KOMTEKINFO JOURNAL
LPPM Universitas Putra Indonesia YPTK Padang
Jl. Raya Lubuk Begalung, Padang, West Sumatera, Indonesia, Zip Code: 25221
Volume: 12, Issue: 3, Page: 124 - 130, September 30th, 2025, e-ISSN: 2502-8758
Available online at website: https://jkomtekinfo.org/ojs/index.php/komtekinfo

Large Language Model Method as a Translator Indonesian Into SQL Language

Candra Adi Putra[1,✉], Syafri Arlis[2], Gunadi Widi Nurcahyo[3]
[1]UIN Syekh Ali Hasan Ahmad Addary Padangsidimpuan, 22730, Indonesia
[2] Universitas Putra Indonesia YPTK, Padang, 25221, Indonesia
[3] Universitas Putra Indonesia YPTK, Padang, 25221, Indonesia

candraadiputra@gmail.com

Abstract

The development of information technology has encouraged the massive implementation of information systems and web-based applications in various sectors, including in the academic environment. However, one of the challenges that are still often faced is the difficulty in extracting or mining information from databases flexibly without having to create additional report modules or write SQL code manually. This problem becomes an obstacle for non-technical users, such as administrative staff or lecturers, who need certain data quickly from academic information systems. In this paper, it is intended to convert Indonesian commands into SQL queries automatically, without the need to add additional programming code. Along with advances in Natural Language Processing (NLP) and Machine Learning technology with the Large Language Model (LLM) method, there is now a new approach that allows users to interact with databases only through commands in natural language. The case study was conducted on the Academic Information System of UIN Padangsidimpuan using a dataset of 1,500 student data. The focus of the research is on the type of Data Query Language (DQL) query in Indonesian form, which is then translated by the model into a SQL command to obtain the desired data. The results showed that this approach was able to achieve results with a Rouge1 conversion precision rate from 0.03 to 0.89. This shows that the integration of LLM technology in academic information systems has great potential in improving data accessibility, operational efficiency, and supporting data-driven decision-making faster and more intuitively, especially for users who do not have a technical background.

Keywords: *NLP, LLM, Indonesian, Information Systems, Python*

1. Introduction

Web-based Information Systems that are generally used by PHP and MySQL technologies are currently used by almost all organizations related to their information system services. Information system services are from just the websites of Organizations, Schools, Universities, Govenerments 1][2], E-Commerce and hotel services, Electronic-based Government Systems and other web-based services [3]. The biggest problem with such a system is that it focuses more on the data input process than on processing data. The general process that is done is that the programmer creates a report menu to extract data from the data that has already been stacked. This is certainly not flexible because every time we request a custom report or just an executive summary, we need to create custom code to display the page.

One of the problems that universities often face is the frequent asking questions about academic information data on campus [4]. The service is already available in the academic information system on the Dashboard page. However, along with the development of new technologies such as Large Languge Models, Chatbots and Natural Language to SQL Technology, there needs to be a service or system that can directly extract data directly from the database. In this case, users can ask questions directly with the database human language without the need to learn SQL query syntax.

The rapid development of information technology has brought major transformations in data management and utilization, especially in relational database systems. In the academic and industrial worlds, the need to access and extract information from databases is becoming increasingly important. However, not all users have the technical ability to write correct and efficient Structured Query Language (SQL) queries. Therefore, the need for a system that can translate natural languages, especially Indonesian, into SQL form is very relevant in the context of inclusive data utilization.

Submitted: 25-07-2025 | Revised: 15-08-2025 | Accepted: 10-09-2025 | Published: 30-09-2025

Research in the field of Machine Learning [5], dan Natural Language Processing (NLP) has shown significant advances in the translation of natural language to SQL, otherwise known as the Text-to-SQL (NL2SQL) approach. One of the initial approaches used was the Statistical Translation Machine which was able to achieve an accuracy of 70% in the domain of student databases. The rule-based approach [6] has also succeeded in converting Indonesian commands into SQL commands with 72.5% accuracy.

