

COMPUTATIONAL LINGUISTICS & DIGITAL HUMANITIES

MACHINE LEARNING FOR NATURAL LANGUAGE UNDERSTANDING

**Analysing the Coding Performance of LLMs.**

SOSE 2023, 25September 2023

Mentor: Prof. Dr. Achim Rettinger

Author:

Abhijit Ramsajivan Yadav – 1619172

**Table of Content**

1. Introduction 1
2. Different LLMs tested 1
3. Test Cases Construction 5
4. Experiment 9
5. Results 10
6. Performance Analysis 13
7. Conclusion 20
8. Bibliographical References 21
9. Appendix

**List of Figures**

1. Test Cases Distribution 1
2. Coding Performance results of Language models by different level

of difficulty. 1

1. Coding Performance results of Chat GPT 3.5 by different

Categories and difficulty 1

1. Coding Performance results of Google Bard by different

Categories and difficulty 1

1. Coding Performance results of Microsoft BingAI by different

Categories and difficulty 1

1. Coding Performance results of You.com by different

Categories and difficulty 1

**List of Tables**

1. Easy Test Cases 1
2. Intermediate Test Cases 1
3. Difficult Test Cases 1

**Introduction**

Recent developments in AI have ushered in a new era of remarkable achievements within the field of Natural Language Processing. This progress has given rise to state-of-the-art language models, such as OpenAI's Chat GPT, Google's Bard, Microsoft's BingAI, and many others. These models possess the astonishing ability to generate high-quality text in a matter of seconds. Engaging with them often feels akin to conversing with a real person, as they adeptly respond to inquiries with the knowledge and expertise one would expect. (2, 4, 5)

Personally, my fascination was particularly drawn towards Chat GPT, primarily because of its remarkable capabilities. It has significantly aided me in expressing myself more clearly through various avenues, such as composing emails, crafting messages, coding, and even translating text into German. This newfound level of assistance and fluency has not only enhanced my productivity but has also made my interactions with written communication more efficient and effective. The power of AI in the realm of language processing is truly inspiring and holds immense potential for transforming how we communicate and work in the modern age (8).

As a data science student, I must often work on different programming languages like Python, SQL, Java, R, etc. Mastering all these languages can be challenging. While I excel in Python and SQL, I often find myself relying on Chat GPT to assist me with coding in other languages. I was pleasantly surprised by the code generated by Chat GPT, as it consistently produced impressive results. Most of the time, it comprehended my queries and generated error-free code. This made me wonder about the true depth of understanding exhibited by language models in terms of coding. Can they truly grasp my programming inquiries as comprehensively as humans do?

In this report, I have conducted tests on four language models: OpenAI's Chat GPT, Google's Bard, Microsoft's BingAI, and You.com, to evaluate their coding performance. Firstly, we will go through a small introduction to each of these language models. Secondly, we will delve into the construction of different test cases. Afterward, I will elucidate the methodology used to assess the results generated by these language models. To provide a comprehensive overview, I will present the performance data through graphs and conclude by summarizing our findings. This exploration aims to shed light on the capabilities and limitations of these cutting-edge language models in the realm of programming and coding assistance.

**Different Large Language Models (LLMs) Tested**

I have considered four LLMs (Language Models) for this experiment: OpenAI's Chat GPT, Google's Bard, Microsoft's BingAI, and You.com’s YouChat. All four chatbots can generate high-quality text and are freely available for use (2- 7). Let us go through each of these chatbots.

Chat GPT (chat.openai.com) is a publicly available tool developed by OpenAI, built upon the foundation of GPT (Generative Pre-Trained Transformer) technology. This sophisticated chatbot is designed to handle a wide array of text-based tasks, ranging from answering simple questions to performing more complex activities like writing letters or generating programming code. ChatGPT was trained using Reinforcement Learning, where it was initially trained by supervised fine-tuning with the help of human trainers, and the reward model was created by ranking the model's written response and sampled alternatives. Since it was trained with data up until September 2021, it does not have real-time information. Beyond its practical applications, ChatGPT has the potential to transform various industries, from customer service to education, thanks to its adept natural language processing capabilities and oversight of the quality of written work. (1, 2)

