems can be swiftly learned by more versatile architectures capable of iterating over all subsets. If there existed an architecture capable of learning arbitrary algorithms from data without a strong bias toward specific algorithmic classes, it could serve as a robust approximation of Solomonoff's induction. Such an architecture might offer superior generalization performance compared to less algorithmically oriented models.

Chatterjee and Zielinski [46] highlight a phenomenon in deep architectures trained with gradient methods: positive feedback loops can emerge due to the paths in the network having gradients proportional to the product of weights along the path. Consequently, paths through edges with already high weights receive higher gradients and thus grow even faster. This amplification effect of depth serves to magnify gradient directions that frequently occur in the dataset. The authors argue that such common gradient directions play a crucial role in generalization and empirically demonstrate that robust estimators of gradients, such as the mean of clipped gradients or gradient median, lead to improved generalization.

4.3 Model architecture for inductive bias

While data-driven methods for controlling inductive bias show promise, certain types of bias may be more effectively managed by explicitly altering the model architecture.

An effective change known to enhance generalization is the incorporation of a world model. For instance, EfficientZero [60] demonstrates the utility of explicitly learning a model of the environment to achieve sample-efficient learning across various games. By leveraging this world model for planning, EfficientZero exhibits impressive performance. Similarly, Li et al. [61] introduce a language model augmented with a value function, enabling control over properties that necessitate planning, such as generated sequence length or the probability of the prompt given the generation. This approach illustrates the potential of integrating world models to enhance the capabilities and flexibility of language models.

In addition to a world model, incorporating a bias for optimization can also prove beneficial. For instance, Amos and Kolter [62] introduce a quadratic programming solver as one of the layers in their model, showcasing its ability to learn Sudoku solely from observing solution examples without explicit rule instructions. Similarly, von Oswald et al. [63] discover that Transformers can internalize gradient descent procedures without architectural modifications. They propose augmenting models with layers capable of solving least squares regression, further

illustrating the potential for integrating optimization biases to enhance model capabilities.

Certain problems may demand more computation than can be accommodated within a single forward pass of the model. In such scenarios, having an adaptive amount of computation at each step proves advantageous. Graves [64] address this challenge by allowing for multiple intermediate predictions before reaching the final output. Banino et al. [65] extend a similar approach to Transformers, proposing a differentiable training method to facilitate adaptability. Alternatively, Dehghani et al. [66] avoid multiple forward passes and instead apply the same set of layers iteratively, thereby enabling the model's depth to adapt dynamically.

Certain types of information are best memorized by the model, and in such cases, the addition of an explicit external memory can be beneficial. Khandelwal et al. [67] propose linearly interpolating the outputs of a large language model with predictions from k-nearest neighbors, aiding in the prediction of rare patterns such as factual knowledge. Guu et al. [68] take a different approach by feeding retrieved documents to another neural network, enabling a more flexible combination with model predictions. Borgeaud et al. [69] demonstrate that this approach remains effective even at the scale of trillions of tokens, allowing for equivalent performance with a 25x smaller model. Wu et al. [70] incorporate kNN lookup at the layer level, seamlessly integrating retrieval and attention mechanisms. They also advocate for writing facts to external memory during inference, enabling the model to access definitions of functions or other relevant mathematical objects. Sukhbaatar et al. [71] replace MLP blocks in Transformers with extra tokens added to self-attention, simplifying the architecture and paving the way for the application of additional attention-related techniques to every layer in such models.

H3 [72] serves as another compelling demonstration of deliberate architecture design. The authors meticulously identify the specific capabilities that render Transformers superior to state space models and subsequently craft a new layer designed to precisely enhance these abilities. The resulting model surpasses Transformer baselines in zero- and few-shot performance, showcasing the efficacy of this targeted design approach.

Another research direction emphasizes architectures that promote sparsity. The rationale behind this approach lies in the notion that if different components of the model operate independently, alterations affecting one component will not impact others, resulting in increased robustness to distribution shifts and enhanced interpretability. To this end, Elhage et al. [73] introduce the SoLU activation function, defined as $f(x) = x \cdot \mathrm{softmax}(x)$, which leads to more efficient activations in MLP layers within Transformers. Additionally, Lamb et al. [74] partition the parameters and representations into groups that communicate solely through attention mechanisms, fostering competition among these groups. They demonstrate that implementing this approach for all Transformer layers except the first and last ones leads performance improvements.

Extensive research has been dedicated to designing neural architectures capable of learning and executing algorithms. The neural turing machine [75], for instance, integrates a recursive network with an external memory in a differentiable manner, resulting in qualitatively improved performance on algorithmic tasks. Veličković and Blundell [76] propose leveraging graph neural networks [58], as their architecture is inherently well-suited to operations characteristic of dynamic programming Bellman [59], allowing many algorithms to be formulated in this framework.

In general, if there exists a known symmetry within the data that can be represented as a matrix group, Finzi et al. [77] demonstrate a method to construct a layer that exhibits equivariance to it. Furthermore, Finzi et al. [78] illustrate that for a model architecture incorporating specific biases, it is possible to construct a flexible yet still biased model by ensembling the biased model with a more generalized one, subsequently regularizing the weight assigned to the general model. Similarly, Goldblum et al. [50] advocate for ensembling models with diverse architectures and applying appropriate regularization as a flexible approach to learning which bias to apply based on the data.

4.4 Summary

Simple models tend to exhibit relatively higher generalization capacity, as evidenced by both empirical observations and theoretical estimates. Simplicity, in this context, can be construed in terms of compressibility (or low Kolmogorov complexity), small weight norms, flatness of the loss landscape, or minimal reliance on individual training examples.

Depth plays a crucial role in this phenomenon, with several explanations of its impact. Firstly, the incorporation of nonlinearities enhances model expressiveness, often exponentially, thereby necessitating fewer parameters and reducing complexity relative to the training data. Secondly,

deep architectures naturally capture hierarchical or computational structures inherent in the data. Finally, when trained via gradient descent, deep models naturally prioritize common cases in terms of gradient directions while disregarding outliers.

5 Training process

This section is by far the most difficult one to write since, unfortunately, there are a lot of choices that are made during training of large language models, yet most of the papers on the topic disclose only a fraction of those decisions. Initially, we delve into works that analyze the individual influence of each factor. Subsequently, we explore three major categories of modifications to mainstream gradient-based optimizers that are adapted to enhance generalization. These include methods explicitly targeting solutions within flat basins of the loss landscape, parameter averaging across multiple models, and training models to exhibit resilience against input perturbations.

An appealing property of human generalization lies in its capacity to generate novel insights even without exposure to new data, e.g., by theoretical research. As the availability of training data gradually emerges as a bottleneck, methodologies capable of extracting more knowledge from a fixed dataset through bootstrapping mechanisms assume greater significance.

The outcomes of model training endeavors, encompassing generalization performance, are significantly contingent upon hyperparameters. Consequently, techniques that facilitate automatic hyperparameter adjustment hold considerable promise.

Lastly, we delve into three distinct groups of methods aimed at controlling the model's generalization tendencies. These encompass mechanisms for regulating the simplicity of learned solutions, fostering diversity among solutions, and introducing biases to steer the model's generalization trajectory in a desired direction.

5.1 Role of the optimizer

The extent to which the choice of optimizer, often manifesting as a form of stochastic gradient descent (SGD), influences generalization performance prompts inquiry. Mingard et al. [79] provide insights by demonstrating that randomly sampling parameters as opposed to using SGD has negligible effects. Specifically, their study investigates the probability of a deep neural network trained with SGD converging to a specific input–output mapping on a given dataset. Their findings reveal that this probability aligns closely with the Bayesian posterior probability of the mapping given the dataset. Similarly, Chiang et al. [80] conduct an experiment employing uniform sampling in lieu of Gaussian processes, achieving comparable generalization success rates to those obtained with SGD. Concurrently, Barak et al. [81] explore the task of learning k-sparse parity of n bits. Their study underscores that the performance of SGD, particularly in terms of training dynamics cannot be explained by random sampling. They delineate the following reasons:

1. The distribution of convergence times deviates significantly from geometric patterns, a finding that aligns with the observations of Kalimeris et al. [82]. Their study highlights that models trained with SGD exhibit a tendency to increase in complexity throughout the training process.
2. Deep models trained with SGD demonstrate an ability to master the sparse parity task in a number of iterations approaching asymptotic optimality, a phenomenon challenging to clarify without introducing an explicit sparsity prior. However, it's noteworthy that the inherent sparsity bias may stem from the deep model architecture itself. Mingard et al. [83] provide insights into this aspect, demonstrating that such models exhibit a bias toward simpler functions. Specifically, if the prior assigns equal probability mass to k-sparse functions for all k, the probability of a single k-sparse function will be $\Omega(n^{-k})$.
3. A significant portion of the variance in convergence times can be attributed to the initialization method used in SGD.
4. Phase transitions, exemplified by phenomenon such as grokking [84], showcase distinct qualitative shifts in training behavior across different stages. Hu et al. [85] offer an explicit modeling approach for this phenomenon using a hidden Markov model, where distinct states correspond to various phases in the training process.

5.2 Flatness

Since at least Hochreiter and Schmidhuber [43], there has been a hypothesis suggesting that flat minima in the loss landscape correspond to better generalizing solutions. Various techniques have been developed to identify such minima, often involving modifications to the loss function.

