Understanding Large Language Models: Learning Processes and AGI (Artificial General Intelligence) Manifestations

R&I

**Abstract**

This paper explores the diverse methods used to teach Large Language Models (LLMs) and the various inferencing techniques employed with these models. While the primary focus lies in understanding these aspects, we also observe that LLMs demonstrate capabilities beyond their explicit training, leading to emergent behaviors and sparks of Artificial General Intelligence (AGI). We delve into specific techniques such as next token prediction, in-context learning (ICL), chain of thought generation, and more. By shedding light on these teaching and inferencing processes, we gain valuable insights into the remarkable capabilities and limitations of LLMs in their pursuit of AGI.

# Introduction

Large Language Models (LLMs) have emerged as a groundbreaking advancement in Artificial Intelligence (AI). These models, built upon powerful transformer architectures and trained on vast amounts of data, have demonstrated remarkable capabilities in language understanding, generation, and reasoning. In this paper, we explore the inner workings of LLMs and their potential towards Artificial General Intelligence (AGI). By understanding the learning processes and manifestations of AGI in LLMs, we gain valuable insights into the capabilities and limitations of these models [[1,](#_bookmark0) [2,](#_bookmark1) [3].](#_bookmark2)

# Next Token Prediction and Transformer Architecture

The transformer architecture’s purpose is to convert sequences of continuous inputs into sequences of continuous outputs. This architecture contains two main blocks, the Encoder and Decoder. The generation of output in both the Encoder and Decoder does not depend on convolutions or recurrence but on a stack of components such as multi-head attention and a fully connected feed- forward network.

## Attention is all you need.

One key feature of the transformer architecture is the attention mechanism, which prioritizes essential elements of the input while discarding non-critical ones. Multi-head attention is used to process multiple input sequences in parallel.

The specific attention mechanism used in Transformers architecture is self-attention. This mechanism automatically adjusts the weight of different elements of the input sequence according to their influence on the output. This is particularly important in language processing tasks, where the meaning of words in a sentence can change based on the context.

The most used attention mechanism integrated into the transformer architecture is the scaled-dot product mechanism, which consists of two parts:

* Self-attention: This function takes the Query, Key, and Value of words in a sentence and finds the scaled dot product attention score (***figure*** ***1***).
* Multi-head attention: This function linearly projects the Query, Key, and Value h times, using different learned projections each time. Attention is calculated for each projection in parallel, producing h outputs. These outputs are then concatenated to produce a result (***figure*** ***2***).

A picture containing text, screenshot, font, line

Description automatically generatedA picture containing text, screenshot, diagram

Description automatically generated

figure 2: Multi-head attention

Figure 1: Self-attention (scaled-dot product attention)

Transformer functionality in a nutshell: The input sequence fed to Encoder, and the Encoder’s processed output is again fed to Decoder, which also takes the previous timestamp output from itself. The Decoder’s output is then converted into output probabilities for each part of the input sequence.

Transformer application in machine translation – an example:

* For each word in input sentence a matrix is created. Its value is the linear weights of each character.
* Positional weights of words are attached to this linear matrix.
* The final matrix is fed into the first layer of the Encoder.
* Self-attention is calculated at each head of the encoder layer, and the output is transferred to the next Encoder layer as input.
* After all layers have completed execution, the output is fed to the Decoder, where similar processing occurs.
* In each Decoder layer, there is an additional layer called masked multi- head attention, which receives the output of the previous layer of Decode as input.
* Finally, the output from the Decoder layers goes through a linear layer, a fully connected network that projects the output into a high-dimensional space where each dimension stands for a word, generating a logits vector (raw prediction vector).
* The logits vector is fed into a SoftMax layer, generating probability values for each index.
* The word corresponding to the index with the highest probability is then predicted from the word vocabulary as the prediction for the input word.
* This process continues for all words in the sentence.

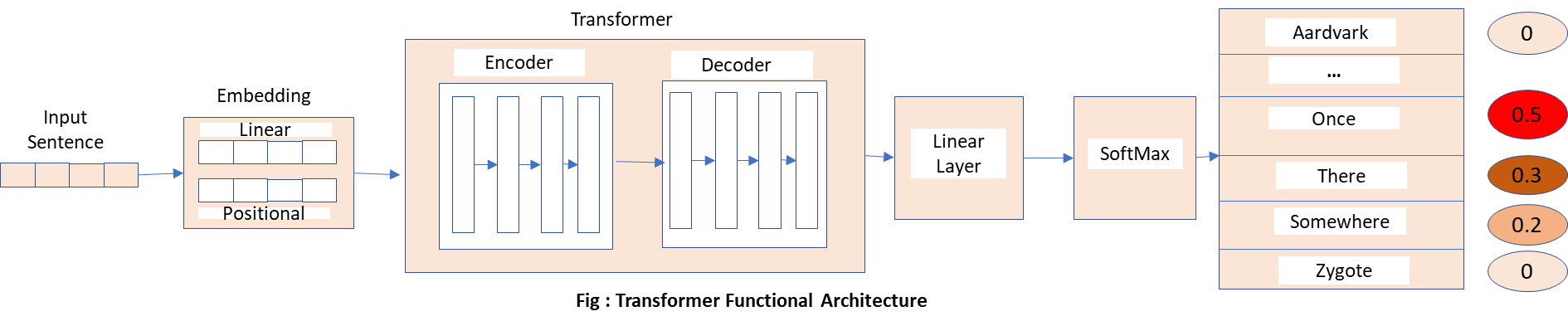


Figure 3: Transformer functional Architecture

Additionally, recent research has explored the integration of external knowledge into attention mechanisms. Techniques such as knowledge-enhanced attention mechanisms or graph attention networks have been proposed to leverage external knowledge graphs or structured data to augment the model's understanding and improve its performance on specific tasks.

## Training and Fine-Tuning Techniques:

Most Large Language Models (LLM) are generative pre-trained transformers, trained with large data sets of unlabeled raw internet text trillions of words which were used to predict the next word as described in the section 2.1. These are loads of unpublished books corpus (i.e., 7000 unpublished books) were used to train the initial GPT model. The pretrained models are fine-tuned in a supervised learning and reinforcement learning from human feedback (RLHF) manner using demonstrations data manner by users or contractors – ideal assistant responses as human feedback in a loop to minimize harmful, untruthful, and biased outputs [13, 16, 17, 18].

The process of RLHF involves three steps:

* First, Dataset of human-generated prompts and responses are collected to fine-tune the LLM model. It involves assembling human-written prompts with suitable human-written responses by asking labelers to write down the correct outputs. These generated datasets are used to further fine-tune the LLM using supervised learning [19].
* Secondly,multiple responses to the same prompt are sampled from the model, for each of a large set of prompts. These responses are then presented to human feedback providers, who rank them according to their preference. The ranked data is then used to train a reward model. The reward model predicts which output humans would prefer [19].
* Thirdly, the reward model is used to further fine-tune and improve the supervised fine-tuned (SFT) model. Human preferences are used as reward signal to fine tune the model. The outcome of this step is called policy model [19].

In summary, RLHF involves using human feedback to generate a reward signal that guides the fine-tuning process. The model is initially fine-tuned using supervised learning with human- provided demonstrations, after which a reward model is trained by ranking different model responses based on quality. The model is then further fine- tuned using Proximal Policy Optimization with the reward model guiding the optimization.

This iterative process allows the model to gradually improve its performance over time, guided by continuous feedback from humans. Simply put,*” LLMs are efficient in-context learners”* [20]. It is important to note that RLHF requires careful implementation to avoid pitfalls such as reward hacking, where the model learns to exploit flaws in the reward model to achieve a higher reward without performing the task better [[5].](#_bookmark4)

## 2.3 In-Context Learning and Its Forms: Weights Frozen

In-Context Learning (ICL) [4,5,6] is a capability observed in Large Language Models (LLMs) where they can perform tasks for which they have seen only a few examples. This capability occurs without any further training or weight adjustment after initial training. ICL encompasses various learning types, including

zero-shot learning, few-shot learning, demonstration learning, and prompting. Despite their differences, all these methods share a common underlying principle: they adapt the model’s output based on a few examples provided in the input context [4, 5, 6].

