VOL. 10. NO. 2 NOVEMBER 2024
P-ISSN: 2685-8223 | E-ISSN: 2527-4864
DOI: 10.33480 /jitk.v10i2.5053

JITK (JURNAL ILMU PENGETAHUAN DAN TEKNOLOGI KOMPUTER)

AUTOMATION OF THE BERT AND RESNET50 MODEL INFERENCE CONFIGURATION ANALYSIS PROCESS

Medi Noviana[1]; Sunny Arief Sudiro[2*]

Master of Electrical Engineering[1]
Universitas Gunadarma, Indonesia[1]
www.gunadarma.ac.id[1]
medinoviana.mn@gmail.com[1]

Master of Electrical Engineering[2]
STMIK Jakarta STI&K, Jakarta, Indonesia[2]
www.jak-stik.ac.id[2]
sunny@staff.jak-stik.ac.id[2*]

(*) Corresponding Author
(Responsible for the Quality of Paper Content)

Abstract—*Inference is the process of using models to make predictions on new data, performance is measured based on throughput, latency, GPU memory usage, and GPU power usage. The models used are BERT and ResNet50. The right configuration can be used to maximise inference. Configuration analysis needs to be done to find out which configuration is right for model inference. The main challenge in the analysis process lies in its inherent time-intensive nature and inherent complexity, making it a task that is not simple. The analysis needs to be made easier by building an automation programme. The automation programme analyses the BERT model inference configuration by dividing 10 configurations namely bert-large_config_0 to bert-large_config_9, the result is that the right configuration is bert-large_config_2 resulting in a throughput of 12.8 infer/sec with a latency of 618 ms. While the ResNet50 model is divided into 5 configurations, namely resnet50_config_0 to resnet50_config_4, the result is that the right configuration is resnet50_config_1 which produces a throughput of 120.6 infer/sec with a latency of 60.9 ms. The automation programme has the benefit of facilitating the process of analysing the inference configuration.*

Keywords: *artificial intelligence, BERT, configuration, inference, ResNet50*

Intisari—*Inferensi adalah proses penggunaan model untuk membuat prediksi terhadap data baru, performanya diukur berdasarkan throughput, latency, penggunaan memori GPU, dan penggunaan daya GPU. Model yang digunakan adalah BERT dan ResNet50. Konfigurasi tepat bisa digunakan untuk memaksimalkan inferensi. Analisis konfigurasi perlu dilakukan guna mengetahui konfigurasi mana yang tepat untuk inferensi model. Tantangan utama dalam proses analisis terletak pada sifat bawaannya yang padat waktu dan kompleksitas yang melekat, menjadikannya tugas yang tidak sederhana. Analisis tersebut perlu dipermudah dengan cara membangun program otomatisasi. Program otomatisasi melakukan analisis konfigurasi inferensi model BERT dengan membagi 10 konfigurasi yaitu bert-large_config_0 hingga bert-large_config_9, hasilnya konfigurasi yang tepat adalah bert-large_config_2 menghasilkan throughput sebesar 12,8 infer/sec dengan latency 618 ms. Sedangkan pada model ResNet50 dibagi 5 konfigurasi yaitu resnet50_config_0 hingga resnet50_config_4, hasilnya konfigurasi yang tepat adalah resnet50_config_1 menghasilkan throughput 120,6 infer/sec dengan latency 60,9 ms. Program otomatisasi memiliki manfaat yakni memudahkan proses analisis konfigurasi inferensi.*

Kata Kunci: *kecerdasan buatan, BERT, konfigurasi, inferensi, ResNet50*

Accredited Rank 2 (Sinta 2) based on the Decree of the Dirjen Penguatan RisBang Kemenristekdikti No.225/E/KPT/2022, December 07, 2022. Published by LPPM Universitas Nusa Mandiri

VOL. 10. NO. 2 NOVEMBER 2024
P-ISSN: 2685-8223 | E-ISSN: 2527-4864
DOI: 10.33480/jitk.v10i2.5053

INTRODUCTION

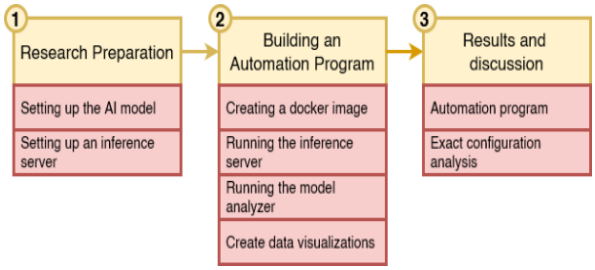
Artificial Intelligence (AI) is an information and communication technology that has emerged in the last decade. The use of AI is not limited to the telecommunications industry, but also extends to the government sectors [1], banking, manufacturing, agriculture, and even Food Industry [2]. Artificial intelligence (AI) models are created using programs or algorithms that are trained with specific data to make decisions without human intervention. This definition highlights the fact that AI models are designed to operate independently of human decision-making processes [3]. Inference is the process of using an AI model to make predictions or decisions based on new data [4]. The BERT and ResNet50 models are among the AI models used to detect fake news. They encode sentences in news content using the BERT model and represent images in the news content using the ResNet50 model [5]. The BERT and ResNet50 models are appropriate research materials because they accept different types of input data. The BERT model takes text data as input [6], while the ResNet50 model takes image data as input [7].

