iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 12 **Issue:** IV **Month of publication:** April 2024

DOI: https://doi.org/10.22214/ijraset.2024.60445

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue IV Apr 2024- Available at www.ijraset.com

Fine-Tuning Language Models for Enhanced Diagram Generation: A Deep Learning Approach

Moksh Mishra[1], Samar Sheikh[2], Tosh Tonpe

Abstract: ***Large Language Models (LLMs) have demonstrated significant prowess in natural language understanding across diverse contexts. However, their performance in specialized do- mains often falls short due to a lack of domain-specific knowledge. In this paper, we present a novel approach named LLM-FD (Fine-tuned Diagrams) designed to enhance LLMs with the ability to generate diagrams, leveraging fine-tuning techniques to imbue the model with domain-specific knowledge related to diagrammatic representation. Our methodology involves training on a dataset of annotated diagrams, enabling the model to synthesize and comprehend complex diagrammatic structures. We evaluate the performance of LLM-FD on a series of diagram synthesis benchmarks, showcasing its proficiency in generating accurate and contextually relevant diagrams. Our findings underscore the potential of fine-tuning LLMs for diagrammatic tasks and high- light the significance of domain-specific knowledge incorporation in enhancing model capabilities. The implementation code and pretrained models are openly available to foster further research and application development in diagram synthesis tasks.***

Index Terms: ***Large Language Models (LLMs), Fine-tuning, Mermaid Diagrams, Domain-specific Knowledge, Model En- hancement***

I. INTRODUCTION

In recent years, Transformer models have made significant advancements in text understanding and generation. Open source frameworks like Hugging Face's Transformers library have played a crucial role in democratizing access to these powerful models. For example, BERT and GPT, two popular transformer architectures, have been widely used for vari-ous tasks such as language translation and text summariza- tion. These models have shown remarkable capabilities in processing and generating text, showcasing the potential of Transformers in improving how we interact with language. By leveraging pre-trained transformer models and fine-tuning them with domain-specific datasets, researchers aim to en- hance the capabilities of these models further. Fine-tuninginvolves adjusting the parameters of the model to specializeit for a particular task, such as generating Mermaid syntax from natural language prompts. This approach allows for the customization of transformer models to cater to specific needs, expanding their utility beyond traditional text-basedapplications. Transfer learning, a technique where a model trained on one task is adapted for another related task, plays a crucial role in this research. By fine-tuning pre-trained modelswith datasets containing text-to-Mermaid syntax mappings,researchers can effectively teach the model to understand and generate diagrams based on textual inputs. This process not only enhances the model's performance but also bridges the gap between natural language understanding and diagram generation, offering a seamless way to communicate complex concepts visually. Real-world applications of this research can revolutionize how technical professionals create visual repre- sentations from textual descriptions. Overall, the integrationof transformer models with diagram generation tasks holds immense potential in simplifying and enhancing the way we communicate complex ideas. By combining the strengths of natural language understanding with diagram visualization, re-searchers are paving the way for a more intuitive and efficient approach to creating visual content from textual inputs.

II. LITERATURE REVIEW

In recent years, the field of Artificial Intelligence haswitnessed a surge in research focused on harnessing pre-trained language models (LLMs) for diverse applications. This literature review aims to synthesize and analyze the current body of knowledge surrounding language model adaptationand its implications for specialized tasks, such as diagramsynthesis. Our main aim and focus in this research is: Language Model Fine-Tuning: An overview of method- ologies and approaches for fine-tuning pre-trained languagemodels to adapt them to specific tasks and domains.

Diagram Generation: Existing research on generating struc- tured diagrams and flowcharts from natural language descrip- tions, highlighting challenges and advancements in this area

Comparative Analysis: A comparative analysis of task- specific language models versus generic LLMs, assessing performance metrics, scalability, and applicability in real- world scenarios.

A. Fine tuning LLMs

[4] In our research, leveraging fine-tuning of Large Lan- guage Models (LLMs) like gemma-7b offers a strategic advan-tage. By customizing pre-trained models through fine-tuning, we can optimize performance for specific NLP tasks critical toour study, such as sentiment analysis or text classification. This approach significantly reduces the overhead associated with developing custom models from scratch, enabling efficient adaptation of LLMs to our domain-specific requirements. Fine-tuning mitigates common challenges like overfitting and underfitting, ensuring robust model performance across diverse datasets relevant to our research objectives.

