Bridging AutoML and LLMs: Towards a Framework for Accessible and Adaptive Machine Learning

Rafael Duque[1][a], Cristina Tîrnăucă[1][b], Camilo Palazuelos[1][c], Abraham Casas[2][d], Alejandro López [2][e] and Alejandro Pérez[2][f]

[1]*Department of Mathematics Statistics and Computer Science, University of Cantabria, Avenida de los Castros S/N, Santander, Spain*

[2]*Centro Tecnológico CTC, Parque Científico y Tecnológico de Cantabria, Santander, Spain*

Keywords: Automated Machine Learning, Large Language Models.

Abstract: This paper introduces a framework architecture that integrates Automated Machine Learning with Large Language Models to facilitate machine learning tasks for non-experts. The system leverages natural language processing to help users describe datasets, define problems, select models, refine results through iterative feedback, and manage the deployment and ongoing maintenance of models in production environments. By simplifying complex machine learning processes and ensuring the continued performance and usability of deployed models, this approach empowers users to effectively apply machine learning solutions without deep technical knowledge.

1 INTRODUCTION

Automated Machine Learning (AutoML) has emerged as a transformative approach to democratize access to machine learning (ML). By automating the selection of models, hyperparameter tuning, and feature engineering, AutoML systems allow users with minimal expertise to leverage powerful ML techniques (Karmaker et al., 2021). AutoML solutions streamline the ML pipeline, reducing the time and effort required to develop robust models while maintaining or even improving their predictive performance.

Large Language Models (LLMs) represent a significant advancement in natural language processing (NLP). Trained on massive datasets, these models can generate human-like text, perform complex reasoning tasks, and adapt to a wide range of applications, from translation to summarization (Fan et al., 2024). The flexibility of LLMs lies in their ability to learn and generalize across diverse tasks with minimal fine-tuning, making them an invaluable tool for both research and practical applications. Their performance on benchmark tasks has pushed the boundaries of what was previously thought achievable in NLP (Chang et al., 2024).

The integration of LLMs into AutoML systems has the potential to empower non-expert users, enabling individuals without a data science background to effectively utilize AutoML tools. LLMs can enhance AutoML by interpreting natural language problem descriptions, suggesting suitable algorithms, and explaining model outputs to users in an intuitive manner (Tornede et al., 2023). Furthermore, LLMs can assist in automating the creation of custom data pipelines and model configurations based on specific user requirements. This synergy between LLMs and AutoML holds the potential to make ML even more accessible and efficient, further closing the gap between technical complexity and practical usability.

Designing framework architectures that integrate Large Language Models (LLMs) with AutoML to meet the needs of non-expert users is a challenging task. This involves ensuring that LLMs can understand user instructions, suggest appropriate solutions, and guide them through the process. At the same time, the framework needs to deliver high-quality results while keeping the experience easy and accessible for users from different fields. To address this

[a] https://orcid.org/0000-0001-8636-3213
[b] https://orcid.org/0000-0002-7129-2237
[c] https://orcid.org/0000-0003-4132-9550
[d] https://orcid.org/0000-0002-7060-9298
[e] https://orcid.org/0000-0002-3262-0605
[f] https://orcid.org/0000-0002-4262-9275

Duque, R., Tîrnăucă, C., Palazuelos, C., Casas, A., López, A. and Pérez, A.
Bridging AutoML and LLMs: Towards a Framework for Accessible and Adaptive Machine Learning.
DOI: 10.5220/0013448500003929
In *Proceedings of the 27th International Conference on Enterprise Information Systems (ICEIS 2025) - Volume 1*, pages 959-964
ISBN: 978-989-758-749-8; ISSN: 2184-4992

challenge, this article presents a proposal that aims to consolidate the advancements made by the scientific community in this area, offering a comprehensive approach to bridge existing gaps.