To support the success of artificial intelligence, basic infrastructure in the form of high-specification computers or servers capable of carrying out tasks quickly is required [8]. Such infrastructure may impact the performance of the inference process [9]. The performance measures are based on the parameters throughput (infer/sec) which is the total number of requests completed during BERT or ResNet50 model inference and latency (ms) which is the total time from the request received by the server until the response is sent during AI model inference [10]. The inference performance is supported by the amount of memory (MB) and power (W) utilised by the GPU [11]. The performance of inference is constrained by the computing specifications of the computer/server [12]. To maximize performance, it is important to use the appropriate configuration for the inference process on computers/servers with limited computing specifications [13]. NVIDIA is a major provider of GPUs and has developed open-source software called NVIDIA Triton Inference Server to enhance the inference performance of AI models using GPUs [14].

The Triton Inference Server is optimised for performance through proper configuration. To maximize results, an analysis must be conducted to determine the appropriate configuration for BERT and ResNet50 model inference through Triton Inference Server. The analysis should be automated to simplify and expedite the process [15]. The automation program aids researchers in conducting exact configuration analysis in BERT and ResNet50 model inference, facilitating multiple stages with a single run. It produces a graph indicating performance results, aiding in determining the appropriate configuration.

MATERIALS AND METHODS

The program automation research of configuration analysis process in BERT and ResNet50 model inference is carried out in several stages. Figure 1. shows the general stages of the research.



Source: (Research Result, 2023)

Figure 1. Research Stages.

The research stagesfrom Figure 1 carried out are:

1. Research preparation, the research utilized BERT and ResNet50 AI models and the NVIDIA Inference Server for inference, ensuring the necessary research materials were prepared.
2. Building an automation program, an automation program consists of four stages: generating a Docker image, executing an inference server, measuring inference performance, and creating data visualizations based on the results.
3. Result and discussion, the text discusses the testing of an automation program on BERT and ResNet50 models, detailing the creation of the program, its size, successful run characteristics, and execution time.

The first stage is research preparation. The AI model was obtained by exporting files through the Python programming language. The Python file can be freely downloaded from the Internet. The Python file for exporting the BERT model is different from the ResNet50 model. Next, the configuration files for the BERT and ResNet50 models were prepared. The structure of the contents of the configuration file was determined by the NVIDIA Triton Inference Server, so that it only needs to be adapted to the model used and the required circumstances. The name of the configuration file that NVIDIA Triton Inference Server can read is config.pbtxt. It is also necessary to create a repository structure to store model files and configuration files. The repository is needed because the NVIDIA Triton Inference Server

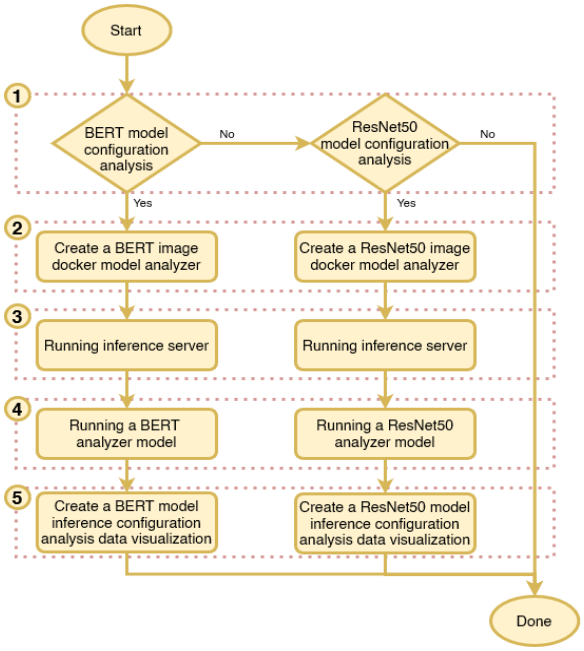
will read the repository to perform inference on the model based on the configuration file. Figure 2. shows a model repository containing model files and configuration files. The repository is read by Triton to run the inference server.

The inference server used is the NVIDIA Triton inference server. The first thing to do is to pull the Docker image of NVIDIA Triton from the official NVIDIA web site. After that, Triton can be run through Docker using the Triton image that was pulled. The Triton inference server that is run will read the model repository that has been created.

```
model_repository/
├── bert-large/
│   ├── config.pbtxt
│   └── 1/
│       └── model.pt
└── resnet50/
    ├── config.pbtxt
    └── 1/
        └── model.savedmodel/
            └── savedmodel.pb
```

Source: (Research Result, 2023)
Figure 2. Model Repository

The next stage is to build an automation program. The Python-based automation program, wrapper.py, is designed for inference configuration analysis on BERT and ResNet50 models, and is available as a .py file. Figure 3. shows the flow of the automation program for configuration analysis of BERT and ResNet50 inference models using Triton Inference Server.