The field of natural language processing (NLP) has witnessed rapid evolution, driven by the proliferation of transformer- based models, particularly large language models (LLMs). A recent survey by [5] provides a comprehensive analysis of text classification techniques, focusing on transformer-based approaches across diverse benchmarks in NLP applications. The study expands the scope of text classification beyond unimodal inputs to encompass multimodal data integration, incorporating text, numeric, and columnar data.

The survey evaluates transformer models' performance across 358 datasets and 20 applications, challenging assumptions of universal superiority and highlighting nuances related to accu- racy, cost, and safety. Notably, the paper underscores emerging challenges such as computational requirements, model size, biases, and ethical implications inherent in LLMs for textclassification tasks.

This review emphasizes the necessity of nuanced model un- derstanding and holistic deployment strategies for transformer-based models in real-world NLP applications. It provides insights into future research directions, including multimodal classification, hierarchical modeling, and addressing legal and ethical concerns associated with LLMs in text classification. Overall, this survey serves as a foundation for advancing transformer-based text classification and navigating the evolv- ing landscape of NLP applications.

Transfer learning via fine-tuning has emerged as a solution, allowing the adaptation of pre-trained models to specific tasks.However, selecting fine-tunable layers remains a critical issue. To address this, [6] presents the Differential Evolution based Fine-Tuning (DEFT) method for optimal layer selection under specified constraints. This method was evaluated in osteosar- coma identification from medical imaging datasets, demon- strating superior performance compared to conventional train-ing methods and manually selected fine-tuning approaches. DEFT achieved classification accuracies surpassing existing methods by up to 32.75%, showcasing its efficacy in optimiz- ing deep learning model adaptation for medical image analysis.This research underscores the significance of tailored fine- tuning strategies in enhancing model performance for critical medical applications.

B. Related Work and Gaps identified

1) UML Generator – Use Case and Class Diagram Generation from Text Requirements

The research focuses on automating the creation of Uni-fied Modeling Language (UML) diagrams from require-ment texts using Natural Language Processing (NLP). Software engineering practices have evolved, emphasiz-ing object-oriented design patterns, where UML plays a crucial role in modeling user requirements. The UML Generator, consisting of five modules accessible via a web interface, extracts use cases, actors, classes, and attributes from user-provided text. NLP techniques, including text tokenization and part-of-speech tagging, identify key elements like actors and classes. The system utilizes XML rule-based approaches to refine identified elements and employs machine learning models (Weka) to rate use cases and establish associations.

Results demonstrate the UML Generator's evolution, showing its ability to produce customized use case and class diagrams from input scenarios efficiently. The sys- tem's intelligence, introduced through NLP and custom algorithms, enhances accuracy and usability. Certain features like recognizing multiplicities remain for future development, highlighting novel approaches in UML diagram automation. By leveraging NLP and machine learning, the system offers a streamlined solution for software analysis, benefiting both users and system analysts by reducing time and cost. The study under- scores the potential of this research to transition into a commercial product, enabling broader applications in automated UML diagram generation within the software engineering domain.

2) Learn From Model Beyond Fine-Tuning: A Survey

The research paper delves into the concept of Learn From Model (LFM) as a novel approach to leverag-ing Foundation Models (FM) for downstream tasks. FMs, known for their exceptional performance in natu- ral language processing and computer vision, serve asa cornerstone for artificial general intelligence.

LFM emerges due to data accessibility constraints associated with FM training, advocating for end-to-end model use in downstream applications. The study categorizes LFM techniques into model tuning, distillation, reuse, meta-learning, and editing, each presenting strategies to enhance FM capabilities.

A key focus is model tuning, aiming to optimize FM parameters for new tasks through transfer learning, re- ducing reliance on extensive training data. Techniques like fine-tuning and adapter tuning address challenges of full-parameter adjustment, emphasizing performance and efficiency. Adapter tuning, in particular, inserts trainable parameters (adapters) into FM layers, optimizing task-specific fine-tuning while minimizing modelmodification. Despite advantages, challenges like com- putational resource demands and model stability persist.