The structure of this article is as follows. Section 2 reviews related work in the field of AutoML and LLM integration, highlighting existing frameworks and their limitations. In Section 3, we present the architecture of our proposed framework, detailing its design and key components. Section 4 provides a discussion of the framework's strengths, including its adaptability and usability for non-expert users, as well as its potential impact compared to existing approaches. Finally, Section 5 concludes the paper, summarizing the main contributions and outlining future directions for research and development.

2 RELATED WORK

Recent advancements in AutoML frameworks have focused on enhancing usability and performance through the integration of LLMs. Thus, AutoM3L (Luo et al., 2024) stands out as a comprehensive system designed for multimodal ML tasks, addressing the limitations of earlier frameworks like AutoGluon. It leverages LLMs to automate main steps such as feature engineering, model selection, and pipeline assembly, thus reducing the steep learning curve for non-expert users. The framework achieves competitive performance on various multimodal datasets, showcasing the potential of LLMs to streamline complex ML processes.

Aliro (Choi et al., 2023) presents a user-friendly AutoML tool specifically designed for the biomedical and healthcare domains. It combines an intuitive web interface with the power of LLMs to assist users in automating data analysis and ML tasks. By providing a conversational interface, Aliro allows users to interact dynamically with their data, offering code execution and model recommendations. This approach significantly lowers the barriers for researchers without programming expertise, enhancing accessibility to advanced ML tools.

GizaML (Sayed et al., 2024) introduces a meta-learning framework tailored for automated time-series forecasting. It employs an LLM-based meta-model to recommend optimal ML configurations based on dataset characteristics. The framework's ability to handle complex tasks such as feature extraction and hyperparameter optimization highlights its effectiveness in real-time applications. GizaML's focus on time-series data addresses a critical need in domains like energy and financial forecasting, where traditional AutoML tools often fall short.

The integration of LLMs into AutoML frameworks is further exemplified by systems like SmartML(Maher and Sakr, 2019), TPOT (Le et al., 2020), and JarviX (Liu et al., 2023). SmartML and TPOT use meta-learning and genetic programming to optimize pipelines, but they lack advanced user interaction features. JarviX addresses this gap by utilizing LLMs to guide users through a rule-based system for data visualization and statistical analysis. JarviX combines a vectorized domain knowledge repository and fine-tuned models like Vicuña and GPT-4 to generate exploratory insights via text or voice input, culminating in comprehensive reports. It further integrates H2O-AutoML pipelines for customized model training, enhancing its analytical depth and flexibility.

These advancements illustrate the significant progress in making AutoML frameworks more accessible to non-expert users through the integration of LLMs. However, existing frameworks like AutoM3L (Luo et al., 2024) focus primarily on automating core steps such as feature engineering and model selection, while systems like GizaML (Sayed et al., 2024) and JarviX (Liu et al., 2023) are domain-specific or dependent on particular technologies. Our proposed framework aims to address these gaps by offering a more flexible, user-friendly approach that enables iterative model refinement and provides explainability, all while being independent of specific technologies. This flexibility and emphasis on explainability could make this framework distinct from the existing solutions. The following sections will present this framework in detail, outlining its design and capabilities.

3 STRUCTURE OF THE FRAMEWORK

This section presents the architecture of a user-friendly framework that combines AutoML and LLMs (see Figure 1), designed to empower users without deep expertise in ML to leverage advanced ML techniques for data analysis, problem definition, model training, and optimization. The framework uses natural language processing (NLP) through LLMs to guide the user through each step of the process, simplifying the configuration and refinement of ML workflows.