Source: (Research Result, 2023)
Figure 3. Inference Configuration Analysis Automation Program Workflow

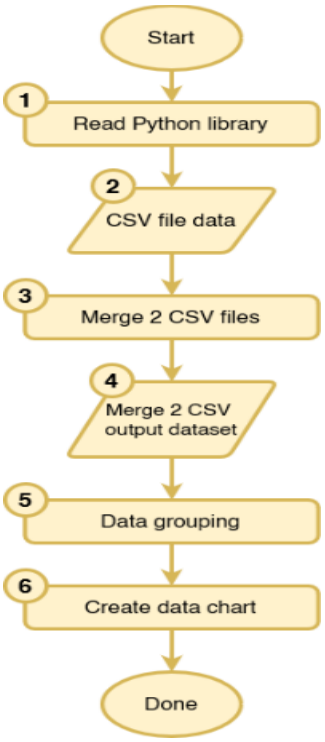
The flow of the program from figure 3 is:

1. Model selection, the automation program can perform inference configuration analysis on one of the models, namely BERT and ResNet50. If you choose the BERT model, the next process will be related to the BERT model, if you choose the ResNet50 model, the next process will be related to the ResNet50 model, if you do not choose from the two models, the process will not do anything or finish immediately.
2. Creating a docker image, The NVIDIA model analyzer is used for configuration analysis on BERT or ResNet50 models. The tool is a Docker image that needs modifications to suit the analysis material, creating a Dockerfile for each model, ensuring automatic execution.
3. Run the inference server, the program will automatically run BERT and ResNet50 model inference because NVIDIA Triton is multi-model, i.e. it can perform inference on several different models. The automation program includes an inference server using Triton Inference Server, which infers BERT and ResNet50 models using Docker. The model analyser is used to profile inference configuration and simulate configurations, measuring inference performance and server-side performance of the GPU. The NVIDIA Geforce GTX 1070Ti GPU is used, and the network uses ethernet LAN.
4. Run the model analyser, The program automatically runs configuration analysis on BERT or ResNet50 model inference using a Docker image. The model analyzer profiles inference configurations, dividing them into multiple configurations for simulated inference. Performance measurement data is stored in CSV files, containing both inference measurement and server-side data.
5. Visualising the analysis data, The automation program visualizes configuration analysis results for BERT or ResNet50 inference models, including throughput, latency, GPU memory usage, and power usage, using linear graphs. This data visualization aids in analyzing the correct configuration for inference performance.

The data visualisation feature was developed separately from the main program using Python, creating two files for BERT model inference data and ResNet50 model inference data. Pandas and Matplotlib libraries were used to process data from CSV files and map it into graphs. Figure 4. shows the workflow of the data visualisation feature built using the Python programming language.

Accredited Rank 2 (Sinta 2) based on the Decree of the Dirjen Penguatan RisBang Kemenristekdikti No.225/E/KPT/2022, December 07, 2022. Published by LPPM Universitas Nusa Mandiri

VOL. 10. NO. 2 NOVEMBER 2024
P-ISSN: 2685-8223 | E-ISSN: 2527-4864
DOI: 10.33480/jitk.v10i2.5053



Source: (Research Result, 2023)
Figure 4. Program Workflow to Create Data Visualisation.

The flow of the data visualisation generation program is:

1. Read Python libraries, calling libraries needs to be done at the beginning of the program because the rest of the program code can only run if the libraries have been defined.
2. CSV file data, CSV files that have been obtained from the results of configuration analysis by the model analyser need to be read using the Pandas library so that data are obtained.
3. Merge 2 CSV files, the two CSV files are merged with the provisions of the merge index using the "Model Config Path" column which contains the names of configurations after the profiling process of the model analyser.
4. Dataset resulting from merging 2 CSV files, the result of the merge process produces one dataset containing data from two CSV files. The merge process needs to be done in order to be able to make the process of making data visualisation in the form of graphs.
5. Data grouping, the dataset still contains units of data so it is necessary to group the data based on the index. As a result, one configuration has a data group containing measurement data.
6. Create data graphs, data visualisation is made in the form of data graphs. The resulting graph is a line graph with the parameters being the measurement results of throughput, latency, the amount of GPU memory usage, and the amount of GPU power usage.