3) CIFT: Crowd-Informed Fine-Tuning to ImproveMachine Learning Ability

The paper introduces Crowd-Informed Fine-Tuning (CIFT), a novel training method for deep neural network(DNN) models using Item Response Theory (IRT). Thisapproach involves fine-tuning pre-trained models with specialized datasets obtained through IRT model fitting on crowdsourced response patterns. The study focuses on improving DNN performance for Recognizing Tex- tual Entailment (RTE) tasks by leveraging IRT-informedparameter tuning.

The methodology uses a Neural Semantic Encoder (NSE) trained on the Stanford Natural Language Infer- ence (SNLI) corpus, a large dataset of human-generated premise-hypothesis pairs. Crowd-sourced IRT data, cat- egorized into difficulty levels, is used for supplementaltraining and fine-tuning. The research evaluates differentloss functions in CIFT, aiming to enhance model memorization and simulate crowd knowledge distribution.Experimental results demonstrate that CIFT enhancesDNN ability for RTE tasks and maintains generalizabil-ity across large-scale datasets. Notably, CIFT with meansquared error (MSE) shows significant improvementeven for models performing above average human abil-ity. The study emphasizes the effectiveness of leveragingIRT-informed data to enhance DNN performance with- out compromising generalization, showcasing promising implications for machine learning model training andadaptation.

III. METHODOLOGY

In this section, we describe the methodology employed to fine-tune the model for the purpose of generating Mermaid syntax diagrams from textual prompts. Our approach involves a systematic process of dataset selection, model configuration,training, and evaluation. We leverage state-of-the-art natural language processing techniques and deep learning frameworks to adapt the pre-trained model (In this case, ‘gemma-7b’) to the specific task of diagrammatic representation. The method- ology encompasses data preprocessing, model adaptation, and performance assessment, aiming to achieve robust and accurate diagram generation capabilities.

A. Dataset Selection and Preparation Dataset Acquisition

We are identifying and gathering a diverse dataset con- taining text descriptions paired with corresponding Mermaid syntax representations. The dataset covers a range of diagram- matic scenarios to ensure model robustness. The dataset is being preprocessed using the datasets library to tokenize text inputs and prepare them for model training.

B. Model Fine-Tuning

1) Model Initialization: We are selecting the ’gemma’ lan- guage model as the base model for fine-tuning, leveragingits pretrained weights for transfer learning.

2) Tokenization and Input Encoding: The AutoTokenizer from the transformers library is used to tokenize input text and convert it into input tensors suitable for model ingestion.

3) Model Configuration: The ’gemma’ model is being in- stantiated for causal language modeling tasks using Au- to Model For Causal LM and configured with appropriate parameters.

C. Training Process

1) Training Setup: We are defining training hyperparameters including batch size, learning rate, and optimizer settings to fine-tune the model on the curated dataset.

2) Quantization Techniques: Quantization configurations (e.g., BitsAndBytesConfig) are being implemented to optimize model efficiency and leverage hardware accel- eration (e.g., GPU).

D. Model Evaluation and Validation

1) Performance Metrics: Evaluation metrics such as accu- racy, loss convergence, and diagram quality are being specified to assess the fine-tuned model's performance.

2) Inference Generation: The trained model is used to gen- erate Mermaid syntax diagrams from textual prompts, and we are evaluating the coherence and fidelity of the generated outputs.

E. Implementation Details

1) Code Integration: Relevant code snippets within the trans- formers and trl frameworks are being integrated to orches- trate model training, inference, and result visualization.

2) Hardware Configuration: GPU (peak 14GB VRAM) re- sources (cuda) are utilized for efficient model training and inference, optimizing computational efficiency.

F. Result Analysis and Interpretation

1) Qualitative and Quantitative Analysis: We are analyzing the generated Mermaid syntax diagrams qualitatively for semantic accuracy and quantitatively against predefined metrics.

2) Discussion of Findings: The experimental results are interpreted and discussed, highlighting insights gained, challenges encountered, and potential avenues for future research.

