Systems & Control Transactions

Research Article Peer Reviewed Conference Proceeding
ESCAPE 35 European Symposium on Computer Aided Process Engineering
Ghent, Belgium. 6-9 July 2025
Jan F.M. Van Impe, Grégoire Léonard, Satyajeet S. Bhonsale,
Monika E. Polańska, Filip Logist (Eds.)

PSE PRESS

Empowering LLMs for Mathematical Reasoning and Optimization: A Multi-Agent Symbolic Regression System

Shaurya Vats[a], Sai Phani Chatti[a*], Aravind Devanand[a], Sandeep Krishnan[a], and Rohit Karanth Kota[a]

[a] Siemens Technology and Services Pvt. Ltd
* Corresponding Author: Bhanu-venkata.chatti@siemens.com.

ABSTRACT

Understanding data with complex patterns is a significant part of the journey toward accurate data prediction and interpretation. The relationships between input and output variables can unlock diverse advancement opportunities across various processes. However, most AI models attempting to uncover these patterns are not explainable or remain opaque, offering little interpretation. This paper explores an approach in explainable AI by introducing a multi-agent system (MaSR) for extracting equations between features using data. We developed a novel approach to perform symbolic regression by discovering mathematical functions using a multi-agent system of LLMs. This system addresses the traditional challenges of genetic optimization, such as random seed generation, complexity, and the explainability of the final equation. We utilize the in-context learning capabilities of LLMs trained on vast amounts of data to generate accurate equations more quickly. This study presents research on expanding the reasoning capacities of large language models alongside their mathematical understanding. The paper serves as a benchmark in understanding the capabilities of LLMs in mathematical reasoning and can be a starting point for solving numerous complex tasks using LLMs. The MaSR framework can be applied in various areas where the reasoning capabilities of LLMs are tested for complex and sequential tasks. MaSR can explain the predictions of black-box models, develop data-driven models, identify complex relationships within the data, assist in feature engineering and feature selection, and generate synthetic data equations to address data scarcity, which are explored as further directions for future research in this paper.

Keywords: Symbolic regression, Large Language Models, Multi-Agent Systems

INTRODUCTION

The foundation of any machine learning algorithm lies in understanding the mathematics behind it. The entire field of artificial intelligence is built upon mathematical principles, using data as its starting point. Accurately uncovering and interpreting relationships within data is a critical step for advancing tasks in data science and its applications. These relationships often take the form of equations that serve as governing rules, applicable across a wide range of domains, including simulations and industrial processes. To address this critical challenge, we developed a multi-agent system using large language models (LLMs) to effectively uncover these mathematical relationships, enabling advancements across various domains such as simulations and industrial processes.

SYMBOLIC REGRESSION: AN OVERVIEW

One powerful technique for discovering such relationships is Symbolic Regression[1,2], which identifies mathematical expressions that best describe the relationships between variables in a dataset. This technique can be implemented through various approaches, including genetic optimization[1,2], reinforcement learning[5], recurrent neural networks (RNNs)[5], generative adversarial networks (GANs)[6], and emerging methods. Despite differences in implementation, these methods follow a common set of steps that define symbolic regression as a process: finding, evaluating, and refining relationships to generate interpretable equations.

Symbolic Regression (SR) plays a crucial role in Process Engineering by enabling the discovery of mathematical models that accurately represent complex chemical processes. In process systems, understanding the underlying relationships between variables is essential for optimization, control, and design. SR offers a powerful tool for automatically deriving these models from experimental or simulated data without requiring prior knowledge of the system's functional form. This capability is particularly beneficial in fields like chemical engineering, where processes can be highly nonlinear, dynamic, and data-intensive. By using SR, engineers can gain insights into the fundamental behavior of a process, leading to improved efficiency, better decision-making, and enhanced predictive capabilities for process design and operation.

FRAMING SYMBOLIC REGRESSION AS A SEARCH PROBLEM

Symbolic regression can be framed either as a learning task[4] or a search problem, where the objective is to identify the optimal equation or parameters within the domain of possibilities. By leveraging **large language models (LLMs)**, the symbolic regression process can be significantly enhanced. LLMs, with their extensive domain knowledge and in-context learning capabilities, have the potential to act as the "brain" of the process, streamlining and optimizing the discovery of symbolic equations. This approach shifts reliance from purely data-driven metrics to a more knowledge-driven framework. The domain of equations can be constrained by the context of the problem statement and the space within which the equation is being searched.