Let's denote the data distribution as $p(x, y)$, and the model's loss with parameters θ as $\mathcal{L}(y \mid x, \theta)$, typically representing the negative log-likelihood. The standard optimization process can then be expressed as:

$$\arg\min_{\theta}\left(\mathbb{E}_{x,y\sim p(x,y)}[\mathcal{L}(y \mid x, \theta)] + \lambda\|\theta\|^2\right) \tag{1}$$

Foret et al. [86] try to identify minima that exhibit flatness in all directions, drawing motivation from PAC Bayesian generalization bounds:

$$\arg\min_{\theta}\left(\mathbb{E}_{x,y\sim p(x,y)}\left[\max_{\|\epsilon\|\leq\rho}\mathcal{L}(y \mid x, \theta+\epsilon)\right] + \lambda\|\theta\|^2\right) \tag{2}$$

Orvieto et al. [45] concentrate on expected sharpness, constructing their loss function as a smoothed version of the original:

$$\arg\min_{\theta}\left(\mathbb{E}_{x,y\sim p(x,y)}\left[\mathbb{E}_{\epsilon\sim\mathcal{N}(0,\sigma^2 I)}\mathcal{L}(y \mid x, \theta+\epsilon)\right]\right) \tag{3}$$

Chaudhari et al. [87] propose a similar approach:

$$\arg\min_{\theta}\left(\mathbb{E}_{x,y\sim p(x,y)}\left[-\log\int_{\mathbb{R}^n}\exp(-\mathcal{L}(y \mid x, \theta+\epsilon) - \frac{\gamma}{2}\|\epsilon\|^2)d\epsilon\right]\right) \tag{4}$$

Additionally, Ishida et al. [88] suggest a heuristic solution empirically shown to improve generalization:

$$\arg\min_{\theta}\left(\mathbb{E}_{x,y\sim p(x,y)}\left[|\mathcal{L}(y \mid x, \theta) - b|\right]\right) \tag{5}$$

Both 2 and 3 optimize for large loss basins, just in different ways. Regarding the connection with 4, Yang et al. [89] demonstrate that some monotonically increasing function of negative local entropy serves as an upper bound for the Hessian norm, which is directly optimized in 3. As for 5, Liu et al. [90] show that when the loss surpasses the flooding threshold b and the model alternates between gradient descent and gradient ascent, the dynamics become equivalent to minimizing the norm of the gradient. After SGD converges close to a local minimum, the expected gradient norm becomes proportional to the Hessian norm and the amount of stochasticity in SGD updates, implying optimization for flatness.

5.3 Averaging

The concept of using trajectory averaging for faster convergence is not novel, as evidenced by Polyak and Juditsky [91] and the literature referenced therein. However, recent research has proposed various approaches to leverage such averaging and has also clarified its correlation with generalization.

Izmailov et al. [92] discovered that SGD with cyclical or constant learning rate schedules tends to converge to the vicinity of high-performing networks but often remains at the boundary rather than reaching the center. To address this, they proposed averaging the model weights from multiple points during training. This method was found to give solutions closer to the center of the corresponding loss basin, resulting in enhanced generalization without adding overhead to either training or inference. Building upon this, Lu et al. [93] applied the technique to fine-tune pre-trained models and noted improvements in generalization.

Zhang et al. [94] proposed a two-level optimizer where the outer loop advances in the direction of the average weight updates computed in the inner loop. While their primary focus was on enhancing convergence, they also noted improved generalization. A slight variation of this approach is presented in Zhang et al. [95], where the inner loop trains multiple copies of the model from the same starting point and subsequently averages their parameters. They suggest that this method can be combined with other techniques, such as sharpness-aware minimization [86] or ensembling, to achieve superior performance.

Mandt et al. [96] examine SGD samples as a Markov chain, offering a concise proof that Polyak averaging [91] with a constant learning rate is optimal among all stochastic gradient methods. They demonstrate that asymptotically, its efficacy remains unaffected by batch size or learning rate adjustments.

5.4 Adversarial training

While methodologies discussed in the previous section concentrate on the flatness of the loss concerning model parameters, adversarial training can be interpreted as striving for flatness of the loss concerning the inputs. However, the relationship between adversarial training and generalization is complex, and it can have both positive and negative effects.

Goodfellow et al. [97] clarify that adversarial examples arise due to the overly linear behavior of the model, which attempts to generalize in directions irrelevant to the task at hand. Essentially, if the model behaves linearly in the vicinity of a particular input and the input space is high-dimensional, perturbations in the direction of the gradient can significantly influence the model's output. While the impact from each dimension may be small, their cumulative effect grows linearly with the number of dimensions.

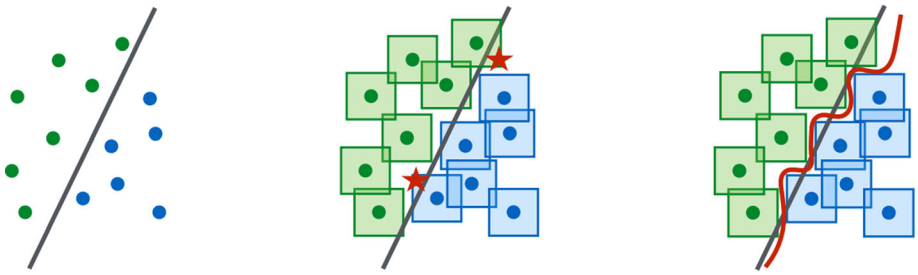

Fig. 1 Decision boundary in adversarial training. Figure from Madry et al. [101]

Consequently, this introduces the first source of the robustness-generalization trade-off: linear extrapolation facilitates easy generalization but compromises robustness. Raghunathan et al. [98] demonstrate this trade-off using a toy example and observe that access to ample unlabeled data points can help mitigate it. On the theoretical front, Bubeck et al. [99] demonstrate that achieving robust learning can be exponentially challenging computationally, while Schmidt et al. [100] illustrate that it can also be hard from an information theoretic sense.

Madry et al. [101] highlight another challenge in building robust models: as the perturbation size increases, the decision boundary becomes more complex, necessitating a higher-capacity model to accommodate it, as illustrated in Fig. 1. Paradoxically, if there exists a less general but more robust way to predict the data, the model might transition to it, thereby compromising generalization once again. Additionally, they observe that augmenting model capacity, such as increasing depth, inherently enhances robustness, even in the absence of adversarial training. Conversely, models with insufficient capacity fail to derive benefits from adversarial training and struggle to learn effectively under such conditions.

Adversarial training can also serve as a form of regularization, as demonstrated by Goodfellow et al. [97], who observed improved generalization on the MNIST dataset after its application. This phenomenon arises because a network exhibiting large gradients with respect to many input features at certain training examples will lack robustness; perturbing any of these features will result in significant shifts in the outputs. Consequently, the network is compelled to either rely on linear combinations with small coefficients (in terms of L1 norm) or adopt highly non-linear behaviors, ensuring that only a few features are pivotal for each example. The former effect promotes sparsity, a beneficial form of simplicity, while the latter reduces linearity, which, in certain scenarios, helps prevent overly simplistic solutions (similar in spirit to Sect. 5.7).

5.5 Bootstrapping

If one aims to train a model for a task with limited or unavailable data, some form of bootstrapping becomes necessary.

Amini et al. [102] provide a comprehensive overview of methods that leverage a trained classifier and unlabeled data to enhance performance further. Similarly, Huang et al. [103] propose an analogous approach adapted for large language models, employing prompting techniques and aggregating multiple predictions to derive more reliable answers from the model. Expanding on this, Jung et al. [104] demonstrate the feasibility of sampling unlabeled data from a language model itself. Furthermore, Wang et al. [105] use such an approach to generate an instruction tuning dataset from only 175 seed examples annotated by humans, showcasing that a model fine-tuned on this dataset outperforms strong baseline models.

In certain scenarios, the intrinsic nature of the problem permits the model to enhance itself solely through self-interaction. A notable instance is AlphaGo Zero [106], a reinforcement learning system that attained superhuman performance solely via self-play. Extending this concept, Bricman and Feeney [107] propose training language models to advocate for various positions in arguments against other language models. They argue that such a dynamic will foster automated truth-seeking capabilities and substantially augment the language model's abilities far beyond its initial state.

5.6 Meta-optimization

The emergent abilities of Transformers highlight the potential of optimizing a sufficiently flexible model on diverse datasets to achieve various beneficial generalizations. Therefore, it stands to reason that enhancing the flexibility of the optimization process itself can similarly give gains in generalization.

One category of approaches revolves around optimizing the hyperparameters of existing optimizers. For instance, Chandra et al. [108] calculate the gradients of the training loss with respect to hyperparameters and subsequently utilize gradient-based optimizers to automatically adjust

these hyperparameters during training. However, as highlighted by Wu et al. [109], there exists a trade-off between short-term and long-term performance. For instance, a high learning rate may consistently overshoot in each iteration but achieve greater overall progress by moving faster. Consequently, when such meta-gradients are computed with a short horizon, it often results in hyperparameters biased toward short-term performance, leading to learning rates significantly lower than optimal by orders of magnitude.

One potential approach to address this credit assignment challenge is by leveraging multiple training runs across various models to train a model capable of optimizing hyperparameters optimally. Almeida et al. [110] approach this as a reinforcement learning problem, employing Proximal Policy Optimization (PPO) [111] to train an LSTM model. This LSTM model takes unitless metrics regarding the current training state, such as the log-ratio of validation and training performance or the cosine similarity between gradient and momentum, as input and outputs adjustments to be made to the hyperparameters.

Yang et al. [89] take a direct approach to meta-optimization by optimizing the model parameters themselves. They leverage local entropy [87] and Hessian norm to identify meta-optimizers that lead to enhanced generalization performance for the optimized models. Expanding on this idea, Kirsch and Schmidhuber [112] push the boundaries further by replacing each model weight with an LSTM and meta-learning the entire training dynamics. Their study demonstrates that such a system can learn to imitate back-propagation, and the optimization algorithm learned through meta-learning on one dataset generalizes effectively to training on another dataset.

Peebles et al. [113] adopt an approach to optimization by learning a generative model for the joint distribution of model parameters and various metrics. This strategy enables them to sample models exhibiting low loss or other desirable properties. While this approach may not be entirely practical at present, the concept of sampling models with good generalization, rather than solely focusing on training methodologies to achieve such models, presents an interesting avenue for further research. For instance, Decision Transformer [114] demonstrated impressive performance in model-free offline reinforcement learning by leveraging a Transformer to learn the joint distribution of policies and their associated rewards.

5.7 Avoiding solutions that are too simple

One crucial inductive bias in machine learning is simplicity, a principle often encapsulated by Occam's razor. This principle suggests that, given two features with equal predictive power, the simpler one is preferred. Numerous studies have demonstrated that deep neural networks exhibit some form of Occam's razor, particularly when trained using gradient descent methods [51, 83]. While simplicity can serve as a useful tool to prevent overfitting in many cases, there exists an inherent tension between simplicity and complexity in achieving generalizable solutions. In scenarios where overly simplistic solutions may exist, it becomes imperative to avoid them, as they tend to rely on spurious correlations from the training data and may consequently fail to generalize well to unseen test data.