The framework begins with an **Input Layer** (see Figure 1) where the user interacts with the system through a natural language interface. The LLM acts as a virtual assistant, helping the user describe the dataset and define the problem to be solved. The user uploads their dataset, which could be in various for-

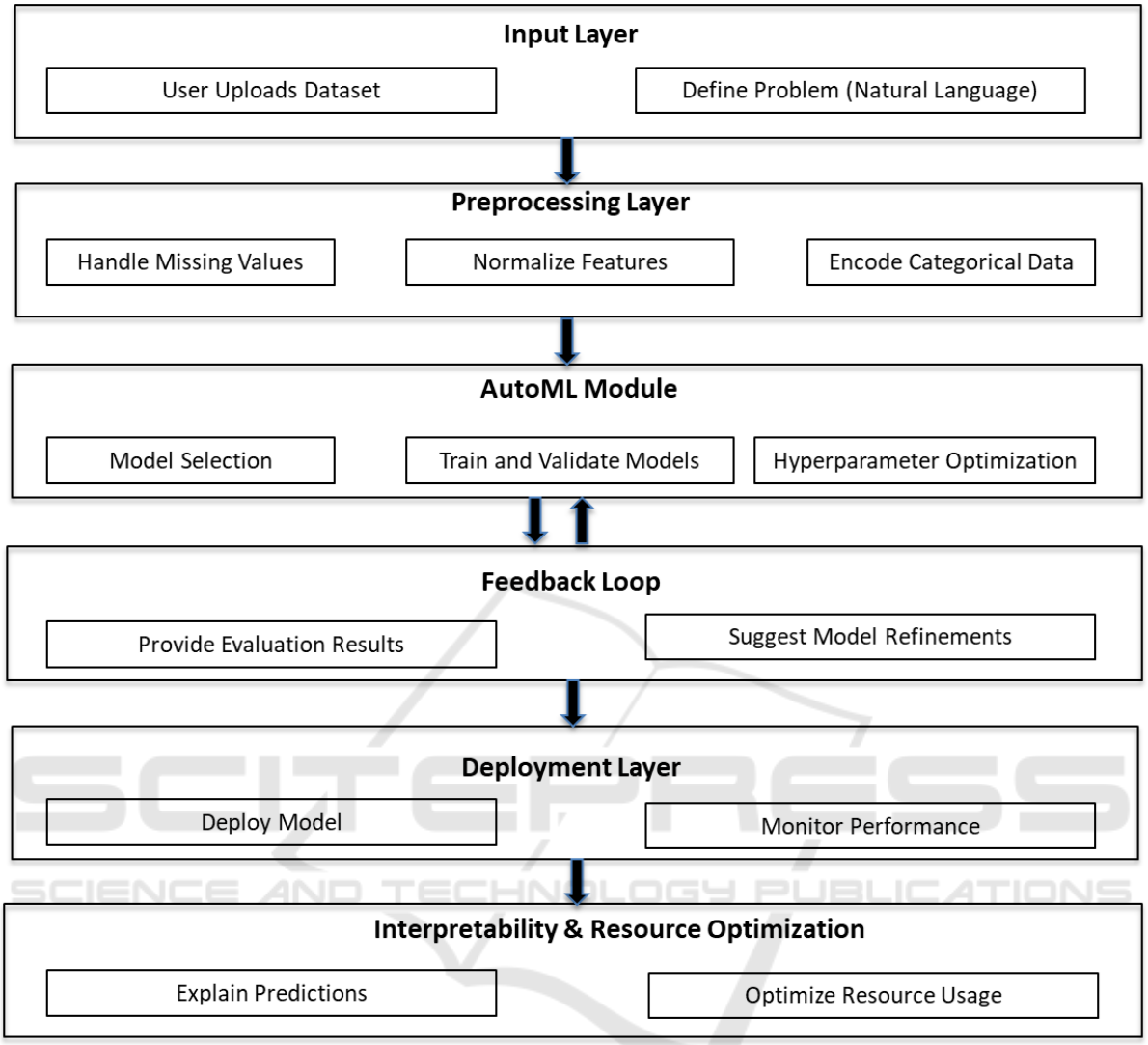

Figure 1: Architecture of the framework.

mats, such as CSV files, text documents, or databases. The LLM processes the dataset and generates a summary that highlights the features, the types of data (e.g., numerical, categorical, or text), and any potential relationships between variables. The LLM can automatically detect missing values, identify outliers, and suggest necessary preprocessing steps.