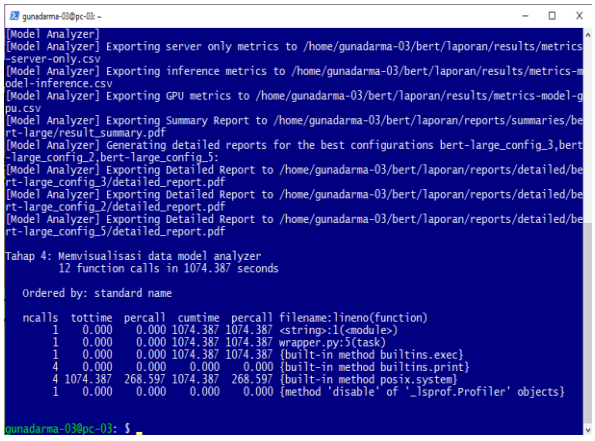
Features that have been created such as creating docker images, running inference servers, running model analysers, and creating data visualisation programs are repackaged in one program using the Python programming language so that all these features can be run automatically with just one run.

RESULTS AND DISCUSSION

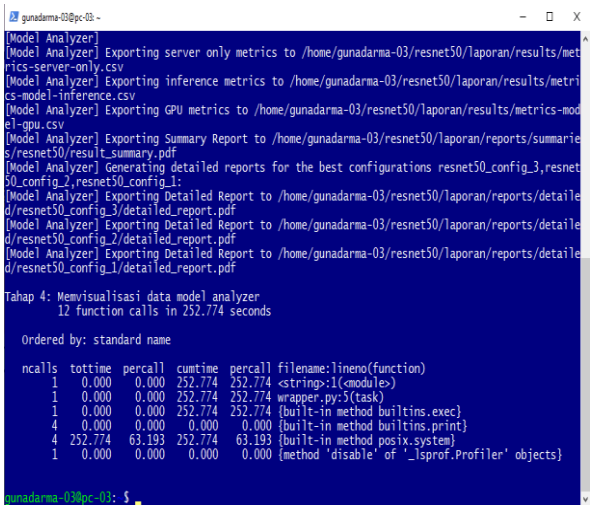
The research developed an automation program for analyzing BERT and ResNet50 model inference configurations, which was discussed in detail, along with the visual analysis of the results.

The automation program includes four features: Docker image creation, inference server operation, model analyzer operation, and data visualization from BERT and ResNet50 model inference configurations. The 2.3 KB file size separates data visualization from the main program, allowing Triton to be used for the process.

Measuring the speed of the running automation program is done by calculating the total time needed to run the functions. The method used to calculate the length of time when the automation program runs is using Python's cProfile library. Figure 5. shows the results of measuring the length of time the automation program runs on the BERT model inference configuration analysis built using the Python programming language. The 12 functions executed with the total time required to run the function is 1074.387 seconds or 17.9 minutes.



Source: (Research Result, 2023)
Figure 5. Total Time Automation Program for Analysis BERT Model Inference Configuration



Source: (Research Result, 2023)

Figure 6. Total Time Automation Program for Analysis Resnet50 Model Inference Configuration

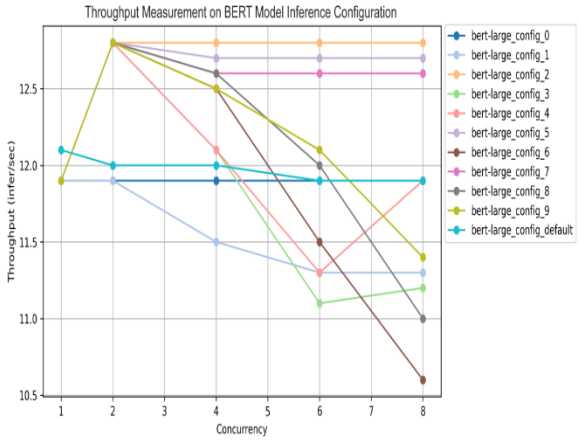
Figure 6. shows the results of measuring the time taken for the automation program to run for the ResNet50 model inference configuration analysis built using the Python programming language. The functions executed are the same as the BERT model, which is 12 with the total time required to run the functions is 252.774 seconds or 4.2 minutes.

The test of the automation program for BERT and ResNet50 model inference configuration analysis is run using NVIDIA Geforce GTX 1070Ti GPU processing. The program will run in 4 stages, namely, creating a docker image, running the inference server, running the model analyser, and creating data visualisation.

1. The first stage, namely the creation of a Docker image, is done in just 0.1 seconds for the BERT model and the ResNet50 model, because there are not many commands in the Dockerfile so that the image build process can be completed very quickly.
2. The second stage is to run the inference server in background process and only display the Docker container ID. The process needs to be run in a background because the Triton Inference Server will continue to be active waiting for input from the client so it needs to end the process first in order to run the next stage.
3. The third stage is to run the model analyser for each model inference configuration. This stage is run using the Docker image that was created in the first stage. This stage profiled the configurations used for BERT and ResNet50 model inference, and simulated the profiled configurations to measure inference performance.
4. The fourth stage is to create data visualisation. The data used is the inference performance measurement data that has been done in the third stage. The resulting visualisation is 4 graphs for each model, namely visualisation of throughput, visualisation of latency, visualisation of GPU memory usage, and visualisation of GPU power usage.