INTRODUCTION TO MASR FRAMEWORK

To explore this potential, we introduce **MaSR: Multi-Agent Symbolic** Regression, a novel framework that employs LLMs to execute and coordinate the various steps of symbolic regression. In this system, multiple agents, each powered by an LLM, collaborate to handle tasks such as searching for relationships, optimizing equations, and explaining results. By integrating LLMs as the core of the process, MaSR not only automates symbolic regression but also provides interpretable and robust equations for practical applications.

PROBLEM AND EXISTING SOLUTIONS

Discovering relationships and hidden patterns within datasets is essential for optimization, prediction, forecasting, and other applications.

Challenges with Existing Models

- The models commonly used for these purposes are often black-box models.
- They are not always well-suited to the specific data being used, as they rely on a limited set of predefined models.
- There is often no mathematical relationship available to effectively link features for tasks like feature selection, synthetic data generation, or feature engineering.
- A lack of physics-based or domain-specific equations ensures constraints are not always adhered to in applications.
- Many transparent models available today are nonlinear and difficult to interpret.
- Traditional models require large amounts of data to perform predictions and learn from the data effectively.

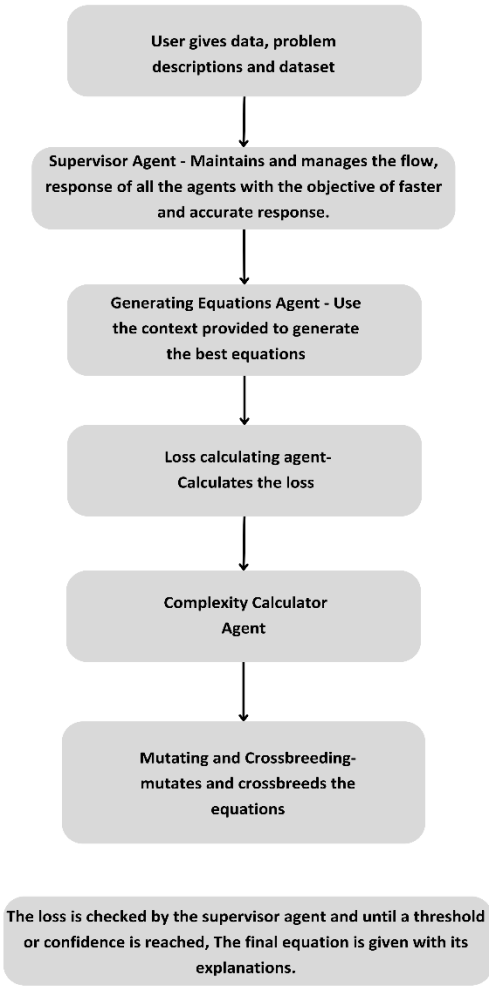

Figure 1. Step-wise process of MaSR framework.

Our goal is to derive mathematical equations and

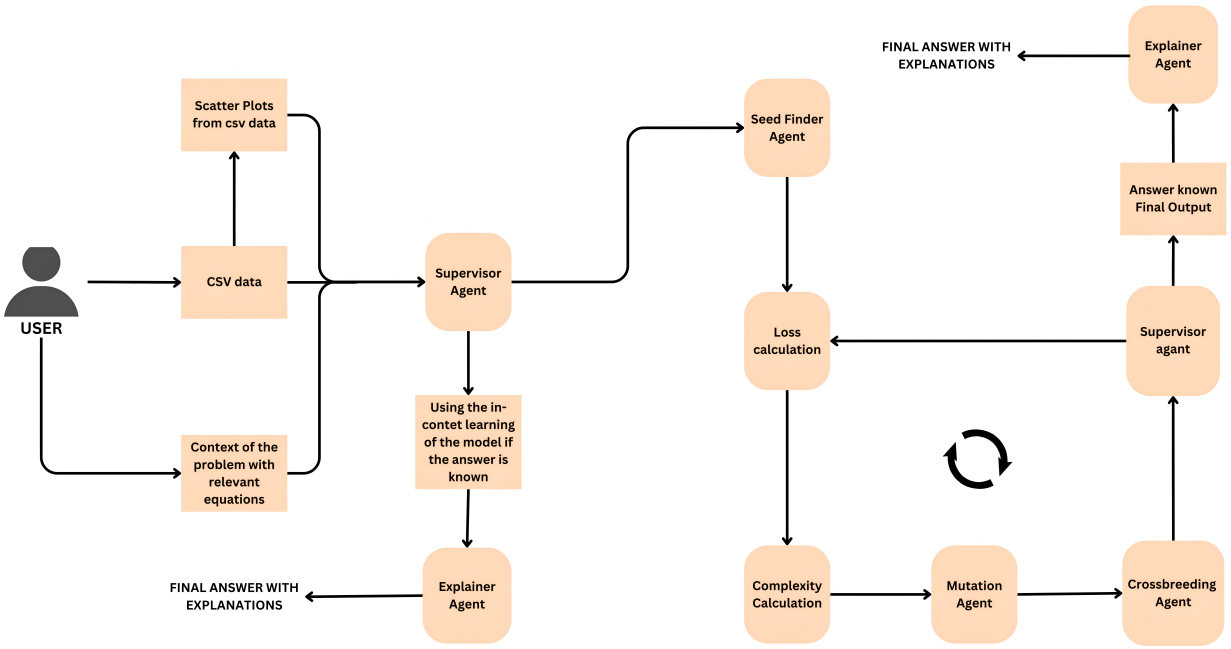

Figure 2: This image illustrates the complete workflow of a multi-agent symbolic system. The process begins with the user providing context and a CSV file to the system. The CSV data is transformed into scatter plots, which are then passed to the supervisor agent along with the original CSV data and user-provided context. If the supervisor agent already has knowledge of a relevant equation within its in-context learning capabilities, it directly returns the equation. Otherwise, the data and context are forwarded to the seed finder agent, followed by the loss finder agent, the complexity calculator agent, and finally the mutation and cross-breeding agent. Throughout the process, the supervisor agent continuously evaluates the equation against a loss threshold. Once an equation meets the specified conditions, it is passed to the explanation module, which provides detailed explanations for the equation. The final output includes both the optimal equation and its corresponding explanations.

relationships from data, creating an explainable white-box model for various applications.

Existing Solutions and Their Limitations

The problem has been addressed using genetic optimization, with several variations and libraries available, including PySR[1], gplearn[2], and SINDy[9]. However, these libraries have the following limitations[8]:

- Random generation of seed functions.
- Managing equation complexity.
- Ensuring the explainability of the final equation.
- Lack of domain-specific knowledge integration.

These limitations are being tackled separately by different methods, such as Pareto-Symbolic Regression[10] and Symbolic GPT[4].

However, no single method currently solves all these problems simultaneously, particularly the challenge of making the equations explainable

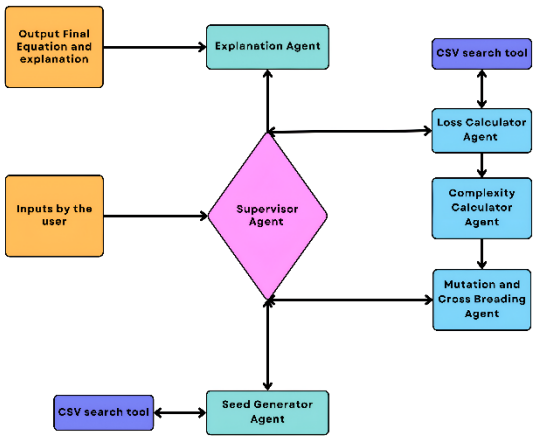

Figure 3. Interaction between different agents of MaSR Framework.

MASR FRAMEWORK

The MaSR framework divides the task into multiple steps, with each step performed by a dedicated agent equipped with specific tools and knowledge bases. These agents collaborate by discarding less accurate or less insightful equations based on metrics such as mean squared error and equation complexity.

The process mimics genetic optimisation and is designed entirely using LLMs to derive relationships from the data.

Foundation of the MaSR Framework

The framework is built upon the reasoning capabilities of large language models (LLMs), leveraging their in-context learning. Its architecture relies on the in-context learning of LLMs and their ability to effectively identify the seed generation agent. The prompts used in each agent provide the necessary context and instructions. A **sample prompt** from one of the agents (seed finder) is provided below:

"""

Task:

You are a mathematical equation generator. Your goal is to derive the accurate and complex equations that describe the relationship between the target variable ['target_variable'] and the independent variables ['independent_variables'], based on the provided scatter plot,csv file data and domain knowledge.

Context:

You have knowledge of ['domain'], including relevant physical and mathematical principles such as dimensional analysis and proportionality. You also have access to:

Data points from a **CSV file** that illustrate the values of all the variables.

scatter plots representing the relationship between the independent variables and the target variable.

Equations: ['Equations'].

Available operators: ['Operators'].

Dimensional information: ['Dimensions'].

Instructions:

1. **Analyze the Data**: Start by analyzing the scatter plot and the data points from the CSV file to understand the relationships between the target variable and the independent variables.

2. **Use Domain Knowledge**: Leverage your understanding of ['domain']}, including fundamental physical and mathematical concepts like distance calculation, to guide your equation formulation.

3. **Generate Equations**: If no existing equation is found, generate up to ['number_of_functions'] possible equations using the operators provided. Start with basic forms and build progressively more complex equations. Incorporate the general proportional relations and dimensional constraints provided.

Output:

Provide the final ['number_of_functions'] functions as mathematical equations that best describe the relationship between ['target_variable'] and ['independent_variables'].

If an existing equation from the domain of ['domain'] is found, present it as the final solution.

Output Format:

Return the entire mathematical equations as strings, formatted like this: `"f1(x) = ..."`, `"f2(x) = ..."`, etc as complex and creative as possible.

You must not output python code and must not use mnemonics in the equation it must be from ['target_variable']

and ['independent_variables'].

"""

Agents and Their Roles

- Seed Function Generator Agent: Generates candidate equations based on the relationships between target and independent variables using the provided data and context.
- Loss Calculator Agent: Evaluates each eqution by calculating the loss to determine its fit to the data.
- Complexity Calculator Agent: Assesses and scores

the complexity of each equation, prioritising simpler models.

- Equation Mutating Agent: Modifies equations to reduce loss and complexity, aiming to improve accuracy.
- Supervisor Manager Agent: Manages the entire process, ensuring effective collaboration among agents to identify the best equation.
- Explainer Agent: Provides detailed insights into the final equation's behaviour and the relationships between the target and independent variables across the domain.

Framework Overview

The MaSR framework employs a systematic multiagent system of LLMs to discover accurate mathematical relationships from data. These relationships can be utilised for modelling, feature engineering, and feature selection. The framework addresses limitations in existing methods and includes the following key components:

- Hierarchical Structure: A supervisor agent oversees the process, ensuring proper flow of outputs between agents and delivering the final equation with its explanation.
- Agent Responsibilities: Agents handle specific tasks such as seed function generation, loss calculation, complexity evaluation, mutation, and explanation.

Explainability from MaSR?

The framework explains the final equation by breaking down its terms and coefficients. A simplified example is provided below to demonstrate the output of the MaSR framework:

'''

To understand this equation, let's define its components first.
MD_power: This term represents some measure that depends only on MD, whose value ranges from 3 to 95.
SG130: This term stands for another measure depending solely on SG130, ranging from 10 to 570.
TemperatureMD: This variable is equal to the expression (3 * MD_power + 0.845 * SG130) x 0.118 + 36.375), *where MD_power and SG130 take on any values within their respective ranges.*
Now, let's analyze the equation based on the above definitions.

Firstly, observe the multiplicative factor (3) associated with MD_power; since it has a positive sign, increasing MD_power results in higher values assigned to TemperatureMD. Consequently, when working within the context of this problem, larger MD_power values lead to greater temperatures.
Next, see the additive component (0.845) linked to SG130; because this term also carries a positive sign, raising SG130 leads to increased values attributed to TemperatureMD. Therefore, more significant values of SG130 imply hotter conditions according to the problem definition.

Note the coefficient (0.118) applied directly onto the sum of the previous two terms. Since this constant is positive too, enlarging the outcome originating from the sum generates amplified values of TemperatureMD. Hence, augmenting the total output by a substantial quantity raises the observed temperature.
Lastly, consider the additive term (+36.375). Regardless of changes in either MD_power or SG130, this constant consistently contributes a minimum of 36.375 toward the calculation of TemperatureMD.
In conclusion, the given equation defines the interplay among factors affecting certain environments' temperatures, namely MD_power and SG130. By raising both MD_power and SG130, the resulting temperature increases proportionally due to the additive properties present in the equation.
'''

RESULTS

The paper primarily aims to introduce a novel approach to performing a traditional method. The framework can be adapted with new agents, data sources, and methods to address specific problems. The current architecture was tested against various applications and compared with traditional genetic optimization methods.

1. We evaluated the performance of genetic optimization methods such as gplearn and MaSR to derive the equations governing the physics of industrial chillers. These equations were used as a loss function of ANNs to create a Physics Informed Neural Network (PINNs). Our key observations were as follows:

- The framework's performance, in terms of metrics like the R^2 score, was 2% better than the traditional methods.
- The initial equation selected by the seed generation agent adhered to the physics of cooling in chillers.
- The MaSR framework converged to the final equation more quickly and was effective even with a limited or scarce dataset.
- Combining MaSR with traditional genetic

optimization algorithms can enhance the overall performance of Symbolic Regression.

- The final equation derived by MaSR provided an interpretable explanation and aligned with known proportionalities.

2. We validated the framework using 20 Feynman equations, successfully identifying all of them based on the in-context learning capabilities of the underlying models. This method can accurately determine the governing physical equation when it is uncertain. In contrast, other genetic optimization methods without an embedded learning mechanism fail in this regard.

KEY ADVANTAGES OF MASR

The main differences between our framework and existing solutions are as follows:

Utilizing LLMs: Utilises the in-context knowledge base and domain-specific expertise of LLMs to generate accurate equations quickly, even with sparse data.

- Explainability: Provides detailed explanations and insights from the final equations, fostering transparency and interpretability.
- Guided Seed Generation: Addresses the issue of random seed generation in current methods by guiding the process toward more accurate equations swiftly and efficiently.
- Simplified Complexity: Ensures the final equations are concise and easy to understand, enhancing their explainability and usability.

CHALLENGES AND LIMITATIONS

The major challenge we faced during the implementation process was accurately enabling the agents to collaborate efficiently while adhering to the workflow. We were able to partially address this by using well-crafted prompts and improving the conditions under which an agent reaches out to another. However, the issue of agents fully understanding and consistently maintaining the workflow for complex tasks remains unresolved and requires further attention.

Another significant issue we encountered was stopping the process when agents reached optimal conditions. To address this, a clear stopping criterion must be defined and strictly enforced through implementation.

In this study, the large language models (LLMs) we explored ranged from multimodal models like Gemini to GPT-based models such as GPT-3.5-Turbo and open-source models like LLaMA 3. We chose to focus on Gemini due to its ability to effectively interpret and process data. To support this, we utilized a combination of Retrieval-Augmented Generation (RAG), structured data in prompts, and scatter plots as data sources. However, passing numerical data to LLMs remains an area that requires significant attention. Ensuring that LLMs can effectively use numerical data in reasoning tasks was a primary objective of this study.

The CrewAI framework we tested for open-source models did not yield the desired results, leading us to switch to LangGraph. We also implemented the system using a structured workflow and conditions crafted for LLMs. Limitations we faced during the implementation included the token size restrictions of both open-source and GPT models, as well as subscription-related challenges with GPT.

CONCLUSION

This framework explores the combination of logical reasoning with data-driven processes, testing the capabilities of LLMs in reasoning tasks and creating explainable, insightful procedures. The complex tasks performed by the LLMs demonstrate that the system can serve as a starting point for executing intricate

reasoning and tasks requiring coordination. Notably, the multi-agent symbolic regression represents a significant step towards understanding LLM behaviour in such processes.

The method shows promising performance in scenarios where the physics of the system is unknown or when the governing equations are dictated by physical laws embedded in the data. It also examines the reasoning and data interpretation capabilities of LLMs, revealing a key limitation: the inability to fully integrate numerical data into their reasoning and responses. Enhancing the suitability for these tasks involves presenting data in various forms, such as CSV files stored as vector databases, screenshots of data included in prompts, or scatter and insight plots.

The architecture of the supervisor agent has proven to be valuable in aiding complex reasoning, as evidenced by our experiments. Frameworks like CrewAI and LangGraph assist in creating multiagent systems, but the use of a set of LLMs as agents—designed to conditionally exchange responses with other agents and collectively decide on final outputs—has shown promising results. With minor adjustments to prompts and structure, this framework can be adapted to other complex tasks involving data analysis. It holds significant potential across the diverse applications of symbolic regression in today's world.

REFERENCES

1. Interpretable Machine Learning for Science with PySR and SymbolicRegression.jl https://arxiv.org/abs/2305.01582

2. gplearn-https://github.com/trevorstephens/gplearn
3. In-Context Symbolic Regression: Leveraging Language Models for Function Discovery https://arxiv.org/html/2404.19094v1
4. SmbolicGPT: A Generative Transformer Model for Symbolic Regression https://arxiv.org/abs/2106.14131
5. Generating Symbolic Reasoning Problems with Transformer GANshttps://arxiv.org/abs/2110.10054
6. Langgraph https://www.langchain.com/ langgraph
7. A Comparison of Recent Algorithms for Symbolic Regression to Genetic Programming https://arxiv.org/html/2406.03585v1
8. Discovering governing equations from data: Sparse identification of nonlinear dynamical systems https://arxiv.org/abs/1509.03580
9. AI Feynman 2.0: Pareto-optimal symbolic regression exploiting graph modularityhttps://arxiv.org/abs/2006.10782