Using natural language, the user defines the problem they want to solve. For instance, the user might describe a goal such as "predicting future sales" or "classifying customer reviews." The LLM translates this high-level description into a specific ML task, such as regression or classification, and prepares the system to set up the AutoML process accordingly.

Once the problem is defined, the framework moves into the **Preprocessing Layer** (see Figure 1). At this stage, the system ensures the dataset is appropriately prepared for modeling. The LLM offers guidance on the preprocessing steps that should be applied to the data. It recommends handling missing data, normalizing numerical features, encoding categorical variables, or tokenizing text for NLP tasks. The LLM provides explanations for each preprocessing technique, ensuring that users understand the purpose of these steps. The system can automatically apply these transformations based on the dataset's characteristics, allowing for a streamlined preparation process.

The next stage of the framework involves configuring the AutoML process, where the LLM plays a main role in helping the user select appropriate settings. For example, if the user is unsure whether to select a decision tree or a neural network, the LLM

can explain the advantages of each and recommend which model might be most suitable for the task at hand.

The AutoML system autonomously selects and evaluates a range of ML models based on the configuration. The LLM helps users understand the selection process, explaining how AutoML identifies the best models for the problem and adjusts hyperparameters to maximize performance. Users are not required to understand the underlying technicalities; the LLM simplifies these concepts into user-friendly language.

The **AutoML Module** (see Figure 1) is the core component of the framework, responsible for automating the ML workflow. This layer handles the bulk of the computational processes, allowing users to focus on higher-level decisions. Based on the configuration, the AutoML module trains multiple ML models using the dataset. This could include models like Random Forests, Support Vector Machines, or neural networks. The system automatically splits the data into training and validation sets, ensuring proper model evaluation.

The AutoML system performs **hyperparameter optimization** (see Figure 1) to fine-tune model parameters and improve the accuracy of the trained models. Methods like grid search, random search, or Bayesian optimization are applied to find the best possible configuration for each model. After training, the models are evaluated using appropriate metrics. For classification tasks, common evaluation metrics include accuracy, precision, recall, and F1-score. For regression, metrics such as mean squared error (MSE) or R-squared are used. The system generates an evaluation report that allows the user to assess model performance.

Following the initial model evaluation, the framework enters a feedback loop that integrates the LLM to refine and optimize the process. The LLM reviews the evaluation results and provides the user with an easy-to-understand analysis of model performance. For example, it can explain why a model performed well or poorly based on certain features or hyperparameter settings. The LLM also offers visualizations to help the user understand the model's strengths and weaknesses.

Based on the evaluation metrics, the LLM suggests adjustments to improve model performance. For instance, if the model performs poorly on certain subsets of data, the LLM might recommend using additional features, applying different preprocessing techniques, or altering the hyperparameter settings. The user can ask the LLM specific questions like, "What can we do to improve the model?" or "Why did the model underperform?" The LLM interprets the user's queries and helps refine the AutoML pipeline accordingly, guiding the user through the process of improving the model.

Once the LLM provides feedback and suggestions, the framework enters an iterative process of model reconfiguration. The AutoML system automatically applies any suggested changes, such as retraining the model with different hyperparameters or additional data features. The LLM ensures that the user is aware of all adjustments and their implications. The training, evaluation, and refinement cycle continues until the best model is identified, ensuring that the user receives a model with optimal performance tailored to their problem.

After the model has been trained and optimized, the framework moves to the **Deployment Layer** (see Figure 1), where the model is put into production. The selected model is deployed into a production environment. The deployment process is automated, ensuring that the user does not need to manage the intricacies of model deployment. The LLM can help explain the deployment process and provide guidance on how to integrate the model into existing systems.

The system continuously monitors the model's performance in the real world. The LLM alerts the user to potential issues, such as data drift or declining accuracy, and suggests corrective actions. The LLM can also assist in retraining the model if performance degradation occurs over time.