The configuration for BERT model inference is divided into 11 configurations, including the default configuration. The division was done during the profiling process. Each configuration produces different measured values. The difference in value is taken into consideration to choose which configuration is appropriate for BERT model inference.

Figure 7. shows the results of throughput measurements on the BERT model inference configuration.



Source: (Research Result, 2023)

Figure 7. Throughput Measurement Results on BERT Model Inference Configuration

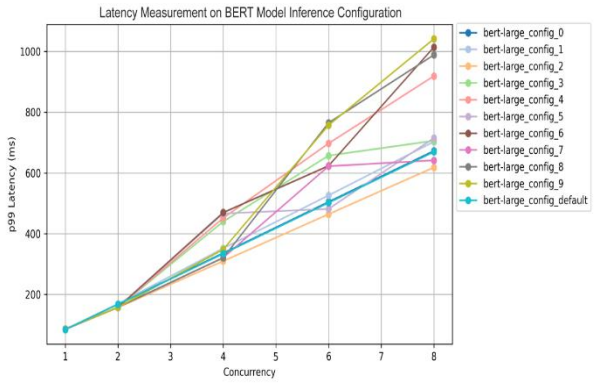
The bert-large_config_2 configuration is the configuration that produces the highest throughput for BERT model inference. The throughput value in bert-large_config_2 at concurrency 2, 4, 6, and 8 is the same or stable at a higher number than other configurations. Concurrency 1 in the bert-large_config_2 configuration is 11.9 infer/sec, and concurrency 2 to concurrency 8 is 12.8 infer/sec.

The bert-large_config_1 configuration is the configuration that produces the lowest throughput value. Concurrency 1 and concurrency 2 are worth 11.9 infer/sec, concurrency 4 is worth 11.5 infer/sec, concurrency 6 and concurrency 8 are worth 11.3 infer/sec.

Figure 8. shows the results of the latency measurement against concurrency in the BERT model inference configuration.

Accredited Rank 2 (Sinta 2) based on the Decree of the Dirjen Penguatan RisBang Kemenristekdikti No.225/E/KPT/2022, December 07, 2022. Published by LPPM Universitas Nusa Mandiri

VOL. 10. NO. 2 NOVEMBER 2024
P-ISSN: 2685-8223 | E-ISSN: 2527-4864
DOI: 10.33480/jitk.v10i2.5053



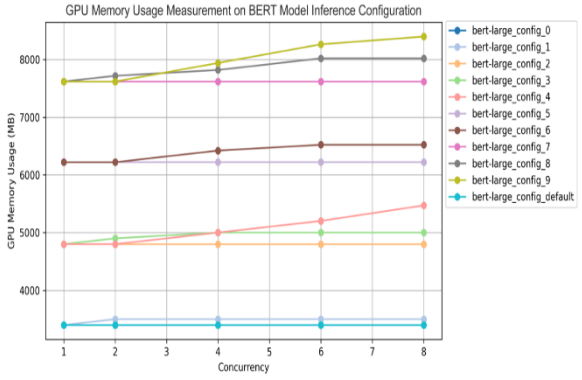
Source: (Research Result, 2023)

Figure 8. Latency Measurement Results on BERT Model Inference Configuration

The configuration that produces the lowest latency for BERT model inference is the bert-large_config_2 configuration. Concurrency 1 is 85 ms, concurrency 2 is 156 ms, concurrency 4 is 309 ms, concurrency 6 is 464 ms, and concurrency 8 is 618 ms.

The configuration that produces the highest latency for BERT model inference is the bert-large_config_9 configuration. The latency value in the bert-large_config_9 configuration is concurrency 1 worth 85 ms, concurrency 2 worth 157 ms, concurrency 4 worth 348 ms, concurrency 6 worth 757 ms, and concurrency 8 worth 1041 ms.

Figure 9. shows the results of measuring GPU memory usage against concurrency in the BERT model inference configuration.



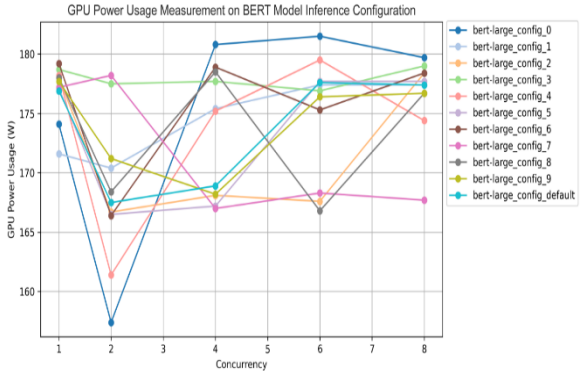
Source: (Research Result, 2023)

Figure 9. Measurement Results of GPU Memory Usage on BERT Model Inference Configuration

The configuration that results in the lowest GPU memory usage is the configuration bert-large_config_default and bert-large_config_0. The value of the GPU storage usage in the configurations bert-large_config_default and bert-large_config_0 from concurrency 1 to concurrency 8 is 3399 MB.