Understanding the mechanism that enables ICL is an active area of research. Several theories have been proposed, with contributions from researchers at Microsoft, MIT, Stanford, and Google [14]. Researchers from Microsoft and Beijing University have proposed a theory based on meta-gradient optimization [15]. They model this process mathematically and argue that it is equivalent to gradient descent optimization in fine-tuning models.

Conversely, researchers from MIT and Google suggest that Transformer- based ICL can be modeled as the implicit implementation of linear regression models in their hidden activations. They have derived an equation for optimal weights from demonstration pairs. Stanford researchers have approached in- context learning from a Bayesian model perspective. Their observations indicate that training examples provide signals for Bayesian inference, and prompts contribute noisy evidence. They have found a correlation between in-context learning performance and the term frequencies of the pretraining dataset.

There has been a trend towards using pre-trained language models in more flexible and task-agnostic methods for downstream transfer. However, these approaches still require task-specific datasets and fine-tuning, which come with limitations. Achieving robust performance on a task typically requires fine- tuning on a dataset of examples specific to that task, ranging in number from thousands to hundreds of thousands.

ICL can be evaluated under three conditions: ”few-shot learning,” where we allow as many demonstrations as will fit into the model’s context window ;”one-shot learning,” where we allow only one demonstration; and ”zero-shot learning,” where the model predicts the answer given only a natural language description of the task. In these settings, no gradient updates are performed. Results have shown various degrees of performance ranging from outstanding to unsatisfactory depending on the downstream task analyzed.

Furthermore, recent research has investigated the impact of different pre- training objectives on downstream tasks. Studies have compared masked language modeling, where the model predicts missing tokens, with other objectives like permutation-based training or denoising autoencoding. The findings suggest that the choice of pre-training objective can significantly affect the model’s performance on specific tasks, highlighting the importance of understanding the trade-offs and selecting appropriate objectives for desired outcomes.

Controversial aspects in ICL research revolve around the optimal size of the pre-training corpus and the potential biases encoded in the training data. Questions arise regarding the balance between model capacity and the generalizability of learned representations, as well as the fairness and equity implications of training on diverse but potentially biased data sources. Addressing these concerns requires ongoing research, data curation efforts, and developing techniques to mitigate biases in LLMs [[8,](#_bookmark7) [9].](#_bookmark8)

# Chain of Thought Generation

Chain of thought generation in LLMs involves enabling the model to generate coherent responses or perform reasoning processes that involve sequential, branching, recursive, or reflective thinking.

To execute chain of thought generation, LLMs leverage their contextual understanding, often combined with techniques such as beam search or sampling, to explore different paths or sequences of reasoning. The model generates a sequence of tokens step by step, considering the preceding context and the desired problem-solving strategy.

Recent developments in chain of thought generation focus on improving the models’ ability to manage complex and multi-step tasks. Techniques like reinforcement learning or curriculum learning have been explored to guide the generation process and encourage the model to produce more accurate and con- textually appropriate responses. Reinforcement learning frameworks employ reward signals to fine-tune the model’s behavior and encourage desirable generation patterns.

Controversies in chain of thought generation revolve around striking a balance between the model’s flexibility and adherence to instructions. Ensuring that the model generates consistent and accurate responses while avoiding errors or nonsensical outputs is an ongoing challenge. Researchers are actively exploring methods to mitigate error propagation and improve the models’ ability to handle ambiguous instructions or poorly defined problem-solving tasks [[8,](#_bookmark7) [9].](#_bookmark8)

# Emergent Behaviors

Emergent behaviors in larger language models are the result of their increased scale, training data, and complex architectures. As models grow larger, they exhibit capabilities that were not explicitly trained for, highlighting the emergence of novel and unexpected behaviors.

Recent research has focused on analyzing the factors contributing to the emergence of these behaviors. Studies have explored the relationship between model size and performance, investigating how scaling affects the model’s language understanding, reasoning, and creative capabilities. Findings suggest that larger models have a higher capacity to capture complex linguistic patterns, perform fine-grained reasoning, and generate diverse and contextually appropriate responses.

Critically evaluating references, recent studies have highlighted the trade-offs associated with scaling LLMs. While larger models demonstrate enhanced performance on various benchmarks and tasks, they also raise concerns regarding resource consumption, computational requirements, and environmental impact. Researchers are actively exploring techniques for model compression, distillation, and parameter sharing to mitigate these concerns while maintaining the models’ emergent behaviors [[5,](#_bookmark4) [6,](#_bookmark5) [7].](#_bookmark6)

In-depth analysis of emergent behaviors reveals insights into the model’s inner workings and their relationship with AGI. These behaviors indicate that LLMs possess a degree of creative problem-solving, abstraction, and adaptation abilities. They provide a glimpse into the potential of LLMs as steppingstones towards AGI, while also raising questions about the ethical implications and responsible deployment of such powerful models.

# Sparks of AGI

Sparks of AGI in LLMs represent their potential to exhibit AGI-like behaviors in certain aspects. Recent research has focused on understanding and enhancing these sparks to push LLMs closer to AGI.

Technical investigations into sparks of AGI have explored areas such as generalization capabilities, transfer learning, and reasoning mechanisms. Studies have investigated techniques like meta-learning or few-shot learning to enhance the models’ ability to generalize from limited examples and adapt to new tasks with minimal training. Reinforcement learning approaches have been employed to enable models to reason and make intelligent extrapolations, improving their problem-solving and decision-making abilities.

Future research directions involve pushing the boundaries of the sparks of AGI exhibited by LLMs. Areas like explainability, causality, and common- sense reasoning remain open challenges. Exploring ways to incorporate external knowledge sources, such as structured databases or ontologies, could enhance the models’ understanding and reasoning abilities.

Predictions and hypotheses about the future of LLMs revolve around their potential for continual learning, lifelong adaptation, and the ability to acquire new knowledge efficiently [[10,](#_bookmark9) [11,](#_bookmark10) [12,](#_bookmark11) [4,](#_bookmark3) [13].](#_bookmark12) As LLMs continue to evolve, advancements in unsupervised learning, reinforcement learning, and model architectures are expected to enable them to tackle increasingly complex intellectual tasks and approach AGI-like capabilities [[5,](#_bookmark4) [6,](#_bookmark5) [7].](#_bookmark6)

# Conclusion

In conclusion, a deep understanding of the technical intricacies of Large Language Models (LLMs) and their journey towards Artificial General Intelligence (AGI) is essential for unlocking their potential while addressing associated challenges. By incorporating recent developments, engaging with controversial aspects, providing in-depth analysis, critically evaluating references, discussing theoretical implications and future directions, and offering predictions and hypotheses, we can advance our knowledge of LLMs and push the boundaries of AGI research.

# References

1. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. *arXiv preprint arXiv:1706.03762*.
2. SchaefferBrando, R., MirandaSanmi, M., Koyejo (2023). Are Emer- gent Abilities of Large Language Models a Mirage? Retrieved from *https://arxiv.org/abs/2304.15004*.
3. O’Connor, R., (2023). Emergent Abilities of Large Language Models. Retrieved from: [*https://www.assemblyai.c*](http://www.assemblyai.com/blog/emergent-abilities-of-)*om/blo*[*g/emergent-abilities-of-*](http://www.assemblyai.com/blog/emergent-abilities-of-) *large-language-models/*.
4. Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., (“Leveraging pre-trained language models for mining microbiome-disease ...”)