Clark et al. [115] address this challenge by employing an ensemble approach, where a smaller capacity model is trained alongside the main model. The smaller model primarily captures superficial patterns, while the larger one learns features that generalize better. Similarly, Nam et al. [116] observe that superficial features are often learned initially. To mitigate this, they propose training a biased model by gradually increasing the weight assigned to correctly predicted examples. Concurrently, they train the main, unbiased model on the more challenging instances.

The concept of simplicity can be customized for specific architectures or tasks. In the field of language modeling, a crucial aspect is the model's ability to capture long-distance dependencies. For instance, [117] augment the effective context length of a language model by computing the difference between output logits of a short-context model and those of the main model. Another approach by Chuang et al. [118] delves into the disparities among layers in a deep model. They observe that outputs from the final layers tend to exhibit greater factual correctness. By strategically selecting one of the earlier layers for comparison, they enhance factuality even further. Specifically, they identify the layer with the highest Jensen-Shannon divergence of next-word distributions from the final layer and then subtract their logits.

5.8 Learning diverse solutions

If a model generalizes in a wrong manner, but superficial features are challenging to differentiate in terms of simplicity, it may prove beneficial to train a diverse set of models. Each model could rely on different features, increasing the likelihood that at least one of them captures the essential ones. Teney et al. [119] achieved such diversity by approximating feature importance through loss gradients and penalizing the dot product of these gradients for all pairs of models. Consequently, they obtained models that were all reasonably accurate, but concurrently made predictions based on different input characteristics.

Another approach involves modifying the training procedure of a single model to incentivize the utilization of multiple features. For instance, Pezeshki et al. [120]

identified that ridge regression introduces undesired coupling between features. They introduced spectral decoupling, which entails applying L2 regularization to the output logits instead of the network weights. The authors derived this approach by analyzing the model's behavior in the NTK [121] regime. However, an intuitive analogy can be drawn to methods from the previous section. Consider the model as an ensemble of two: one that leverages superficial features and another based on more critical ones. After training the biased part, regularization on outputs will compel the unbiased model to focus on more challenging examples. This is because increasing activations for easy answers will no longer lead to a reduction in loss.

5.9 Controlling a known bias

To mitigate biases similar to those induced by simplicity, one approach involves training a distinct model specifically designed to capture the biased features and patterns. Subsequently, the residuals of this specialized model can be used to train the primary model [122]. Alternatively, the primary model can be trained as part of an ensemble alongside the biased model [123], thereby reducing the impact of the bias on the final predictions.

In the domain of computer vision, Touvron et al. [124] demonstrated enhanced data efficiency for vision Transformers by training them not only to predict the correct answers but also to predict the answers provided by a convolutional network. Building upon this work, Ren et al. [125] illustrated that employing two distinct teacher architectures gives even greater benefits.

Naturally, for a model to effectively adopt a particular inductive bias, it must possess the flexibility to learn it. For instance, the LSTM architecture tends to derive fewer advantages from LIME pre-training [28] compared to the Transformer architecture [126].

5.10 Summary

The influence of gradient-based optimizers on successful generalization is moderate, with the model architecture playing a more pivotal role by imposing an appropriate simplicity prior. Understanding the behavior of models during training-including the increasing complexity of represented functions, initialization dependence, and phase transitions-is crucial.

Various approaches aim to incentivize convergence to flatter solutions, despite being formulated in different ways, they optimize for similar properties.

Averaging models in parameter space proves effective, especially when models stem from the same training trajectory or have a common ancestor, such as in fine-tuning scenarios.

While adversarial training may enhance generalization, it demands increased model capacity to learn a more non-linear solution, potentially compromising performance for fixed parameter counts.

Bootstrapping methods encompass generating training data from scratch or from seed examples, leveraging language model reasoning to enhance data quality, and employing model outputs in self-interaction to serve as primary training data sources.

Meta-optimizers offer avenues for improved generalization within the same training budget, through hyperparameter selection for existing optimizers and direct optimization of models. Basic notions like back-propagation can even be parametrized and learned. Moreover, it may be feasible to generate model parameters in a single pass using another model, allowing control over different properties, including generalization.

When simple solutions are deemed inadequate for a task, training a less capable model alongside the main one can capture overly simplistic features and patterns, preventing the main model from relying on them. Alternatively, if undesired bias capture via an auxiliary model is uncertain, the training process can incentivize the model to diversify its feature usage for predictions. Finally, known useful biases represented by specific model architectures or algorithms can be imparted onto models with different architectures, provided they exhibit sufficient flexibility.

6 Inference

Even without updating model parameters, it is possible to influence performance by adjusting how the model is utilized. A prominent example of this is zero-shot and few-shot learning, where simply adding examples to the prompt can enable the model to grasp the task contextually. Further advancements can be made by augmenting the model with memory, additional tools, and sophisticated prompting schemes.

These systems, often referred to as language agents, have garnered significant research attention recently. For a comprehensive overview, one may refer to [127, 128]. In this section, we aim to underscore the pivotal role of such systems in enhancing the capabilities of raw language models, particularly in terms of their generalization.

6.1 In-context learning

In-context learning [129], as exemplified by techniques such as chain-of-thought prompting, offers means to instill desired biases into models without explicit training. For

instance, presenting the model with examples of reasoning in similar tasks can prompt it to employ similar reasoning in its responses [130]. Even the simple addition of phrases like "let's think step by step" before providing answers can give significant effects [131]. In a demonstration by Jojic et al. [132], models executing algorithms step-by-step within prompts showcase the ability to solve specified tasks.

In their work Min et al. [133] integrate pre-training with in-context learning. Specifically, they train the model using in-context learning demonstrations, enhancing its ability to adapt and learn in-context during inference. This approach can be interpreted as an inductive bias toward acquiring further inductive biases.

Garg et al. [134] demonstrate that Transformers exhibit the capacity to learn functions that generate data, including linear functions, two-layer neural networks, or decision trees, following pre-training with corresponding in-context learning examples. This finding suggests a meta-level inductive bias, focusing on the acquisition of specific classes of inductive biases, but ones less prevalent in natural language tasks.

Kirsch et al. [135] have shown that various models can be trained as in-context learners provided they possess a sufficiently large capacity and are exposed to a diverse array of tasks during pre-training. Interestingly, they note that the critical dimension for effectiveness in this context is not the number of parameters, but rather the size of model state (memory).

6.2 Tool use

A popular way to augment agent capabilities in some domain is to connect it to relevant tools. Parisi et al. [136] allow models to invoke tools and show how to bootstrap model to use tools effectively from a few demonstrations of tool use. Komeili et al. [137] allow the model access to internet search and show that it leads to better performance than retrieval-based methods. Gao et al. [138] augment language models with the access to Python interpreter. Schick et al. [139] use question answering, calculator, Wikipedia search, machine translation and calendar as tools. Patil et al. [140] fine-tune a language model to effectively use a vast array of tools via their APIs, including pre-trained models from HuggingFace, TorchHub, and TensorHub. Liu et al. [141] use language model to convert a given task into a formal language and then employ a specialized solver for planning tasks.

6.3 Memory

Memory is a critical aspect of a learning system, particularly influencing its generalization performance, as demonstrated in [135]. Furthermore, a model with enhanced memory capabilities can generate longer outputs without sacrificing coherence, offering advantages for approaches that extend a model's abilities at the expense of longer inference times, as discussed in the next sections. However, numerous models either possess a finite context window or struggle to access information from distant time steps, prompting various efforts to augment them with external memory mechanisms.

Zhou et al. [142] propose RecurrentGPT, which operates similarly to LSTM but utilizes natural language processing for all operations instead of linear algebra. In another approach, Packer et al. [143] conceptualize memory as a tool accessible to the system, enabling functions such as keyword-based search. Additionally, Chen et al. [144] assist the system in handling long documents by constructing a tree of summaries first, allowing the system to navigate it to locate relevant information.

6.4 Mutli-step reasoning

Wang et al. [145] use a technique where multiple solutions are sampled from a model, and the most frequent answer is selected. Since all correct solutions to a problem should give the same correct answer, while incorrect solutions may diverge, this aggregation method tends to enhance the accuracy of proposed solutions. Building upon this idea, Yao et al. [146] introduce the concept of a tree of thoughts, where the model can branch into different lines of reasoning at any point, instead of sampling multiple chains of reasoning and aggregating them afterward. Extending further, Besta et al. [147] advance from trees to arbitrary graphs, enabling the aggregation of ideas even before reaching the final output or refining an idea rather than deriving conclusions from it. Lastly, Sel et al. [148] discover that presenting in-context examples of tree search over ideas is sufficient to encourage a language model to use a similar technique. Furthermore, because the model is not bound to strictly follow the algorithm and can apply heuristics, it performs even better than the original algorithm. For a visual comparison of these approaches, refer to Fig. 2.

6.5 Reflection

Madaan et al. [149] demonstrate that a simple iterative process of generation and self-reflection can enhance the quality of outputs from the same underlying language model. Meanwhile, Shinn et al. [150] employ language models to simulate reinforcement learning algorithms. Instead of relying on gradient descent, they solve the credit assignment problem through self-reflection. To preserve the insights gained, they use an external memory. These

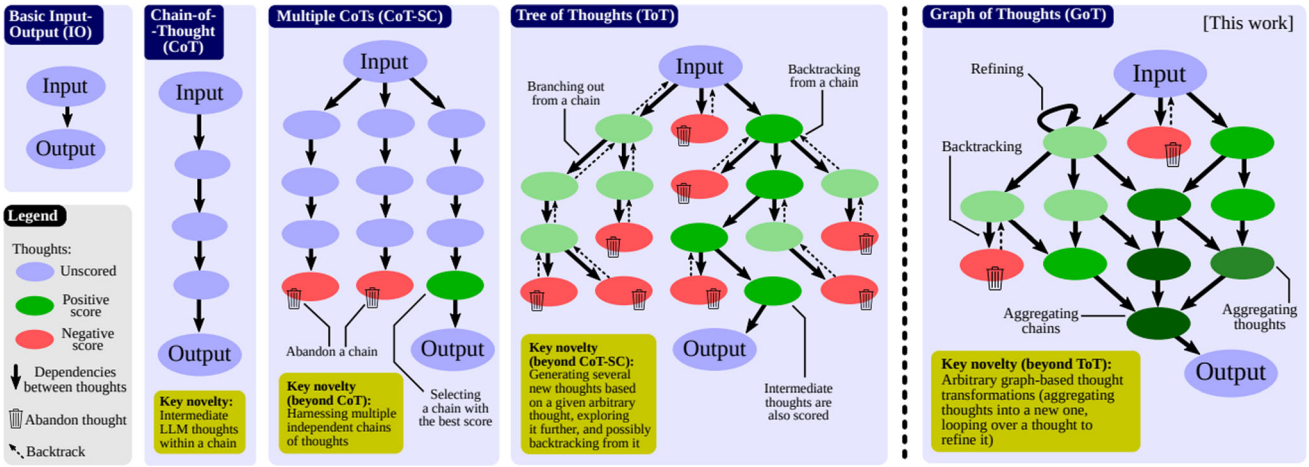

Fig. 2 Prompting strategies for multi-step reasoning. Figure from Besta et al. [147]

approaches allow for a trade-off between longer inference times and improved output quality, where quality encompasses generalization. By engaging in more extensive "thinking," a model may uncover deeper structures within a task, even if said task lies outside the distribution originally encountered by the model.

Zhao et al. [151] integrate external memory storage, reflective analysis of failed attempts, reuse of previous successful strategies, and examination of commonalities among successive attempts. This integrated approach results in an agent capable of experiential learning and adept at generalizing across tasks.

Zelikman et al. [152] present an approach where an agent is represented as a Python function. This function takes an initial solution to the task, a utility function specific to the task, and a language model as inputs, and iteratively refines the solution. Through recursive application of this agent to optimize its own code, they provide a proof-of-concept for self-improvement in language model-based agents.

6.6 Summary

The upper limit of what Transformers can achieve through in-context learning remains uncertain. However, a notable achievement lies in their ability to emulate reasoning strategies and even basic machine learning algorithms. Furthermore, refining in-context learning capabilities is feasible through effective pre-training strategies.

To address the limitations of modern language models, leveraging external tools and memory proves beneficial. Additionally, employing sophisticated prompting techniques enhances robustness against inadvertent reasoning errors and enables learning from experience without re-training.

The comparative table presents the key results of various studies that were considered in this review (Appendix A).

7 Discussion

This paper provides a review of the expanding field of research focused on understanding the generalization capabilities of neural networks. Within this domain, numerous approaches have emerged, operating at various levels of abstraction and targeting different components of the training pipeline. These components encompass diverse aspects such as model architecture, data distribution, optimization techniques, and modifications made during inference. A wide array of strategies has been proposed, each offering insights into enhancing generalization performance. These strategies span from structural modifications to the neural network architecture to adjustments in the distribution of training data.

A critical need exists for the development of theories that clarify the causal mechanisms underlying observed phenomenon. Establishing such theoretical foundations could pave the way for principled approaches aimed at systematically improving generalization across diverse neural network architectures and tasks. As the field continues to evolve, the pursuit of such theories becomes increasingly imperative.

Appendix A Comparison table of different studies

Authors	Models	Data	Results
Irrelevant data			
Papadimitriou and Jurafsky [6]	Pre-training Long Short-Term Memory (LSTM) networks	Structured, non-linguistic data such as MIDI music, Java code, or nested parentheses	Transfer learning as a method for analyzing the encoding of grammatical structure in neural language models; improve test performance on natural language, despite no overlap in surface form or vocabulary
Lu et al. [8]	Fine-tuning only the input and output embeddings of a pre-trained Generative Pre-trained Transformer 2 (GPT-2) model; Frozen Pre-trained Transformer (FPT)	Natural language data; the datasets provided by TAPE	Investigate the capability of a transformer pre-trained on natural language to generalize to other modalities with minimal fine-tuning; language pre-trained Transformers can obtain strong performance on a variety of non-language tasks
Sinha et al. [10]	Masked language model (MLM)	Sentences with randomly shuffled word order	MLM's success can largely be explained by it having learned higher-order distributional statistics that make for a useful prior for subsequent fine-tuning
Irrelevant data			
Krishna et al. [11]	Different T5 models; PG models	The training data from a completely artificial language, comprising random n-grams	Pre-training on documents consisting of character n-grams selected at random nearly matches the performance of models pre-trained on real corpora
Maennel et al. [12]	Several network architectures, including VGG16 and ResNet-18, on CIFAR-10 and ImageNet	Natural image data with entirely random labels	An alignment between the principal components of network parameters and data takes place when training with random labels
Chowers and Weiss [13]	Deep Convolutional Neural Networks (CNNs)	Different dtasets, for example, ImageNet, CIFAR-10	The energy distribution is very different from that of the initial weights and is remarkably consistent across random initializations, datasets, architectures and even when the CNNs are trained with random labels
Mehta et al. [14]	Different pre-trained Transformer models (D-BERT: DistilBERT, BERT-b: BERT- base, RoBERTa: RoBERTa-base, BERT-L: BERT-Large)	Different datasets, for example, MNIST, CIFAR-10	Investigate existing methods in the context of large, pre-trained models and evaluate their performance on a variety of text and image classification tasks
Irrelevant data			
Neyshabur et al. [15]	RI (random initialization), P (pre-trained model), RI-T (model trained on target domain from random initialization), P-T (model trained/fine-tuned on target domain starting from pre-trained weights)	IMAGENET pre-training; CHEXPERT; three sets from DOMAINNET as downstream transfer learning tasks	New tools and analyses to address fundamental questions what enables a successful transfer and which part of the network is responsible for that
Internal representations			
Gurnee and Tegmark [16]	The Llama-2 family of models	Datasets: world, US, NYC places; three temporal datasets (historical figures, artworks, news headlines)	Identify individual "space neurons" and "time neurons" that reliably encode spatial and temporal coordinates

Authors	Models	Data	Results
Li et al. [17]	GPT model	A simple board game, Othello	Evidence of an emergent nonlinear internal representation of the board state; produce "latent saliency maps" that help explain predictions
Jin and Rinard [18]	Transformer model	A synthetic corpus of programs written in a domain-specific language for navigating 2D grid world environments	Evidence that language models of code can learn to represent the formal semantics of programs
Turner et al. [20]	LLMs (LLaMA-3, OPT, GPT-2, and GPT-J)	The Stanford IMDb Large Movie Review Dataset; RealToxicityPrompts; LAMA ConceptNet; OpenWebText	Investigate activation engineering: modifying activations at inference time to predictably alter model behavior
High-quality data			
Eldan and Li [22]	GPT-3.5 and GPT-4	TinyStories—a synthetic dataset of short stories that only contain words that a typical 3 to 4-year-olds usually understand	A framework which uses GPT-4 to grade the content generated by these models as if those were stories written by students and graded by a (human) teacher
Gunasekar et al. [23]	phi-1—a Transformer-based language model for code	CodeTextbook; CodeExercises dataset	phi-1 attains accuracy 50.6% on HumanEval and 55.5% on MBPP
Li et al. [24]	phi-1.5—a 1.3 billion parameter model	A combination of phi-1's training data (7B tokens) and newly created synthetic, "textbook-like" data (roughly 20B tokens)	phi-1.5 exhibits the ability to "think step by step" or perform some rudimentary in-context learning
Surkov and Yamshchikov [25]	GLUE, SuperGLUE, CLUE, and RussianSuperGLUE	The data of GLUE, SuperGLUE, CLUE, RussianSuperGLUE; all the benchmarks in Papers With Code which belong to the NLP field	A theoretical instrument and a practical algorithm to calculate similarity between benchmark tasks
Learning inductive bias from data			
McCoy et al. [26]	Model in framework	Different data	A framework for giving particular linguistic inductive biases to a neural network model
Wu et al. [28]	LIME models, HAT	Synthetic datasets	Hypothesis that Transformer networks are flexible enough to learn inductive bias from suitable generic tasks
Learning inductive bias from data			
Lindemann et al. [29]	Pre-training models	Synthetic data	Show how a structural inductive bias can be efficiently injected into a seq2seq model by pre-training it to simulate structural transformations on synthetic data
Mukherjee et al. [30]	Orca, a 13-billion parameter model that learns to imitate the reasoning process of LFMs (large foundation models); GPT-4	Large-scale and diverse imitation data with judicious sampling and selection	Show that learning from step-by-step explanations, whether these are generated by humans or more advanced AI models, is a promising direction to improve model capabilities and skills
Demonstrations of instruction following and problem solving			
Wei et al. [32]	137B parameter pre-trained language model FLAN	Over 60 NLP datasets verbalized via natural language instruction templates	Number of fine-tuning datasets, model scale, and natural language instructions are key to the success of instruction tuning
Ouyang et al. [33]	Fine-tune GPT-3 using supervised learning; InstructGPT	Collect a dataset of labeler demonstrations	InstructGPT models show improvements in truthfulness and reductions in toxic output generation while having minimal performance regressions on public NLP datasets

Authors	Models	Data	Results
Demonstrations of instruction following and problem solving			
Ye et al. [34]	FLIPPED model	The subset of T0 meta-training datasets	FLIPPED gives particularly large improvements on tasks with unseen labels, outperforming T0-11B by up to +20% average F1 score
Chen et al. [35]	A variety of base LMs	A setup of question answering (QA) with a Google search API	Propose FireAct, a novel approach to fine-tuning LMs with trajectories from multiple tasks and prompting methods, and show having more diverse fine-tuning data can further improve agents
Wang et al. [36]	Tk-INSTRUCT—a Transformer model trained to follow a variety of in-context instructions	Instruction-following datasets; SUPER-NATURAL INSTRUCTIONS, a benchmark of 1,616 diverse NLP tasks and their expert-written instructions	Tk-INSTRUCT outperforms existing instruction-following models such as InstructGPT by over 9% on our benchmark despite being an order of magnitude smaller
Chan et al. [37]	An embedder; a causal Transformer model	Data consist of a mix of 'bursty' and 'non-bursty' sequences	In-context learning traded off against more conventional weight-based learning, and models were unable to achieve both simultaneously
Factors related to generalization			
Chatterjee and Zielinski [46]	Deep architectures trained with gradient methods	Different data	Exploration of the interaction of the gradients of different examples during training
Goldblum et al. [50]	GPT-2, GoogLeNet models	CIFAR-10; CIFAR-100; ImageNet	Architectures designed for a particular domain, such as computer vision, can compress datasets on a variety of seemingly unrelated domains
Role of depth			
Von Oswald et al. [56]	Transformers	Linear regression datasets	Trained Transformers become meta-optimizers, i.e., learn models by gradient descent in their forward pass
Xu et al. [57]	Graph neural networks (GNNs)	Dataset generation	GNNs align with DP (dynamic programming) and to solve different tasks
Model architecture for inductive bias			
Li et al. [61]	A language model augmented with a value function; BLEU; AdverSuc	The OpenSubtitles (OSDb) dataset	Strategy to manipulate the behavior of a neural decoder that enables it to generate outputs that have specific properties of interest
Amos and Kolter [62]	OptNet—a network architecture	Dataset of puzzles	OptNet—a network architecture that integrates optimization problems (specifically in the form of quadratic programs) as individual layers in larger end-to-end trainable deep networks
Model architecture for inductive bias			
von Oswald et al. [63]	Transformers	A large compilation of various English text datasets including parts of Wikipedia, arXiv, and code	Standard next-token prediction error minimization gives rise to a subsidiary learning algorithm, this process corresponds to gradient-based optimization of a principled objective function, which leads to strong generalization performance on unseen sequences