The configuration that produces the highest GPU memory usage is the bert-large_config_9 configuration. The value of GPU memory usage in the bert-large_config_9 configuration is concurrency 1 and concurrency 2 worth 7616 MB, concurrency 4 worth 7939 MB, concurrency 6 worth 8262 MB, and concurrency 8 worth 8397 MB.

Figure 10. shows the results of measuring GPU power usage against concurrency in the BERT model inference configuration.

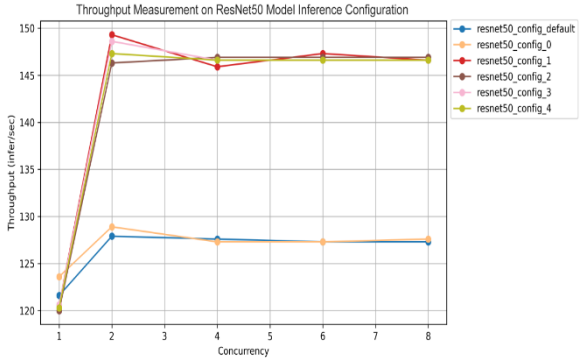


Source: (Research Result, 2023)

Figure 10. GPU Power Usage Measurement Results on BERT Model Inference Configuration

The configuration that produces the lowest GPU power usage is bert-large_config_2. The value of GPU power usage in the bert-large_config_2 configuration is concurrency 1 worth 178 W, concurrency 2 worth 166 W, concurrency 4 worth 168 W, concurrency 6 worth 167 W, and concurrency 8 worth 178 W.

The bert-large_config_0 configuration results in high GPU power usage. The value of GPU power usage in the bert-large_config_0 configuration is concurrency 1 worth 174 W, concurrency 2 worth 157 W, concurrency 4 worth 180 W, concurrency 6 worth 181 W, and concurrency 8 worth 179 W.



Source: (Research Result, 2023)

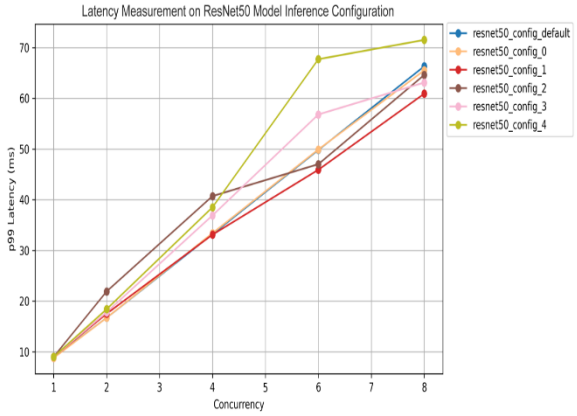
Figure 11. Throughput Measurement Results on ResNet50 Model Inference Configuration

The configuration for ResNet50 model inference is divided into 6 configurations, including the default configuration. The division was done during the profiling process in stage 3 of the automation program. Each configuration was simulated to measure throughput and other parameters. Each configuration produces different measured values. The difference in values is taken into consideration to choose which configuration is appropriate for ResNet50 model inference.

Figure 11. shows the throughput measurement results for the ResNet50 model inference configuration. The configuration that produces the highest throughput is the resnet50_config_1 configuration. The throughput value in the resnet50_config_1 configuration is concurrency 1 worth 120.6 infer/sec, concurrency 2 worth 149.3 infer/sec, concurrency 4 worth 145.9 infer/sec, concurrency 6 worth 147.3 infer/sec, and concurrency 8 worth 146.6 infer/sec.

The configuration that produces the lowest throughput is the resnet50_config_default configuration. The throughput value in the resnet50_config_default configuration is concurrency 1 worth 121.6 infer/sec, concurrency 2 worth 127.9 infer/sec, concurrency 4 worth 127.6 infer/sec, concurrency 6 and concurrency 8 worth 127.3 infer/sec.

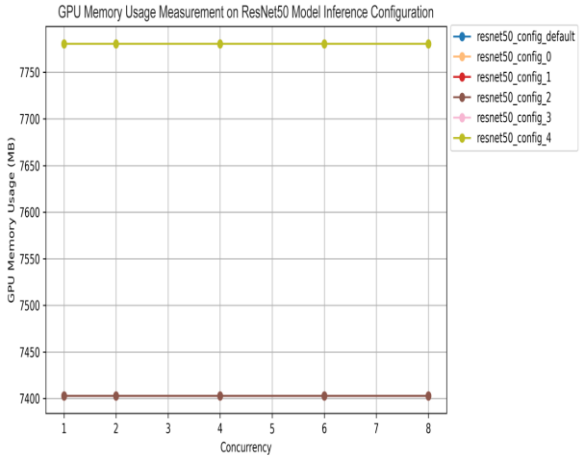
Figure 12. shows the results of the latency measurement against concurrency in the ResNet50 model inference configuration.