As artificial intelligence technology develops, particularly through Deep Learning approaches, various models such as LSTM [7] and gradual parsing frameworks [8] have begun to be applied to improve the accuracy and complexity of language understanding. However, the limitations of handling complex queries and the need for large training data are challenges.

In recent years, Large Language Models (LLMs) such as GPT, T5, and BERT variants have revolutionized the Text-to-SQL approach [9]. Models such as PURPLE [10] manage to achieve an accuracy of up to 87.8% in the popular NL2SQL benchmark. Other approaches, such as nanoT5[11] focus on training efficiency with limited resources such as Personal PCs, making this technology more accessible.

Other studies have also shown the successful application of LLM to non-English languages, such as TUR2SQL for Turkish [12] or Arabic to SQL [13] However, in-depth studies related to Indonesian as an input to NL2SQL are still very limited. This is a research opportunity to explore LLM's ability to understand the unique linguistic structure of the Indonesian language, as well as translate it into the appropriate form of SQL.

This study aims to develop and evaluate the effectiveness of the Large Language Model method in translating Indonesian into SQL query form. By drawing on the latest approaches such as prompt engineering, schema linking, and Retrieval-Augmented Generation (RAG)[14], this research is expected to be able to bridge the gap between ordinary users and database systems.

Leveraging the combination of the power of LLMs such as Google T5[15], Flan-T5, and lighter but efficient models, this study not only aims to improve translation accuracy, but also considers resource efficiency as well as implementation scalability in a variety of contexts, including academic information systems. For LLM Itself it is widely used in foreign languages such as English to SQL or Chinese to SQL [16]. For Indonesian itself, there has been no research related to LLM technology to translate Indonesian to SQL, but there have been studies that have conducted NLP results in Indonesian [17].

Based on the explanation of the previous research, this study aims to apply the LLM model to translate Indonesian to SQL, design an LLM Model with the FLANT-T5 architecture [18] and the Huggingface Transformers library [19][20] to directly communicate with the MySQL Database and test the model in terms of Accuracy compared to the basic model.

2. Methodology

This study uses the Large Language Model with the FLAN T5 Basic Model which is a development of the T5 Model developed by Google. This model has a decoder and encoder as shown in Figure 1.

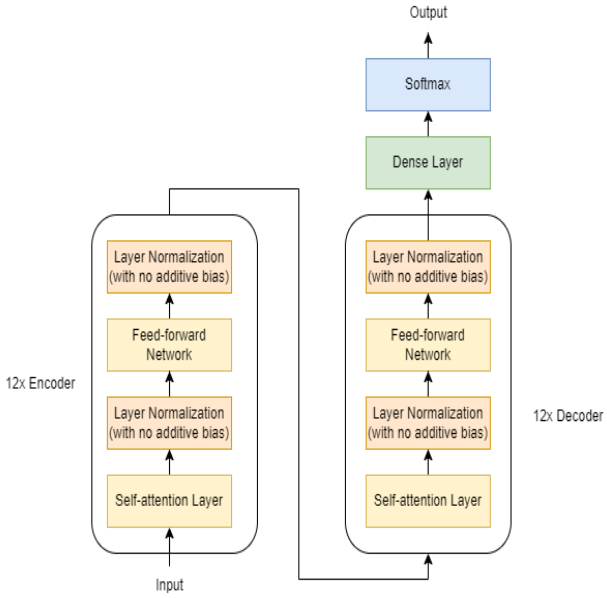

Figure 1. Architectural Diagram of FLAN T5

Flan-T5 is a development of Google's T5 model that has been specially adapted to understand and execute various text-based commands. The model is trained using instructional datasets at scale so that it has good ability to handle different types of natural language processing tasks. In the context of Indonesian to SQL translation, Flan-T5 is used to convert natural language queries into SQL commands that can be executed on a database. The entire process, from input to output, is formatted as text, so the model is very flexible in handling a wide variety of user queries.