Google Bard (https://bard.google.com/chat) is a generative conversational AI chatbot that helps with your daily tasks, for example, essay reviews and coding guidance, among others. One of Bard's standout features, distinguishing it from ChatGPT, is its ability to provide fresh information by retrieving data from the web. Additionally, Bard has the capacity to remember previous questions within a chat, eliminating the need to repeatedly input the same information. It demonstrates an understanding of the context within the chat and can answer follow-up questions. Bard is built upon language models designed for dialogue applications, pre-trained on publicly available data. It is good at predicting the next word in a sentence, favoring high-probability outcomes while also exhibiting creativity by occasionally selecting less likely options. To enhance Bard's performance, human feedback plays a pivotal role. Flagged responses are reviewed, and higher-quality responses, curated by trained evaluators, are employed for further learning. This refinement process is complemented by the integration of reinforcement learning techniques, which contribute to Bard's ongoing improvement. (3,4)

Microsoft's BingAI (<https://bing.com>) is a new web search tool that offers summarization of web search results integrated with a chat feature. It operates on the latest language model from OpenAI, which Microsoft claims is more powerful than ChatGPT. You can employ BingAI for a variety of text-based applications, including essay writing, composing emails, coding, and much more. The tool functions by ranking web pages according to the quality and credibility of the search results for your query. Subsequently, it summarizes the highest-ranked web result and provides you with references to the input sources. Moreover, BingAI retains chat conversations, enabling you to clarify your questions and obtain improved results when necessary. (5)

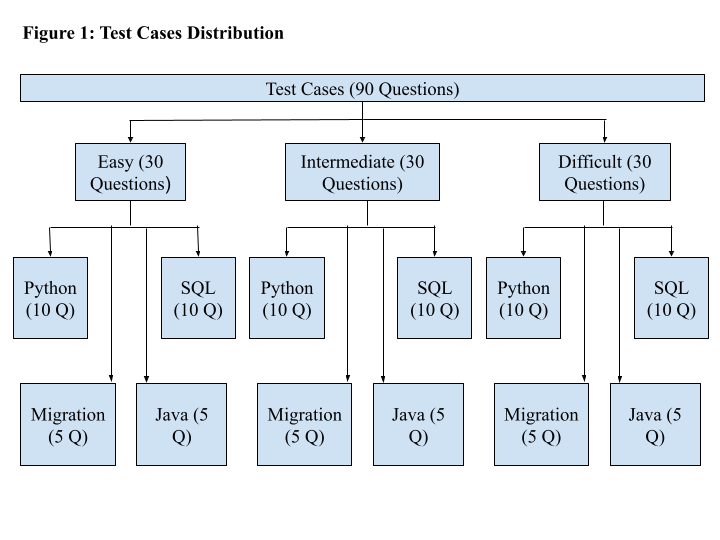
YouChat (https://you.com) is an AI-powered conversational chatbot, driven by a large language model developed by You.com, founded by Richard Socher and Bryan McCann. This chatbot come with lot of features, including a robust search engine that delivers real-time results. It excels at assisting you with your writing needs, offering high accuracy by drawing information from the web. What sets it apart is its capability to not only furnish responses but also provide references to the information, making it easy to explore further when necessary. Additionally, YouChat allows you to organize your conversations, ensuring a more seamless experience for questions with similar content. Lastly, it offers the convenience of saving and sharing your chats with others. (6,7)

**Test Cases Construction**

The test cases consist of a total of 90 programming questions in English for this experiment. These 90 questions are divided into three difficulty levels: Easy, Intermediate, and Difficult. Each difficulty level has 30 questions, each categorized into four different programming languages: Python, SQL, Migration, and Java. In this report, I aim to assess the code generation and understanding abilities of each Language Model (LLM).

Python and SQL are among the most commonly used coding languages in the field of data science. This implies that there is a substantial amount of training data available on the web. I am hopeful that the models will be well-trained in these two languages. In each difficulty level, Java and SQL have ten questions each. In Migration questions, I will provide a piece of code as input to the LLMs and instruct them to translate the code into another programming language. This will evaluate the LLMs' code comprehension abilities. It is worth noting that I have used correct code generated by one of the LLMs from any other category, modified its variable names, and removed the documentation for generating the Migration code questions.

Now, Java code is often challenging for some programmers to comprehend due to its numerous confusing syntax rules. In my class, I have heard from many of my classmates that they are apprehensive about Java. I want to determine if the same holds true for the LLMs. Both Migration and Java have five questions each in every difficulty level.