Source: (Research Result, 2023)

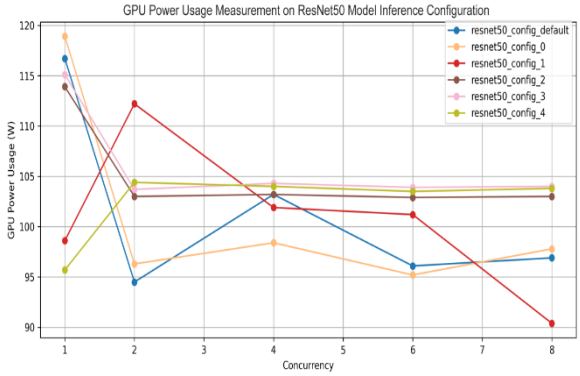
Figure 12. Latency Measurement Results on ResNet50 Model Inference Configuration

The configuration that produces the lowest latency for ResNet50 model inference is the resnet50_config_1 configuration. The latency values in the resnet50_config_1 configuration is concurrency 1 worth 9 ms, concurrency 2 worth 17.5 ms, concurrency 4 worth 33.1 ms, concurrency 6 worth 45.9 ms, and concurrency 8 worth 60.9 ms. The configuration that produces the highest latency for ResNet50 model inference is the resnet50_config_4 configuration. The latency values in the resnet50_config_4 configuration is concurrency 1 worth 9 ms, concurrency 2 worth 18.4 ms, concurrency 4 worth 38.5 ms, concurrency 6 worth 67.7 ms, and concurrency 8 worth 71.5 ms. The results of measuring GPU memory usage against concurrency in the ResNet50 model inference configuration was shown in Figure 13. Perpendicular graph on the Concurrency axis, meaning that all configurations use GPU memory with the same value from concurrency 1 to concurrency 8. The configurations that produce the lowest GPU memory usage are the resnet50_config_0, resnet50_config_1, resnet50_config_2, and resnet50_config_default configurations. The GPU memory usage in these configurations is 7402.9 MB. The resnet50_config_3 and resnet50_config_4 configurations produce the highest GPU memory usage for ResNet50 model inference at 7780.4 MB.



Source: (Research Result, 2023)

Figure 13. GPU Memory Usage Measurement Results on ResNet50 Model Inference Configuration



Source: (Research Result, 2023)

Figure 14. GPU Power Usage Measurement Results on ResNet50 Model Inference Configuration

Accredited Rank 2 (Sinta 2) based on the Decree of the Dirjen Penguatan RisBang Kemenristekdikti No.225/E/KPT/2022, December 07, 2022. Published by LPPM Universitas Nusa Mandiri

VOL. 10. NO. 2 NOVEMBER 2024
P-ISSN: 2685-8223 | E-ISSN: 2527-4864
DOI: 10.33480/jitk.v10i2.5053

Figure 14. shows the results of measuring GPU power usage against concurrency in the ResNet50 model inference configuration. The configuration that produces the lowest GPU power usage is the resnet50_config_0 configuration. The value of GPU power usage in the resnet50_config_0 configuration is concurrency 1 worth 118.9 W, concurrency 2 worth 96.3 W, concurrency 4 worth 98.4 W, concurrency 6 worth 95.2 W, and concurrency 8 worth 97.8 W.

The resnet50_config_3 configuration is a configuration that produces the highest GPU power usage. The value of GPU power usage in the resnet50_config_3 configuration is concurrency 1 worth 115.1 W, concurrency 2 worth 103.7 W, concurrency 4 worth 104.3 W, concurrency 6 worth 103.9 W, and concurrency 8 worth 104 W.

CONCLUSION

The automation program was built using the Python programming language. The form of the automation program is a file with the .py extension, the file name is wrapper.py, and the file size is 1.58 KB. The features of the automation program are modifying the docker image, running the inference server, running the model analyser, and creating visualizations of configuration analysis data for BERT and ResNet50 model inference. The total time required to run the automation program on the BERT model was 17.9 minutes, and the total time required to run the automation program on the ResNet50 model was 4.2 minutes.

The appropriate configuration for BERT model inference is the bert-large_config_2 configuration with a throughput of 12.8 infer/sec, latency of 618.2 ms at the end of concurrency, GPU memory usage of 4800.4 MB, and GPU power usage of 166.7 W. The appropriate configuration for ResNet50 model inference is the resnet50_config_1 configuration with a throughput of 149.3 infer/sec, latency of 60.9 ms at the end of concurrency, GPU memory usage of 7402.9 MB, and GPU power usage of 90.4 W. The research focuses on AI model inference using NVIDIA Triton, using simulation tests on the Triton Inference Server. Parameters include latency, throughput, total GPU memory consumption, and GPU power consumption. Line graphs are displayed, automatic programs are built using Python, and BERT and ResNet50 models are used for analysis.