Interpretability and explainability are critical for understanding how models make decisions, especially for non-experts. Thus, the **Interpretability Layer** (see Figure 1) ensures that users can trust and understand their models. The LLM explains the reasoning behind the model's predictions in a way that is accessible to the user. For instance, it might say, "The model predicted this outcome because there was a high correlation between the values of feature X and thos of the target variable". This enhances the user's confidence in the model's decisions. The LLM also provides insights into the dataset and model predictions, highlighting patterns in the data that the model has identified. This helps the user gain a deeper understanding of the problem domain and the model's behavior.

The **Resource Optimization Layer** (see Figure 1) focuses on ensuring the framework operates efficiently, even with large datasets or complex models. The system manages computational resources, such as GPUs and CPUs, to ensure cost-effective training and inference. The LLM can suggest resource optimization strategies to the user, helping them balance performance with computational cost. The system is designed to scale with the complexity of the task, en-

suring that even large datasets can be processed efficiently without compromising performance.

4 DISCUSSION

The framework offers a solution for empowering non-expert users to leverage ML techniques. This section discusses the key aspects of the framework architecture, such as its user control, modularity, adaptability, and its ability to offer explainability. Additionally, we explore how the framework supports the iterative process of ML, allowing users to refine their models and improve performance continuously:

- **User Control and Interaction.** the framework provides the user with clear steps, ensuring that even those without technical expertise can engage with ML processes effectively. The process begins with the user describing the dataset and defining the problem. The LLM then assists in understanding the dataset and clarifying the problem, translating high-level descriptions into ML tasks. This ensures the user retains control over the workflow, while the system provides valuable support in every step. For instance, once the user defines the task, the LLM interprets it into a clear ML objective, such as classification or regression. This progression allows the user to guide the process without needing to understand the intricacies of the underlying algorithms.
- **Modularity and Adaptability.** the framework's various components, such as dataset understanding, problem definition, preprocessing, and model selection, work together in an adaptable system. As the user proceeds through the workflow, the system applies modular AutoML tools and adjusts the configuration automatically based on user inputs. Each layer of the process, from data preprocessing to model evaluation, is adaptable to different types of datasets and problem domains. Whether the user is dealing with time-series data, images, or text, the framework adapts its methods and tools to fit the task at hand, allowing users to solve a broad range of problems without needing to switch between different technologies or tools.
- **Technology Independence.** the framework's design makes it independent of specific ML technologies or platforms. While the AutoML component utilizes various ML models (decision trees, neural networks, etc.), the user interacts with the system through natural language and does not need to worry about the technical details of the underlying tools. The LLM provides an abstraction layer, allowing the user to remain agnostic to the specifics of the AutoML tools or technologies. This independence from technology means that the framework can easily incorporate new advancements in ML, ensuring the system remains up-to-date without disrupting the user experience.
- **Explainability and Interpretability.** a major challenge in ML is ensuring the results are understandable to non-experts. The proposal addresses this challenge by providing explainability through its LLM interface. The LLM explains model predictions and offers insights into how different features influenced the outcome. This allows the user to better understand the model's behavior and gain confidence in the results. For instance, when a model makes a prediction, the LLM can explain why certain features were important in driving the decision. This transparent communication is crucial for helping users make informed decisions about further improving the model or selecting the best one for deployment. In this way, the framework enables the user to trust and interpret the results, which is especially important when the user lacks the expertise to dive into the technical aspects of the model.
- **Iterative Refinement and User Feedback.** one of the core features of the framework is the continuous feedback loop that allows users to refine their models iteratively. The LLM helps the user understand why certain models performed better than others and makes specific suggestions to improve performance. The iterative cycle of training, evaluation, feedback, and reconfiguration ensures that the model improves over time. The framework's design supports ongoing refinement based on user interactions.