Authors	Models	Data	Results
Graves [64]	Recurrent neural networks	The Hutter prize Wikipedia dataset	Adaptive Computation Time (ACT)—an algorithm that allows recurrent neural networks to learn how many computational steps to take between receiving an input and emitting an output
Banino et al. [65]	Transformers	The bAbI question answering dataset; the English Question Answer dataset	PonderNet—a new algorithm that learns to adapt the amount of computation based on the complexity of the problem at hand
Dehghani et al. [66]	The Universal Transformer	The WMT14 En-De dataset	The Universal Transformer (UT)—a parallel-in-time self-attentive recurrent sequence model which can be cast as a generalization of the Transformer model
Model architecture for inductive bias			
Khandelwal et al. [67]	kNN-LMs	WIKITEXT-103 LM; BOOKS is the Toronto Books Corpus; WIKI-3B is English Wikipedia	kNN-LMs—an extension a pre-trained neural language model (LM) by linearly interpolating it with a k-nearest neighbors (kNN) model; kNN-LM achieves a new state-of-the-art perplexity of 15.79—a 2.9 point improvement with no additional training
Guu et al. [68]	Retrieval-Augmented Language Model pre-training (REALM)	CoNLL-2003 data; Open-QA datasets	Augmentation of language model pre-training with a latent knowledge retriever, which allows the model to retrieve and attend over documents from a large corpus such as Wikipedia
Borgeaud et al. [69]	Retrieval-Enhanced Transformer (Retro)	A 2 trillion token database	Enhancement of auto-regressive language models by conditioning on document chunks retrieved from a large corpus, based on local similarity with preceding tokens
Wu et al. [70]	Extension to the Transformer architecture, called kNN-augmented attention; Transformer XL; Memorizing Transformer	Generic webtext (C4), math papers (arXiv), books (PG-19), code (Github), formal theorems (Isabelle)	Extension of language models with the ability to memorize the internal representations of past inputs
Role of the optimizer			
Mingard et al. [79]	Deep neural networks (DNNs)	MNIST, Fashion-MNIST, IMDb movie review dataset, Ionosphere Dataset	Randomly sampling parameters as opposed to using SGD has negligible effects
Kalimeris et al. [82]	Models trained with SGD	Binary MNIST; CIFAR-10	Key to this work is a new measure of how well one classifier explains the performance of another, based on conditional mutual information
Flatness			
Yang et al. [89]	Neural networks ResNet-20	CIFAR-10	An implicit connection between the local entropy and the Hessian; flatness-aware regularizers to incorporate two metrics into the L2O framework for meta-training optimizers to learn to generalize
Averaging			
Izmailov et al. [92]	State-of-the-art residual networks, PyramidNets, DenseNets, and ShakeShake networks	CIFAR-10, CIFAR-100 and ImageNet	Simple averaging of multiple points along the trajectory of SGD, with a cyclical or constant learning rate, leads to better generalization than conventional training

Authors	Models	Data	Results
Averaging			
Lu et al. [93]	The pre-trained DistilRoBERTa; RoBERTa-Large	The GLUE benchmark	Adaptation of Stochastic Weight Averaging (SWA), a method encouraging convergence to a flatter minimum, to fine-tuning PLMs; this simple optimization technique is able to outperform the state-of-the-art KD methods for compact models
Zhang et al. [94]	Transformer-based neural machine translation models	ImageNet, CIFAR-10/100; the Penn Treebank dataset; the WMT 2014 English-to-German dataset	Lookahead—a new optimization algorithm, that is orthogonal to previous approaches and iteratively updates two sets of weights
Zhang et al. [95]	CNNs and ViTs	CIFAR and ImageNet	Lookaround - a straightforward yet effective SGD-based optimizer leading to flatter minima with better generalization
Adversarial training			
Goodfellow et al. [97]	Deep neural networks	The MNIST dataset	The primary cause of neural networks' vulnerability to adversarial perturbation is their linear nature
Raghunathan et al. [98]	Wide ResNet 40-2 models	CIFAR-10; MNIST	The robustness-generalization trade-off: linear extrapolation facilitates easy generalization but compromises robustness
Bootstrapping			
Huang et al. [103]	Pre-trained large language models	GSM8K; DROP; OpenBookQA; ANLI-A3	LLM is capable of self-improving with only unlabeled datasets
Jung et al. [104]	Impossible-T5; GPT-3.5; ChatGPT	Distilled dataset from 1.5B LMs	IMPOSSIBLE DISTILLATION—a framework for paraphrasing and sentence summarization, that distills a high-quality dataset and model from a low-quality teacher that itself cannot perform these tasks
Meta-optimization			
Chandra et al. [108]	MLPs, CNNs, RNNs	CIFAR-10	Showing how to automatically compute hypergradients with a simple and elegant modification to back-propagation
Wu et al. [109]	MLP network with two hidden layers of 100 units, with ReLU activations; CNN network	MNIST; CIFAR-10	Introduction of a toy problem, a noisy quadratic cost function, on which the authors analyze short-horizon bias by deriving and comparing the optimal schedules for short- and long-time horizons
Avoiding solutions that are too simple			
Clark et al. [115]	LXMERT—a transformer-based model that has been pre-trained on image-caption data; the pre-trained uncased BERT-Base model; ResNet-18	Synthetic MNIST; VQA-CP v2 dataset; the MNLI Hard sentence pair classification dataset; ImageNet	A method that can automatically detect and ignore the dataset-specific patterns (dataset biases)
Nam et al. [116]	MLP with three hidden layers, ResNet-20, and ResNet-18	Colored MNIST; Corrupted CIFAR-10	A failure-based debiasing scheme by training a pair of neural networks simultaneously
Chuang et al. [118]	LLaMA family models	TruthfulQA; FACTOR (News/Wiki)	Decoding by Contrasting Layers (DoLa) approach obtains the next-token distribution by contrasting the differences in logits obtained from projecting the later layers versus earlier layers to the vocabulary space

Authors	Models	Data	Results
Learning diverse solutions			
Teney et al. [119]	A fully connected two-hidden layer MLP classifier	ImageNet-9; MNIST/CIFAR	The simplicity bias can be mitigated and out-of-distribution generalization improved
Pezeshki et al. [120]	A convolutional network with ReLU nonlinearity	CIFAR-10, CIFAR-100, and CIFAR-2; Colored MNIST	Gradient Starvation arises when cross-entropy loss is minimized by capturing only a subset of features relevant for the task, despite the presence of other predictive features that fail to be discovered
Controlling a known bias			
Touvron et al. [124]	Vision Transformer	Imagenet	Competitive convolution-free transformers by training on Imagenet only; a teacher-student strategy specific to Transformers
Ren et al. [125]	Vision Transformer; CivT (cross inductive bias transformer); ResNet; RedNet	ImageNet	A distillation-based method to train vision Transformers; lightweight teachers with different architectural inductive biases (e.g., convolution and involution) to co-advise the student Transformer
In-context learning			
Jojic et al. [132]	GPT-3 family of models	A dataset of 100 random nonrepeating digit sequences of length 5	Execution and description of iterations by regimenting self-attention (IRSA) in one (or a combination) of three ways: 1) Using strong repetitive structure in an example of an execution path of a target program for one particular input, 2) Prompting with fragments of execution paths, and 3) Explicitly forbidding (skipping) self-attention to parts of the generated text
In-context learning			
Min et al. [133]	The pre-trained LM	142 NLP datasets	MetaICL (Meta-training for InContext Learning) - a meta-training framework for few-shot learning where a pre-trained language model is tuned to do in-context learning on a large set of training tasks
Tool use			
Parisi et al. [136]	Transformer-based language models; pre-trained T5 models	Natural Questions (NQ); MathQA	Tool Augmented Language Models (TALM)—combining a text-only approach to augment language models with non-differentiable tools, and an iterative "self-play" technique to bootstrap performance starting from few tool demonstrations
Komeili et al. [137]	T5; BART-Large; BlenderBot variants; Transformer	Collected dataset of human-human conversations	An approach that learns to generate an internet search query based on the context, and then conditions on the search results to finally generate a response
Tool use			
Gao et al. [138]	ProgramAided Language models (PAL)	BIG-Bench Hard	ProgramAided Language models (PAL) - an approach that uses the LLM to read natural language problems and generate programs as the intermediate reasoning steps