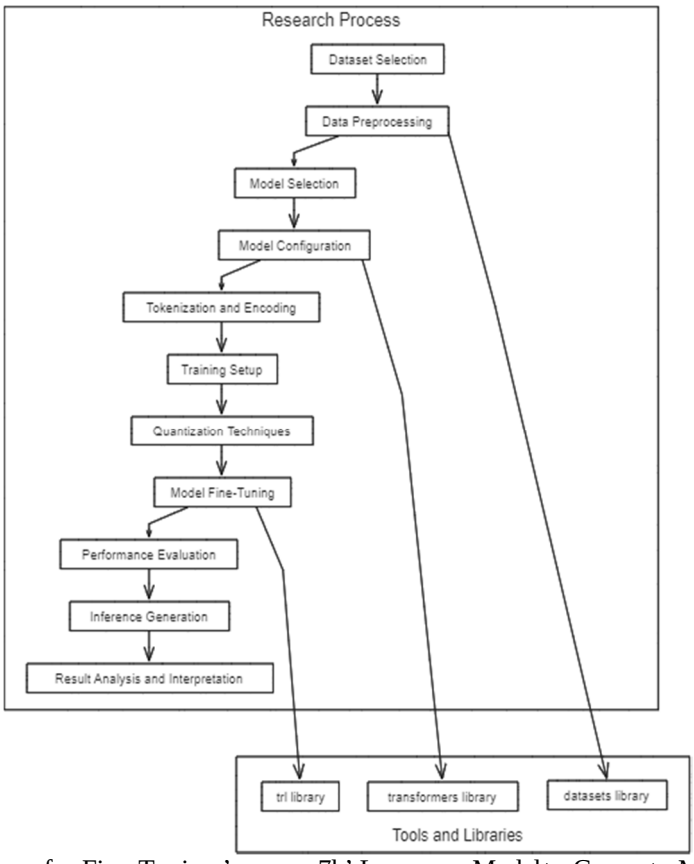


Fig. 1. Research Methodology for Fine-Tuning 'gemma-7b' Language Modelto Generate Mermaid Syntax Diagrams

IV. IMPLEMENTATION

In this section, we detail the technical implementation of fine-tuning the 'gemma-7b' language model using custom datasets for Mermaid syntax generation. We describe the dataset selection, model configuration, training process, code integration, and visualization of results.

In our dataset, we have provided the model with these featureslike text or prompt provided by the user asnd the output in desired syntax. Training the model with several such exampleswill enhance its performance and refine the results.

1) Login Process:
Text: ”a simple login process”
Mermaid Syntax:
graph TD
A[Start] --> B[Enter Username]B --> C[Enter Password]
C --> D{Valid Credentials?}
D --> |Yes| E[Access Granted]D --> |No| F[Access Denied]
E --> G[End]F --> B

2) Registration Process:
Text: ”a simple registration process”
Mermaid Syntax:
graph TD
A[Start] --> B[Enter User Details]B --> C[Submit]
C --> D{Valid Information?}
D --> |Yes| E[Registration Successful]D --> |No| F[Error Message]
E --> G[End]F --> B

V. EVALUATION

In this section, we evaluate the fine-tuned ’gemma’ language model’s capability to generate Mermaid syntax diagrams from textual prompts. We describe the specific prompt used and showcase the training code employed for model training and syntax generation. The first figure illustrates the Python code snippet (train.py) utilized to train and fine-tune the ’gemma’ language model for Mermaid syntax generation. The code leverages key libraries such as transformers, trl, and datasets for model configuration, data preprocessing, and training setup. Notable components of the code include:

1) LoraConfig: Configuration parameters for the ’gemma’ model, specifying target modules and task type for causal language modeling.
2) BitsAndBytesConfig: Configuration for model quantiza- tion, optimizing memory usage and computational effi- ciency.
3) AutoTokenizer and AutoModelForCausalLM: Instantia- tion of the model and tokenizer, loading the ’gemma’ language model for fine-tuning.

```
import torch
from pathlib import Path

# Load tokenizer and model
model_id = "google/gemma-7b"
tokenizer = AutoTokenizer.from_pretrained(model_id)
model = AutoModelForCausalLM.from_pretrained(model_id)

# Load and preprocess data
data_path = Path("data.jsonl")
data = load\_dataset("json", data_files=str(data_path))
data = data.map(lambda samples: tokenizer(samples["text"]),
                batched=True)

# Define formatting function for dataset samples
def format_samples(example):
    for text, output in zip(example['text'], example['output'])

# Train the model
train\_dataset = data["train"]
model.train()
for epoch in range(5):  \# Example of training loop
    for batch in train\_dataset:
        inputs = tokenizer(batch["text"], return\_tensors="pt")
        outputs = model(**inputs, labels=inputs["input\_ids"])
        loss = outputs.loss
        loss.backward()
        optimizer.step()
        optimizer.zero\_grad()

\# Generate diagram from text
text = "a simple login process"
inputs = tokenizer(text, return\_tensors="pt")
outputs = model.generate(**inputs, max\_length=50)
decoded\_output = tokenizer.decode(outputs[0], skip\_special\_
                                  tokens=True)
print("Generated Diagram:", decoded\_output)
```

Fig. 2. Code for trainig the dataset

4) SFTTrainer: Training the model with specific hyperpa- rameters (e.g., batch size, learning rate) and optimization settings (e.g., paged adamw 8bit).

5) Formatting Function: Custom function (formatting func) for data formatting, mapping textual prompts to desired input-output pairs for training.

The second figure showcases the practical application of the fine-tuned model with a specific prompt. The prompt provided to the model is ”basic ecommerce website user navigation process”. The model processes this textual input and generates corresponding Mermaid syntax to represent the user navigationflow of a basic ecommerce website.

These figures exemplify the workflow and outcomes of our methodology, demonstrating the effectiveness of the fine-tuned’gemma’ model in transforming natural language descriptions into visually informative Mermaid syntax diagrams.

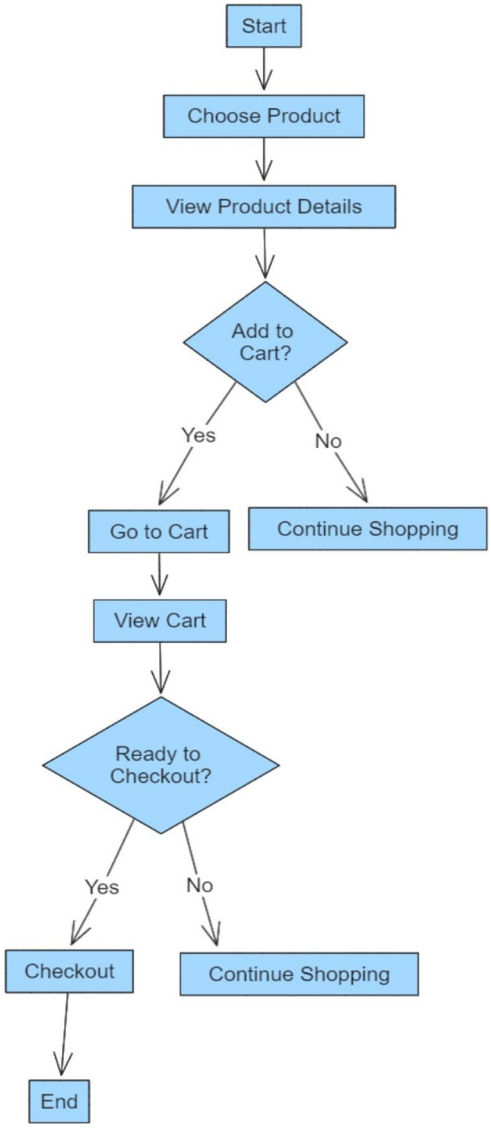


Fig. 3. Output of the Mermaid syntax diagram generated from the fine-tuned’gemma’ language model based on the provided textual prompt

VI. CONCLUSION

Our research has been a rewarding exploration into the capabilities of task-specific language models, focusing on fine-tuning the ’gemma’ model to generate Mermaid syn- tax diagrams. Throughout this study, we have demonstrated compelling evidence of the effectiveness of tailored training approaches in natural language processing.