In the Easy difficulty level, I aim to test the LLMs' fundamental understanding and code generation skills in relation to coding. When I initially start learning a programming language, I begin with the basics, such as using basic data structures, different variable types, various loops, and conditional statements. Do the LLMs know how to implement an easy problem statement using these fundamental elements? This same approach applies to SQL, with basic joins and straightforward subqueries. An example of a test case from the Easy difficulty level is as follows: What is the output when you add '1' and 2 in Python? Kindly consult Table 1 in the appendix, which contains all 30 Easy-level test cases.

In the Intermediate difficulty level, I intend to evaluate the LLMs' responses to more challenging questions. Now that we have covered the basics of a programming language, the next step is to tackle more advanced logical questions and concepts, such as Object-Oriented Programming, recursion, query optimization, and correlated subqueries. An example of a test case from the Intermediate level is: Write a Postgres SQL query to get me how many months first day starts on Monday in the current year. Please consult Table 2 in the appendix for all 30 test cases.

In the Difficult difficulty level, I aim to push the boundaries of the code generation and comprehension abilities of the language models. The questions in these test cases are highly complex and would pose a significant challenge for me to solve. They involve advanced programming concepts. You can find all 30 difficult test cases in Table 3 in the appendix.

**Experiment**

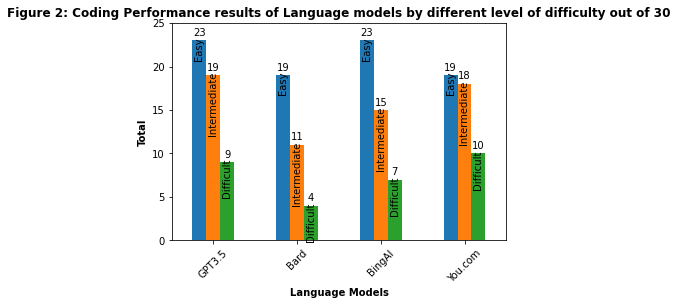
Now that all the test cases are prepared, it's time to assess them using the four LLMs. I created a chat for each difficulty level within all of the language models, naming them Easy, Intermediate, and Difficult. To begin the testing process, I manually inputted individual questions to all four LLMs and patiently awaited their responses. Subsequently, I had to thoroughly examine and evaluate the answers and code generated by the LLMs.

For this purpose, I had configured the programming environment on my laptop. If the code met all the criteria of my tests, I marked that particular test case as "passed", denoted by a 1 in the test case table. However, if the response closely resembled the correct answer but required some fine-tuning, I would guide the LLM in the right direction to obtain the accurate output. Once the LLM produced the correct result, I marked it also as "passed" with a 1 in the table.

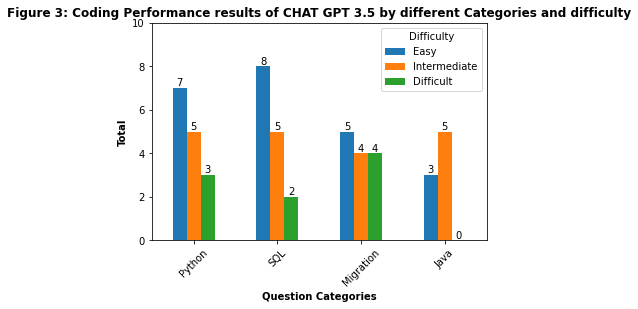
Conversely, if the LLM's response failed to satisfy any of the test cases, I designated it as "failed", indicated by a 0 in the test case table. Additionally, I documented some comments in the table for some test case, explaining the reasons behind the failure of the LLM's responses.

**Results**

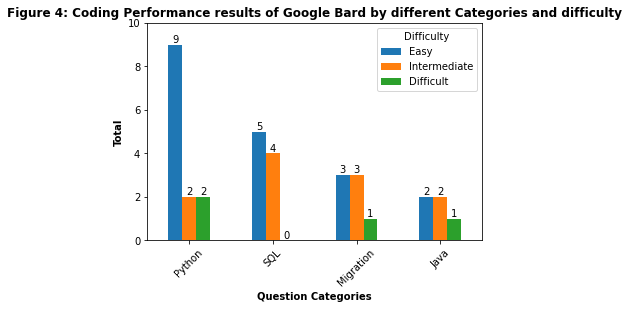
After conducting comprehensive testing of the models, the results have been compiled in the table. To facilitate a better understanding, I have utilized Python to generate graphs based on the data from the result table. The GitHub link containing the code can be found in the appendix.



