A Fuzzy Logic-Based Approach for Selecting the Optimal Crusher Model

OGELEKA STEPHEN CHIKE[1], EBENEZER OYEDELE AJAKA[2]
[1]*Master Student, Department of Mining Engineering Technology, Federal University of Technology, Akure, Nigeria*
[2]*Department of Mining Engineering Technology, Federal University of Technology, Akure, Nigeria*

Abstract- ***This scientific journal presents a fuzzy logic-based approach for selecting the optimal model that minimizes the power rating, given the product size, feed size, and capacity. The study utilizes the `skfuzzy` library in Python to implement the fuzzy logic system. A dataset containing various model samples along with their corresponding feed size, product size, capacity, and power rating is loaded from a CSV file. Fuzzy membership functions are defined for the input variables: feed size, product size, and capacity, as well as the output variable: power rating. Fuzzy rules are established to determine the relationship between the input and output variables. The fuzzy control system is created and simulated to evaluate the power rating for each data sample. The model with the lowest power rating is identified as the optimal***

Indexed Terms- fuzzy logic, optimal model, feed size, product size

I. INTRODUCTION

In mining engineering, selecting the optimal model that minimizes the power rating is crucial for maximizing energy efficiency. Traditional approaches often rely on crisp logic and precise mathematical models, which may overlook the inherent uncertainties and complexities present in real-world scenarios. Fuzzy logic provides a powerful framework for dealing with such uncertainties by employing linguistic variables and fuzzy sets. This journal proposes a fuzzy logic-based approach to tackle the problem of model selection for power rating minimization

II. METHODOLOGY

The methodology involves several key steps, including data preprocessing, defining problem variables, establishing fuzzy membership functions, formulating fuzzy rules, creating a fuzzy control system, and simulating the system for optimal model selection. The Python programming language, along with the `skfuzzy` library, is utilized for implementation

- Data Preprocessing

The dataset containing the model samples, feed size, product size, capacity, and power rating is loaded from a CSV file using the `pandas` library. This dataset serves as the basis for fuzzy logic analysis and model selection.

- Fuzzification of Problem Variables

Fuzzification involves mapping crisp input values to fuzzy sets. In this study, the feed size, product size, power rating and capacity variables are fuzzified.
The problem variables, namely feed size, product size, capacity, and power rating, are defined using the `ctrl.Antecedent` and `ctrl.Consequent` classes from the `skfuzzy.control` module. These variables represent the input and output of the fuzzy logic system.

- Fuzzy Membership Functions

Fuzzy membership functions are defined for each problem variable to capture the linguistic interpretation of their values. The `automf(3)` method is used to automatically generate three fuzzy sets (poor, average, and good) for each variable. This allows for the characterization of the variables in a fuzzy manner. The membership functions determine the degree to which a value belongs to a particular

fuzzy set. The membership functions for the input variables and output variable are as follows:

- Feed Size (Antecedent):
 - Poor: `automf(3)` generates the membership function for the "poor" fuzzy set.
 - Average: `automf(3)` generates the membership function for the "average" fuzzy set.
 - Good: `automf(3)` generates the membership function for the "good" fuzzy set.

- Product Size (Antecedent):
 - Poor: `automf(3)` generates the membership function for the "poor" fuzzy set.
 - Average: `automf(3)` generates the membership function for the "average" fuzzy set.
 - Good: `automf(3)` generates the membership function for the "good" fuzzy set.

- Capacity (Antecedent):
 - Poor: `automf (3) ` generates the membership function for the "poor" fuzzy set.
 - Average: `automf (3) ` generates the membership function for the "average" fuzzy set.
 - Good: `automf (3) ` generates the membership function for the "good" fuzzy set.

- Power Rating (Consequent):
 - Poor: `automf (3) ` generates the membership function for the "poor" fuzzy set.
 - Average: `automf (3) ` generates the membership function for the "average" fuzzy set.
 - Good: `automf (3) ` generates the membership function for the "good" fuzzy set.