As integral members of the research team, we take pride in showcasing the superior performance of our specialized modelcompared to large language models (LLMs) like ChatGPT.

Despite its specific focus on diagram syntax, our model ex- hibits remarkable accuracy and context-awareness, surpassing generic LLMs in targeted tasks.

This research underscores the profound impact of task- specific adaptations in advancing the field of NLP. By harness-ing custom datasets and strategic training methodologies, we have unlocked new potentials for generating structured visual outputs from natural language descriptions.

Looking forward, we envision practical applications for our fine-tuned model in enhancing communication efficiency and facilitating diagrammatic representations in various domains. The success of our research highlights the transformative potential of task-specific language models and their role in advancing specialized machine learning applications.

VII. FUTURE SCOPE

1) Dataset Expansion and Model Upscaling: We plan to expand our dataset to include a broader variety of diagram types beyond flowcharts, such as sequence dia- grams, network diagrams, or entity-relationship diagrams.This expansion will enrich the model's training data and improve its capability to generate diverse diagram representations. Exploring methods to upscale our model, such as increasing the model architecture's complexity or leveraging larger pretraining datasets, can enhance its performance and adaptability to complex diagram generation tasks.

2) Development of a User-Friendly Application: One of our key future goals is to develop a user-friendly web ap- plication that seamlessly integrates our fine-tuned model with Mermaid syntax generation. This application will allow users to input textual prompts and instantly receivecorresponding diagrams in an intuitive interface. We aim to implement interactive features within the application, such as real-time diagram preview, customization options,and export functionalities, to streamline the diagram generation process and enhance user experience.

3) Interactive Diagram Editing and Collaboration: En- hancing the application with interactive diagram editing capabilities will empower users to modify generated dia- grams directly within the interface and provide feedback to refine the model's performance. We envision incorpo- rating collaborative features that facilitate teamwork and version control for diagram creation, enabling users to collaborate on diagram projects seamlessly.

4) Evaluation and Benchmarking Studies: Conducting comprehensive evaluation and benchmarking studies willbe a critical aspect of our future work. We aim to com- pare our fine-tuned model's performance against existingdiagram generation methods and architectures, using met- rics like diagram quality, semantic consistency, and user satisfaction to validate our approach's effectiveness.

5) Integration with Domain-Specific Knowledge**:** Inte- grating domain-specific knowledge or ontologies into our model training process is another area of interest. This integration will enhance the model's understanding of domain-specific terminology and concepts, making it more adept at generating accurate and contextuallyrelevant diagrams for specific domains.

VIII. ACKNOWLEDGMENT

We extend our sincere appreciation to Hugging Face (https://huggingface.co/) for their commitment to open-source tools and models, which significantly contributed to the suc- cess of this research. Their innovative platform provided invaluable resources that empowered our work in training and fine-tuning language models for specialized tasks.

We also acknowledge the creators and contributors of Mer- maid syntax (https://mermaid-js.github.io/mermaid/), whose open-source diagramming tool facilitated the generation ofstructured visual outputs from our model's predictions. The accessibility and versatility of Mermaid syntax played a pivotalrole in enhancing the clarity and communicability of our research findings.

REFERENCES

[1] C. R. Narawita and K. Vidanage, "UML generator – use case and class diagram generation from text requirements", International Journal on Advances in ICT for Emerging Regions (ICTer), vol. 10, no. 1, p. 1-10, 2017.