The strength of the Flan-T5 lies in its ability to understand the context of commands in Indonesian, especially when applied to specific domains such as academic information systems. Using datasets containing questions related to academic data—such as student data, courses, grades, and lecture schedules, the model is trained to be able to generate SQL commands that match the structure and schema of the academic database. This allows users to access information from the academic system simply by asking questions in everyday language, without the need to master SQL syntax technically.

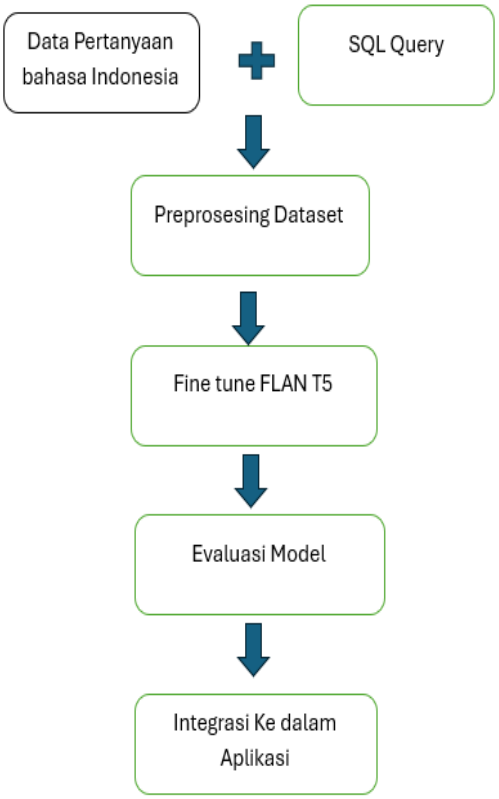

Figure 2. Indonesian to SQL LLM System Diagram

The training process of the Google Flan-T5-based Text-to-SQL model began with the collection stage of two main components, namely data in the form of questions in Indonesian and *ground truth* pairs that represent the appropriate SQL commands. The two types of data are then combined and processed through the pre-processing stage to ensure the quality, uniformity, and readiness of the data in support of the model training process. This pre-processing stage includes normalizing the text, removing invalid data, and rearranging the data into a format that can be understood by the machine learning model.

After the pre-processing process is complete, the data proceeds to the advanced *fine-tuning* stage on the Flan-T5 model. The main objective of this stage is to adapt the model to be able to understand the instructions in Indonesian and generate SQL commands that correspond to the semantic context of the given question. The Flan-T5 model was selected based on its proven ability to follow instructions and complete various natural *language processing tasks*.

Next, an evaluation process was carried out on the performance of the model to assess the extent of the accuracy of converting text into SQL command forms. Evaluation is carried out using metrics such as *exact match accuracy* and *execution accuracy* to ensure that the model output results are not only syntactically correct but also provide appropriate results when executed on the database.

When the model has reached an optimal level of performance, it is implemented in the form of an application programming interface (API) service that is able to receive queries in Indonesian and return the conversion results in the form of SQL commands. This API service is then integrated with the database system, allowing users to query directly without requiring knowledge of SQL syntax. This stage allows the practical application of the model in various information systems, such as academic information systems or other database information systems. To get the results of the Indonesian to SQL model training, it was carried out on Google Collab Pro with an A100 GPU with a dataset of 8000 data with hasl as a train/loss as 0.0208.

3. Results of the Discussion

In this discussion, 3 things will be discussed, namely the arguments of the training process and the evaluation metrics of the training results. The argument will explain the training parameters. The main parameters are epoch and parameters. In this training we only use 2 epochs with 14,000 steps.

3.1 Arguments Training

In the machine learning model training process, the term *epoch* refers to a single full cycle in which the entire training data is used once to update the weight of the model. If the number of *epochs* is set to two, it means that the entire training dataset will be passed to the model twice full. Each *epoch* allows the model to learn data patterns repeatedly, thus hopefully improving the model's generalization ability to new data.