The research on AI model inference is limited and needs further development to improve program results and configuration analysis. Future considerations include replacing AI models with other models, adding AI models, altering concurrency, batch, or GPU cores in the inference configuration file, and using additional libraries or data visualization software.

REFERENCE

[1] Supriyadi Irawan Endang, & Asih Banyu Dianing. "Implementasi Artificial Inteleligence (AI) di Bidang Administrasi Publik pada Era Revolusi Industri 4.0," Jurnal RASI, vol. 2, no. 2, pp. 12–22. January 2021, doi: doi:10.52496/rasi.v2i2.62,

[2] N. N. Misra et al., "IoT, Big Data, and Artificial Intelligence in Agriculture and Food Industry," *IEEE Internet of Things Journal*, vol. 9, pp. 6305–6324, May 2022, doi: 10.1109/JIOT.2020.2998584.

[3] Nugroho Sasongko, "Strategi Nasional Kecerdasan Artifisial Indonesia Strategi Nasional Kecerdasan Artifisia," Badan Pengkajian dan Penerapan Teknologi, July 2020.

[4] Sergio P. Perez et al., "Training and inference of large language models using 8-bit floating point", available online at: https://arxiv.org/pdf/2309.17224, 2023.

[5] Long Ying, Hui Yu, Jinguang Wang, Yongze Ji, and Shengsheng Qian, "Multi-Level Multi-Modal Cross-Attention Network for Fake News Detection," *IEEE Access*, vol. 9, pp. 132363–132373, 2021, doi: 10.1109/ACCESS.2021.3114093

[6] Subakti Alvin, Murfi Hendri, & Hariadi Nora. The performance of BERT as data representation of text clustering. Journal of Big Data, 9. February 2022, doi:10.1186/s40537-022-00564-9.

[7] Kirandeep, Ramanpreet Kaur, and Vijay Dhir, "Image Recognition using ResNet50," *European Chemical Bulletin*, vol. 12, pp. 7533-7538, July 2023.

[8] Hanqiu Chen, Yahya Alhinai, Yihan Jiang, Eunjee Na, and Cong Hao, "Bottleneck Analysis of Dynamic Graph Neural Network Inference on CPU and GPU," in *2022 IEEE International Symposium on Workload Characterization (IISWC)*, pp. 130-145, November 2022, 10.1109/IISWC55918.2022.00021.

[9] Chunrong Yao et al., "Evaluating and analyzing the energy efficiency of CNN inference on high-performance GPU," *Concurrency and Computation: Practice and Experience*, vol. 33, no. 6, pp. e6067,

October 2020, doi: https://doi.org/10.1002/cpe.6064.

[10] Erqian Tang, Svetlana Minakova, and Todor Stefanov, "Energy-Efficient and High-Throughput CNN Inference on Embedded CPUs-GPUs MPSoCs," in *Embedded Computer Systems: Architectures, Modeling, and Simulation.*: Springer International Publishing, pp. 127–143, 2022, doi: https://doi.org/10.1007/978-3-031-04580-6_9.

[11] Seungbeom Choi, Sunho Lee, Yeonjae Kim, Jongse Park, Youngjin Kwon, Jaehyuk Huh:, "Multi-model Machine Learning Inference Serving with GPU Spatial Partitioning," CoRR abs/2109.01611, 2021.

[12] Ali Jahanshahi, Hadi Zamani Sabzi, Chester Lau, and Daniel Wong, "Gpu-Nest: Characterizing Energy Efficiency of Multi-Gpu Inference Servers," *IEEE Computer Architecture Letters*, vol. 19, pp. 139–142, July 2020, doi: 10.1109/LCA.2020.3023723

[13] Hanif Abdullah Muhammad, & Shafique Muhammad. Cross-Layer Optimizations for Efficient Deep Learning Inference at the Edge. Embedded Machine Learning for Cyber-Physical, IoT, and Edge Computing,". Springer Nature Switzerland, (225–248) doi:10.1007/978-3-031-39932-9_9, October 2023.

[14] Jiacong Fang, Qiong Liu, and Jingzheng Li, "A Deployment Scheme of YOLOv5 with Inference Optimizations Based on the Triton Inference Server," in *2021 IEEE 6th International Conference on Cloud Computing and Big Data Analytics (ICCCBDA)*, April 2021.

[15] Hohman Fred, Wang Chaoqun, Lee Jinmook, Görtler Jochen, Moritz Dominik, Bigham Jeffrey, Zhang Xiaoyi, "Talaria: Interactively Optimizing Machine Learning Models for Efficient Inference. CHI." *In Proceedings of the CHI Conference on Human Factors in Computing Systems*, pp. 1-19, 2024.

Accredited Rank 2 (Sinta 2) based on the Decree of the Dirjen Penguatan RisBang Kemenristekdikti No.225/E/KPT/2022, December 07, 2022. Published by LPPM Universitas Nusa Mandiri