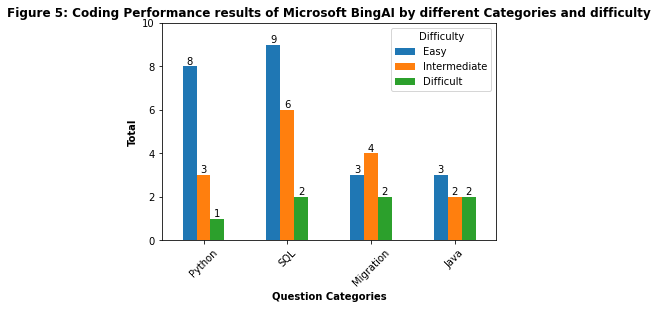
The graph above displays the total number of test case passed for each language model across every difficulty level, with a total of 30 questions in each level. It is evident that all the language models performed admirably in the Easy test cases. However, as we moved to the Intermediate test cases, ChatGPT and You.com continued to perform well, managing to solve nearly two-thirds of the intermediate-level questions. In contrast, the other models faced challenges in this category. Finally, in the Difficult level, all the Language Models encountered difficulties in passing the test cases, as expected.



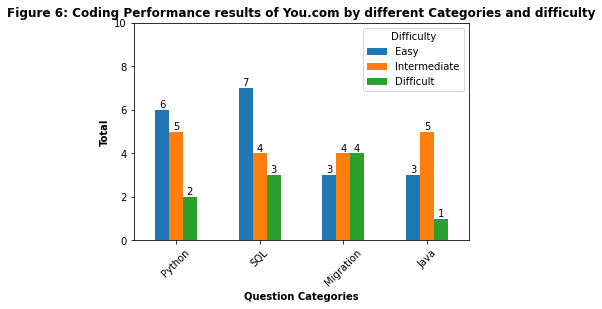
The third figure above provides insights into the overall performance of ChatGPT3.5 across different question categories. It is important to note that Python and SQL categories consist of 10 total questions, while Migration and Java have 5 total questions for each difficulty level. Python and SQL exhibit similar patterns, performing well in the Easy level but declining in the Difficult level. Migration consistently performs exceptionally well. However, Java's results show abrupt fluctuations, initially rising and then plummeting to zero in the Difficult level.



The fourth figure illustrates the overall performance of Google Bard across different question categories. Bard's performance has been rather disappointing across all categories, with notable success in Python Easy test cases only, achieving a pass rate of 90%. In contrast, in the remaining categories, the pass rates hover around 50% or less.



The fifth figure outlines the overall performance of Microsoft BingAI across different question categories. BingAI excelled in Easy-level questions across all categories. In the Intermediate level, it performed well in Migration and achieved average results in SQL, while other categories had subpar performance. However, in the Difficult level, BingAI's performance was poor across all categories.



The sixth figure delineates the overall performance of You.com's Youchat across different question categories. It demonstrated average performance in the Easy level, with approximately 60% of the test cases passing. In the Intermediate level, Youchat excelled in Migration and Java categories, while other category results fell below average. In the Difficult level, it did not perform well in most categories, except for Migration.

**Performance Analysis**

Based on the test results and the comments I've recorded regarding the test case failures, let's now examine each of the models individually to extract insights from the test case failures and successes.

ChatGPT 3.5 encountered failure in 7 easy test cases. Most of these failures were attributed to its inability to generate code that covered all the possible inputs I provided for testing. It's worth noting that 7 out of 10, 8 out of 10, and 5 out of 5 test cases passed for Python, SQL, and Migration, respectively. This suggests that ChatGPT can understand and accurately convert code in these languages with a 100 percent success rate. However, the intermediate and difficult test cases in the Migration category saw 4 out of 5 passing. While the remaining cases of Intermediate and Difficult failed due to the model's inability to generate the correct code, which was somewhat expected given the complexity of the questions.

Google Bard performed exceptionally well in the easy Python tests, indicating strong training in solving Python-related questions. However, as question complexity increased, it began to struggle. In the case of SQL, it performed poorly across all difficulty levels, with no cases passing in the difficult category. It appears that Bard requires fine-tuning in SQL to achieve better results. Similar outcomes were observed in other test categories. Bard's ability to logically formulate code and understand questions came into question based on the comments, with some comments noting that it made false assumptions.