Meanwhile, *the step* is a one-time *forward pass* and *backward pass* process for one *batch* of data. The number of *steps* in an *epoch* can be calculated by dividing the total amount of data by the batch *size*, then multiplying the result by the number of *epochs*. For example, if the *batch size* is 32 and the dataset has 224,000 sample data, then the number of *steps* per *epoch* is 224,000 ÷ 32 = 7,000 *steps*. With training for two *epochs*, the total weight update steps carried out were 7,000 × 2 = 14,000 *steps*.

The model training process begins with the preparation stage of datasets that can come from various sources such as CSV files, JSON, and query results from SQL databases. The dataset first goes through *the preprocessing* and tokenization stages, so that each text is converted into a numerical representation that can be processed by the model. The tokenization results are

then separated into training data and validation data to ensure that the model can be evaluated objectively during the training process.

3.2 Training Process

The initial process begins with collecting Indonesian text data pairs and their corresponding SQL pairs. Next, the technologization process is carried out, followed by training. The training stages are shown in Figure 3.

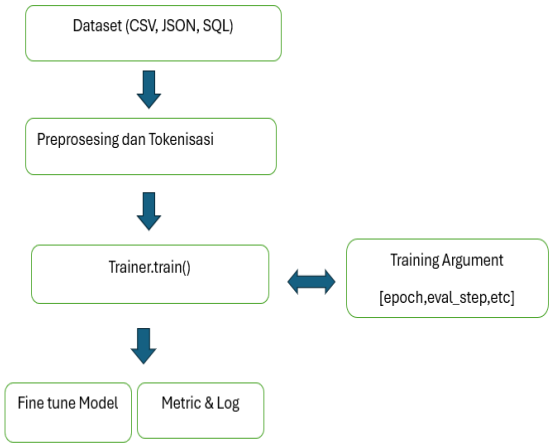

Figure 3. Model Training Process

The next stage is to configure the training parameters using the TrainingArguments object. These parameters include setting a *learning rate* of 5×10^{-3}, the number of *epochs* twice full rounds of all data, the *training* and evaluation batch size of 16 each, and the application of *a weight decay* of 0.01 to reduce the risk of *overfitting*. In addition, the training process is designed to log every 50 steps (*logging_steps=50*) and evaluate every 500 steps (*eval_steps=500*), so that the model's performance progress can be monitored periodically. These parameters, along with the model to be trained, are then combined in the Trainer object that is responsible for carrying out the training and evaluation process.

The training begins with the invocation of the trainer. train() method, which runs an iterative cycle that includes *a forward pass* to calculate the prediction, a loss value calculation, *a backward pass* to calculate the gradient, and an update of the model weight based on the gradient. During training, the model is periodically evaluated against validation data to measure its ability to generalize. Training results, including *model checkpoints*, evaluation metrics such as *loss* and ROUGE values, and log files, are automatically stored in an output directory named after the training *timestamp*, e.g../sql-training-<timestamp>. This process ensures the reproducibility of experiments and facilitates the analysis of training results at a later stage.

3.3 Metric Training Results

A training loss diagram is a tool to visualize model learning. Training loss visualizes how well the model fits the training data. Figure 4 Show the training loss Decreases smoothly indicate that model learning patterns that generalize well.

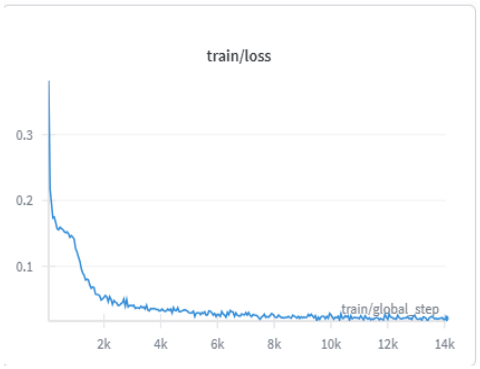

Figure 4. Diagram/loss diagram during training

A train/loss *value* of **0.0208** indicates that the trained model has a very low rate of prediction errors against the training data. This figure indicates that the model has been able to study the patterns and relationships between variables in the dataset well, so that the difference between the prediction results and the actual value is very small. In the context of model performance evaluation, a low loss value generally indicates that the parameter optimization process is running effectively, and the model is in a stable training stage.

