engineering proceedings

MDPI

Proceeding Paper

Applying Existing Large Language Models for Print Circuit Board Routing †

Kangkang Zhang [1,*], Huailong Zhang [1], Aobo Li [1], Zhiping Yang [2] and Xiuqin Chu [1]

[1] School of Electronic Engineering, Xidian University, Xi'an 710000, China; xqchu@mail.xidian.edu.cn (X.C.)
[2] PCB Automation Inc., Rolla, MO 65409, USA
* Correspondence: 22021211668@stu.xidian.edu.cn
† Presented at the 2024 International Conference on Science and Engineering of Electronics (ICSEE'2024), Wuhan, China, 22–26 November 2024.

Abstract: Large language models (LLMs), such as GPT-4.0 and Gemini, have achieved excellent performance on natural-language tasks, and they also show high expectations for logical reasoning. In the realm of print circuit board (PCB) routing, complex routing scenarios still rely on manual routing performed by seasoned engineers, which consumes significant human resources and time. This paper proposes an approach using few-shot and chain-of-thought training LLMs to tackle this issue, enabling LLMs to assist engineers in design tasks with a small number of samples. We tested the performance of LLMs in different routing scenarios with a few examples, validating the applicability of this method. Furthermore, we explored fine-tuning techniques to enhance the effectiveness of the few-shot learning approach, to overcome the limitation of scarce real-world PCB cases, and we employed code synthetic cases to fine-tune the model in place of actual PCB scenarios, ultimately improving the LLMs' capability to manage intricate routing tasks. The results validate the feasibility and effectiveness of this method, offering a promising avenue for reducing the manual burden in PCB design.

Keywords: LLMs; PCB routing; few-shot; chain-of-thought

check for updates

Academic Editor: Fei Wang

Published: 18 February 2025

Citation: Zhang, K.; Zhang, H.; Li, A.; Yang, Z.; Chu, X. Applying Existing Large Language Models for Print Circuit Board Routing. *Eng. Proc.* **2025**, *86*, 2. https://doi.org/10.3390/engproc2025086002

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Large language models, such as GPT-4.0 [1] and Gemini [2], are regarded as a revolutionary breakthrough in the field of artificial intelligence. They have over 12 billion parameters, and are trained to solve complex tasks. In past years, LLMs have achieved excellent performance on natural language tasks, and have created high expectations for their logical reasoning capability. Gemini, for example, uses 100 billion LLMs and was developed by Google AI, and has demonstrated exceptional capabilities in diverse domains [3–5].

One exciting aspect of LLMs is the possibility for zero, one, and few-shot training [6]. The method proposed by OpenAI shows a much easier way to train LLMs; unlike traditional fine-tuning using a sizeable number of on-task training examples (which typically require 500–10,000 examples), it only need several input–output pairs to obtain the targeted outcome. An example is shown in Figure 1.

Another exciting aspect of LLMs is chain-of-thought [7]. This entails guiding language models through a step-by-step process, prompting them to explicitly generate a series of intermediate thoughts or steps leading to a final solution. This enables the model to decompose the problem into smaller components, facilitating the derivation of the correct answer.

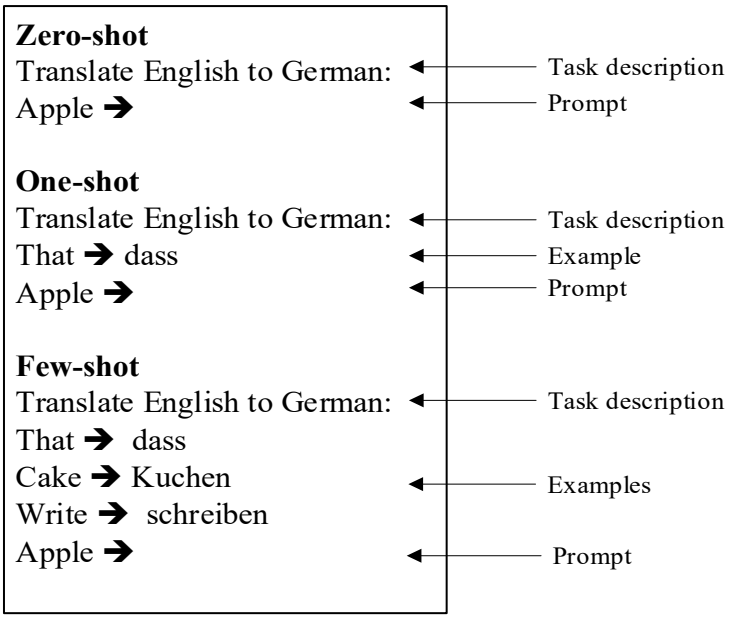

Figure 1. Zero-shot, one-shot, and few-shot examples.

In the realm of PCB routing, the intricate task of routing traces between components are often transformed into maze-solving problems [8], which can then be solved using traditional pathfinding algorithms or reinforcement learning algorithms to achieve optimal routing paths [9,10]. The intricacies of PCB routing scenarios and the existence of diverse routing rules, such as escape routing [11], river routing [12], differential pair routing, and serpentine routing, necessitate the development of specialized algorithms for each scenario; this requirement impedes the efficiency of PCB routing.

Few-shot learning possesses a distinctive attribute: its capability to address problems with specific scenarios. For PCB routing, a small set of routed examples, with some task description, are provided to LLMs to enable it to comprehend the task and generate responses to novel problems. To enhance the model's comprehension capabilities, this paper introduces chain-of-thought, a structured approach to decomposing PCB routing examples into manageable steps, ultimately leading to improved routing performance.

In scenarios with limited training data or poor few-shot performance, this paper proposes an auxiliary method, which uses a small set of synthetic examples (typically easily generated using simple code). This fine-tuning enhances LLMs' understanding of the problem's critical aspects and improves few-shot prediction capabilities.

2. PCB Data Processing

2.1. IPC-2581 Data Format for PCB Extraction

PCBs are extensively used in electronic devices, and their design and manufacturing processes involve the exchange of large amounts of data. To ensure interoperability and accuracy in data exchange, IPC-2581 [13] is selected, which is used as a widely recognized and supported industry standard.

IPC-2518 files adopt a hierarchical structure, with PCB information primarily located in Ecad -> CadData -> Step section. For PCB routing purposes, the following critical data items need to be extracted: component package information (pads' location and size); logical netlist; and existing trace.

2.2. Chain-of-Thought

To achieve the chain-of-thought, this paper decomposes the routing problem into several steps. First the problem description is provided, including an introduction to the routing task, namely finding a path from the starting point to the ending point without touching the regions occupied by obstacles. Following this, there are descriptions of the starting and ending points, along with the line width and obstacles. The starting and ending points are represented by two-dimensional arrays, while obstacles are represented by the points of their four vertices.

The expected output is an array describing the routing trace, where each element, excluding the starting and ending points, represents a turning point.

3. Zero-Shot, One-Shot, and Few-Shot

LLMs have demonstrated remarkable achievements across various domains. Hence, this paper investigates the zero-shot, one-shot, and few-shot capabilities of Gemini, ChatGPT, and Claude, observing that the few-shot setting yields the most prominent results.

3.1. Training Methods and Standards

For PCB routing, the primary objective is to ensure the connectivity between pins while avoiding any contact with obstacles. To assess the routing results, we employ these criteria and compare them against traditional routing algorithms, A* and JPS, to evaluate the practicality of using large language models for routing tasks.

This study leverages existing large language models for few-shot testing, interfacing through exposed API functions, as is shown in Figure 2. The processed data, as shown in Table 1, are input into the LLMs to produce the routing results, and the prompt is adjusted according to the evaluation criteria. This approach is notably efficient, offering rapid performance while consuming less power.

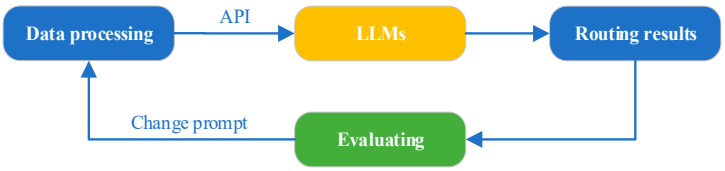

Figure 2. Training process.

Table 1. Example of input and output.

Input	Output
Input1: From start to end, cannot touch barrier area Input2: [4.02890, 6.40000], [4.39630, 5.52320] Input3: [[[4.37500, 5.91500], [4.37500, 6.02500], [4.62500, 6.02500], [4.62500, 5.91500]], 0.005]	From [4.02890, 6.40000] to reach [4.39630, 5.52320], cannot touch the area [[4.37500, 5.91500], [4.37500, 6.02500], [4.62500, 6.02500], [4.62500, 5.91500]], the line width is 0.005, so the trace is [4.02890, 6.40000], [4.18950, 5.77320], [4.34950, 5.77320], [4.40200, 5.52320]

3.2. Zero-Shot

Consider a simple routing scenario, in which a trace needs to be routed from a starting point to an end point and there is an obstacle between them. The trace needs to be routed around the obstacle, as is shown in Figure 3.

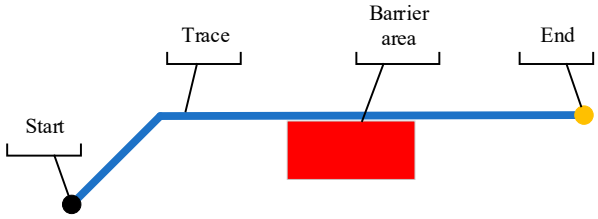

Figure 3. An example of a simple routing case where the trace circumvents obstacles.

Using only task description and prompts, the predicted paths could reach from the starting point to the end point, but failed to avoid obstacles or exhibited a rather tortuous routing, rendering them unsuitable for PCB routing applications, as is shown in Figure 4.

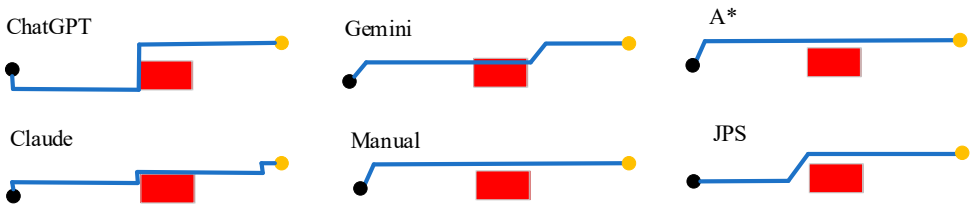

Figure 4. The results predicted by ChatGPT, Gemini, Claude, A*, and JPS.

3.3. One-Shot

Two similar routing scenarios were created. One scenario was used as the example case, and the other was used as the test case. The LLMs were then prompted to generate a routing solution for the test case.

When LLMs were provided with only a single input case for prediction, it often tended to mimic the input case, resulting in an output that lacked generalization to different scenarios. This limitation became even more pronounced when the two scenarios were not very similar. Even if numerous task descriptions were provided and the chain-of-thought method was employed, the LLMs still struggled to generate results that were applicable to PCB routing.

3.4. Few-Shot

A set of multiple similar routing scenarios were created. One of these scenarios was designated as the test case, while the others were used as examples, as shown in Figure 5.

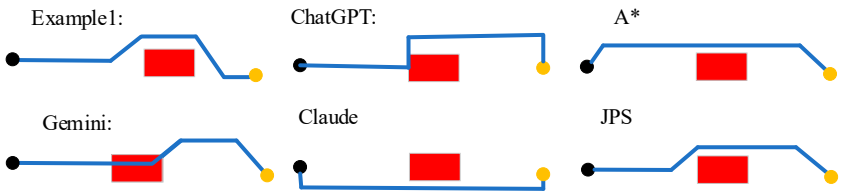

Figure 5. Predicted trace of one-shot with ChatGPT, Gemini, Claude, A*, and JPS.

By controlling the number of input cases and observing the prediction results of large language models, it was found that with the increase in the number of input cases, the prediction results of LLMs became better and better, changing from simple imitation to an understanding of the routing problem, as is shown in Figure 6

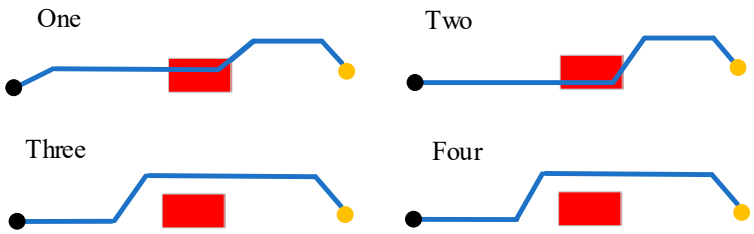

Figure 6. Traces of few-shot with a different number of examples using Gemini.

Few-shot learning requires a small number of labeled examples to achieve good generalization performance. A good example can provide more information and help LLMs learn more effectively. When the relevance of the examples is not that high, LLMs need more examples to learn the underlying patterns. In addition, providing appropriate task description can help the model focus on the important aspects of the examples and improve the learning efficiency, as is shown in Figure 7.

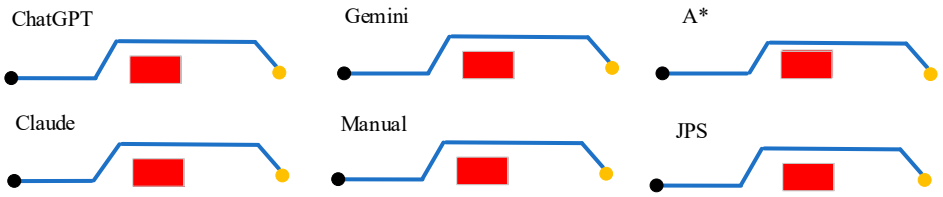

Figure 7. Traces of few-shot with ChatGPT, Gemini, Claude, A*, and JPS.

3.5. Real PCB Case

In simple routing problems, the performance of the few-shot method is excellent. However, in real routing scenarios, the problems are often more complex, involving considerations such as multiple wires and so forth.

To evaluate the few-shot method in a practical setting, a PCB is designed using the Cadence platform. On this PCB, a chip is connected to eight controllers, each of which is linked to three lines. These lines must avoid resistor pads areas. We have manually connected seven of them, and the PCB file was exported in IPC-2581 format. Subsequently, the routing information was extracted from the file.

The extracted data are used for few-shot training; seven groups are inputted as the training data to predict the un-routed group. We restored the predicted data back to the original IPC-2581 file and displayed it in Cadence. As is shown in Figure 8, the results avoid obstacles and achieve the same performance as with manual routing.

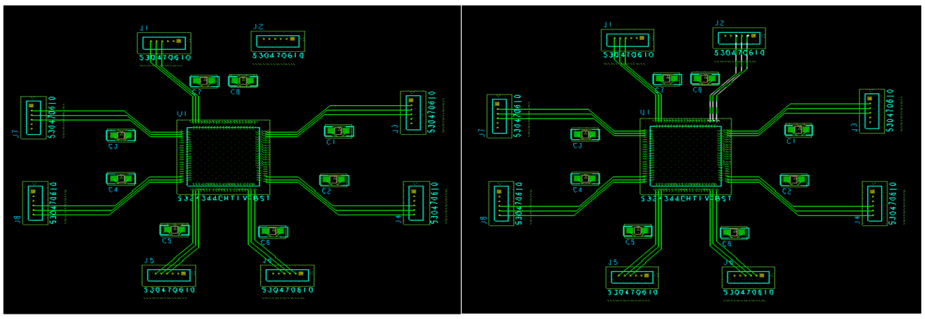

Figure 8. Left picture is the initial PCB layout; partial routing is performed. Right picture is the PCB layout after few-shot routing.

4. Fine-Tuning

In the process of PCB routing, especially in the initial stage, the scarcity of real-project cases poses a significant challenge to the performance of LLMs. To address this issue, this paper employs a fine-tuning strategy to adapt the LLMs to the domain of PCB routing.

4.1. The Fine-Tuning Method

Fine-tuning is a method in which an existing pre-trained model, whether it be open-source or proprietary, is further trained using a diverse set of specific examples. Subsequently, the updated weights are saved as a new model checkpoint.

In the field of natural language processing (NPL), various fine-tuning methods achieve state-of-the-art performance in vertical domains, particularly in fields such as medicine [14], finance [15], and law [16]. Research has conclusively shown that appropriate fine-tuning can lead to superior performance in specialized domains [17], particularly in the context of zero-shot.

This paper presents a novel approach for fine-tuning LLMs to address the challenging problem of PCB routing. The proposed method utilizes synthetic examples as a substitute for real-world PCB examples to fine-tune the models using GPT3.5 Turbo, as is shown in Figure 9, thereby enhancing its performance. Furthermore, a comprehensive workflow for auto routing using the fine-tuned LLMs is designed and successfully achieves a better performance.

System:
You are a professional PCB designer. You need to complete the routing task of the PCB
User:
start and end points[13, 31][71, 12]
four vertices of rectangular obstacles:[[32, 16], [32, 28], [56, 28], [56, 16]] line width: 1
Assistant
trace points:[[13, 31], [15, 29], [27, 17], [27, 14], [28, 13], [29, 12], [71, 12]]

Figure 9. Fine-tuning data example of GPT3.5 Turbo.

To generate samples for fine-tuning, we adopted code synthesis. We randomly selected two points in a plane and generated obstacles of certain size between the two points, finding the optimal path using the traditional path planning algorithm JPS, which can minimize the occurrence of right-angle turns and keep the trace close to obstacles, thereby enhancing the utilization of routing resources. We generated 300 groups of data using this method, and saved the data in the few-shot format.

We divided these 300 sets of cases into two groups: one group was used for few-shot fine-tuning, and the other was reserved for testing to evaluate the learning ability of the LLMs after fine-tuning with this algorithm, as is shown in Figure 10.

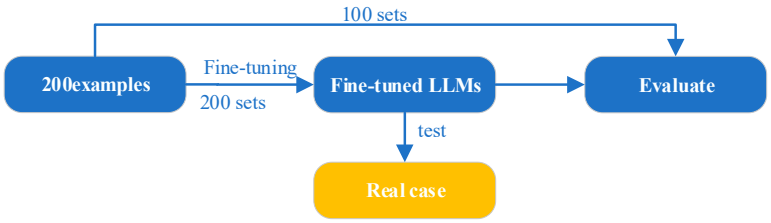

Figure 10. The fine-tuning solution.

4.2. Evaluation

We employed the recommended settings for fine-tuning GPT3.5 Turbo, with three epochs, a batch size of one, and an LR multiplier of two. The final training loss achieved was 0.3099.

To assess the performance of the fine-tuned model, we examined the connectivity and obstacle avoidance of the model's predictions and found that all results met the requirements, as is shown in Table 2.

Table 2. Evaluation results.

	Connectivity	Avoidance
Percentage	100%	100%

4.3. Test Zero-Shot

Following fine-tuning, LLMs underwent zero-shot and one-shot testing, with the testing content remaining consistent with the third section.

The performance of zero-shot was significantly improved, resulting in a reduction in turning points in the trace, as is shown in Figure 11.



Figure 11. Predicted trace of zero-shot with the fine-tuned model.

The performance of one-shot also conformed to routing tasks. The aforementioned examples provide compelling evidence that training LLMs on synthetic examples can effectively enhance their problem-solving capabilities for routing tasks, as is shown in Figure 12.

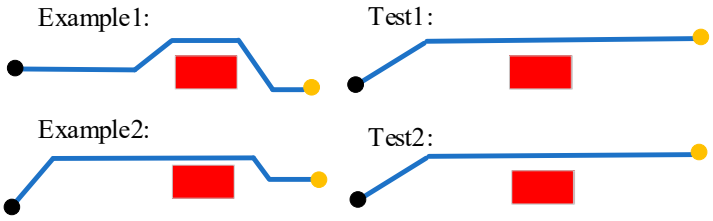

Figure 12. Predicted trace of one-shot with the fine-tuned model.

5. Conclusions

This paper investigates the performance of few-shot methods for PCB routing, and compares the performance of zero-shot, one-shot, and few-shot methods with traditional algorithms and manual results. Few-shot had the best performance, and could achieve the effect of manual routing and traditional algorithms in simple scenarios. This paper also

proposes a fine-tuning scheme to fine-tune LLMs, which further improves the performance of few-shot with fewer cases. The experiments presented in this study offer a valuable reference for leveraging large models in the implementation of PCB routing, and demonstrate the feasibility of utilizing such models for this purpose. If a large model is fine-tuned using a large amount of PCB design files, it is possible to achieve the automatic routing of more complex scenarios.

Author Contributions: Conceptualization, K.Z. and Z.Y.; methodology, K.Z.; software, H.Z.; validation, K.Z., A.L. and H.Z.; formal analysis, K.Z.; investigation, K.Z.; resources, K.Z.; data curation, K.Z.; writing—original draft preparation, K.Z.; writing—review and editing, X.C.; visualization, K.Z.; supervision, X.C.; project administration, X.C.; funding acquisition, X.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Natural Science Foundation of China under grants 62171338 and 62471353, and in part by the Key Research and Development Program of Shaanxi under grant 2024GX-ZDCYL-05-02.

Institutional Review Board Statement: Not applicable

Data Availability Statement: Data are contained within the article.

Acknowledgments: The authors would like to thank Candence for providing software support.

Conflicts of Interest: The authors declare no conflict of interest. PCB Automation Inc. has no commercial conflict of interest.

References

1. GPT-4 Technical Report. Available online: https://arxiv.dosf.top/abs/2303.08774? (accessed on 1 January 2025).
2. Saeidnia, H.R. Welcome to the Gemini Era: Google DeepMind and the Information Industry. Library Hi Tech News, 14 February 2023, Ahead-of-Print. Available online: https://www.emerald.com/insight/content/doi/10.1108/lhtn-12-2023-0214/full/html?trk=public_post_comment-text (accessed on 1 January 2025).
3. Lee, G.G.; Latif, E.; Shi, L.; Zhai, X. Gemini pro defeated by gpt-4v: Evidence from education. *arXiv* **2023**, arXiv:2401.08660.
4. Liu, M.; Chen, C. An evaluation of gpt-4v and gemini in online vqa. *arXiv* **2023**, arXiv:2312.10637.
5. McIntosh, T.R.; Susnjak, T.; Liu, T.; Watters, P.; Halgamuge, M.N. From google gemini to openai q*(q-star): A survey of reshaping the generative artificial intelligence (ai) research landscape. *arXiv* **2023**, arXiv:2312.10868. [CrossRef]
6. Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language models are few-shot learners. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 1877–1901.
7. Wang, X.; Wei, J.; Schuurmans, D.; Le, Q.; Chi, E.; Narang, S.; Chowdhery, A.; Zhou, D. Self-consistency improves chain of thought reasoning in language models. *arXiv* **2022**, arXiv:2203.11171.
8. Cong, J.; Madden, P.H. Performance Driven Multi-Layer General Area Routing for PCB/MCM Designs. In Proceedings of the 35th Annual Design Automation Conference, San Francisco, CA, USA, 15–19 June 1998.
9. Lee, C.Y. An algorithm for path connections and its applications. *IRE Trans. Electron. Comput.* **1961**, *EC-10*, 346–365. [CrossRef]
10. Peshkin, L.; Savova, V. Reinforcement Learning for Adaptive Routing. In Proceedings of the 2002 International Joint Conference on Neural Networks, Honolulu, HI, USA, 12–17 May 2002; IEEE: New York, NY, USA, 2002; IJCNN'02 (Cat. No. 02CH37290); Volume 2.
11. Ozdal, M.M.; Wong, M.D. Simultaneous Escape Routing and Layer Assignment for Dense PCBs. In Proceedings of the IEEE/ACM International Conference on Computer Aided Design, San Jose, CA, USA, 7–11 November 2004; IEEE: New York, NY, USA, 2004; ICCAD-2004; pp. 822–829.
12. Pinter, R.Y. River routing: Methodology and analysis. In *Third Caltech Conference on Very Large Scale Integration*; Springer: Berlin/Heidelberg, Germany, 1983; pp. 141–163.
13. IPC-DPMX (IPC-2581) Consortium. Available online: https://www.ipc2581.com/ (accessed on 1 January 2025).
14. Thirunavukarasu, A.J.; Ting, D.S.; Elangovan, K.; Gutierrez, L.; Tan, T.F.; Ting, D.S. Large language models in medicine. *Nat. Med.* **2023**, *29*, 1930–1940. [CrossRef]
15. Wu, S.; Irsoy, O.; Lu, S.; Dabravolski, V.; Dredze, M.; Gehrmann, S.; Kambadur, P.; Rosenberg, D.; Mann, G. Bloomberggpt: A large language model for finance. *arXiv* **2023**, arXiv:2303.17564.

16. Cui, J.; Li, Z.; Yan, Y.; Chen, B.; Yuan-CoRR, L. Chatlaw: Open-source legal large language model with integrated external knowledge bases. *arXiv* **2023**, arXiv:2306.16092.
17. Ge, Y.; Hua, W.; Mei, K.; Tan, J.; Xu, S.; Li, Z.; Zhang, Y. Openagi: When llm meets domain experts. In Proceedings of the Thirty-Seventh Annual Conference on Neural Information Processing Systems (NeurIPS 2023), New Orleans, LA, USA, 10–16 December 2023. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.