5 CONCLUSIONS

This work presents an architecture for a framework that offers an accessible solution for users without deep expertise in ML. By allowing users to engage in tasks like dataset description, problem definition, and model refinement through natural language, the framework simplifies the process of ML. The integration of AutoML with LLMs makes it possible for non-experts to navigate complex workflows with ease, bridging the gap between advanced ML techniques and user-friendly interfaces.

The framework also ensures iterative improvement and clarity at each step, allowing users to refine their models based on feedback from the system.

By continuously guiding users through each stage of the process, the system fosters a deeper understanding of ML principles while enabling effective decision-making without requiring advanced technical knowledge.

Future work will focus on building ML models and experimenting with the proposal to assess its effectiveness in real-world scenarios. This will include testing the framework with diverse datasets and problem types to ensure its scalability and adaptability. Moreover, LLMs may have limitations, such as high computational costs, biases, and the potential for generating incorrect information. These issues were not addressed in this paper but could be considered in future work.

ACKNOWLEDGEMENTS

This work has been partially supported by grants PID2022-139237NB-I00 and PID2023-146243OB-I00 funded by MICIU/AEI/10.13039/501100011033 and by "ERDF/EU". This research was also partially developed in the project FUTCAN-2023/TCN/018 that was co-financed from the European Regional Development Fund through the FEDER Operational Program 2021-2027 of Cantabria through the line of grants "Aid for research projects with high industrial potential of excellent technological agents for industrial competitiveness TCNIC".

REFERENCES

Chang, Y., Wang, X., Wang, J., Wu, Y., Yang, L., Zhu, K., Chen, H., Yi, X., Wang, C., Wang, Y., et al. (2024). A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology*, 15(3):1–45.

Choi, H., Moran, J., Matsumoto, N., Hernandez, M. E., and Moore, J. H. (2023). Aliro: an automated machine learning tool leveraging large language models. *Bioinformatics*, 39(10):btad606.

Fan, L., Li, L., Ma, Z., Lee, S., Yu, H., and Hemphill, L. (2024). A bibliometric review of large language models research from 2017 to 2023. *ACM Transactions on Intelligent Systems and Technology*, 15(5):1–25.

Karmaker, S. K., Hassan, M. M., Smith, M. J., Xu, L., Zhai, C., and Veeramachaneni, K. (2021). Automl to date and beyond: Challenges and opportunities. *ACM Computing Surveys (CSUR)*, 54(8):1–36.

Le, T. T., Fu, W., and Moore, J. H. (2020). Scaling tree-based automated machine learning to biomedical big data with a feature set selector. *Bioinformatics*, 36(1):250–256.

Liu, S.-C., Wang, S., Lin, W., Hsiung, C.-W., Hsieh, Y.-C., Cheng, Y.-P., Luo, S.-H., Chang, T., and Zhang, J. (2023). Jarvix: A llm no code platform for tabular data analysis and optimization. *arXiv preprint arXiv:2312.02213*.

Luo, D., Feng, C., Nong, Y., and Shen, Y. (2024). Autom3l: An automated multimodal machine learning framework with large language models. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 8586–8594.

Maher, M. M. M. Z. A. and Sakr, S. (2019). Smartml: A meta learning-based framework for automated selection and hyperparameter tuning for machine learning algorithms. In *EDBT: 22nd International conference on extending database technology*.

Sayed, E., Maher, M., Sedeek, O., Eldamaty, A., Kamel, A., and El Shawi, R. (2024). Gizaml: A collaborative meta-learning based framework using llm for automated time-series forecasting. In *EDBT*, pages 830–833.

Tornede, A., Deng, D., Eimer, T., Giovanelli, J., Mohan, A., Ruhkopf, T., Segel, S., Theodorakopoulos, D., Tornede, T., Wachsmuth, H., et al. (2023). Automl in the age of large language models: Current challenges, future opportunities and risks. *arXiv preprint arXiv:2306.08107*.