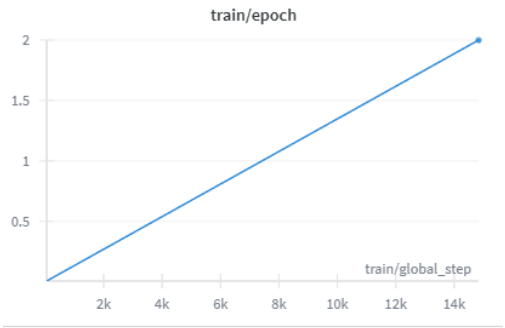

Figure 5. Train/Epoch with an epoch value of 2 and step 14k

Figure 5 shows that the model training process is carried out twice full cycles on the entire dataset (*2 epochs*) with a total of about 14,000 steps of weight updates. One *epoch* means that the entire training data has been used once to update the model parameters, so with two *epochs* it means that the model processes the entire dataset twice. Meanwhile, *the step* represents a one-time

forward pass and *backward pass* process on a single *batch* of data. The total number of *steps* is affected by the size of the dataset, *the batch size*, and the number of *epochs* used.

With a total of 14,000 *steps* in two *epochs*, it can be concluded that each *epoch* consists of about 7,000 *steps*. This indicates that the training process is intensive with many parameter updates, which allows the model to learn data patterns gradually and repeatedly. This information is important for evaluating the duration of the training, the computational needs, and the degree of convergence of the model. Understanding the relationship between *epoch* and *steps* helps researchers determine the optimal training settings so that the model performs best without overfitting or *underfitting*.

The results of the training showed a comparison of performance between the original model and *the fine-tuned* result model using the ROUGE evaluation metric. ROUGE itself is a method to measure the results of model's work. Rough stands for Recall-Oriented Understudy for Gisting Evaluation is a Recall-Based Summary Evaluation. In this training, the rough results can be seen in table 1

Table 1. Data rouge Model before fine tune

Performance	Value
rouge -1	0,0312
rouge -2	0,005
rouge -L	0,0315
rouge -LSum	0,0317

In the original model, the value of ROUGE-1 was 0.0312, ROUGE-2 was 0.005, ROUGE-L was 0.0315, and ROUGE-Lsum was 0.0317. These very low values indicate that the model has not been able to produce outputs that have meaningful similarities with the reference text.

Table 2. Data rouge Model after Finetune

Metric	Weighted Average
rouge -1	0,8968
rouge -2	0,8520
rouge -L	0,8905
rouge -LSum	0,8877

After the *fine-tuning* process, the evaluation values increased significantly: ROUGE-1 reached 0.8968, ROUGE-2 reached 0.8520, ROUGE-L reached 0.8905, and ROUGE-Lsum reached 0.8877. This improvement shows that the *fine-tuned* model can produce outputs that are very similar to the reference text in terms of word choice and sentence structure. This spike in value is an indication that *fine-tuning* is successfully adapting the model to the desired domain and task, so that it performs much more optimally than the initial model.

4. Conclusion

Based on the results of the training, there was a very significant difference between the performance of the original model and the model after *fine-tuning*. The original model obtained only a ROUGE-1 value of 0.0312, ROUGE-2-2 of 0.0050, ROUGE-L of 0.0315, and ROUGE-Lsum of 0.0317, indicating that the output of the model has almost no meaningful similarity to the reference text. This value indicates that the initial model cannot understand or reproduce the language patterns that are in accordance with the task given.

In contrast, the *fine-tuned* results model showed a remarkable spike in performance, with ROUGE-1 at 0.8968, ROUGE-2 at 0.8520, ROUGE-L at 0.8905, and ROUGE-Lsum at 0.8877. This improvement illustrates that after the *fine-tuning process*, the model can produce text that is very similar to the reference text in terms of word selection, word order, and sentence structure. A ROUGE value close to 1.0 indicates a very high degree of conformity between the model results and the reference.

Thus, it can be concluded that the *fine-tuning* process succeeded in significantly improving the model's ability to understand and generate text according to the desired domain or task. The high ROUGE score on *the fine-tuned* model proves that the adaptation of the model to a specific dataset can optimize the prediction quality, so that the model is ready for implementation in real scenarios with an excellent level of accuracy. In this case, the model results can be used in applications for Indonesian to SQL translation in relational database-based applications.

The biggest challenge of using models is the need for large datasets so that the resulting model can be used across domains from different databases if the model is to be used in different database information systems. The model produced in this study only focuses on models for academic information systems databases. If the model wants to be used in other data systems such as Ecommerce or health, then training needs to be redone with the appropriate dataset.

References

1. Setiawan, A., Samsugi, S., & Alita, D. (2023). Rancang Bangun Sistem Informasi Akademik SMK TAMAN SISWA 1 Tanjung Karang BERBASIS WEB. *J. Inform. Dan Rekayasa Perangkat Lunak*, *4*(1), 53-59.

2. Murtia, S., Saputra, R., & Ramadani, N. (2025). Village Information System Design with Mobile Development Life Cycle Approach. *Jurnal KomtekInfo*, 86-93.]

3. Mantu, A. M., Tatuhey, E. L., & Thamrin, R. M. (2024). Rancang Bangun Platform E-commerce berbasis Website pada Media Cell. *Jutisi: Jurnal Ilmiah Teknik Informatika dan Sistem Informasi*, *13*(3).

4. Halwa, E. N., & Marwati, A. (2021). Analisis Sistem Informasi Akademik Universitas Sunan Giri Surabaya Menggunakan Metode Pieces. *Jurnal Ilmiah Manajemen Informasi dan Komunikasi*, *5*(2), 55-66..

5. Ramadhanu, A., Zaky, M. R., Isra, M., Nengsih, N. S. W., & Sularno, S. (2023). Penerapan Machine Learning Untuk Menentukan Tingkat Kepuasan Tamu Hotel Dymens Menggunakan Metode Vader. *Jurnal Teknologi Dan Sistem Informasi Bisnis*, *5*(3), 337-343.

6. El Boujddaini, F., Laguidi, A., & Mejdoub, Y. (2024, May). A survey on text-to-sql parsing: From rule-based foundations to large language models. In *International Conference on Connected Objects and Artificial Intelligence* (pp. 266-272). Cham: Springer Nature Switzerland.[2]

7. Zhang, H. (2024). Application of LSTM-Based Seq2Seq Models in Natural Language to SQL Conversion in Financial Domain. *Science, Technology and Social Development Proceedings Series*, *2*, 10-70088.[3]

8. Shen, R., Sun, G., Shen, H., Li, Y., Jin, L., & Jiang, H. (2023). Spsql: Step-by-step parsing-based framework for text-to-sql generation. *arXiv preprint arXiv:2305.11061*.[4]

9. Xusheng, L., Yeteng, A., Jingxian, L., Huimin, Z., Yumeng, Z., Min, L., ... & Huiqin, L. (2023, January). Research on BERT-based Text2SQL Multi-task Learning. In *2023 IEEE 3rd International Conference on Power, Electronics and Computer Applications (ICPECA)* (pp. 864-868). IEEE.[5]

10. Ren, T., Fan, Y., He, Z., Huang, R., Dai, J., Huang, C., ... & Wang, X. S. (2024, May). Purple: Making a large language model a better sql writer. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)* (pp. 15-28). IEEE.[6]

11. Nawrot, P. (2023). nanot5: A pytorch framework for pre-training and fine-tuning t5-style models with limited resources. *arXiv preprint arXiv:2309.02373*.[7]

12. Kanburoğlu, A. B., & Tek, F. B. (2023, September). TUR2SQL: A cross-domain Turkish dataset for Text-to-SQL. In *2023 8th International Conference on Computer Science and Engineering (UBMK)* (pp. 206-211). IEEE.[8]

13. Heakl, A., Mohamed, Y., & Zaky, A. B. (2024). Araspider: Democratizing arabic-to-sql. *arXiv preprint arXiv:2402.07448*.

14. Zhao, X., Zhou, X., & Li, G. (2024). Chat2data: An interactive data analysis system with rag, vector databases and llms. *Proceedings of the VLDB Endowment*, *17*(12), 4481-4484.[9]

15. Wong, A., Pham, L., Lee, Y., Chan, S., Sadaya, R., Khmelevsky, Y., ... & Ferri, M. (2024, April). Translating Natural Language Queries to SQL Using the T5 Model. In *2024 IEEE International Systems Conference (SysCon)* (pp. 1-7). IEEE.[10]

16. Chen, Y., Huang, S., Zhuan, Z., & Zhou, E. (2021, November). Research on the Technology of Generating Single-Table sql Query Sentences in Chinese Natural Language. In *2021 2nd International Conference on Intelligent Computing and Human-Computer Interaction (ICHCI)* (pp. 48-52). IEEE. [11]

17. Cahyawijaya, S., Winata, G. I., Wilie, B., Vincentio, K., Li, X., Kuncoro, A., ... & Fung, P. (2021). IndoNLG: Benchmark and resources for evaluating Indonesian natural language generation. *arXiv preprint arXiv:2104.08200*. [12]

18. Chung, H. W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., ... & Wei, J. (2024). Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, *25*(70), 1-53.

19. Meena, A., Kaur, P., Singh Bains, S., Bagri, A., & Agrawal, S. (2024, June). Cross-Language Question-Answering System Using Hugging-Face Transformers. In *International Conference on Intelligent Computing and Big Data Analytics* (pp. 316-329). Cham: Springer Nature Switzerland.

20. Srihari, C., Sunagar, S., Kamat, R. K., Raghavendra, K. S., & Meleet, M. (2022, August). Question and answer generation from text using transformers. In *International Symposium on Intelligent Informatics* (pp. 201-210). Singapore: Springer Nature Singapore.

Biographies of Authors

	**Candra Adi Putra** Candra Adi Putra is an educational staff member at UIN Syekh Ali Hasan Ahmad Addary Padangsidimpuan who currently serves as an Information Systems Analyst. He holds a bachelor's degree from STMIK AKAKOM Yogyakarta majoring in Informatics Engineering. He is currently pursuing a master's degree in informatics engineering at Universitas Putra Indonesia YPTK Padang. His interests in Information Technology include Operating Systems, Cloud Computing, Web Programming, and Data Science, as well as Writing.
	**Syafri Arlis** Syafri Arlis was born in Padang on October 23, 1986. His undergraduate study was completed in 2009 at Universitas Putra Indonesia YPTK. He completed his master's degree at Putra Indonesia University, YPTK Padang. He currently serves as a lecture in the Informatics Engineering study program at the Universitas Putra Indonesia YPTK Padang. Teaching history that has been carried out starting from 2011 until now, such as databases and digital image processing. Published research history places more emphasis on the atificial neural network and digital image processing. He can be contacted at email: syafri_arlis@upiyptk.ac.id.
	**Gunadi Widi Nurcahyo** was born in Temanggung, 14 March 1969. He was graduated bachelor's degree in informatics management at Universitas Putra Indonesia YPTK Padang in 1992. He completed his Master and PhD in Computer Science at Universiti Teknologi Malaysia in 2003. Scopus Id is 57200563356. E-mail: gunadiwidi@yahoo.co.id