... & Amodei, D. (2020). Language models are few-shot learners. *Advances in neural information processing systems, 33*, 1877-1901.

1. Henderson, P., Islam, R., Bachman, P., Pineau, J., Precup, D., & Meger, D. (2018). Deep reinforcement learning that matters. *arXiv preprint arXiv:1709.06560*.
2. Jaderberg, M., Mnih, V., Czarnecki, W. M., Schaul, T., Leibo, J. Z., Silver, D., & Kavukcuoglu, K. (2017). Human-level performance in first- person multiplayer games with population-based deep reinforcement learn- ing. *arXiv preprint arXiv:1707.06203*.
3. Parisotto, E., & Salakhutdinov, R. (2017). Neural map: Structured memory for deep reinforcement learning. *arXiv preprint arXiv:1702.08360*.
4. Li, J., Monroe, W., Ritter, A., Jurafsky, D., & Galley, M. (2016). Learning to ask: Neural question generation for reading comprehension. *arXiv preprint arXiv:1606.00930*.
5. Wiseman, S., Shieber, S., & Rush, A. M. (2017). Challenges in data-to- document generation. *arXiv preprint arXiv:1707.08052*.
6. Grefenstette, E., Stenetorp, P., & Susskind, J. (2019). Generalization from Simulation: An Examination of Gradients in Neural Sequence-to-Sequence Models. *arXiv preprint arXiv:1906.03152*.
7. Santoro, A., Bartunov, S., Botvinick, M., Wierstra, D., Lillicrap, T., & others (2016). Meta-learning with memory-augmented neural networks. *arXiv preprint arXiv:1605.06065*.
8. Yang, Z., Dai, Z., Salakhutdinov, R., & Cohen, W. W. (2019). “End-to-end learning of semantic role labeling using recurrent neural networks.” (“End-to-end learning of semantic role labeling using recurrent neural ...”) *arXiv preprint arXiv:1901.11214*.
9. Amodei, D., Olah, C., Steinhardt, J., Christiano, P., Schulman, J., & Man´e, D. (2016). Concrete problems in AI safety. *arXiv preprint arXiv:1606.0*
10. *Akyurek, E., Schuurmans, D., Andreas, J., Ma., T. and Zhou, D. What Learning Algorithm is In-context Learning? Investigations with Linear Models, ICLR, 2023*
11. *Dai, D., Sun, T. and Li, D. Hao, Y. and Ma, S. and Sui, Z. and Wei, F. What Can GTP Learn in-Context? Language Models Implicitly Perform Gradient Descent as Meta-Optimizers, ACL 2023, July 2023*
12. *Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., ... & Lowe, R. (2022). Training language models to follow instructions with human feedback. Advances in Neural Information Processing Systems, 35, 27730-27744.*
13. *Bai, Y., Jones, A., Ndousse, K., Askell, A., Chen, A., DasSarma, N., ... & Kaplan, J. (2022). Training a helpful and harmless assistant with reinforcement learning from human feedback. arXiv preprint arXiv:2204.05862.*
14. *Ramponi, Marco ( 2022, Dec 23), How ChatGPT Actually Works. Retrieved from* [*https://www.assemblyai.com/blog/how-chatgpt-actually-works/*](https://www.assemblyai.com/blog/how-chatgpt-actually-works/)
15. *Sagiraju, S. (2023). Unlocking the Power of Human Feedback: Benefits of RLHF. Retrieved from* [*https://appen.com/blog/unlocking-the-power-of-human-feedback-the-benefits-of-rlhf/*](https://appen.com/blog/unlocking-the-power-of-human-feedback-the-benefits-of-rlhf/)Kwon, M., Xie, S. M., Bullard, K., & Sadigh, D. (2023). Reward design with language models. *arXiv preprint arXiv:2303.00001*.
16. Kwon, M., Xie, S. M., Bullard, K., & Sadigh, D. (2023). Reward design with language models. *arXiv preprint arXiv:2303.00001*.