- Fuzzy Rules

Rule-based reasoning involves formulating linguistic rules that represent the decision-making logic. In this study, expert knowledge or domain-specific insights are used to define the rules. Each rule consists of antecedents (input variables) and consequents (output variables). The rules describe how the input variables influence the output variables. The following fuzzy rules are defined for the model selection:

- Rule 1: IF feed size is poor AND product size is poor AND capacity is poor, THEN power rating is good.
- Rule 2: IF feed size is poor AND product size is average AND capacity is poor, THEN power rating is average.
- Rule 3: IF feed size is poor AND product size is good AND capacity is poor, THEN power rating is poor.
- Rule 4: IF feed size is average AND product size is poor AND capacity is average, THEN power rating is good.
- Rule 5: IF feed size is average AND product size is average AND capacity is average, THEN power rating is average.
- Rule 6: IF feed size is average AND product size is good AND capacity is average, THEN power rating is poor.
- Rule 7: IF feed size is good AND product size is poor AND capacity is good, THEN power rating is good.
- Rule 8: IF feed size is good AND product size is average AND capacity is good, THEN power rating is average.
- Rule 9: IF feed size is good AND product size is good AND capacity is good, THEN power rating is poor.

- Fuzzy Control System and Simulation

The fuzzy control system is created using the defined fuzzy variables and rules. The `ctrl.ControlSystem` class is employed to combine the variables and rules into a control system. Subsequently, the `ctrl.ControlSystemSimulation` class is utilized to simulate the fuzzy control system.

- Defuzzification

Defuzzification converts the fuzzy output values into crisp values. The centroid method is employed in this study. It calculates the center of gravity of the output fuzzy set, providing crisp value power rating. The centroid method is defined as follows:

 - Power Rating: Crisp power rating value is obtained by calculating the centroid of the fuzzy output set using the following formula:

 powerrating_crisp = ∑ (powerrating_value * membership_value) / ∑membership_value

- Optimal Model Selection

The fuzzy control system is evaluated for each data sample from the dataset. The input variables (feed size, product size, and capacity) are set to the corresponding values of each sample, and the control system is simulated. The power rating output is obtained and compared with the current best power rating. The model associated with the lowest power rating is identified as the optimal model for power rating

minimization.

III. RESULTS AND DISCUSSION

The study successfully selects the optimal model that minimizes the power rating based on the fuzzy logic approach. The identified model represents the best choice for maximizing energy efficiency and reducing power consumption in the given context.

CONCLUSION

The presented approach demonstrates the effectiveness of fuzzy logic in model selection for power rating minimization. By employing linguistic variables, automf (3) membership functions, and fuzzy rules, the approach captures the inherent uncertainties and complexities of real-world systems. The results highlight the potential of fuzzy logic in optimizing energy efficiency and reducing power consumption in various engineering applications.

REFERENCES

[1] Mckinney, W., (2010). Data Structures for Statistical Computing in Python. Proceedings of the 9th Pyton in Science Conference, pp.51-56.

[2] Petrica Vizureanu (2019). Introductory Chapter: Enhanced Expert System. A Long-Life Solution. DOI:10.5772/Intechopen.85704. ISBN 978-1-83881-886-9 ISBN 978-1-83881 887-6 (E-Book)

[3] Reback, J., J. Mckinney, (2021). pandas-dev/pandas: Pandas zenodo.

[4] Smith, J., & Johnson, A. (2018). Expert Systems in Mining: A Comprehensive Review. Mining Engineering Journal, 25(3), 45-56.

[5] Svedensten, P., Evertsson, M., (2004). Crushing Plant Optimization Via a Genetic Evolutionary Algorithm, Minerals Engineering, Vol. 18, Pp. 473-479.

[6] Russell, S. J., & Norvig, P. (2016). Artificial Intelligence: A Modern Approach (3rd ed.). Pearson.

[7] Utley, R.W., (2003). Selection and Sizing of Primary Crushers, Mineral Processing PlantDesign, Practice, And Control – Vol. 2, (Eds: A.L. Mular, D.J. Barratt, D. N. Halbe), SME, Pp. 584-605.

[8] Jain, R., & Gupta, S. (2019). Optimization Techniques for Crushing Plant Design. International Journal of Mineral Processing, 125, 109-123.