Authors	Models	Data	Results
Memory			
Zhou et al. [142]	RECURRENTGPT; RNN	A diverse set of genres of novels including science fiction, romance, fantasy, horror, mystery, and thriller novels	RECURRENTGPT—a languagebased simulacrum of the recurrence mechanism in RNNs
Packer et al. [143]	MemGPT (MemoryGPT)	Multi-Session Chat dataset; augmented MSC dataset, nested KV retrieval dataset, and a dataset of embeddings for 20M Wikipedia articles	Virtual context management—a technique drawing inspiration from hierarchical memory systems in traditional operating systems which provide the illusion of an extended virtual memory via paging between physical memory and disk
Chen et al. [144]	Stable Beluga 2; MEMWALKER; LLaMA 2 Chat	QuALITY, SummScreenFD, and GovReport from the SCROLLS benchmark	MEMWALKER—a method that first processes the long context into a tree of summary nodes
Mutli-step reasoning			
Wang et al. [145]	UL2; GPT-3; LaMDA-137B; PaLM-540B	The Math Word Problem Repository; CommonsenseQA	A decoding strategy, self-consistency, to replace the naive greedy decoding used in chain-of-thought prompting
Yao et al. [146]	GPT-4	Game of 24; for a creative writing task where the input is 4 random sentences; 5×5 mini crosswords	"Tree of Thoughts" (ToT) - a framework for language model inference which generalizes over the popular "Chain of Thought" approach to prompting language models, and enables exploration over coherent units of text ("thoughts") that serve as intermediate steps toward problem solving
Besta et al. [147]	GPT-3.5; Llama-2	Different prompts	Graph of Thoughts (GoT) - a framework that advances prompting capabilities in large language models (LLMs) beyond those offered by paradigms such as Chain of Thought or Tree of Thoughts (ToT)
Reflection			
Madaan et al. [149]	GPT-3.5 and GPT-4	A single LLM as the generator	SELF-REFINE - an approach for improving initial outputs from LLMs through iterative feedback and refinement
Reflection			
Shinn et al. [150]	CoT (GT); ReAct	HotPotQA; multi-step tasks in common household environments in AlfWorld; code writing tasks in competition-like environments with interpreters and compilers in HumanEval; MBPP; LeetcodeHard	Reflexion—a framework to reinforce language agents not by updating weights, but instead through linguistic feedback; Reflexion achieves a 91% pass@1 accuracy on the HumanEval coding benchmark
Zhao et al. [151]	ReAct and Act as main baselines planning LLM agents	HotpotQA; ALFWorld and WebShop; FEVER	Experiential Learning (ExpeL) agent that autonomously gathers experiences and extracts knowledge using natural language from a collection of training tasks
Zelikman et al. [152]	GPT-4	Code generation using LMs as meta-optimizers	Demonstration that LMs like GPT-4 are capable of improving code that leverages the LM itself

Funding Open Access funding enabled and organized by Projekt DEAL. Mrs. Bykova work was supported by the grant for research centers in the field of AI provided by the Analytical Center for the Government of the Russian Federation (ACRF) in accordance with the agreement on the provision of subsidies (identifier of the Agreement 000000D730321P5Q0002) and the agreement with HSE University No. 70-2021-00139.

Data availability This is a review paper; thus, it has no accompanying data.

References

1. He K, Zhang X, Ren S, Sun J (2015) Delving deep into rectifiers: surpassing human-level performance on imagenet classification. In: Proceedings of the IEEE international conference on computer vision, pp 1026–1034
2. Shlegeris B, Roger F, Chan L (2022) Language models seem to be much better than humans at next-token prediction. https://www.alignmentforum.org/posts/htrZrxduciZ5QaCjw/language-models-seem-to-be-much-better-than-humans-at-next
3. Villalobos P, Sevilla J, Heim L, Besiroglu T, Hobbhahn M, Ho A (2022) Will we run out of data? an analysis of the limits of scaling datasets in Machine learning. arXiv preprint arXiv:2211.04325
4. Hendrycks D, Mazeika M, Woodside T (2023) An overview of catastrophic AI risks. arXiv preprint arXiv:2306.12001
5. Han X, Zhang Z, Ding N, Gu Y, Liu X, Huo Y, Qiu J, Yao Y, Zhang A, Zhang L et al (2021) Pre-trained models: past, present and future. AI Open 2:225–250
6. Papadimitriou I, Jurafsky D (2020) Learning music helps you read: Using transfer to study linguistic structure in language models. arXiv preprint arXiv:2004.14601
7. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780
8. Lu K, Grover A, Abbeel P, Mordatch I (2021) Pretrained Transformers as universal computation engines. arXiv preprint. arXiv:2103.05247
9. Radford A, Wu J, Child R, Luan D, Amodei D, Sutskever I et al (2019) Language models are unsupervised multitask learners. OpenAI blog 1(8):9
10. Sinha K, Jia R, Hupkes D, Pineau J, Williams A, Kiela D (2021) Masked language modeling and the distributional hypothesis: order word matters pre-training for little. arXiv preprint arXiv:2104.06644
11. Krishna K, Bigham J, Lipton ZC (2021) Does pretraining for summarization require knowledge transfer? arXiv preprint arXiv:2109.04953
12. Maennel H, Alabdulmohsin IM, Tolstikhin IO, Baldock R, Bousquet O, Gelly S, Keysers D (2020) What do neural networks learn when trained with random labels? Adv Neural Inf Process Syst 33:19693–19704
13. Chowers R, Weiss Y (2023) What do CNNs learn in the first layer and why? A linear systems perspective
14. Mehta SV, Patil D, Chandar S, Strubell E (2021) An empirical investigation of the role of pre-training in lifelong learning. arXiv preprint arXiv:2112.09153
15. Neyshabur B, Sedghi H, Zhang C (2020) What is being transferred in transfer learning? Adv Neural Inf Process Syst 33:512–523
16. Gurnee W, Tegmark M (2023) Language models represent space and time. arXiv preprint arXiv:2310.02207
17. Li K, Hopkins AK, Bau D, Viégas F, Pfister H, Wattenberg M (2022) Emergent world representations: exploring a sequence model trained on a synthetic task. arXiv preprint arXiv:2210.13382
18. Jin C, Rinard M (2023) Evidence of meaning in language models trained on programs. arXiv preprint arXiv:2305.11169
19. Mikolov T, Chen K, Corrado G, Dean J (2013) Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781
20. Turner A, Thiergart L, Udell D, Leech G, Mini U, MacDiarmid M (2023) Activation addition: steering language models without optimization. arXiv preprint arXiv:2308.10248
21. Nanda N (2023) Actually, othello-GPT has a linear emergent world representation. https://www.neelnanda.io/mechanistic-interpretability/othello
22. Eldan R, Li Y (2023) Tinystories: How small can language models be and still speak coherent English? arXiv preprint arXiv:2305.07759
23. Gunasekar S, Zhang Y, Aneja J, Mendes CCT, Del Giorno A, Gopi S, Javaheripi M, Kauffmann P, Rosa G, Saarikivi O, et al (2023) Textbooks are all you need. arXiv preprint arXiv:2306.11644
24. Li Y, Bubeck S, Eldan R, Del Giorno A, Gunasekar S, Lee YT (2023) Textbooks are all you need ii: phi-1.5 technical report. arXiv preprint arXiv:2309.05463
25. Surkov MK, Yamshchikov IP (2024) Vygotsky distance: measure for benchmark task similarity. arXiv preprint arXiv:2402.14890
26. McCoy RT, Grant E, Smolensky P, Griffiths TL, Linzen T (2020) Universal linguistic inductive biases via meta-learning. arXiv preprint arXiv:2006.16324
27. Finn C, Abbeel P, Levine S (2017) Model-agnostic meta-learning for fast adaptation of deep networks. In: International conference on machine learning. PMLR, pp 1126–1135
28. Wu Y, Rabe MN, Li W, Ba J, Grosse RB, Szegedy C (2021) LIME: learning inductive bias for primitives of mathematical reasoning. In: International conference on machine learning. PMLR, pp 11251–11262
29. Lindemann M, Koller A, Titov I (2023) Injecting a structural inductive bias into a seq2seq model by simulation. arXiv preprint arXiv:2310.00796
30. Mukherjee S, Mitra A, Jawahar G, Agarwal S, Palangi H, Awadallah A (2023) Orca: progressive learning from complex explanation traces of GPT-4. arXiv preprint arXiv:2306.02707
31. Yi S, Goel R, Khatri C, Cervone A, Chung T, Hedayatnia B, Venkatesh A, Gabriel R, Hakkani-Tur D (2019) Towards coherent and engaging spoken dialog response generation using automatic conversation evaluators. arXiv preprint arXiv:1904.13015
32. Wei J, Bosma M, Zhao VY, Guu K, Yu AW, Lester B, Du N, Dai AM, Le QV (2021) Finetuned language models are zero-shot learners. arXiv preprint arXiv:2109.01652

33. Ouyang L, Wu J, Jiang X, Almeida D, Wainwright C, Mishkin P, Zhang C, Agarwal S, Slama K, Ray A et al (2022) Training language models to follow instructions with human feedback. Adv Neural Inf Process Syst 35:27730–27744
34. Ye S, Kim D, Jang J, Shin J, Seo M (2022) Guess the instruction! making language models stronger zero-shot learners. arXiv preprint arXiv:2210.02969
35. Chen B, Shu C, Shareghi E, Collier N, Narasimhan K, Yao S (2023) Fireact: toward language agent fine-tuning. arXiv preprint arXiv:2310.05915
36. Wang Y, Mishra S, Alipoormolabashi P, Kordi Y, Mirzaei A, Arunkumar A, Ashok A, Dhanasekaran AS, Naik A, Stap D, et al (2022) Super-naturalinstructions: generalization via declarative instructions on 1600+ NLP tasks. arXiv preprint arXiv:2204.07705
37. Chan S, Santoro A, Lampinen A, Wang J, Singh A, Richemond P, McClelland J, Hill F (2022) Data distributional properties drive emergent in-context learning in Transformers. Adv Neural Inf Process Syst 35:18878–18891
38. Arora S, Ge R, Neyshabur B, Zhang Y (2018) Stronger generalization bounds for deep nets via a compression approach. In: International conference on machine learning. PMLR, pp 254–263
39. Lotfi S, Finzi M, Kapoor S, Potapczynski A, Goldblum M, Wilson AG (2022) PAC-bayes compression bounds so tight that they can explain generalization. Adv Neural Inf Process Syst 35:31459–31473
40. Bartlett P (1996) For valid generalization the size of the weights is more important than the size of the network. In: Advances in neural information processing systems, vol 9
41. Wei C, Ma T (2019) Data-dependent sample complexity of deep neural networks via Lipschitz augmentation. In: Advances in neural information processing systems, vol 32
42. Kawaguchi K, Kaelbling LP, Bengio Y (2017) Generalization in deep learning. arXiv preprint arXiv:1710.05468
43. Hochreiter S, Schmidhuber J (1997) Flat minima. Neural Comput 9(1):1–42
44. Bahri D, Mobahi H, Tay Y (2021) Sharpness-aware minimization improves language model generalization. arXiv preprint arXiv:2110.08529
45. Orvieto A, Kersting H, Proske F, Bach F, Lucchi A (2022) Anticorrelated noise injection for improved generalization. In: International conference on machine learning. PMLR, pp 17094–17116
46. Chatterjee S, Zielinski P (2022) On the generalization mystery in deep learning. arXiv preprint arXiv:2203.10036
47. Bousquet O, Elisseeff A (2000) Algorithmic stability and generalization performance. In: Advances in neural information processing systems, vol 13
48. Wolpert DH (1996) The lack of a priori distinctions between learning algorithms. Neural Comput 8(7):1341–1390
49. Solomonoff RJ (2009) Algorithmic probability: theory and applications. In: Information theory and statistical learning, pp 1–23
50. Goldblum M, Finzi M, Rowan K, Wilson AG (2023) The no free lunch theorem, Kolmogorov complexity, and the role of inductive biases in machine learning. arXiv preprint arXiv:2304.05366
51. Valle-Perez G, Camargo CQ, Louis AA (2018) Deep learning generalizes because the parameter-function map is biased towards simple functions. arXiv preprint arXiv:1805.08522
52. Cover TM (1965) Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. IEEE Trans Electron Comput 3:326–334
53. Eldan R, Shamir O (2016) The power of depth for feedforward neural networks. In: Conference on learning theory. PMLR, pp 907–940
54. Gallicchio C, Scardapane S (2020) Deep randomized neural networks. In: Recent trends in learning from data: tutorials from the INNS big data and deep learning conference (INNSBDDL2019). Springer, pp 43–68
55. Hinton GE, et al (1986) Learning distributed representations of concepts. In: Proceedings of the 8th annual conference of the cognitive science society, vol 1. Amherst, MA, p 12
56. Von Oswald J, Niklasson E, Randazzo E, Sacramento J, Mordvintsev A, Zhmoginov A., Vladymyrov M (2023) Transformers learn in-context by gradient descent. In: International conference on machine learning. PMLR, pp 35151–35174
57. Xu K, Li J, Zhang M, Du SS, Kawarabayashi K-i, Jegelka S (2019) What can neural networks reason about? arXiv preprint arXiv:1905.13211
58. Scarselli F, Gori M, Tsoi AC, Hagenbuchner M, Monfardini G (2008) The graph neural network model. IEEE Trans Neural Netw 20(1):61–80
59. Bellman R (1966) Dynamic programming. Science 153(3731):34–37
60. Ye W, Liu S, Kurutach T, Abbeel P, Gao Y (2021) Mastering atari games with limited data. Adv Neural Inf Process Syst 34:25476–25488
61. Li J, Monroe W, Jurafsky D (2017) Learning to decode for future success. arXiv preprint arXiv:1701.06549
62. Amos B, Kolter JZ (2017) Optnet: differentiable optimization as a layer in neural networks. In: International conference on machine learning. PMLR, pp 136–145
63. Oswald J, Niklasson E, Schlegel M, Kobayashi S, Zucchet N, Scherrer N, Miller N, Sandler M, Vladymyrov M, Pascanu R, et al (2023) Uncovering mesa-optimization algorithms in transformers. arXiv preprint arXiv:2309.05858
64. Graves A (2016) Adaptive computation time for recurrent neural networks. arXiv preprint arXiv:1603.08983
65. Banino A, Balaguer J, Blundell C (2021) Pondernet: learning to ponder. arXiv preprint arXiv:2107.05407
66. Dehghani M, Gouws S, Vinyals O, Uszkoreit J, Kaiser Ł (2018) Universal transformers. arXiv preprint arXiv:1807.03819
67. Khandelwal U, Levy O, Jurafsky D, Zettlemoyer L, Lewis M (2019) Generalization through memorization: nearest neighbor language models. arXiv preprint arXiv:1911.00172
68. Guu K, Lee K, Tung Z, Pasupat P, Chang M (2020) Retrieval augmented language model pre-training. In: International conference on machine learning. PMLR, pp 3929–3938
69. Borgeaud S, Mensch A, Hoffmann J, Cai T, Rutherford E, Millican K, Van Den Driessche GB, Lespiau J-B, Damoc B, Clark A, et al (2022) Improving language models by retrieving from trillions of tokens. In: International conference on machine learning. PMLR, pp 2206–2240
70. Wu Y, Rabe MN, Hutchins D, Szegedy C (2022) Memorizing transformers. arXiv preprint arXiv:2203.08913
71. Sukhbaatar S, Grave E, Lample G, Jegou H, Joulin A (2019) Augmenting self-attention with persistent memory. arXiv preprint arXiv:1907.01470
72. Fu DY, Dao T, Saab KK, Thomas AW, Rudra A, Ré C (2022) Hungry hungry hippos: towards language modeling with state space models. arXiv preprint arXiv:2212.14052
73. Elhage N, Hume T, Olsson C, Nanda N, Henighan T, Johnston S, ElShowk S, Joseph N, DasSarma N, Mann B, Hernandez D, Askell A, Ndousse K, Jones A, Drain D, Chen A, Bai Y, Ganguli D, Lovitt L, Hatfield-Dodds Z, Kernion J, Conerly T, Kravec S, Fort S, Kadavath S, Jacobson J, Tran-Johnson E, Kaplan J, Clark J, Brown T, McCandlish S, Amodei D, Olah C (2022) Softmax

linear units. transformer circuits thread. https://transformer-circuits.pub/2022/solu/index.html
74. Lamb A, He D, Goyal A, Ke G, Liao C-F, Ravanelli M, Bengio Y (2021) Transformers with competitive ensembles of independent mechanisms. arXiv preprint arXiv:2103.00336
75. Graves A, Wayne G, Danihelka I (2014) Neural turing machines. arXiv preprint arXiv:1410.5401
76. Veličković P, Blundell C (2021) Neural algorithmic reasoning. arXiv preprint arXiv:2105.02761
77. Finzi M, Welling M, Wilson AG (2021) A practical method for constructing equivariant multilayer perceptrons for arbitrary matrix groups. In: International conference on machine learning, pp 3318–3328. PMLR
78. Finzi M, Benton G, Wilson AG (2021) Residual pathway priors for soft equivariance constraints. Adv Neural Inf Process Syst 34:30037–30049
79. Mingard C, Valle-Pérez G, Skalse J, Louis AA (2021) Is SGD a Bayesian sampler? Well, almost. J Mach Learn Res 22(1):3579–3642
80. Chiang P-y, Ni R, Miller DY, Bansal A, Geiping J, Goldblum M, Goldstein T (2022) Loss landscapes are all you need: neural network generalization can be explained without the implicit bias of gradient descent. In: The 11th international conference on learning representations
81. Barak B, Edelman B, Goel S, Kakade S, Malach E, Zhang C (2022) Hidden progress in deep learning: SGD learns parities near the computational limit. Adv Neural Inf Process Syst 35:21750–21764
82. Kalimeris D, Kaplun G, Nakkiran P, Edelman B, Yang T, Barak B, Zhang H (2019) SGD on neural networks learns functions of increasing complexity. In: Advances in neural information processing systems, vol 32
83. Mingard C, Skalse J, Valle-Pérez G, Martínez-Rubio D, Mikulik V, Louis AA (2019) Neural networks are a priori biased towards Boolean functions with low entropy. arXiv preprint arXiv:1909.11522
84. Power A, Burda Y, Edwards H, Babuschkin I, Misra V (2022) Grokking: generalization beyond overfitting on small algorithmic datasets. arXiv preprint arXiv:2201.02177
85. Hu MY, Chen A, Saphra N, Cho K (2023) Latent state models of training dynamics. arXiv preprint arXiv:2308.09543
86. Foret P, Kleiner A, Mobahi H, Neyshabur B (2020) Sharpness-aware minimization for efficiently improving generalization. arXiv preprint arXiv:2010.01412
87. Chaudhari P, Choromanska A, Soatto S, LeCun Y, Baldassi C, Borgs C, Chayes J, Sagun L, Zecchina R (2019) Entropy-SGD: biasing gradient descent into wide valleys. J Stat Mech Theory Exp 2019(12):124018
88. Ishida T, Yamane I, Sakai T, Niu G, Sugiyama M (2020) Do we need zero training loss after achieving zero training error? arXiv preprint arXiv:2002.08709
89. Yang J, Chen T, Zhu M, He F, Tao D, Liang Y, Wang Z (2023) Learning to generalize provably in learning to optimize. In: International conference on artificial intelligence and statistics, pp 9807–9825. PMLR
90. Liu Q, Zheng R, Rong B, Liu J, Liu Z, Cheng Z, Qiao L, Gui T, Zhang Q, Huang X-J (2022) Flooding-X: improving BERT's resistance to adversarial attacks via loss-restricted fine-tuning. In: Proceedings of the 60th annual meeting of the association for computational linguistics (Vol 1: Long Papers), pp 5634–5644
91. Polyak BT, Juditsky AB (1992) Acceleration of stochastic approximation by averaging. SIAM J Control Optim 30(4):838–855
92. Izmailov P, Podoprikhin D, Garipov T, Vetrov D, Wilson AG (2018) Averaging weights leads to wider optima and better generalization. arXiv preprint arXiv:1803.05407
93. Lu P, Kobyzev I, Rezagholizadeh M, Rashid A, Ghodsi A, Langlais P (2022) Improving generalization of pre-trained language models via stochastic weight averaging. arXiv preprint arXiv:2212.05956
94. Zhang M, Lucas J, Ba J, Hinton GE (2019) Lookahead optimizer: k steps forward, 1 step back. In: Advances in neural information processing systems, vol 32
95. Zhang J, Liu S, Song J, Zhu T, Xu Z, Song M (2023) Lookaround optimizer: *k* steps around, 1 step average. arXiv preprint arXiv:2306.07684
96. Mandt S, Hoffman MD, Blei DM (2017) Stochastic gradient descent as approximate Bayesian inference. arXiv preprint arXiv:1704.04289
97. Goodfellow IJ, Shlens J, Szegedy C (2014) Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572
98. Raghunathan A, Xie SM, Yang F, Duchi JC, Liang P (2019) Adversarial training can hurt generalization. arXiv preprint arXiv:1906.06032
99. Bubeck S, Lee YT, Price E, Razenshteyn I (2019) Adversarial examples from computational constraints. In: International conference on machine learning. PMLR, pp 831–840
100. Schmidt L, Santurkar S, Tsipras D, Talwar K, Madry A (2018) Adversarially robust generalization requires more data. In: Advances in neural information processing systems, vol 31
101. Madry A, Makelov A, Schmidt L, Tsipras D, Vladu A (2017) Towards deep learning models resistant to adversarial attacks. arXiv preprint arXiv:1706.06083
102. Amini M-R, Feofanov V, Pauletto L, Devijver E, Maximov Y (2022) Self-training: a survey. arXiv preprint arXiv:2202.12040
103. Huang J, Gu SS, Hou L, Wu Y, Wang X, Yu H, Han J (2022) Large language models can self-improve. arXiv preprint arXiv:2210.11610
104. Jung J, West P, Jiang L, Brahman F, Lu X, Fisher J, Sorensen T, Choi Y (2023) Impossible distillation: from low-quality model to high-quality dataset & model for summarization and paraphrasing. arXiv preprint arXiv:2305.16635
105. Wang Y, Kordi Y, Mishra S, Liu A, Smith NA, Khashabi D, Hajishirzi H (2022) Self-instruct: aligning language model with self generated instructions. arXiv preprint arXiv:2212.10560
106. Silver D, Schrittwieser J, Simonyan K, Antonoglou I, Huang A, Guez A, Hubert T, Baker L, Lai M, Bolton A et al (2017) Mastering the game of go without human knowledge. Nature 550(7676):354–359
107. Bricman P, Feeney T (2023) Elements of computational philosophy. https://compphil.github.io/. Accessed 01 Nov 2023
108. Chandra K, Xie A, Ragan-Kelley J, Meijer E (2022) Gradient descent: the ultimate optimizer. NeurIPS
109. Wu Y, Ren M, Liao R, Grosse R (2018) Understanding short-horizon bias in stochastic meta-optimization. arXiv preprint arXiv:1803.02021
110. Almeida D, Winter C, Tang J, Zaremba W (2021) A generalizable approach to learning optimizers. arXiv preprint arXiv:2106.00958
111. Schulman J, Wolski F, Dhariwal P, Radford A, Klimov O (2017) Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347
112. Kirsch L, Schmidhuber J (2021) Meta learning backpropagation and improving it. Adv Neural Inf Process Syst 34:14122–14134
113. Peebles W, Radosavovic I, Brooks T, Efros AA, Malik J (2022) Learning to learn with generative models of neural network checkpoints. arXiv preprint arXiv:2209.12892
114. 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. Adv Neural Inf Process Syst 34:15084–15097