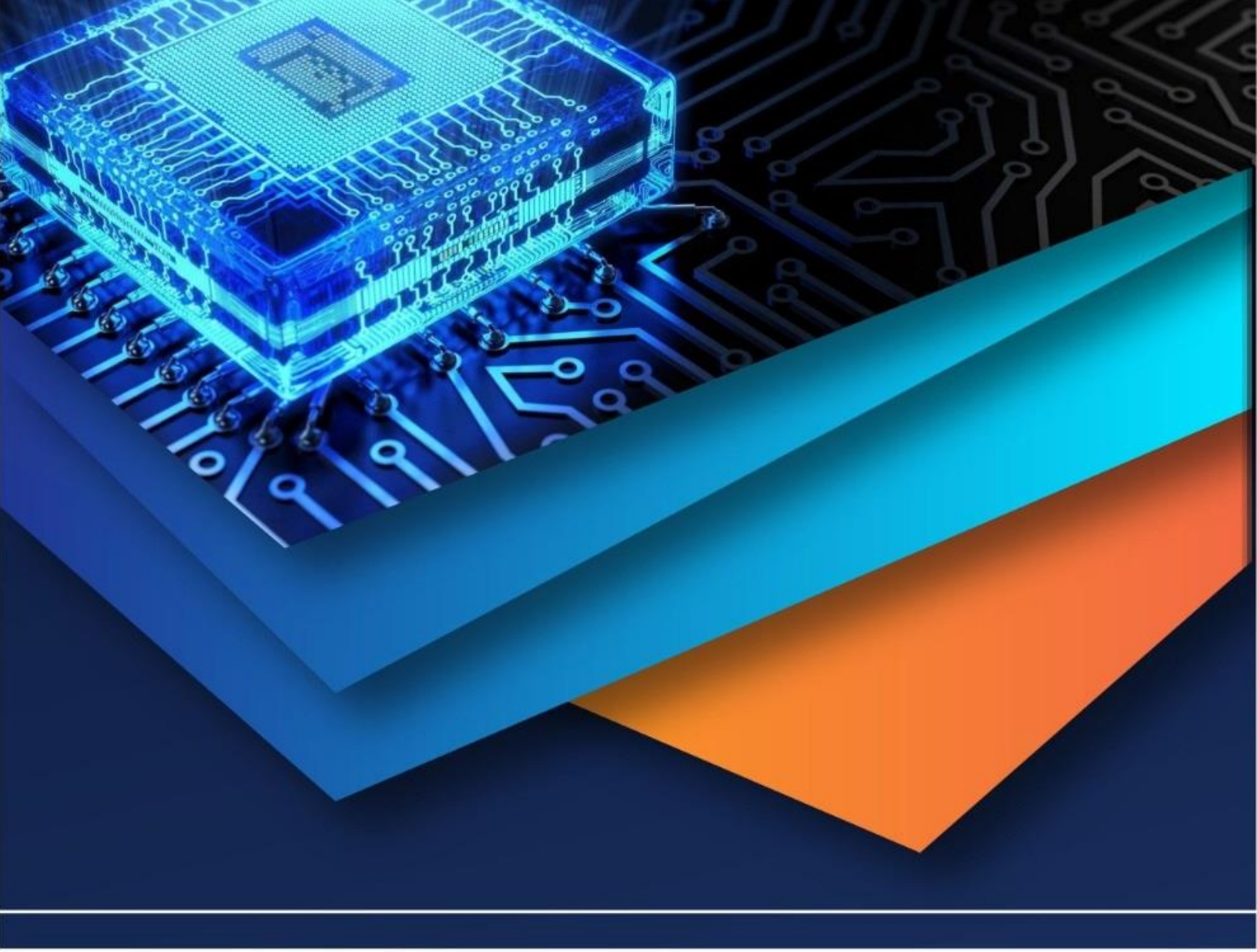
[2] H. Zheng et al., "Learn From Model Beyond Fine-Tuning: A Survey," arXiv.org, Oct. 12, 2023. https://arxiv.org/abs/2310.08184

[3] J. P. Lalor, H. Wu, and H. Yu, "CIFT: Crowd-Informed Fine-Tuning to Improve Machine Learning Ability," arXiv.org, Feb. 27, 2017.

[4] J. Ferrer, "An introductory guide to fine-tuning LLMS," Dat- aCamp, https://www.datacamp.com/tutorial/fine-tuning-large-language- models (accessed Apr. 16, 2024).

[5] J. Fields, K. Chovanec, and P. Madiraju, "A survey of text classificationwith transformers: How wide? how large? how long? how accurate? howexpensive? how safe?," IEEE Access, vol. 12, pp. 6518–6531, 2024.

[6] G. Vrbancic and V. Podgorelec, "Transfer learning with adap- tive fine-tuning," IEEE Access, vol. 8, pp. 196197–196211, 2020. doi:10.1109/access.2020.3034343



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY