LLM-Analytica: Facilitating Large Language Models for Industrial Analytics

Anonymous Author(s)

ABSTRACT

The rise of Industry 4.0 has led to significant advances in real-time process monitoring and predictive maintenance, aided by machine learning and deep learning tools developed over the past decade. However, on account of a steep learning curve, usage of these tools remains a prerogative of a limited set of users who are proficient in programming. There is a need for good and easy to use analytics platforms that can be used by practitioners in manufacturing industries. This need has unfortunately remained a challenge. The tool handling capability of LLMs holds a new promise, but their performance for manufacturing domain is often poor and largely untested. We introduce LLM-Analytica, a framework for developing end-to-end workflows for industrial analytics designed to perform tasks like process optimization, fault detection and diagnosis, and predictive maintenance for maintaining and improving the plant KPIs such as efficiency, productivity, product quality, reliability, etc. We have integrated 60+ expert-designed modules and used iterative prompting for pipelining to help LLM-Analytica augment the performance of LLMs for industrial analytics. The effectiveness of LLM-Analytica for automating a wide array of industrial analytics tasks is demonstrated and evaluated using expert feedback. This work is expected to accelerate industrial analytics activities and the development of digital twins thereby helping the industry in improving efficiency.

CCS CONCEPTS

• **Applied computing → Engineering**; **Physical sciences and engineering**.

KEYWORDS

Large Language Models, Tool Usage, Industrial Analytics

ACM Reference Format:

Anonymous Author(s). 2018. LLM-Analytica: Facilitating Large Language Models for Industrial Analytics. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation emai (Conference acronym 'XX)*. ACM, New York, NY, USA, 6 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

In the last few years, large language models (LLMs) have transformed various sectors by augmenting natural language tasks [1, 3, 4, 8, 10, 17, 23, 24]. Recent research shows the potential of enhancing the problem-solving ability of LLMs across domains to handle complex textual data and understand the context. But they struggle with seemingly simple domain specific tasks [11, 15, 19]. This is because most of the LLMs available focus on basic language tasks but ignore the usage of domain specific tools. Tool learning, as described by [20], aims to make LLMs more effective in using various tools to do complex tasks. When the LLMs are combined with external tools, they will be more useful and let them act as mediators between end users and different applications [5]. [20] aimed to harness the capabilities of LLMs for effectively interacting with various tools (APIs) and accomplishing complex tasks. [18] combined text-only and a self-play approaches for iterative bootstrap example of tool-use with progressively higher quality. [21] introduced Toolformer, which showed language models can teach themselves to use a range of external tools, including a calculator, a Q&A system, a search engine, a translation system, and a calendar for different tasks. Yao, et al [25] proposed ReAct, Synergizing Reasoning and Acting in Language Models, a method for prompting LLMs in a manner that invokes their ability to reason about problems as well as execute predefined actions which have external effects. For instance, these actions may include interacting with a Wikipedia API or performing tasks within a simulated text-based environment. Bran, et al[2] introduced ChemCrow, an LLM chemistry agent designed to accomplish tasks involved in organic synthesis, drug discovery and materials design. [13] proposed GeneGPT augmenting web APIs for biotechnology information using chain-of-thought approach. Even though open-source LLMs like Llama [23] have become very flexible through instruction tuning, they find it difficult to carry out more advanced tasks, like properly using tools to follow complex human domain specific instructions [7, 22]. Previous studies on tool usage for complex tasks often focused on simulating tool-use capabilities within LLMs, typically restricted to single-tool instructions and limited scenarios. However, real-world situations may demand the integration of multiple tools with sequential or concurrent dependencies on the output of preceding tools to address intricate tasks effectively.

The emergence of Industry 4.0 has revolutionized manufacturing and process industries by integrating a plethora of sensors into industrial processes, resulting in the generation of vast volumes of real-time sensor data. This data holds invaluable insights into the operational health and status of equipment, serving as a critical resource for plant personnel. Industrial analytics workflows refer to the systematic approach followed for applying artificial intelligence (AI) techniques particularly machine learning (ML) or deep learning (DL), to solve industrial problems such as predictive maintenance, anomaly detection and diagnosis, health monitoring, forecasting and more. They involves several steps that are typically followed to develop and deploy AI solutions in an industrial setting. The typical steps involved in solving industrial problems using AI include data gathering, data preprocessing, feature selection, model training and

Conference acronym 'XX, June 03–05, 2018, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/18/06
https://doi.org/XXXXXXX.XXXXXXX

evaluation, model optimization and tuning, deployment, continuous monitoring, and maintenance. By following these systematically, AI techniques for industrial analytics can be leveraged to tackle complex problems, optimize operations, and drive innovation in various industrial domains. Limited research has been conducted on utilizing LLMs for industrial applications [6, 9, 12, 14, 16]. LLMs face difficulties with industrial analytics tasks as they often lack the specific knowledge or understanding of industrial processes and terminology, making it difficult for them to accurately interpret and analyze industrial data. Furthermore, industrial analytics tasks often require more than just language understanding – they may involve complex numerical calculations, data manipulation, and domain-specific reasoning, which LLMs may not excel at without specialized training or adaptation.To overcome this restriction, a potential feasible approach involves enhancing LLMs by integrating them with dedicated external tools or modules developed for executing industrial analytics workflows. These dedicated tools offer precise solutions, which could help in mitigating the intrinsic limitations of LLMs for manufacturing industry domains and improving their overall effectiveness and consistency, reliability, and safety. By integrating LLMs with external tools, consistency checks can be added to enforce uniformity in generated responses and provide benchmarks and metrics to evaluate the consistency of LLM outputs. Accuracy and reliability are critical in industrial settings to maintain operational efficiency and prevent errors. External tools like quality assurance systems or validation algorithms can be combined with LLMs to validate the accuracy of generated text. For example, LLM outputs can be compared against established databases, specifications, or regulatory standards to ensure compliance and reduce the likelihood of misinformation. Given the presence of hazardous conditions or sensitive data in industrial environments, LLM outputs can be assessed for potential risks or regulatory infringements.

Inspired by the successful LLM based applications in other fields, we propose LLM-Analytica, an LLM framework for industrial analytics workflow orchestration. It is designed to streamline the workflows for various industrial analytics tasks across areas such as process optimization, predictive and preventive maintenance, equipment health monitoring and fault detection and diagnosis, optimization etc. involving single-tool and multi-tool scenarios with sequential dependencies on the output of preceding tools to cover real-time complex scenarios using iterative prompting. Iterative prompting allows refining the prompts used to interact with the tools and facilitates transfer of output from one tool to another in scenarios involving multiple tools, resulting in more accurate and effective outcomes.

2 METHODOLOGY: COMBINING LANGUAGE MODELS WITH EXTERNAL TOOLS

The industrial analytics engine, LLM-Analytica, represents a sophisticated framework empowered by LLMs harnesses the power of multiple expert designed tools for manufacturing industry equipment and process analytics. The objective is to empower the language model with the capability to utilize various tools through the mechanism of function calls. LLM-Analytica operates by providing specific instructions to LLM instances (Claude-2 and Claude-3) to carry out designated tasks. These instructions include a list of tool names, descriptions of their functions, and details of the required input. Once provided with these instructions, the LLM is tasked with responding to user prompts using the provided tools as needed. We have integrated over 60+ tools, spanning data preprocessing — such as data cleaning, transformation, outlier analysis, and imputation — to descriptive analytics, covering correlation analysis, statistical methods, data visualization, and feature selection. Additionally, we've incorporated more than 20 regression and classification prediction algorithms, along with semi-supervised and unsupervised anomaly detection and diagnosis techniques like Mahalanobis distance, one-class SVM, Elliptic Envelope, and Autoencoders [26] etc. Moreover, our tool repository includes root cause identification methods [27] and process optimization algorithms. Each of these algorithms has been expertly tailored to suit the unique requirements of industrial datasets and analytics activities. The model follows a systematic approach as shown in Fig. 1. The steps and dependencies outlined in the prompt are identified first, followed by selecting and utilizing appropriate tools for analytics. After identifying the necessary steps, the LLM requests the corresponding tools and their required inputs. The inputs and outputs of each function call can include flat files, past historic records and maintenance logs. They are also not limited solely to text sequences. The program then attempts to execute the requested functions by passing the provided input. If multiple tools are required and have dependencies, the results are combined with the original prompt and presented to the LLM for further analysis as shown in Fig. 2. The process begins by initiating a loop that involves calling LLM with a tool use prompt containing tool specifications and user input. Upon receiving the completion of one tool from LLM-Analytica, it is examined to determine if there are other tools are identified. If another tools are identified, the tool name and parameters are extracted from the prompt, and the respective tool is invoked. Subsequently, the results are formatted and appended to the prompt. This loop iterates, incorporating the updated prompt, until the final output is generated, prompting the termination of the loop. At this stage an Anthropic model, such as claude-2.1 and claude-3-opus-20240229 is employed.

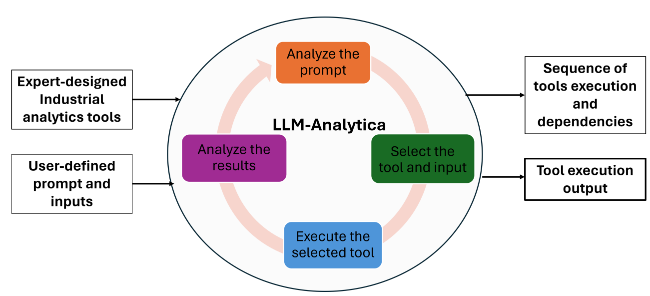


Figure 1: An overview of LLM-Analytica

Figure 3 illustrates an example of an outlier detection tool. Within this function, outlier_detection is invoked upon querying outlier analysis, and outlier_detection_description is supplied to the LLM to indicate the availability of the outlier analysis tool.

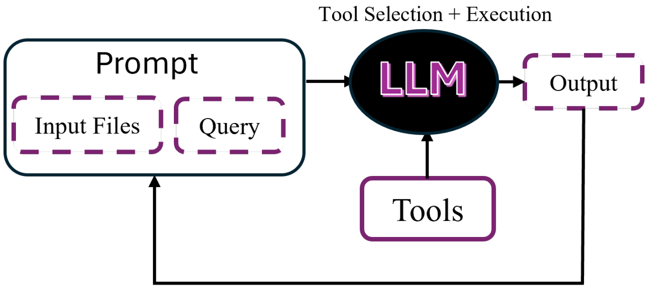


Figure 2: LLM-Analytica Inference Methodology

```
def outlier_detection(filename: str):
  "Perform outlier detection"
  df = pd.read_csv(filename)

  for column in df.columns.tolist():
    fig, ax = plt.subplots(figsize =(5, 3))
    plt.title(column)
    ax.boxplot(df.loc[:,column])
    plt.savefig(column.jpg', dpi=250)
    plt.close()

  out = []
  for column in df.columns.tolist():
    Q1, Q3 = np.nanpercentile(df[column], [25, 75])
    IQR = Q3- Q1
    lower_range = Q1 - (1.5 * IQR)
    upper_range = Q3 + (1.5 * IQR)
    out.append(df[(df[column] > upper_range) | (df[column] < lower_range) ].count())
    df[(df[column] > upper_range) | (df[column] < lower_range) ] = np.nan

  dic = {"Columns Name" :df.columns.tolist(), "No of outliers" :out}
  pd.DataFrame(out).to_csv(+ 'outlier_summary.csv', index=df.columns.tolist())
  df.to_csv('outlier_detection_output.csv', index=False)
  return 'outlier_detection_output.csv'

outlier_detection_description = """
<tool_description>
<tool_name>outlier_detection</tool_name>
<description>
perform outlier analysis and replace outlier values with NaN in given dataframe </description>
<parameters>
<parameter>
<name>filename</name>
<type>string</type>
<description>filename to read and perform outlier detection</description>
</parameter>
</parameters>
</tool_description>
"""
```

Figure 3: Outlier Detection Tool with Syntax

3 RESULTS AND DISCUSSION

We evaluated LLM-Analytica using two distinct types of prompts. The first prompt is for providing textual descriptions of tasks, either utilizing a single tool or multiple tools. The Claude model assesses the necessity of using a tool, determining whether the request requires invoking a tool or it can be handled without tool. If a tool call is identified, then the tool name and parameters are extracted, and the respective tool function is invoked. Figure4 depicts an example of single tool usage using textual prompting. In this case, input data was provided in a csv file and both the models were able to read the input data and perform outlier analysis for all the columns of data. Here, tools_list refers to the list of available tool names and input_data_filename refers to the path and name of input data file. Figure 5 depicts an example of multiple tool usage using textual prompting. Even when prompted with multiple tool requests, both models successfully identified dependencies and invoked the necessary tools, seamlessly passing the output of the previous tool to the next one.

System Prompt: In this environment, you act as a data scientist with access to a range of tools that can be utilized to address the user's queries as necessary.
The available tools are listed below:
<tools>
{tools_list}
</tools>

User Prompt: In {input_data_filename} file, perform outlier analysis.

Figure 4: Example of single tool use using textual description

System Prompt: In this environment, you act as a data scientist with access to a range of tools that can be utilized to address the user's queries as necessary.
The available tools are listed below:
<tools>
{tools_list}
</tools>

User Prompt: In {input_data_filename} file, perform the following steps.
1. Perform outlier analysis.
2. Perform Imputation.
3. Perform feature selection for target variable = "target_variable_name"
4. Build the random forest model using the selected features in the previous step.

Figure 5: Example of multiple tool use using textual description

In a scenario involving multiple tools, LLM-Analytica accurately identified the steps and their dependencies and sequentially queried the tools. It began with the outlier analysis tool, followed by the imputation tool, the feature selection tool and model building tool, effectively utilizing the output of each tool in the subsequent step. The process begins by initiating a loop that involves calling Claude model with a tool use prompt containing tool specifications and user input. Subsequently, the results are formatted and appended to the prompt. This loop iterates, incorporating the updated prompt, until model generates a final output, prompting the termination of the loop. LLM-Analytica effectively applied both Claude-2.1 and Claude-3-opus-20240229 models to accurately utilize tools for basic user inputs, including outlier analysis, imputation, correlation analysis, and visualization-related simple tasks described in a textual format. To the best of our understanding, given the absence of a benchmark dataset, we conducted a manual evaluation of the performance of both Claude models. However, we refrained from comparing them with existing benchmark datasets mentioned in cited literature due to the distinct nature of the current use case. The model's performance was evaluated on the basis of accuracy in tool selection, precise identification of dependencies, and the extent of successful workflow execution. The second type entails presenting historical or maintenance reports and instructing the LLM-Analytica to replicate the methodologies outlined in these reports for other datasets. Prompts used in our experiments are as shown in Fig. 6. The term report_file_path denotes the directory path and filename where the historical report is stored.

System Prompt: In this environment, you act as a data scientist with access to a range of tools that can be utilized to address the user's queries as necessary. These tools can be accessed as follows: You are advised to call only one function at a time, awaiting the results before proceeding to invoke another function.
The available tools are listed below:
<tools>
{tools_list}
</tools>
</tools>

User Prompt: In the file located at {report_file_path}, read the file and extract pertinent details regarding descriptive, data preprocessing, predictive and prescriptive analytics activities. Specifically, focus on information pertaining to processes, steps, and significant actions undertaken. Identify any references to tool names utilized in these activities within the {report_file_path}.
Instructions:
1. Extract information about ML problem formulation and steps applied related to descriptive, data preprocessing, predictive and prescriptive analytics as described in the provided text.
2. Identify the steps, dependencies, and tools necessary for execution.
3. In the {input_data_filename} file, for the {target_variable_name}, follow the identified steps and dependencies, and execute the requisite tools.

Sample Report given as Input: XYZ product Demand Forecast
Executive Summary: XYZ Terminal situated in Nashville, USA, supplies XYZ to five plants in the region. ABCDEFG a management consulting company, approached YYY for PoC to explore the XYZ demand forecast for XYZ terminal using data-based approach. YYY USA along with YYY Research, explored different techniques to forecast XYZ demand. A report containing descriptive analysis and predictive analysis was submitted to ABCDEFG.
Introduction: XYZ Terminal is the facility for re-gasifying the liquified natural gas (XYZ) shipped in by XYZ tanker from the production zones. A conventional terminal has four functions: 1. Berthing of XYZ tankers and unloading or reloading of cargos 2. Storage of XYZ in cryogenic tanks 3. Regasification of XYZ and 4. Send out of this gas into the transmission grid.Invalid source specified..
ABCDEFG a consulting company to XYZ Terminal approached YYY for exploration of data-based forecast of the total XYZ demand which would help it for scheduling of the XYZ tankers. Demand dataset for the period 1st Jan 2021- 19th Sep 2021 was provided by the client. Descriptive analysis followed by Predictive analysis was performed to build the model for forecasting the total gas demand of the terminal.
Detailed Description of our Solution Approach: For building the prediction model using for given dataset, descriptive analysis followed by data preprocessing and predictive analysis was performed.
- Descriptive Analysis: Descriptive analysis was performed by calculating basic statistics and correlation analysis.
a. Basic statistics: calculate basic statistics like mean, standard deviation, variance, minimum and maximum values for the variables in given dataset.
b. Correlation analysis: calculate correlations between variables.
- Data Preprocessing: data is preprocessed by detecting outliers, imputing missing values, feature selection and building random forest model.
a. Outlier Detection: Identified and removed the outlier values in the given dataset.
b. Imputation: For the outlier detected data, removed outlier values and missing values were imputed.
c. Feature selection: To build the prediction model, important features are selected in the data.
- Predictive Analysis: Based on descriptive analysis, for predicting the XYZ demand, Random Forest regression algorithm was used. Details of model building, and results are mentioned in the results and discussion section of this project.

Results and Discussion:
Descriptive Analysis: Following insights were derived from the descriptive analysis:
- Demand of plant A is dominant and shares 56% in the total demand, however, demand of the other plants B, C, D and E are not negligible.

Predictive Analysis: For predictive analysis, prediction model were built using random forest regressor (machine learning) algorithm.
Results and Discussion:
……..
…….

Figure 6: Example of multiple tool use using historical report description

Claude models were able to handle simple tasks like data preprocessing or descriptive analytics, as well as more complex tasks or workflows with many steps that rely on each other. They could pick out important analytics steps and their dependency from the input report. Subsequently, these models could systematically execute the corresponding tools for each identified step, showcasing their proficiency in automating the analytics workflow. The report contained details about XYZ product, but the workflow was tested on weather_history dataset. The Claude-3 model accurately identified the steps outlined in the input report and performed them on the weather_history dataset. Actions identified from Claude-3 models for the given sample input are shown in Fig. 7. The Claude-2 model also identified the steps and dependency outlined in the report. However, when applied to the weather_history dataset, Claude-2 overlooked the correlation analysis step.

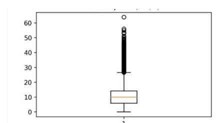
```
partial completion:  Based on the text, here are the key details I extracted related to the descriptive, data preprocessing and predictive analytics:

Descriptive Analysis:
- Calculated basic statistics like mean, standard deviation, variance, minimum and maximum values
- Performed correlation analysis to identify correlations between variables

Data Preprocessing:
- Outlier detection - identified and removed outlier values
- Imputation - imputed missing values and outliers detected
- Feature selection - selected important features to build prediction model

Predictive Analysis:
- Built random forest model to predict target variable (demand)

Tools Used:
- basic_statistics
- correlation_analysis
- outlier_detection
- imputation
- feature_selection
- random_forest_regressor

Steps and dependencies:

1. Perform descriptive analysis
- Invoke basic_statistics tool
- Invoke correlation_analysis tool

2. Preprocess data
- Invoke outlier_detection tool
- Invoke imputation tool
- Invoke feature_selection tool

3. Build predictive model
- Invoke random_forest_regressor tool
```

Figure 7: LLM-Analytica Identified actions for sample input using Claude-3 Model

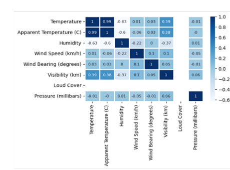
After execution of identified tools, the output of each step is stored in the specified output directory. In case of descriptive analytics basic statistics and correlation analysis were performed. In basic statistics, mean, max, min, standard deviation, variance of each variable were calculated and saved in a csv file as shown in Fig 8(a). In correlation analysis, correlation between variables in identified and correlation values are saved in csv file along with the heatmap as shown in Fig. 8 (c). In data preprocessing, outlier analysis, imputation of missing values and selection of important features were performed. In outlier analysis, Outliers were removed from the dataset and replaced with NaN values. Outlier removed data along with Outliers summary was saved in csv files. It also created the visualization of outliers for each variable. Sample visualization for one variable is shown in Fig 8(b). For Imputation, data was imputed using the forward fill method and saved in csv file. During the feature selection stage, significant features relevant to the target variable were chosen using feature importance scores derived using the Random Forest algorithm. The data of the selected features was then stored in a CSV file. In predictive analytics, the Random Forest regression model was built using the data for selected features. The model's performance on test data is recorded in a JSON file, along with a parity plot illustrating the comparison between actual and predicted values as shown in Fig 8 (d).

Parameter	Mean	SD	Variance	Min	Max
Temperature	11.93268	9.551546	91.23204	-21.8222	39.90556
Apparent Temperature (C)	10.85503	10.69685	114.4225	-27.7167	39.34444
Humidity	0.734899	0.195473	0.03821	0	1
Wind Speed (km/h)	10.81064	6.913571	47.79746	0	63.8526
Wind Bearing (degrees)	187.5092	107.3834	11531.2	0	359
Visibility (km)	10.34732	4.192123	17.5739	0	16.1
Loud Cover	0	0	0	0	0
Pressure (millibars)	1003.236	116.9699	13681.96	0	1046.38

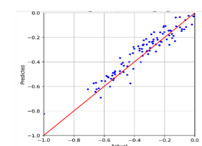
(a) Basic Statistics output



(b) Outlier Analysis for Wind Speed(km/h) feature



(c) Correlation Analysis Heatmap



(d) Actual vs Predicted values for Target variable

Figure 8: Final output obtained from LLM-Analytica with tools

The performance of both Claude models was assessed without additional tool augmentation. However, the models were incapable of implementing the identified dependencies and producing the anticipated output and visualizations. We have tasted other workflows related to fault detection and diagnosis and forecasting of KPIs. While conventional LLMs possess significant code generation capabilities, their outputs often tend to be rudimentary and inflexible, resulting in increased iteration to achieve desired results. Conversely, integrating LLMs with augmented tools enhances the efficiency of result generation by providing more refined and tailored outputs, reducing the need for extensive iterations to achieve desired results, meeting user requirements more efficiently. The external tools often provide domain-specific features and algorithms that enhance the accuracy and relevance of the output. Also, by leveraging the augmented tools, LLMs can identify patterns and faults anomalies in real-time and improve process efficiency and productivity. Overall, the synergy between LLMs and external tools in industrial process analytics offers enhanced data insights, real-time decision-making capabilities, and improved operational efficiency, ultimately driving better outcomes and competitiveness in industrial settings.

4 FUTURE WORK

We are currently strategizing the implementation of fine tuning of open-source large language models such as Llama and Mistral as alternatives to commercial LLMs. Results of the fine tuning evaluation will be added subsequently.

5 CONCLUSION

A framework, LLM-Analytica, for integrating industrial analytics tools with language models was presented. It requires two types of prompts: textual prompts to model tool usage and historical reports from industrial processes. Given a historical report as an input, Large Language Models (LLMs) retrieve intricate analytics workflow details and identify appropriate tools and their dependencies. Subsequently, they execute the workflow components in the requisite order, resulting in systematic execution of the entire workflow and generation of precise results. LLM-Analytica consistently surpassed non-augmented LLMs in producing the expected outputs. This streamlined methodology indicates the potential of LLMs to enhance efficiency and accuracy in carrying out complex analytics activities. Our findings suggest that augmenting the language models with tools enables them to outperform larger non-augmented models.

REFERENCES

[1] Rohan Anil, Andrew M. Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, Eric Chu, Jonathan H. Clark, Laurent El Shafey, Yanping Huang, Kathy Meier-Hellstern, Gaurav Mishra, Erica Moreira, Mark Omernick, Kevin Robinson, Sebastian Ruder, Yi Tay, Kefan Xiao, Yuanzhong Xu, Yujing Zhang, Gustavo Hernandez Abrego, Junwhan Ahn, Jacob Austin, Paul Barham, Jan Botha, James Bradbury, Siddhartha Brahma, Kevin Brooks, Michele Catasta, Yong Cheng, Colin Cherry, Christopher A. Choquette-Choo, Aakanksha Chowdhery, Clément Crepy, Shachi Dave, Mostafa Dehghani, Sunipa Dev, Jacob Devlin, Mark Díaz, Nan Du, Ethan Dyer, Vlad Feinberg, Fangxiaoyu Feng, Vlad Fienber, Markus Freitag, Xavier Garcia, Sebastian Gehrmann, Lucas Gonzalez, Guy Gur-Ari, Steven Hand, Hadi Hashemi, Le Hou, Joshua Howland, Andrea Hu, Jeffrey Hui, Jeremy Hurwitz, Michael Isard, Abe Ittycheriah, Matthew Jagielski, Wenhao Jia, Kathleen Kenealy, Maxim Krikun, Sneha Kudugunta, Chang Lan, Katherine Lee, Benjamin Lee, Eric Li, Music Li, Wei Li, YaGuang Li, Jian Li, Hyeontaek Lim, Hanzhao Lin, Zhongtao Liu, Frederick Liu, Marcello Maggioni, Aroma Mahendru, Joshua Maynez, Vedant Misra, Maysam Moussalem, Zachary Nado, John Nham, Eric Ni, Andrew Nystrom, Alicia Parrish, Marie Pellat, Martin Polacek, Alex Polozov, Reiner Pope, Siyuan Qiao, Emily Reif, Bryan Richter, Parker Riley, Alex Castro Ros, Aurko Roy, Brennan Saeta, Rajkumar Samuel, Renee Shelby, Ambrose Slone, Daniel Smilkov, David R. So, Daniel Sohn, Simon Tokumine, Dasha Valter, Vijay Vasudevan, Kiran Vodrahalli, Xuezhi Wang, Pidong Wang, Zirui Wang, Tao Wang, John Wieting, Yuhuai Wu, Kelvin Xu, Yunhan Xu, Linting Xue, Pengcheng Yin, Jiahui Yu, Qiao Zhang, Steven Zheng, Ce Zheng, Weikang Zhou, Denny Zhou, Slav Petrov, and Yonghui Wu. 2023. PaLM 2 Technical Report. arXiv:2305.10403 [cs.CL]

[2] Andres M Bran, Sam Cox, Oliver Schilter, Carlo Baldassari, Andrew White, and Philippe Schwaller. 2023. Augmenting large language models with chemistry tools. In *NeurIPS 2023 AI for Science Workshop*. https://openreview.net/forum?id=wdGIL6lx3l

[3] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., 1877–1901. https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf

[4] Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, Harsha Nori, Hamid Palangi, Marco Tulio Ribeiro, and Yi Zhang. 2023. Sparks of Artificial General Intelligence: Early experiments with GPT-4. arXiv:2303.12712 [cs.CL]

[5] Tianle Cai, Xuezhi Wang, Tengyu Ma, Xinyun Chen, and Denny Zhou. 2024. Large Language Models as Tool Makers. arXiv:2305.17126 [cs.LG]

[6] WANG Chen, LIU Yan-yi, GUO Tie-zheng, LI Da-peng, HE Tao, LI Zhi, YANG Qing-wen, WANG Hui-han, and WEN Ying-you. 2023. Systems Engineering Issues for Industry Applications of Large Language Model. *Applied Soft Computing* 151 (12 2023), 111165. https://doi.org/10.1016/j.asoc.2023.111165

[7] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An Open-Source Chatbot Impressing GPT-4 with 90%* ChatGPT Quality. https://lmsys.org/blog/2023-03-30-vicuna/

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:1810.04805 [cs.CL]

[9] Tao Fan, Yan Kang, Guoqiang Ma, Weijing Chen, Wenbin Wei, Lixin Fan, and Qiang Yang. 2023. FATE-LLM: A Industrial Grade Federated Learning Framework for Large Language Models. arXiv:2310.10049 [cs.LG]

[10] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. 2022. Training Compute-Optimal Large Language Models. arXiv:2203.15556 [cs.CL]

[11] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of Hallucination in Natural Language Generation. *Comput. Surveys* 55, 12 (March 2023), 1–38. https://doi.org/10.1145/3571730

[12] Jian Jia, Yipei Wang, Yan Li, Honggang Chen, Xuehan Bai, Zhaocheng Liu, Jian Liang, Quan Chen, Han Li, Peng Jiang, and Kun Gai. 2024. Knowledge Adaptation from Large Language Model to Recommendation for Practical Industrial Application. arXiv:2405.03988 [cs.IR]

[13] Qiao Jin, Yifan Yang, Qingyu Chen, and Zhiyong lu. 2024. GeneGPT: Augmenting large language models with domain tools for improved access to biomedical information. *Bioinformatics (Oxford, England)* 40 (02 2024). https://doi.org/10.1093/bioinformatics/btae075

[14] Arpan Kar, Varsha Ps, and Shivakami Rajan. 2023. Unravelling the Impact of Generative Artificial Intelligence (GAI) in Industrial Applications: A Review of Scientific and Grey Literature. *Global Journal of Flexible Systems Management* 24 (09 2023). https://doi.org/10.1007/s40171-023-00356-x

[15] Mojtaba Komeili, Kurt Shuster, and Jason Weston. 2021. Internet-Augmented Dialogue Generation. arXiv:2107.07566 [cs.AI]

[16] Zongjie Li, Wenying Qiu, Pingchuan Ma, Yichen Li, You Li, Sijia He, Baozheng Jiang, Shuai Wang, and Weixi Gu. 2024. An Empirical Study on Large Language Models in Accuracy and Robustness under Chinese Industrial Scenarios. arXiv:2402.01723 [cs.CL]

[17] OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang,

Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. 2024. GPT-4 Technical Report. arXiv:2303.08774 [cs.CL]

[18] Aaron Parisi, Yao Zhao, and Noah Fiedel. 2022. TALM: Tool Augmented Language Models. arXiv:2205.12255 [cs.CL]

[19] Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP Models really able to Solve Simple Math Word Problems?. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tur, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou (Eds.). Association for Computational Linguistics, Online, 2080–2094. https://doi.org/10.18653/v1/2021.naacl-main.168

[20] Yujia Qin, Shengding Hu, Yankai Lin, Weize Chen, Ning Ding, Ganqu Cui, Zheni Zeng, Yufei Huang, Chaojun Xiao, Chi Han, Yi Ren Fung, Yusheng Su, Huadong Wang, Cheng Qian, Runchu Tian, Kunlun Zhu, Shihao Liang, Xingyu Shen, Bokai Xu, Zhen Zhang, Yining Ye, Bowen Li, Ziwei Tang, Jing Yi, Yuzhang Zhu, Zhenning Dai, Lan Yan, Xin Cong, Yaxi Lu, Weilin Zhao, Yuxiang Huang, Junxi Yan, Xu Han, Xian Sun, Dahai Li, Jason Phang, Cheng Yang, Tongshuang Wu, Heng Ji, Zhiyuan Liu, and Maosong Sun. 2023. Tool Learning with Foundation Models. arXiv:2304.08354 [cs.CL]

[21] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language Models Can Teach Themselves to Use Tools. *ArXiv* abs/2302.04761 (2023). https://api.semanticscholar.org/CorpusID:256697342

[22] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford Alpaca: An Instruction-following LLaMA model. https://github.com/tatsu-lab/stanford_alpaca.

[23] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open Foundation and Fine-Tuned Chat Models. arXiv:2307.09288 [cs.CL]

[24] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022. Emergent Abilities of Large Language Models. arXiv:2206.07682 [cs.CL]

[25] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing Reasoning and Acting in Language Models. arXiv:2210.03629 [cs.CL]

[26] Kalyani Zope, Kuldeep Singh, Sri Harsha Nistala, Arghya Basak, Pradeep Rathore, and V. Runkana. 2019. Anomaly Detection and Diagnosis in Manufacturing Systems: A Comparative Study of Statistical, Machine Learning and Deep Learning Techniques. https://papers.phmsociety.org/index.php/phmconf/article/view/815

[27] Kalyani Zope, Tanmaya Singhal, Sri Harsha Nistala, and V. Runkana. 2021. Bayesian network-based root cause analysis and fault pathway identification in complex industrial processes. https://papers.phmsociety.org/index.php/phmconf/article/view/3050