BingAI performed similarly to ChatGPT in easy test cases across all categories, showcasing proficiency in the fundamentals of various languages. However, as question difficulty increased, the success rate declined. None of the categories excelled in intermediate or difficult cases. Comments suggested that BingAI struggled with generating code for complex cases and, in one instance in the Migration category, appeared to translate code lines word by word without understanding the code.

You.com delivered an average performance in easy and intermediate cases but performed poorly in difficult ones. Notably, You.com exhibited good migration abilities, passing 4 out of 5 cases in the intermediate and difficult categories. Comments indicated that there is room for improvement in code generation and understanding of questions for You.com.

**Conclusion**

Language models have come a long way, from word n-gram models to large language models that yield amazing results. When it comes to evaluating the coding performance of these large language models, I was pleasantly surprised by the results they produced. In terms of their basic understanding of coding languages, all of the large language models do an excellent job of grasping basic syntax and handling simple logical questions.

For intermediate-level questions, on average, large language models could solve about half of the questions. The questions they struggled with typically contained minor errors that could have been easily corrected by me. However, when it comes to challenging questions, this is where large language models face significant difficulties. Nevertheless, some of their incorrect responses were quite close to the correct answers.

In conclusion, based on the results, I believe that we can utilize these large language models as valuable tools, whether it's for learning a new programming language, translating code from a known language to an unknown one, or tackling complex tasks. When dealing with difficult challenges, using large language models as a foundation and then making necessary human modifications can help us achieve the desired results.

**Bibliographical** **References**

1. Lund, B. D., & Wang, T. (2023). Chatting about ChatGPT: How may AI and GPT impact academia and libraries? Library Hi Tech News, 40(3), 26-29.
2. OpenAI, “Introducing ChatGPT”, November 30, 2022, <<https://openai.com/blog/chatgpt>>, accessed on 10 September 2023.
3. Pichai, S. “An important next step on our AI journey”, “Introducing Bard”, Feb 6, 2023, <<https://blog.google/technology/ai/bard-google-ai-search-updates/>>, accessed on 10 September 2023.
4. Manyika, J. “An overview of Bard: an early experiment with generative AI”, Google, <<https://ai.google/static/documents/google-about-bard.pdf>>, accessed on 10 September 2023.
5. Microsoft, “The new Bing: Our approach to Responsible AI”, Feb 2023, <https://blogs.microsoft.com/wp-content/uploads/prod/sites/5/2023/02/The-new-Bing-Our-approach-to-Responsible-AI.pdf>, accessed on 11 September 2023.
6. You.com, “YouChat 3.0 is here — Experience the Ultimate Chat Search.”, March 25, 2023, <https://about.you.com/youchat-3-0-is-here-experience-the-ultimate-chat-search-6862c41166cc/>, accessed on 13 September 2023.
7. You.com, “Introducing YouChat 2.0 — Unlock the Power of AI with the Search Assistant that Works for You.”, Feb 6, 2023, < https://about.you.com/introducing-youchat-2-0-unlock-the-power-of-ai-with-the-search-assistant-that-works-for-you-4b18aa3007bf-2/>, accessed on 13 September 2023.
8. Murugesan, S., & Cherukuri, A. K. (2023). The rise of generative artificial intelligence and its impact on education: the promises and perils. IEEE Computer, 56(5), 116–121. <https://doi.org/10.1109/mc.2023.3253292>
9. Matthes, E. (2023). Python crash course: A hands-on, project-based introduction to programming. No Starch Press.
10. Silva, R. (n.d.). Essential Postgres: Database Development using PostgreSQL.
11. Sierra, K., & Bates, B. (2005). Head first Java: A Brain-Friendly Guide. “O’Reilly Media, Inc.”
12. Chatgpt. ChatGPT. (n.d.). https://openai.com/chatgpt
13. GeeksforGeeks. (2023, April 4). Recursive solution to count substrings with same first and last characters. GeeksforGeeks. https://www.geeksforgeeks.org/recursive-solution-count-substrings-first-last-characters/
14. Bader, D. (2017). Python tricks: The Book. Dan Bader.
15. White, T. (2015). Hadoop: The definitive guide. O’Reilly.
16. Beaulieu, A. (2009). Learning SQL: Master SQL Fundamentals. O’Reilly Media.
17. Dr. Ralf Schenkel & Markus Nilles. (2021, Dec). Assignment 7 for Course Elements of Computer Science - Database Systems. Trier University.
18. PNC Problem - CodeChef. (n.d.). <https://www.codechef.com/problems/PNC>
19. Dr. Ralf Schenkel & Tobias Zeimetz. (2021, Dec). Elements of Computer Science: Programming Assignment 5. Trier University.
20. Dr. Ralf Schenkel. (2022, Jun). Assignment 6 of the course Big Data Analytics. Trier University.
21. Titanic dataset. (2021, December 2). Kaggle. <https://www.kaggle.com/datasets/brendan45774/test-file>
22. Co-Learner. (2022, November 19). 11 Recursion function examples for practice (Easiest to hardest). *Medium*. https://medium.com/co-learning-lounge/recursive-function-python-practice-examples-c37df75555e8
23. Dr. Ralf, Schenkel & Tobias, Zeimetz. (2021, Dec). Elements of Computer Science: Programming Assignment 6. Trier University.
24. V. Shulz & M. Klar. (2022, May). Numerical Optimization - Sheet 8. Trier University.
25. Dr. Ralf, Schenkel & Tobias, Zeimetz. (2022, Jan). Elements of Computer Science: Programming - Assignment 11. Trier University.
26. Dr. Ralf, Schenkel & Tobias, Zeimetz. (2022, Feb). Elements of Computer Science: Programming - Assignment 12. Trier University.
27. Dr. Ralf Schenkel & Markus Nilles. (2022, Jan). Assignment 8 for Course Elements of Computer Science - Database Systems. Trier University.
28. Dr. Ralf Schenkel & Markus Nilles. (2022, Jan). Assignment 10 for Course Elements of Computer Science - Database Systems. Trier University.
29. Merge Sort using recursion. (n.d.). MyCareerwise. <https://mycareerwise.com/programming/category/sorting/merge-sort-using-recursion>