115. Clark C, Yatskar M, Zettlemoyer L (2020) Learning to model and ignore dataset bias with mixed capacity ensembles. arXiv preprint arXiv:2011.03856
116. Nam J, Cha H, Ahn S, Lee J, Shin J (2020) Learning from failure: de-biasing classifier from biased classifier. Adv Neural Inf Process Syst 33:20673–20684
117. Malkin N, Wang Z, Jojic N (2021) Coherence boosting: when your pretrained language model is not paying enough attention. arXiv preprint arXiv:2110.08294
118. Chuang Y-S, Xie Y, Luo H, Kim Y, Glass J, He P (2023) DoLa: decoding by contrasting layers improves factuality in large language models. arXiv preprint arXiv:2309.03883
119. Teney D, Abbasnejad E, Lucey S, Hengel A (2022) Evading the simplicity bias: training a diverse set of models discovers solutions with superior OOD generalization. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 16761–16772
120. Pezeshki M, Kaba O, Bengio Y, Courville AC, Precup D, Lajoie G (2021) Gradient starvation: a learning proclivity in neural networks. Adv Neural Inf Process Syst 34:1256–1272
121. Jacot A, Gabriel F, Hongler C (2018) Neural tangent kernel: convergence and generalization in neural networks. In Advances in neural information processing systems, vol 31
122. He H, Zha S, Wang H (2019) Unlearn dataset bias in natural language inference by fitting the residual. arXiv preprint arXiv:1908.10763
123. Clark C, Yatskar M, Zettlemoyer L (2019) Don't take the easy way out: ensemble based methods for avoiding known dataset biases. arXiv preprint arXiv:1909.03683
124. Touvron H, Cord M, Douze M, Massa F, Sablayrolles A, Jégou H (2021) Training data-efficient image transformers & distillation through attention. In: International conference on machine learning. PMLR, pp 10347–10357
125. Ren S, Gao Z, Hua T, Xue Z, Tian Y, He S, Zhao H (2022) Co-advise: cross inductive bias distillation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 16773–16782
126. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I (2017) Attention is all you need. In: Advances in neural information processing systems, vol 30
127. Xi Z, Chen W, Guo X, He W, Ding Y, Hong B, Zhang M, Wang J, Jin S, Zhou E, et al (2023) The rise and potential of large language model based agents: a survey. arXiv preprint arXiv:2309.07864
128. Wang L, Ma C, Feng X, Zhang Z, Yang H, Zhang J, Chen Z, Tang J, Chen X, Lin Y, et al (2023) A survey on large language model based autonomous agents. arXiv preprint arXiv:2308.11432
129. Brown T, Mann B, Ryder N, Subbiah M, Kaplan JD, Dhariwal P, Neelakantan A, Shyam P, Sastry G, Askell A et al (2020) Language models are few-shot learners. Adv Neural Inf Process Syst 33:1877–1901
130. Wei J, Wang X, Schuurmans D, Bosma M, Xia F, Chi E, Le QV, Zhou D et al (2022) Chain-of-thought prompting elicits reasoning in large language models. Adv Neural Inf Process Syst 35:24824–24837
131. Kojima T, Gu SS, Reid M, Matsuo Y, Iwasawa Y (2022) Large language models are zero-shot reasoners. Adv Neural Inf Process Syst 35:22199–22213
132. Jojic A, Wang Z, Jojic N (2023) GPT is becoming a Turing machine: here are some ways to program it. arXiv preprint arXiv:2303.14310
133. Min S, Lewis M, Zettlemoyer L, Hajishirzi H (2021) MetaICL: learning to learn in context. arXiv preprint arXiv:2110.15943
134. Garg S, Tsipras D, Liang PS, Valiant G (2022) What can Transformers learn in-context? A case study of simple function classes. Adv Neural Inf Process Syst 35:30583–30598
135. Kirsch L, Harrison J, Sohl-Dickstein J, Metz L (2022) General-purpose in-context learning by meta-learning Transformers. arXiv preprint arXiv:2212.04458
136. Parisi A, Zhao Y, Fiedel N (2022) Talm: Tool augmented language models. arXiv preprint arXiv:2205.12255
137. Komeili M, Shuster K, Weston J (2021) Internet-augmented dialogue generation. arXiv preprint arXiv:2107.07566
138. Gao L, Madaan A, Zhou S, Alon U, Liu P, Yang Y, Callan J, Neubig G (2023) Pal: program-aided language models. In: International conference on machine learning. PMLR, pp 10764–10799
139. Schick T, Dwivedi-Yu J, Dessì R, Raileanu R, Lomeli M, Zettlemoyer L, Cancedda N, Scialom T (2023) Toolformer: Language models can teach themselves to use tools. arXiv preprint arXiv:2302.04761
140. Patil SG, Zhang T, Wang X, Gonzalez JE (2023) Gorilla: Large language model connected with massive apis. arXiv preprint arXiv:2305.15334
141. Liu B, Jiang Y, Zhang X, Liu Q, Zhang S, Biswas J, Stone P (2023) LLM+P: Empowering large language models with optimal planning proficiency. arXiv preprint arXiv:2304.11477
142. Zhou W, Jiang YE, Cui P, Wang T, Xiao Z, Hou Y, Cotterell R, Sachan M (2023) RecurrentGPT: interactive generation of (arbitrarily) long text. arXiv preprint arXiv:2305.13304
143. Packer C, Fang V, Patil SG, Lin K, Wooders S, Gonzalez JE (2023) MemGPT: towards LLMs as operating systems. arXiv preprint arXiv:2310.08560
144. Chen H, Pasunuru R, Weston J, Celikyilmaz A (2023) Walking down the memory maze: beyond context limit through interactive reading. arXiv preprint arXiv:2310.05029
145. Wang X, Wei J, Schuurmans D, Le Q, Chi E, Narang S, Chowdhery A, Zhou D (2022) Self-consistency improves chain of thought reasoning in language models. arXiv preprint arXiv:2203.11171
146. Yao S, Yu D, Zhao J, Shafran I, Griffiths TL, Cao Y, Narasimhan K (2023) Tree of thoughts: deliberate problem solving with large language models. arXiv preprint arXiv:2305.10601
147. Besta M, Blach N, Kubicek A, Gerstenberger R, Gianinazzi L, Gajda J, Lehmann T, Podstawski M, Niewiadomski H, Nyczyk P, et al (2023) Graph of thoughts: Solving elaborate problems with large language models. arXiv preprint arXiv:2308.09687
148. Sel B, Al-Tawaha A, Khattar V, Wang L, Jia R, Jin M (2023) Algorithm of thoughts: enhancing exploration of ideas in large language models. arXiv preprint arXiv:2308.10379
149. Madaan A, Tandon N, Gupta P, Hallinan S, Gao L, Wiegreffe S, Alon U, Dziri N, Prabhumoye, S, Yang Y, et al (2023) Self-refine: iterative refinement with self-feedback. arXiv preprint arXiv:2303.17651
150. Shinn N, Cassano F, Labash B, Gopinath A, Narasimhan K, Yao S (2023) Reflexion: language agents with verbal reinforcement learning. arXiv preprint arXiv:2303.11366
151. Zhao A, Huang D, Xu Q, Lin M, Liu Y-J, Huang G (2023) ExpeL: LLM agents are experiential learners. arXiv preprint arXiv:2308.10144
152. Zelikman E, Lorch E, Mackey L, Kalai AT (2023) Self-taught optimizer (STOP): recursively self-improving code generation. arXiv preprint arXiv:2310.02304