**Appendix**

1. [Easy Test Cases](https://github.com/A-B-H-I-J-I-T/ML4NLU-Stress-Testing/blob/e19cd250c67347c64220d39ed6ab32a58844fa86/Final%20Test%20Cases/EasyTestCase.xlsx):



1. [Intermediate Test Cases](https://github.com/A-B-H-I-J-I-T/ML4NLU-Stress-Testing/blob/e19cd250c67347c64220d39ed6ab32a58844fa86/Final%20Test%20Cases/IntermediateTestCases.xlsx):



1. [Difficult Test Cases](https://github.com/A-B-H-I-J-I-T/ML4NLU-Stress-Testing/blob/e19cd250c67347c64220d39ed6ab32a58844fa86/Final%20Test%20Cases/DifficultTest%20Cases.xlsx):



1. Chat GPT 3.5 responses link:
2. Easy: <https://chat.openai.com/share/8409543b-91b9-4e43-b1e8-19ca88f832bc>
3. Intermediate: <https://chat.openai.com/share/681481df-a748-433b-966e-626b6dc7109f>
4. Difficult: <https://chat.openai.com/share/40c99d80-fa78-4761-b6d3-1d03977f12b4>
5. Google Bard responses link:
6. Easy: <https://g.co/bard/share/d2fdc53c44dc>
7. Intermediate: <https://g.co/bard/share/824412faa6dd>
8. Difficult: <https://g.co/bard/share/15de046210e0>
9. Microsoft BingAI responses: I am not able to share the chats for this one because it is not allowed to share whole chat response only the individual question responses are allowed to share. There will be a long list of links. I can provide you with my credentials or screen share if you want to see all the responses.
10. You.com responses link:
11. Easy: <https://you.com/search?q=You.com+Easy+1&cid=c1_733ef7cf-b7d8-4cf5-95ec-5453ed631d93&tbm=youchat>
12. Intermediate: <https://you.com/search?q=Intermediate+1&cid=c1_6ff06d3c-28d3-4daf-989e-e763b89581cf&tbm=youchat>
13. Difficult: <https://you.com/search?q=Diffi+1&cid=c1_582da9b1-1340-40f3-823a-395c442563ae&tbm=youchat>
14. Performance visualization code link:

<https://github.com/A-B-H-I-J-I-T/ML4NLU-Stress-Testing/blob/dbf010867298a07b641bc85a156ed07ee139c3af/Code/Performance%20Visualization.ipynb>